Automatic Grammar Induction from Free Text Using Insights from Cognitive Grammar

A thesis submitted in partial fulfilment
of the requirement for the degree of Doctor of Philosophy

Vigneshwaran Muralidaran

June 2022

Language & Communication / Computer Science & Informatics

School of English, Communication and Philosophy and School of Computer Science and Informatics

Cardiff University

Acknowledgements

I started my PhD journey with fascinating ideas about languages and with the curiosity to explore those ideas further in a rigorous research environment. The exposure that I gained in the last four years at Cardiff University has given me a concrete understanding of academia and has led me to systematically develop my ideas. The journey has been challenging for me on both personal and professional fronts with lifelong takeaways in both.

I am deeply grateful to my advisors Prof. Irena Spasic and Dr. Dawn Knight for their constant trust in me and for the support they have offered me throughout. I am especially thankful to Irena for motivating me to work towards paper submissions, for giving regular feedback and comments and for encouraging me towards various professional and academic pursuits. When I had to make major revisions to the thesis and work towards resubmission, Irena spent a lot of time and effort helping me with editing the thesis and was instrumental in steering this research work to completion. If not for her support and motivation, I am sure this thesis would have never reached completion. I am very thankful to her. The resubmission period was one of the hardest times of my life and I am indebted to my undergraduate Professors Sri Adinarayanan and Smt Smrithirekha for providing me a home under the tranquil foothills of Iyvar malai and accepting me unconditionally during this phase. This gave me the much-needed space to freely work towards finishing the thesis.

I would fail in my duty if I do not remember with gratitude the generous support of my uncles, Mr. Bhavani Sankar and Mr. Krishnamurthy who offered me the initial financial help needed to move to Cardiff and start my PhD study without any hassle. I am grateful to them for their trust and confidence in me. I am indebted to my parents, brother and in-laws for being pillars of support and encouragement even when there were no visible signs of progress from my end. My wife, Vishnupriya (Priya), is the most important person who stayed with me and remained extremely supportive during some of the very difficult times during these years. Priya is the one who provided me with immense confidence even during the times when I had given serious thought about giving up on my research. If I have managed to finish this thesis today, I owe a great deal of it to her. Life in Cardiff has been memorable due to my colleagues Steve, Laura and Jenny and my friends Roshan, Shifani and their lovely kids.

I thank the University for their PhD studentship and the Welsh government for funding the project on developing a Welsh stemmer. I must also not forget to mention the School of English, Communication and Philosophy and the School of Computer Science and Informatics for providing me many part-time jobs during the four years I spent in Cardiff that helped me manage my survival there.

Abstract

Automatic identification of the grammatical structure of a sentence is useful in many Natural Language Processing (NLP) applications such as Document Summarisation, Question Answering systems and Machine Translation. With the availability of syntactic treebanks, supervised parsers have been developed successfully for many major languages. However, for low-resourced minority languages with fewer digital resources, this poses more of a challenge. Moreover, there are a number of syntactic annotation schemes motivated by different linguistic theories and formalisms which are sometimes language specific and they cannot always be adapted for developing syntactic parsers across different language families.

This project aims to develop a linguistically motivated approach to the automatic induction of grammatical structures from raw sentences. Such an approach can be readily adapted to different languages including low-resourced minority languages. We draw the basic approach to linguistic analysis from usage-based, functional theories of grammar such as Cognitive Grammar, Computational Paninian Grammar and insights from psycholinguistic studies. Our approach identifies grammatical structure of a sentence by recognising domain-independent, general, cognitive patterns of conceptual organisation that occur in natural language. It also reflects some of the general psycholinguistic properties of parsing by humans - such as incrementality, connectedness and expectation.

Our implementation has three components: Schema Definition, Schema Assembly and Schema Prediction. Schema Definition and Schema Assembly components were implemented algorithmically as a dictionary and rules. An Artificial Neural Network was trained for Schema Prediction. By using Parts of Speech tags to bootstrap the simplest case of token level schema definitions, a sentence is passed through all the three components incrementally until all the words are exhausted and the entire sentence is analysed as an instance of one final construction schema. The order in which all intermediate schemas are assembled to form the final schema can be viewed as the parse of the sentence. Parsers for English and Welsh (a low-resource minority language) were developed using the same approach with some changes to the Schema Definition component. We evaluated the parser performance by (a) Quantitative evaluation by comparing the parsed chunks against the constituents in a phrase structure tree (b) Manual evaluation by listing the range of linguistic constructions covered by the parser and by performing error analysis on the parser outputs (c) Evaluation by identifying the number of edits required for a correct assembly (d) Qualitative evaluation based on Likert scales in online surveys.

Contents Page No.

Relevant publications

Muralidaran, V. and Sharma, D.M., 2016, April. Construction grammar based annotation framework for parsing Tamil. In International Conference on Intelligent Text Processing and Computational Linguistics (pp. 378-396). Springer, Cham.

Muralidaran, V., Spasić, I. and Knight, D., 2020, October. A Cognitive Approach to Parsing with Neural Networks. In International Conference on Statistical Language and Speech Processing (pp. 71-84). Springer, Cham.

Muralidaran, V., Spasić, I. and Knight, D., 2020. A systematic review of unsupervised approaches to grammar induction. Natural Language Engineering, pp.1-43.

List of figures Page no.

List of tables Page no.

Algorithm

List of abbreviations and acronyms

Abbreviation / Acronym	Expansion
CorCenCC	Corpws Cenedlaethol Cymraeg Cyfoes (National Corpus of Contemporary Welsh)
POS	Parts of Speech
NLP	Natural Language Processing
CG	Cognitive Grammar
CxG	Construction Grammar
CPG	Computational Paninian Grammar
ANN	Artificial Neural Networks
CML	Computational Linguistics
RQ	Research Question
ACM	Association of Computing Memory
DBLP	Database Systems and Logic Programming
IEEE	Institute of Electrical and Electronic Engineers
CFG	Context Free Grammar
ADIOS	Automatic Distillation of Structure
PCFG	Probabilistic Context Free Grammar
DOP	Data Oriented Parsing
DMV	Dependency Model with Valence
CL	Cognitive Linguistics
STM	Short Term Memory
WSJ	Wall Street Journal
CEG	Cronfa Electroneg o Cymraeg (Welsh Electronic Fund)
NLTK	Natural Language Toolkit

Chapter 1 Introduction

1.1 Background

A natural language is a complex system that humans use to encode meaning in both spoken and written form. Language is a structured system that allows the speakers to construct sensible sentences from words. In this sense, it provides a framework for meaningful communication with one another. It is also a creative system that allows expression of an infinite number of concepts from a finite number of words. Languages convey highly structured information, but the raw digital representation of natural language data is unstructured. Therefore, its innate structure needs to be reconstructed and added to such representation explicitly. The explicit structure can be inferred by employing algorithms for sentence boundary detection, word tokenisation, parts-of-speech (POS) tagging, named entity recognition, syntactic parsing, semantic role labelling, etc.

Natural language processing (NLP) is a field of computer science and artificial intelligence that is concerned with developing methods that facilitate the analysis of natural language expressions in both written and spoken form. The ways in which the speech and text processing is approached are very different and this project focuses specifically on text processing. In a text document, the words are organised into larger units such as phrases or chunks. These chunks are further organised into clauses, which in turn form more complex syntactic units such as sentences. The sentences are rooted into the context of the surrounding sentences. A text is made of related, coherent and structured groups of sentences that form a unified whole by being both lexically and topically coherent instead of being a random collection of sentences. Therefore, text can be processed at various levels starting from lexical and morphological level over syntactic and semantic level, and ultimately at the high level of discourse and pragmatics.

This project deals specifically with grammatical analysis of a text at sentence level. There are two different schools of linguistics that differ in their way of describing and analysing a formal grammar of a sentence. These are generative school and functional-cognitive school of linguistics. According to the generative school of thought, a grammar is understood as a set of formal rules that defines how the words and phrases are arranged to generate well-formed sentences. A sentence is well-formed if it conforms to the grammatical rules of a given language. A sentence can be well-formed at various levels, e.g., phonological level (sound patterns of the language), morphological level (word formation patterns), syntactic level (rules of arrangement of words) and semantic level (meaning conveyed by the sentence). Within the generative school of thought, it is assumed that well-formedness at each level is independent

of other levels. For example, the famous sentence ‘Colourless green ideas sleep furiously’, originally coined by Noam Chomsky in his book 'Syntactic Structures' (Chomsky, 1957) is syntactically well-formed but semantically ill-formed.

In contrast to this, the functional-cognitive school assumes that the grammatical structure of a sentence cannot be understood independently of its meaning. In other words, all linguistic artefacts, starting from lexicon all the way to the grammar, are analysed with respect to the meaning they convey. Therefore, the meaning of a composite expression not only includes the semantic structure that represents its composite sense but also the composite path, i.e., the way in which individual components of the expression are arranged to form the composite expression. Under this view, two expressions such as *pig meat* and *pork* that express the same composite meaning have different composite paths and therefore are semantically different (Langacker, 2008 p. 39). In the same way, *green ideas* and *black intuitions*, even though their composite meaning does not emerge, are analysed as meaningful and nonsynonymous. Both schools of thought and their relevance for the field of computational linguistics as well for this particular project are described in more detail in Chapter 2.

Regardless of a particular school of thought, the linguistic task of assigning an appropriate syntactic structure to a sentence is called syntactic parsing. One of the goals of NLP is to automate this task. There are two dominant approaches to syntactic parsing in NLP, namely constituency and dependency parsing, which support two different types of grammatical analysis. Constituency parsing is based on the application of context-free rules to arrive at the best possible parse tree of a given sentence. An example of a constituency parsing for the sentence ‘The children ate the cake with a spoon’ is shown in Figure 1.1. The parse tree was generated by the syntax tree generator online[1].

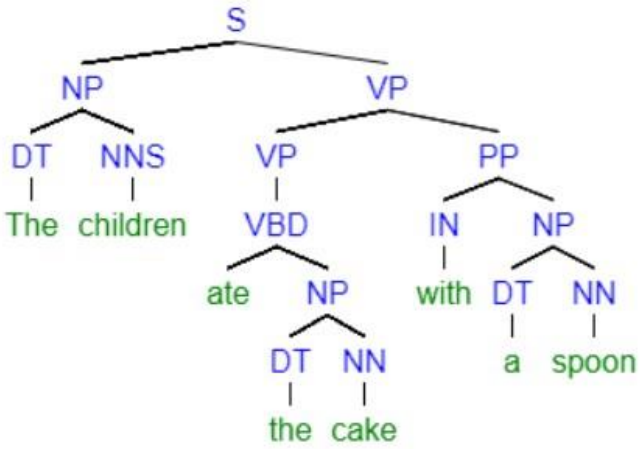


Figure 1.1 Constituency parse

[1] https://github.com/mshang/syntree

On the other hand, dependency parsing works by identifying the set of grammatical relations that hold between the individual words in a sentence. The dependency relations between words in a sentence are labelled and marked from head to its dependants. The labels are obtained from a fixed inventory of grammatical relations specified by the dependency grammar. An example of a dependency parse of the same sentence 'The children ate the cake with a spoon' is shown in Figure 1.2. The parse tree has been created automatically using spaCy's built-in dependency visualiser[2].

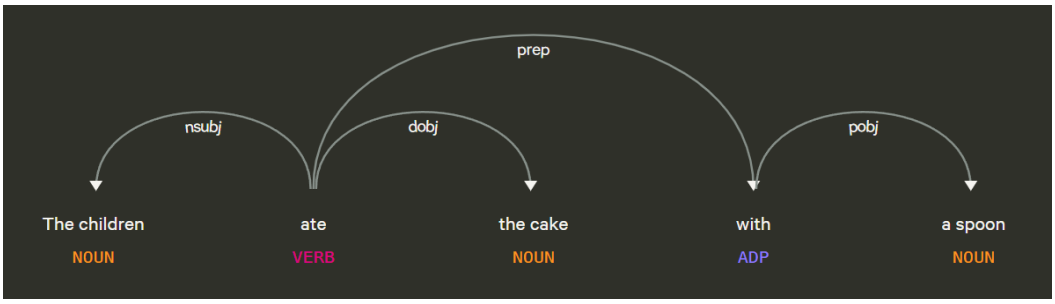


Figure 1.2 Dependency parse

The choice of a particular type of parsing depends on both a given natural language and the type of downstream NLP tasks. For example, constituency parsing is more suitable for fixed-word order languages such as English. However, it does not cope well with free word order languages where the position of words can be freely altered without altering the meaning of the sentence while preserving grammar (Covington, 1990 p.7). Conversely, dependency parsers are better equipped for dealing with free-word order languages (Debusmann, 2000). In order to annotate and learn these constituency and dependency trees, different annotation schemes have been proposed and treebanks created for different languages. Examples include Penn Treebank for constituency structures in English (Taylor et al., 2003), Stanford Dependencies for English dependency analysis (Silveira et al., 2014), NEGRA (Skut et al., 1997) and TüBa-D/Z treebanks for German (Telljohann et al., 2004), PDT scheme for Prague Dependent Treebanks (Hajic et al., 2012), Computational Paninian Grammar for dependency parsing Indian languages (Bharati and Sangal, 1993), etc. It has been shown that the accuracy of a parser depends on the different decisions taken while developing the annotation scheme used in the treebank (Maier, 2006). These decisions often depend on the type of a given natural language. One way of facilitating the development of syntactic parsers that are capable of providing consistent performance across a wide range of languages, would be to develop a parsing approach that is unsupervised and language-independent, thus requiring very little annotation. The functional-cognitive school of linguistics provides the foundational principles based on which such a parsing strategy can be developed.

[2] https://explosion.ai/demos/displacy

In contrast to the key paradigms of formal linguistics, which views syntactic structures as independent of semantics, functional approaches formulate grammar from a meaning-oriented perspective. Functional theories analyse the grammatical structure of a language as do formal theories, but it also analyses the entire communicative situation including the purpose of speech event, participants, discourse, context, etc (Bates and McWhinney, 1982; Dik, 1987, 1991). There are various schools of functional approaches to grammar based on the type of functional analysis they engage in. Amongst the functional approaches, cognitive linguistics maintains a school of thought where a language is treated as an integral part of human cognition, which operates on the same principles as other cognitive faculties such as perception, attention, memory, spatial and visual processing (Evans, 2006). There are various branches of this paradigm that give rise to different theories such as gestalt-psychology based orientation (Talmy, 1975), a cognitive discourse-based orientation (Langacker, 2001), cognitive sociolinguistic orientation and psycholinguistic orientation (Król-Markefka, 2014). We draw inspiration from construction grammar and cognitive grammar approaches, which subscribe to the idea that the knowledge of a language is based on a 'collection of form and function pairings.

There are different types of functional approaches to grammar such as Cognitive Grammar (Langacker, 1987), Construction Grammar (Goldberg, 2003), Prague functionalism (Danes, 1987), Systemic Functional Grammar (Matthiessen and Halliday, 2009), Functional Discourse Grammar (Hengeveld and Mackenzie, 2006) and so on depending on the type of functional analysis conducted in these frameworks. In cognitive grammar, a language is understood as a symbolic system all the way from lexicon to grammar. Here, lexicon, morphology and syntax differ only in the schematic complexity of the corresponding symbolic expressions. The basic units of analysis are symbolic expressions or common constructions. More theoretical details of how language can be treated fully as a symbolic system can be understood from (Langacker, 2008).

Based on our understanding from Cognitive Grammar and Construction Grammar, we propose the use of construction schemas to symbolise different types of syntagms and paradigms. A construction is defined as either a linguistic expression or a schema abstracted from linguistic expressions that captures their commonality at any level of specificity. Practically, a construction schema acts as a meaningful template that maps to some semantic conceptualisation. Any linguistic expression that satisfies the given conceptualisation is said to instantiate the construction schema. To aid interpretability of the

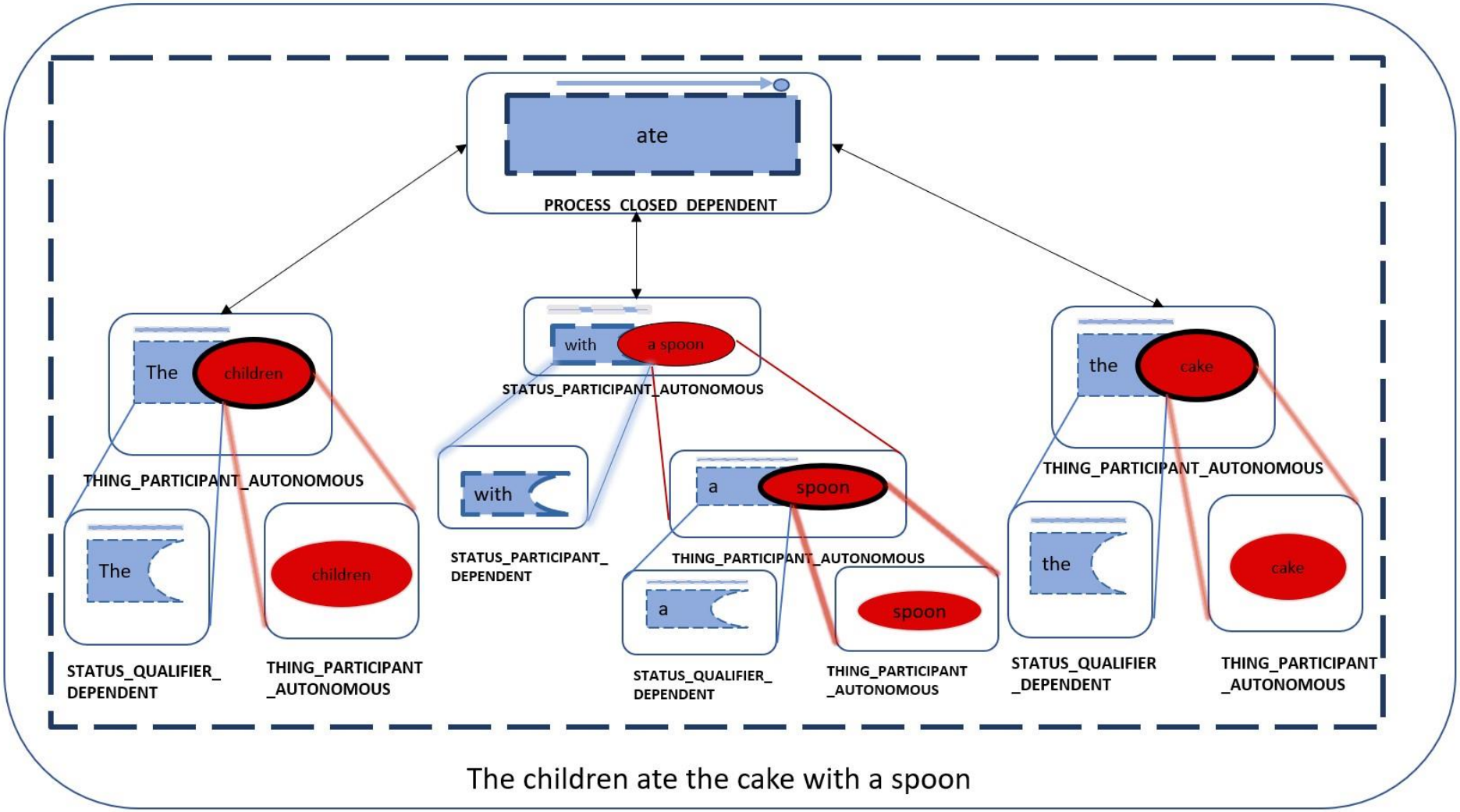


Figure 1.3 Parsing as an assembly of construction schemas

construction schemas, we labelled them based on the commonalities observed from various constructions along three dimensions that correspond to the following questions: (1) What is the content of a construction? This axis of analysis is called the *composition* of a construction (2) How does it interact with other constructions? This axis of analysis is called the *interaction* of a construction (3) Is the construction autonomous or dependent? This axis of analysis is called the *autonomy* of a construction.

Every part of a sentence can be analysed along the three axes. A sentence and its parts can be analysed as instances of various schemas at the same time and in each analysis the parts may be assembled differently. All possible assemblies can be potentially valid and therefore the same sentence can be assembled in more than one valid way. One of the feasible full parses of the sentence 'The children ate the cake with a spoon' using the proposed approach is shown in Figure 1.3.

1.2 Motivation

As we indicated in the previous section, one way of facilitating the development of syntactic parsers that are capable of providing consistent performance across a wide range of languages, would be to develop a parsing approach that is unsupervised and language independent, thus requiring very little annotation. One of the key advantages of such an approach is that it facilitates the development of NLP tools in minoritised languages, in which the resources and native speakers who may take on the role of manual annotators may not be readily available. Wales, where Cardiff University is situated, has a dual language policy that gives equal status to both English and Welsh. All public services in Wales are provided bilingually and people are able to use the language of their choice. This encourages the use of Welsh in all aspects of public life. Digital technology plays an important role in this by giving the freedom for Welsh speakers to use their first language. This effort can be supported by the availability of NLP tools such as search engines, chat bots, machine translation systems etc.

Welsh can be understood as a low-resourced language because of few publicly available NLP tools as well as large annotated corpora that can be used to fast track their development (Cunliffe et al., 2022). Some of the most recent developments include the Universal Dependencies Treebank for Welsh (Heinecke and Tyers 2019), which was developed as the third Celtic language treebank within the Universal Dependencies project (Nivre et al., 2016).

The treebank contains 10,756 tokens with a universally defined set of POS tags and dependency relations. In comparison to Penn treebank which has accrued 7 million words of POS tagged text in its eight years of operation (Taylor et al., 2003 pp. 5-22), the Welsh treebank is comparatively small.

Language Technologies Unit research group at Bangor University has driven the development of open-source NLP resources. Some of the basic NLP tools developed there include a Welsh POS tagger, lemmatiser are corpora-related research tools that have been developed and released by this group (Prys et al., 2021). In Cardiff, the University of South Wales developed Welsh Natural Language Toolkit that contained a suite of open-source NLP tools for Welsh including a sentence splitter, tokenizer, POS tagger and morphology analyser[3]. Cardiff University developed an alternative rule-based context-sensitive POS tagger, which features POS tags that are specific to Welsh (Neale et al., 2018). Lancaster University developed a semantic tagger, which was successfully adapted for Welsh (Piao et al., 2018). We complemented these open-source projects by developing a Welsh stemmer (Muralidaran et al, 2021) based on Porter's stemming algorithm (Porter, 2001). However, none of these projects developed a parser for the Welsh language. A syntactic parser is one of the basic steps in a standard NLP pipeline that processes a natural language text to determine the grammatical structures at the sentence level. The research on developing syntactic parsers for English has matured. There is now an opportunity to develop NLP tools in Welsh that not only mimic their equivalents in English but go beyond to explore novel approaches to the corresponding problems. Given the dual language policy in Wales as well as the goal of developing a language-independent approach to parsing, the two languages present an ideal opportunity to test our approach on these two structurally different languages.

1.3 Research aims and objectives

In this research we focus on automatically inducing a grammatical structure from raw sentences in an attempt to facilitate higher-level analysis of linguistic data by unifying syntactic and semantic aspects of word organisation. The advantage of this unification is that any grammatical analysis can then invoke the same syntactic frame at different levels of semantic abstraction. The idea of continuity between lexicon, grammar and semantics is not in itself novel. It has been extensively discussed in theoretical linguistics literature (Bates and

[3] https://sourceforge.net/projects/wnlt/

McWhinney, 1982; Dik, 1987, 1991; Langacker, 1987, 2008, 2009; Harrison et al., 2014 pp. 1-16). Lexicon-grammar continuum implies that there is no sharp boundary between lexical items and grammatical constructions. Therefore, words, phrases, clauses and entire sentences can have the same construction format only differing in their internal complexity (Langacker, 2008, 2009). Syntax-semantics continuum implies that the syntactic structures in a language are meaningful and therefore we can discover patterns of how syntactic structures symbolise semantic concepts. This thesis takes a fresh look at these concepts by using theoretical knowledge from cognitive linguistics to inform the development of a practical implementation of a parser that operates on the syntax-semantics continuum. Specifically, such a parser integrates both syntax and semantics to learn meaningful grammatical structures directly from the text data.

This work draws its basic outlook of linguistic analysis from usage-based, functional theories of grammar such as Cognitive Grammar (CG) (Langacker, 2008, 2009), Computational Paninian Grammar (CPG) (Bharati and Sangal 1993; Sangal et al. 1995; Kesidi et al. 2013; Das et al. 2017) and psycholinguistic studies (MacDonald et al 1994; McRae et al. 1998; Staub, 2015). The proposed approach identifies grammatical structures in a text by recognising general cognitive patterns of conceptual organisation that we use intuitively to interpret the structure of naturally occuring sentences. In other words, the parser reflects some of the general psycholinguistic properties of parsing by humans such as incrementality, connectedness and expectation. The motivation behind such an approach is that by integrating syntax and semantics, the corresponding parser minimises its commitment to a language-specific grammar and by basing its development on an unsupervised approach it becomes readily applicable to different languages including the low-resourced minority ones.

A parser can be developed either using explicit hand-written rules or the rules that are inferred from data using machine learning. The former method requires knowledge elicitation from linguistic experts, which presents a major bottleneck in efficiently bootstrapping a parser. In addition, such parsers may encode the bias from these experts and therefore may not generalise well. Machine learning has long been seen as a solution bottleneck. However, supervised approaches to machine learning suffer from data annotation bottleneck. Instead of eliciting knowledge from experts, they are now required to manually annotate the data. The annotation task itself may seem easier than that of knowledge elicitation as experts need to answer the question *What* instead of *How*. However, supervised machine learning requires

large amounts of annotated data in order to infer the underlying rules, which again poses an annotation bottleneck in efficiently bootstrapping a parser. On the other hand, unsupervised machine learning approaches can learn from unlabelled data with minimal to no human supervision. This is advantageous in the context of minority languages especially because unlabelled digital texts are relatively easier to acquire than labelled corpora which are expensive to construct. Unsupervised approaches to parsing are based on an assumption that the probabilistic patterns of a word organisation can be generated by hidden language models. Despite sophisticated inferencing approaches, the performance of unsupervised parsers is still considerably lower than that of supervised counterparts. This leads us to the main research question in this project: Can an unsupervised learning approach be used to develop a syntactico-semantic parser? Here we hypothesise that the parsing rules can be learnt from raw data without human supervision. To test this hypothesis, we have set the following objectives:

RO1: To utilise existing body of knowledge and identify research gaps in this area. Specifically, a systematic literature review is to be conducted in order to survey the existing approaches to unsupervised learning of parsing rules in terms of their theoretical underpinnings, practical implementations and evaluation.

RO2: To derive a list of requirements for an unsupervised syntactico-semantic parser that combines insights from the functional-cognitive schools of grammar and experiments on online sentence processing by humans.

RO3: To use the findings from RO1 and RO2 to design and implement a practical approach to unsupervised parsing that takes the advantages of prevalent parsing approaches in computational linguistics and NLP while introducing novelty based on the lessons learnt from the psycholinguistic properties of online sentence parsing by humans and the functional-cognitive theories.

RO4: To evaluate the effectiveness of the approach proposed in RO3 and see how it generalises across structurally different languages. The two languages spoken in Wales, Welsh and English, fit this requirement. Given the unsupervised nature of the proposed approach, a novel evaluation framework will be designed to address the lack of ground truth.

1.4 Research contributions

1.4.1. Systematic literature review

This research is founded on the results of a systematic literature review, which focused on understanding the prevalent approaches to unsupervised grammar induction. Its results have been published in the following journal article:

> Vigneshwaran Muralidaran, Irena Spasić, Dawn Knight (2020) A systematic review of unsupervised approaches to usage-based grammar induction. Natural Language Engineering, Vol. 27, No. 6, pp. 647-689

The theoretical and psycholinguistic studies identified in the review revealed the limitations of the prevailing grammar induction methods, questioned the suitability of parser evaluation by comparing its output against a gold standard and supported the idea that sequential, incremental, bottom-up approaches to grammar induction are closer to how humans acquire and process grammar. These insights helped position our research within a wider context of usage-based, incremental, and sequential systems of grammar induction in contrast to the formal, non-incremental and hierarchical view of grammar adopted by most unsupervised methods. The details of this systematic review are presented in Chapter 2.

1.4.2. Proposal of construction schemas

A novel research contribution of this project is a proposal of construction schemas for Welsh and English based on the knowledge engineering involving language experts and based on the synthesis of ideas from the systematic literature review and three other fields:

a) Cognitive Grammar and Construction Grammar
b) Computational Paninian Grammar
c) Psycholinguistic studies conducted on online sentence processing by humans

Cognitive and Construction Grammar: Cognitive Grammar (CG) and Construction Grammar (CxG) are grammatical theories that belong to the functional-cognitive paradigm in theoretical linguistics in which the shape of a linguistic structure is defined by usage and governed by the human cognition in general. These theories helped us recognise that no fundamental difference can be made between lexicon, morphology and syntax as they form a continuum along the spectrum of symbolic complexity. In this view, the entire grammar is treated as a

symbolic system where the overall structure of a sentence is motivated by a relatively small set of domain independent cognitive abilities such as categorisation, focus, comparison, schematisation, grouping, reification and so on. These abilities, typically used for non-linguistic purposes, are repurposed to support linguistic processing. More insights obtained from these theories are discussed in section 4.1 and its subsections in Chapter 4.

Based on these insights, we carefully defined a finite number of construction schemas by observing the syntagmatic and paradigmatic patterns of the constructions available in a language. First, through our linguistic observations we identified THING, PRONOMINAL, RELATIONSHIP, PROCESS, STATUS, EVENT, OPERATOR, CONTINUATIVE, COMBINATIVE, CLOSED, QUALIFIER, PARTICIPANT, DESCRIPTIVE, AUTONOMOUS, DEPENDENT, JOIN as the basic and generic construction schemas from which more specific schemas could be defined. Next, using these basic schemas, we represented the syntagms (relationship of positioning linguistic items) and paradigms (relationships of substituting the linguistic items in each other's place) of a language's grammar as more specific construction schemas. The list of these construction schemas, their definitions, meaning and examples are given in Chapter 4.

Insights from Computational Paninian Grammar: CPG is a computational linguistic framework where the grammatical relations are analysed neither syntactically nor semantically but as syntactico-semantic relations between nouns and verbs. The syntactico-semantic relations are functionally motivated from the speaker's viewpoint as well as the usage conventions of a language. More details regarding these syntactico-semantic relations are given in section 4.2 and its subsections in chapter 4. CPG provides the insight that a verb is a central part of a sentence, an activity complex, around which other parts of the sentence participate in syntactico-semantic relations. More complex grammatical structures are analysed as relations between many such action-noun complexes.

We incorporated this idea into our schema definitions by dividing the list of schemas into token level, process level and event level definitions. Each and every level of these schema definitions progressively becomes more complex such that token level schemas participate as components in process level schema definitions. Again, process level schemas participate as components in event level definitions and finally the sentence is analysed as relations between many such event level schemas.

Insights from psycholinguistic studies on online sentence processing: Based on the psycholinguistic studies, we identified the functional requirements of a parsing model so that

our parser makes use of the strategies used by the humans in processing sentences. We identified that the parsing model should be incremental, multiple parses should be allowed to compete and one that finishes first should be chosen, grammar should closely reflect the way a sentence is parsed, syntax should be treated as meaningful, the parse should be treated as an action - not as a product. More details of how we arrived at these decisions about the parsing model is discussed in section 4.3 and its subsections in chapter 4.

Based on these ideas, we allowed multiple definitions for a single construction schema so that all valid variations are allowed to compete during schema assembly. The same schema can be reanalysed as another construction schema based on its level of analysis such as token level, process level or event level. Here, parts of sentences are transparently mapped to one or more of schema definitions so that the grammar closely reflects the way the sentence is parsed.

The original contribution of this research is the integration of these ideas into our own construction schema definitions to be used in our parsing model. The core idea behind our definitions of construction schemas is that just like linguistic units organise themselves into different formal structures at syntactic levels, they organise themselves into self-similar construction patterns at different functional levels. By identifying such self-similarities at various levels such as token level, process level and event level, we were able to define potentially innumerable grammatical structures based on a finite number of construction schemas.

1.4.3. Parsing model

We extended our novel research contribution to the parsing model that we developed on top of construction schema definitions. Its early design was presented at an international conference:

> Vigneshwaran Muralidaran, Irena Spasić, Dawn Knight (2020) A deep learning approach to parsing with insights from cognitive grammar, in Proceedings of the 8th International Conference on Statistical Language and Speech Processing, Cardiff, UK, pp. 71-84

By treating a sentence as a miniature version of a text, we were able to recognise that local cohesion plays a significant role in how the grammatical structure of a sentence is organised. A sentence functions as a cohesive whole because its parts are meaningfully connected. In

line with the construction schemas, we analyse every part of a sentence along three axes: composition, interaction and autonomy, which are explained in chapter 4. When two expressions are semantically compatible along all the three axes, they assemble with each other so that they are conceived as components of a composite expression. We proposed an algorithm that reads parts of sentences incrementally and recognises their construction schemas along the three axes. It then assembles any two component schemas into one composite schema if they are compatible and parses a span of text as incremental assembly of components into composites. Within a span of text, multiple running parses are retained and the best parse is ultimately chosen. The construction schema definitions and their patterns of assembly are implemented as dictionary-cum-rules because they are fewer in number, largely language-independent and can be extended to handle language-specific variations. The dictionary specifies definitions of each construction schema in terms of parts of speech and other construction schema. It also defines how the basic schemas can assemble with each other and what will be the resultant schema. Rules are written to identify all possible construction patterns from a span of text based on the dictionary definitions. The rules also allow all possible valid assemblies between these constructions.

The number of possible assemblies increases with the sentence length. In order to constrain the number of possible analyses and choose an optimal pattern of construction assembly, we decided to use Artificial Neural Networks (ANNs). ANNs are computational systems that are inspired by the biological neural networks in the human brain and they are designed to mirror how humans analyse and work. ANNs can be used to model problems involving complex patterns because they learn from examples and apply their learning on other similar events. Unlike traditional programming, information learnt by the ANNs are distributed throughout the network and not on a database. They are also fault-tolerant i.e., even if some information is missing, the functioning of the network will not be affected. This self-adaptive, data-driven, flexible and fault-tolerant nature of neural networks motivated us to use them to recognise the optimal construction assembly that should be chosen in a context. A basic feedforward neural network component was trained to learn all valid patterns of assemblies possible in a span of text and to choose the best parse. A successful parse exhausts all the words in the sentence and ensures local cohesion and assembly at every stage of analysis. We present our approach for parser implementation in chapter 5. Given the lack of suitable evaluation metrics for unsupervised parsing identified in the systematic review, we developed a new evaluation framework, which allowed native speakers to critique the parsing results.

1.5 Thesis organisation

The thesis is organised as follows:

Chapter 2 provides a detailed systematic literature review on unsupervised approaches to parsing, evidence-based insights and findings. This chapter also discusses how the findings motivate the direction of our approach.

Chapter 3 presents the research methodology adopted in the development of the research. In this chapter we present the various stages of the research such as identification of research gaps and parser requirements, development of construction schemas through knowledge engineering, methodology adopted for parser implementation. The chapter also explains how native speakers of Welsh and English were involved in the evaluation of the parser outputs and the methodology used for reporting the results.

Chapter 4 identifies the functional specifications of a syntactico-semantic parser by synthesising the insights obtained from CG, CxG, CPG and psycholinguistic studies. A detailed discussion of the concepts that are useful for identifying the parser specifications is made in this chapter and detailed justifications for this are provided.

Chapter 5 describes the full parser implementation based on the parser specifications identified in chapter 4. Specifically, it contains the details of each component of the parser namely: schema definition component, schema assembly component and schema prediction component. A novel algorithm that makes use of these components to perform incremental cognitive parsing is presented in this chapter. This chapter presents the schema definitions for the English language and includes a discussion on how this component was adapted for use in the Welsh language.

Chapter 6 starts with a discussion on difficulties inherent in measuring the performance of unsupervised parser, then proceeds to present the evaluation methodologies that were adopted to measure the parser's performance. A discussion of the strengths and weaknesses of our evaluation strategies is also provided here. The coverage of the parser, meaningfulness of the intermediate schemas recognised by the parser, and the actual parser outputs on a sample complex sentence are shown in this chapter.

Finally, **Chapter 7** summarises the work done in this research briefly and concludes with remarks about the scope for future work in this area.

Chapter 2 A systematic review of unsupervised approaches to grammar induction

The main deliverable of the thesis is to develop an unsupervised parser for Welsh language. In order to thoroughly understand the available work in the domain of unsupervised parsing, we began our work with a detailed literature review. We published our systematic review and the results obtained from the review in the Natural Language Engineering journal (Muralidaran et al. 2020). The same article is reproduced here.

In the context of natural languages, grammar refers to a system that underlies the ability or capacity of human beings to use natural languages. Different theoretical frameworks have been proposed to formalise the principles of grammar. In the mid-1950s, Noam Chomsky developed the theoretical foundations of generative grammar (Chomsky 1957, 1965), an autonomous, formal system of rules that defines and constrains how lexical items are arranged to create well-formed sentences. This system of rules of well-formedness called syntax was central to generative theory of grammar. The generative view provided an alternative to the behaviourist theories of grammar (Skinner 2014; Bloom et al. 1974; Bloomfield 1962), which were prevalent at the time. There are different schools of the generative grammar tradition such as transformational grammar (Chomsky 1965, 1968; Jackendoff 1977; Radford 1981), generalised phrase structure grammar (Gazdar et al. 1985), lexical functional grammar (Dalrymple 2001; Falk 2011), head-driven phrase structure grammar (Pollard and Sag 1994; Levine and Meurers 2006). These approaches differ in the types of rules and representations that they use to predict grammaticality, but the study of syntactic well-formedness based on certain formal rules of arrangement is central to all of them. They share the view that only syntactic well-formedness is directly accessible for analysis and that meaning or semantics can only be studied insofar as it constitutes a compositional homomorph of syntax. Thus, syntax is treated as an autonomous system independent of meaning. Another idea related to the generative view of language is that the differences found in natural languages are just parametric variations of the universal grammatical principles that are genetically encoded in the human brain. This means that there is an innate, universal grammar hardwired in the brain with abstract properties such as distinguishing a noun from a verb, a content word from a function word and so on (Chomsky 2014, pp. 28- 32). Vocabulary, word order and many other language-specific properties are

parameters that will be set during language acquisition. In this paper we refer to these diverse approaches to grammar as *generative-formal* school of thought.

In contrast, more recently introduced theories of grammars are based on an idea that structure or syntax cannot be analysed independently of meaning or semantics. Functional and cognitive linguistics are proponents of this view. In functional theories of grammar, sentence structures are understood in terms of their functions, which can be semantic (agent, patient etc.), pragmatic (theme and rheme, topic and focus etc.), syntactic (subject, object etc.) or discursive (references, cohesion etc.) (Dik 1987, 1991). These theories explain grammatical structures by grounding their analysis in the communicative situation (Nichols 1984; Bates and McWhinney 1982; Dik 1987,1991; Givón 1983; Matthiessen and Halliday 2009). Cognitive linguistics argues that all knowledge of linguistic phenomena is conceptual in nature and that grammar is not an independent mental faculty but connected to all other general cognitive processes and structures (Evans 2006).

While generative theories imply the existence of a universal grammar, cognitive approaches to grammar treat linguistic structures as cognitive schemas or mappings between form and function that are inductively learnt through real-life language use. Cognitive grammar (Langacker 1987, 2008, 2009) and construction grammar (Östman and Fried 2005; Król-Markefka 2014; Goldberg 2003) are examples of usage-based approaches to describing human linguistic ability. Cognitive grammar argues that all linguistic units from morphemes, grammatical categories to syntactic relations are meaningful symbolic units which evoke different aspects of conceptualisation in the user's mind during language processing (Langacker, 1987, 2008). Cognitive grammar is different from generative grammars in three ways: in its centrality of meaning, meaningfulness of grammar and usage-based nature of grammar (Król-Markefka 2014). Construction grammar is a theory of grammar where the primary units of linguistic analysis are *constructions* that integrate *form* and *content*. *Form* refers to any combination of phonological, morphological or syntactic patterns or templates and *content* broadly refers to the meaning derived from semantics, pragmatics and discourse structure which are analysed in terms of conceptual structures such as image schemas, frames, conceptual metaphors, mental spaces etc (Lakoff 1988; Fauconnier 1994; Hampe and Grady eds. 2005; Lakoff and Johnson 1980). We refer to these different approaches to grammar as *functional-cognitive* school of thought.

Out of the two schools of thought, *generative-formal* school has been highly influential and dominant in theoretical linguistics. The formal grammars based on generative tradition have

found many practical applications as well. Perhaps most prevalently, they are used to describe the syntax of programming languages and compile and interpret code written in such languages (Harrison 1978; Moshier 1988). They have also been successfully applied in NLP to describe and process the syntax of natural languages. Traditionally, grammars used in this context were defined manually, for example using context-free grammar rules, which are then extended for computational implementation. Given the complexity of natural languages, such rules are not exhaustive, and this obviously creates a knowledge engineering bottleneck. In order to address this problem, data-driven methods have been used since the 1990s to extrapolate grammar from large corpora by exploiting their statistical properties (Leech 1993; Briscoe and Waegner 1992; Schabes et al. 1993). The most prominent subclass of such methods, supervised machine learning, requires syntactic categories and relations to be annotated manually beforehand. Although the task of manual annotation is much simpler than that of defining the grammar rules, this involves the development of large annotated treebanks containing millions of words and thousands of sentences (for instance Penn treebank has 3 million words of skeletally parsed text (Taylor et al. 2003)). The sheer volume of training data that should be annotated for supervised learning to perform well still presents a considerable bottleneck for knowledge engineering. Other supervised learning applications where annotations do not require specialised expertise have successfully resolved this problem through crowdsourcing (Cocos et al. 2015). Unfortunately, linguistic expertise is not readily available to attempt a crowdsourcing approach for creating large treebanks, so alternative approaches need to be considered. Unsupervised machine learning methods, which draw inferences from raw or unlabelled data, have started to find applications in grammar induction from text corpora (Klein and Manning 2004).

While the evolution of natural language processing and computational linguistics thus proceeded hand-in-hand with the evolution of generative-formal linguistic theories, computational linguists have often emphasised the divergence of aims between theoretical linguistics (TL) and computational linguistics (CML), sometimes even questioning the relevance of linguistic theories to computational linguistics (Paillet 1973; Jones 2007). This is due to two reasons. Firstly, the generative linguistic theories identify linguistic classes and describe the structural units purely based on their formal properties without functional motivations. Typically, these theories provide a *non-process, descriptive* account of the overall structural properties of language. However, computational linguistics is interested in modelling a *process* account of how linguistic data can be manipulated in specified ways to yield particular results. For instance, in computational linguistics, mechanisms for accessing

and deriving phrase structure rules require additional computational modules which are quite distinct and divorced from the core competence grammar modules described by the generative grammar frameworks. A more straightforward view of grammar, where processing is directly related to linguistic structures and meaning, is preferable. Secondly, with the rise of statistical methods and their usefulness in various computational linguistic tasks, a theory of grammar which is empirically grounded and compatible with statistical learning from linguistic usage is preferred.

Formal linguistic theories and statistical approaches in Computational Linguistics also differ in their views of ambiguity resolution. Linguistic theories focus on the human ability to recognize and form grammatical sentences. They state the formal principles that characterise the human linguistic capacity as a system and thus do not concern themselves with resolving any grammatical ambiguities. However, statistical approaches aim to assign the most probable structure out of all possible grammatical structures for a given utterance. Thus, ambiguity resolution is at the heart of Computational Linguistics. In this context, the functional-cognitive school has the advantage of mapping the linguistic structures to meaning directly. It emphasises usage-based learning, maintains the centrality of meaning in linguistic analysis, treats linguistic structures as form-function mappings called constructions and approaches syntactic well-formedness as successful symbolic assembly of form-function pairs. It holds that it is the meaning that can be accessed directly, and the syntax is learnt inductively through real-life language use. This has implications for grammar induction, which is defined as the process of learning the formal rules from a set of grammatical sentences with or without structural annotations (D'Ulizia et al. 2011). In a cognitive view of grammar, grammatical categories and relations are not available beforehand but are themselves grounded in patterns of usage and conceptualisations associated with them. According to Cognitive grammar, the essence of a grammar lies in conceptualisation whereby a symbolic link is construed between a linguistic form and its meaning. Induction of a sentence structure becomes the task of learning a composite structure as a form-meaning assembly of the components of the sentence. Inducing such form-meaning pairs from raw text can be challenging due to having access to words as symbols. The functional-cognitive school has not successfully explained how exactly this usage-based induction of grammar can be computationally modelled. A possible approach can be to identify primitive form-meaning pairs that can be encoded from basic cognitive profiles or if they can be induced from local statistical dependencies. Unsupervised grammar induction then becomes the task of inducing

grammatical categories and relations by identifying the basic word-meaning pairs and learning the patterns of their assembly.

The influence of *generative-formal* school of thought on language processing is evident from the earlier rule-based parsers to the later supervised models of parsing. A systematic study of *functional-cognitive* influences on unsupervised approaches to grammar induction has the potential to highlight any gap between the grammatical theories and the computational processing models of grammar. While surveying the parser implementations, we look for the theoretical underpinnings of these studies, their evaluation methodologies, identified baselines of evaluation and their relative strengths and weaknesses. Apart from informing us of the state-of-the-art methods and baselines, a thorough literature review can also help us identify domain-independent computational methods that might be usefully adapted to usage-based grammar induction.

This literature review takes the linguistic perspective to grammar induction. There is a statistical perspective of grammar induction that is taken in Machine Learning. It focuses on defining the model as the set of parameters and the ways they are linked together to determine the probability of a grammatical sentence. Here, an objective function is defined to allow selection of a single estimate of the parameters using a search approach that performs such estimation efficiently. These aspects are outside of the scope of the review. Instead, we concentrate on the structural aspects of defining the model through the use of grammar.

2.1 Methodology

Systematic reviews aim to identify, critically appraise, interpret and summarise all currently available evidence in relation to a given research question. Systematic reviews are common in medical and healthcare literature, but they are also becoming increasingly useful for other fields (Petticrew 2001). They follow strict scientific protocols based on explicit and reproducible methods designed to limit bias and random errors. Such protocols include multiple steps, typically: (1) identifying a set of well- defined research questions, (2) defining strict inclusion and exclusion criteria, (3) searching relevant literature databases using a carefully developed set of search terms, (4) assessing the quality of studies, (5) systematic extraction, abstraction and synthesis of evidence by multiple investigators independently. Consequently, they provide reliable conclusions and often identify research gaps to guide future research. We followed the systematic literature review methodology proposed by

Kitchenham and Charters (2007). It involves all the 5 steps of protocols mentioned above. Figure 2.1 shows the steps involved in the systematic review.

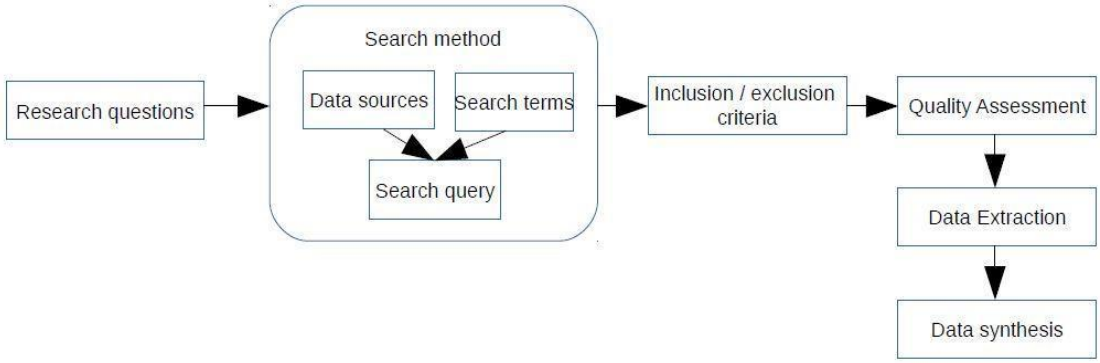


Figure 2.1 Systematic review protocol

2.2. Research questions

This review aims to determine the influences of functional-cognitive school on unsupervised approaches to grammar induction in NLP). It tries to achieve that by conducting a systematic literature review on the state of the art in Computational Linguistics and NLP that lie at the intersection of the following domains: usage-based theories of grammar, unsupervised approaches to computational grammar induction, and grammar representation. The intersection ensures that the study could be about any of the following - unsupervised parser implementation, a computational study or experimental study related to functional-cognitive school of thought, studies related to representation of grammar. The following research questions (RQs) are addressed while synthesising information obtained from unsupervised parser implementations.

RQ1. Which types of grammar theories have been the subjects of grammar induction?

RQ2. Which methods have been employed to support grammar induction?

RQ3. Which features have been used by these methods for learning?

RQ4. How were these methods evaluated?

RQ5. In terms of performance, how do these methods compare to one another?

RQ1 is concerned with the types of grammars that are amenable to automated induction. Particularly we are interested in the ways in which these grammars are represented.

RQ2 aims to identify a range of methods and techniques used to implement grammar induction approaches. Furthermore, we want to examine how these approaches address the notion of grammatical productivity, which is defined as the human capacity to keep creating new grammatical expressions by manipulating a finite set of linguistic resources.

RQ3 focuses on the types of features used by these methods and their utility for grammar induction. We also want to explore the ways in which such features can be extracted together with the associated costs, e.g., in terms of manual effort involved (e.g., for annotation) and the volume of data needed to train the methods.

RQ4 is concerned with the performance of existing grammar induction methods. In particular, unsupervised methods are known to be notoriously difficult to evaluate. To that end, we want to identify which evaluation measures have been used in practice and whether they are transferable across different methods thereby allowing them to be compared and ultimately establish the baseline performance as part of RQ5.

RQ5 aims to consolidate the findings from various studies and interpret the results obtained by comparing their relative strengths and weaknesses. It can help us understand the implications of these results for future research.

2.3. Search strategy

In order to efficiently identify a set of articles relevant to the given research questions, we compiled a list of appropriate search terms. First, we identified a set of relevant domains and then compiled a list of keywords related to each domain. Search queries based on these keywords were tested against relevant literature databases, which include ACM Digital Library, Cardiff University Library Search, DBLP, Google Scholar and IEEExplore. The abstracts and keywords from the retrieved results were screened to check their relevance and identify other pertinent search terms including synonyms and spelling variations. The list of search terms was refined iteratively in this manner until no significant changes could be made. Table

2.1 provides the finalised list of search terms, where a wildcard character was used to address inflection and derivation. Finally, a Boolean OR operator was used to combine the search terms for each domain. These subqueries were then combined using a Boolean AND operator.

Table 2.1 Search query construction

Domain	**Search terms**
Usage-based theories of grammar	gramma* gramma* cat* syntax cognitive construction usage-based form-meaning construal schema* linguistic functionalist pattern
Computational grammar induction	induction inference acquisition parsing learning processing analysis synta* struct* parse chunking unsupervised
Representation of grammatical structure	representation formalism model

	framework finite-state automata incremental trees networks assembl* neural net* graphs shallow pars*
Review papers on grammar induction	survey review bibliographical study meta-analysis

2.4. Selection criteria

The retrieved articles were manually curated with respect to their relevance to the given set of research questions. To formalise the curation process, we defined a set of inclusion and exclusion criteria.

Inclusion criteria

1. The article has to contribute to the field of natural language processing or cognitive linguistics.
2. The article has to focus on grammar induction.

Exclusion criteria

1. Psycholinguistic studies which look at neurobiological factors of linguistic phenomena.
2. Studies which are language-specific or construction-specific.
3. Articles that were not peer reviewed.
4. Certain types of publications such as editorials, informal articles, tutorials and posters.

5. Articles written in a language other than English.

Apart from this, we also followed up the citations from the studies included in our list and if any of those cited studies passed our inclusion criteria, we updated our selection list with these studies as well.

2.5. Quality assessment

All selected articles underwent quality assessment based on the following criteria:

- The research goals, methodology and contribution to the field are clearly defined.
- Data used in experiments are described with sufficient detail.
- The results are reported using theoretically sound evaluation metrics and compared, where appropriate, against a relevant baseline.
- Limitations of the study are carefully analysed.

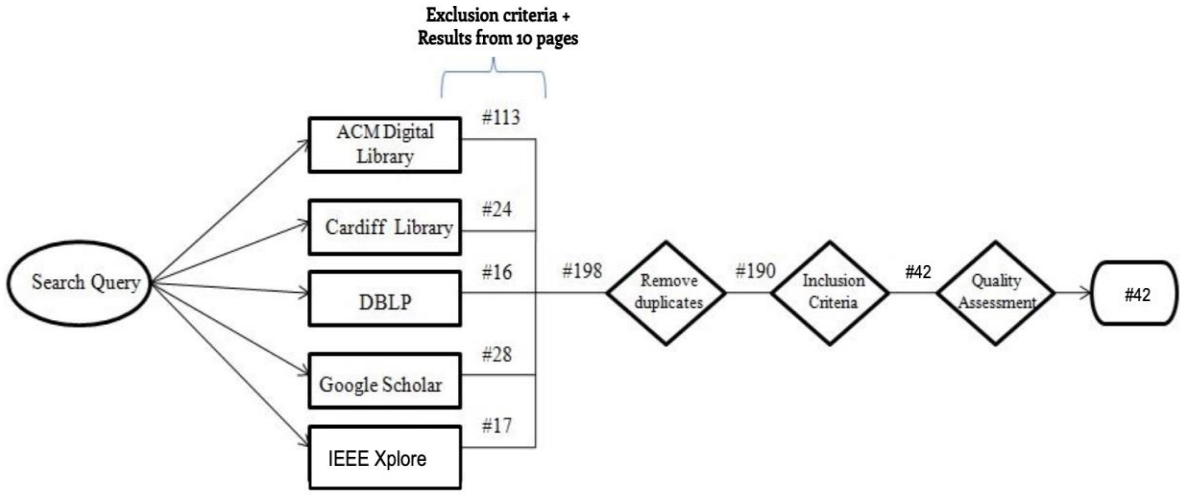


Figure 2.2 Search and selection of papers for systematic review

The results returned by the top ten pages only were chosen and exclusion criteria were applied on them which resulted in 198 articles. A total of 190 articles that remained after removing the duplicates (i.e., the same study retrieved from multiple data sources) were assessed according to inclusion criteria in section 2.4, which reduced the number of articles to 42. All the 42 studies described their goals, methodology, experiments, results and analysis according to the quality assessment criteria listed above. This process of searching and selecting the appropriate papers for systematic review is shown in Figure 2.2. One study, which was known to be relevant, was not retrieved, but was added manually, thus providing a total of 43 studies reviewed here.

2.6. Data extraction

To answer our research questions, evidence was extracted from the final set of 43 articles to review and formatted according to predefined data extraction cards. The cards populated with information from articles describing practical implementations had the following fields: title, grammar theory, grammar representation, computational model, methodology, features, dataset, evaluation metrics, results and summary. The cards populated with information from articles describing theoretical and experimental studies had the following fields: title, aim, method, experiment, conditions, result, insights and summary. This systematic data extraction allowed us to compare the evidence obtained from various studies and support generalisation from the observed evidence. The studies considered in our review and their references can be accessed from Appendix A.

2.7. Data synthesis

Table 2.2 provides a brief summary of the extracted data. Towards generalising the findings, we stratified the articles across different aspects considered in Table 2.2. They help us quantitatively determine the influence of functional-cognitive school on the studies retrieved from the data extraction step. Each study can be from one of these three categories: (a) a computational study or an experimental study on grammar induction or processing (b) an implementation of unsupervised grammar induction (c) a survey or evaluation paper on grammar induction. Each category of study offers different kinds of information. Data obtained from experimental studies on human subjects are *goal, methodology, experiment, result, insights* and *summary*. Data obtained from survey or evaluation papers is *a list of findings from the survey*. Data obtained from implementation studies are *theory of grammar, representation of grammatical productivity, processing grammatical productivity, output representation, evaluation strategy, scale of training, features used for grammar induction, approach or methodology behind the implementation*. The studies used for data extraction and a brief list of findings synthesised from them are included in the Appendix A.

The synthesis of data analysed along these aspects can answer the research questions mentioned in section 2.2. While most of these aspects are self-explanatory, the rationale behind identifying these aspects needs a discussion before we present them in detail later. *Theory of grammar* aspect tries to understand the linguistic school of thought an implementation is influenced by. *Representation of grammatical productivity* and *Processing*

grammatical productivity are two aspects which reveal how a study approaches syntactic productivity computationally. To treat productive structures in natural languages with recursive, hierarchical application of productive formal computational operations can indicate the influence of generative-formal thought on the model of parsing. Through the two aspects said above, we see if the approach to grammatical productivity in a given study is purely in a generative-formal sense or if it explicitly or implicitly is amenable to a functional-cognitive standpoint. *Representation of grammatical productivity* is understood from how the study treats the productive elements of the language in its output: a hierarchical arrangement of productive linguistic units in the form of constituency or dependency trees shows a generative-formal influence. Any computational representation that does not implicitly assume recursion and hierarchy but reveals productivity of language through meaningful assembly of usage patterns indicates functional-cognitive influence. *Processing grammatical productivity* is understood from how the decoding is done in the parsing model. An incremental decoding indicates that grammatical productivity is processed incrementally, even partial non-sentences can have their parse, and not all words of the sentence need be available before parsing begins. Such computational models are compatible with cognitive grammar ideas.

The aspect *Output representation* reveals the various types of grammar outputs found in the data. These output representations can sometimes be formally similar or in some cases intrinsically tied to the grammar itself. For example, the construction slots as discussed in (Dunn 2017a, 2017b) are filled with syntactic phrases and are formally equivalent to a phrase structure in a constituency tree. However, given the overall scheme of the paper, which tries to learn an optimal construction grammar by allowing various levels of representation from lexical to semantic; we see that the slot could potentially be filled with any entity, even entirely functional ones. Thus, we treat the construction slots as a distinct kind of output representation. Similarly, the directed multigraph learnt by the ADIOS implementation (Solan et al. 2004) is directly equivalent to the Context Free Grammar learnt by the system. However, we see that the method itself is amenable to extending learning form-function pairs that it is useful to recognise as different from a regular constituency or dependency tree. *Evaluation strategy* identifies how the performance of the system is judged. *Scale of training* indicates if the learning is fully unsupervised or semi-supervised. Finally, the *approach or methodology* is an aspect that identifies what is the computational method used for grammar induction in a given study.

Another point to note is that many studies can fit in more than one of the categories of the relevant aspect. For example, the same study can show evidence for multiple types of evaluation strategies in different types of experiments; or the same study can be analysed as adopting multiple approaches to grammar induction. In all such cases, we include the study in all the relevant categories and the total counts shown in Table 2.2 reflect that. The details of the actual data extracted can be found in Appendix A.

Table 2.2 Criteria and their values synthesised from 43 studies

Criteria	**Classification**	**Total**
Type of study	Implementation of grammar induction	33
	Theoretical or Experimental studies	7
	Survey or Evaluation paper	3
Theory of grammar	Generative-formal	22
	Functional-cognitive	3
	Theory- neutral	8
Representation of grammatical productivity	Hierarchical	31
	Non-hierarchical	2
Processing grammatical productivity	Incremental	1
	Non-incremental	32
Output representation	Constituency trees	14
	Dependency trees	11
	A directed multi-graph of patterns learnt	4
	Construction slots or Templates	2
	Transient structure	1
	Common cover link set	1

Evaluation strategy*	Comparing against gold standard treebank annotation	25
	Performance of the learner in grammaticality judgement tasks	2
	Agreement between two highly-constrained models on disjoint corpora	1
	Qualitative evaluation	1
	Maximum-coverage, minimum-size, stability measures	3
	Agreement with CFG grammar that generated the learning data	2
Features used**	POS tags	18
	Valence, Direction of attachment	10
	Distributed representation of words	5
	Word alignments	1
	Chunks or Bracketed sequences	1
	Orthographic cues	2
	Heuristic rules	4
	Construction association measures	2
Grammar induction approaches***	Top-down dependency grammar models and their variations	10
	Exemplar based models and their variations	2
	Distribution based	3
	Clustering approaches	2

	Heuristic approaches	2
	Automatic Distillation of Structure (ADIOS)	4
	Probabilistic Context Free Grammar (PCFG)	3
	Chunkers and their extensions	1
	Data Oriented Parsing and variations (DOP)	4
	Construction grammar induction	2

* Discrepancy in total count. Study number 26 in the Appendix A uses 2 different evaluation methods.

** Discrepancy in total count. Different studies use multiple features.

*** Study numbers 10 and 32 in the Appendix A use more than one approach for grammar induction

These different aspects of synthesis are discussed in the following sub-subsections.

2.7.1 Theory of grammar

In the introduction of this chapter, we reviewed different theories used to formalise the notion of a grammar. We classified all studies describing practical implementation of grammar induction with respect to the underlying grammatical theory. We could identify 3 classes of theoretical views underlying the implementation. They are:

A. *Formal-generative* grammar implementations which explicitly learn a system of rules or constraints that describe how lexical items are arranged in order to form a grammatical sentence.
B. *Functional-cognitive* grammar implementations in which the linguistic capacity is explicitly modelled as an inductive process based on language use in practice.
C. *Theory-neutral* implementations are amenable to be adapted to incorporate functional-cognitive insights.

We distinguished between *Functional-cognitive* and *theory-neutral* implementations when the latter did not necessarily learn a usage-based representation of grammar as the output. We found that although domain-independent bottom-up methods can be used to extract usage-based schemas, most of these studies learnt a formal grammar with their output

represented by the likes of constituency or dependency trees. Recognising such theory-neutral implementations thus becomes important because they can be adapted to learn fully usage-based grammar in an unsupervised fashion.

We found that the vast majority (22 out of 33) of the studies showed the influence of formal-generative school of thought in their parsing models. A top-down dependency grammar model for unsupervised parsing exemplifies this view. Given a sequence of words, the model starts with a head, attaches a sequence of arguments to the left or right and generates a dependency tree in a top-down manner. Here, the dependencies are formal syntactic relations which are defined *a priori* by linguistic theories and are learned statistically. Klein and Manning's (2004) corpus-based induction of syntactic structure laid the foundation for a statistical generative model to unsupervised dependency parsing. Overall, we found that there were eleven studies (out of 22) that implemented the top-down dependency model of grammar or its variations (Klein and Manning 2004; Spitkovsky et al. 2010, 2011, 2012; Sangati 2010; Mareček and Žabokrtský 2012a, 2012b; Gillenwater et al. 2011; Boonkwan and Steedman 2011; Headden et al. 2009; Dominguez and Infante-Lopez 2011). The remaining 11 implementations (out of 22) adopted different methods to learn formal grammar. They are described below.

Snyder et al. learn constituency trees using an unordered tree alignment model where word alignments from bilingual corpora with their parts of speech are used as features to loosely bind the parallel trees from different languages (Snyder et al. 2009). An exemplar-based approach to unsupervised parsing was proposed by Dennis (2005) where the parse tree of the target sentence is obtained by aligning it with nearest neighbour exemplar sentences and choosing a constituency tree with minimum cost for alignment. There were three studies that used distributed representations of words in deriving the parsed trees. One was by Brooks (2006) where distributional representation of words was used to segment text into constituents and heuristics were then applied to reduce the number of possible candidates (Brooks 2006). Seginer's algorithm (2007) captures the skewness of syntactic trees in its syntactic representation, restricts the search space by processing utterances incrementally (like humans do) and relies on the Zipfian distribution of words to guide its parsing decisions. The other study was Søgaard's (2011) implementation which presents a very different approach to unsupervised dependency parsing. They explore the view that dependency structure can be treated as a partial order on the nodes in terms of saliency or centrality. Their implementation assigns a dependency structure to a sequence of *n* words in two stages. In

the first stage they decorate n-nodes with word forms and distributional clusters, construct a directed acyclic graph in O(n^2) and rank the nodes using iterative-graph based ranking (Brin and Page 1998). In the second stage, a parse tree is constructed from the ranked list of words (Søgaard 2011). Another implementation of Ponvert et al. (2011) starts with learning chunks using standard probabilistic finite state models and then cascades this chunker to achieve constituent parsing. Two studies used heuristic models (Santamaria and Araujo 2010; Araujo and Santamaría 2010). One study used a clustering approach and explicit formal syntactic features to learn constituency trees (Reichart and Rappoport 2008). Reichart and Rappoport (2008) propose a labelled grammar induction by starting from Parts of Speech tags (POS tags), followed by the induction of initial brackets, subsequently labelling them and finally clustering the label outputs using syntactic features. There were two studies (Jin et al. 2018; Adriaans et al. 2000) that treat parsing as a task of learning the probabilistic context-free grammars. In summary, these 22 studies exhibit the *formal-generative* view either implicitly or explicitly.

Only three studies explicitly considered grammar as a usage-based system and modelled the grammar induction process from this theoretical perspective. The use of a construction grammar induction algorithm is an example of this type of implementation (Dunn 2017a, 2017b). Given a corpus of sentences, this method statistically allows all potential linguistic generalisations from the observed sentences and chooses the optimal inventory of construction slots that can generalise them. In one study, Jonathan Dunn (2017a) demonstrates the learnability and falsifiability of construction grammar. Learnability is the degree to which the optimum set of constructions can be consistently selected from the large set of potential constructions; falsifiability is the ability to make testable predictions about the constructions present in a dataset. The study evaluates these two by performing an induction task. In another study, the same author describes in detail how a construction grammar learner can be implemented and evaluated (Dunn 2017b). An algorithm that achieves this by using frequency and association measures of co-occurrence at multiple levels of analysis (lexical, syntactic and semantic levels) is proposed. Marques and Beuls (2016) propose proof-of-concept evaluation strategies for computational construction grammars. They take inspiration from existing measures in semantic parsing and machine translation and propose two new metrics for this evaluation task. These 3 studies complete discussion of grammar induction or evaluation from an explicit usage-based view thus showing their compatibility with *functional-cognitive* school.

Finally, the remaining eight studies proposed methods that can learn significant usage

patterns bottom up, but they were explicitly not framed to learn usage-based grammars. These methods were used to learn constituency trees and CFG rules and evaluated against formal syntactic annotated trees. The ideas themselves need not be tied to learning formal syntactic trees and can be easily adapted to incorporate ideas from functional-cognitive theories. We call such studies *theory-neutral*. In our review, we found two major implementation methods that were theory-neutral. First method was the Automatic Distillation of Structures (ADIOS) algorithm, which incrementally learns a model of morphosyntax from raw input by distilling structural regularities and contextual cues. At the end of learning, a directed multigraph containing an abstraction of patterns, which can in turn be represented as context-free writing rules, is produced. The second method was Data Oriented Parsing (DOP) where an unsupervised DOP model allows all possible binary trees to be built bottom up and computes the most probable tree from the shortest derivations of sentences. This model can be used to explain both rule-based and exemplar-based properties of a natural language.

There were four studies which used the approach of ADIOS to learn significant syntactic rules from data without any annotation whatsoever (not even POS tags) (Solan et al. 2004; Brodsky and Waterfall 2007; Berant et al. 2007; Edelman et al. 2005). One of these implementations by Brodsky and Waterfall (2007) proposed a method for evaluating the grammar learned by the ADIOS system even in the absence of any gold treebank for quantitative evaluation, or human judgement for qualitative evaluation. Their observation is that when two highly-constrained models, which are trained on disjoint corpora, agree very closely on the acceptability of a given test sentence, it cannot be coincidental. An agreement between two such models indicates that the scoring of acceptability based on this agreement is correct. This was the basis of their evaluation method.

For Data Oriented Parsing and its variations, we identified four implementations in our review (Bod 2006, 2009; Zuidema 2006; Post and Gildea 2013). The authors point out that their implementation methodology is compatible with the usage-based theoretical views such as construction grammar, but they have used the method to learn constituency trees and evaluate them against a gold treebank. By allowing all possible binary tree substructures that combine to form the final parse tree through a formal operation called labelled substitution, DOP obtains the optimal parse by computing the most probable tree from among the shortest derivations of sentences. We consider these eight studies as theory-neutral implementations.

2.7.2 Representing grammatical productivity

We have previously introduced the notion of grammatical productivity as the speaker's capacity to produce novel utterances of virtually limitless length using a finite number of linguistic resources at hand. In our review, we noticed that there were two ways to represent grammatical productivity in all studies considered. They are:

A. Hierarchical arrangement of productive linguistic units, e.g., constituency and dependency trees.
B. Non-hierarchical representation of productive units, e.g., sequence of construction slots.

We found that 31 out of 33 studies represent grammatical productivity hierarchically whereas the remaining two studies do it non-hierarchically (Table 2.2). Here we noticed that the 22 studies that take the generative-formal view of grammar represent this productive capacity of natural language as a constituency tree or a directed dependency tree with either phrasal hierarchy or the head-dependent relations. The three studies on construction grammar implementation and evaluation represent their output as a sequence of construction slots, which is non-hierarchical. The study itself fills the construction slots with phrase structure heads which are indeed hierarchical. This might raise the question as to why this output is considered non-hierarchical. The rationale is that the study indeed allows all types of information, from lexical to semantic, to fill its construction slots and demonstrates the feasibility of learning construction grammar. In the larger scheme of the paper, we recognize that the authors learn the construction grammar with various co-occurrence metrics and therefore the fact that phrase structures are filled as entries in the construction slots does not make it central to the idea of the construction slot itself. The construction slot representation is perfectly compatible with any type of assembly of symbols which are non-hierarchical. It is not inherently motivated to be filled with formal phrasal heads and hence we identify the three studies as showing a non-hierarchical approach to representing grammatical productivity. Finally, the eight studies that are theory neutral in the sense defined in the previous section 2.7.1 represent their output hierarchically. The four DOP implementations learned how phrases should be attached in a constituency hierarchy. In the four ADIOS implementations, the structural patterns distilled from the data are equivalent to the recursive phrase structure rules.

2.7.3 Processing grammatical productivity

The next aspect that we assessed was how the processing of grammatical structure was treated in each study. We identified two approaches:

A. Incremental processing
B. Non-incremental processing

In this study, by incremental processing of grammatical productivity, we mean that the analysis of grammatical structure of a sentence is incrementally updated when new sequences of words are observed; not a model which starts with all the n words w1, w2... wn and which treats parsing as establishing or filling the grammatical relations between those *n* words. As discussed in Section 2.7, in incremental decoding even partial non-sentences can have their parse, not all words of the sentence need be available before parsing begins. When the decoding itself is incremental in an implementation, we call the processing of grammatical productivity incremental. Otherwise, it is non-incremental. Incremental parsing is psycholinguistically motivated. Data synthesis revealed that several experimental and theoretical studies deemed incremental parsing suitable for grammar induction (Cramer 2007; Frank et al. 2012). According to functional-cognitive school, grammatical structure of a sentence is almost entirely overt and it does not conceal a deeper level of grammatical organisation (Langacker 1987 pp.46-47). A sharp distinction between competence (speaker's mental grammar) and performance (sentence production and processing) is also not maintained (Jensen 2014). Cognitive grammar literature also proposes incremental contribution of components to the holistic conceptual structure of a text (Harrison et al. 2014 pp. 22,100). These properties make functional-cognitive school compatible with the psycholinguistic literature. Although generative grammars can also be used to parse a sentence incrementally, the grammar formalism and derivation/parsing strategy are divorced. As mentioned in the beginning of the chapter, 'mechanisms for accessing and deriving phrase structure rules require additional computational modules which are quite distinct and divorced from the core competence grammar modules described by the generative grammar frameworks.'

In our review, we found that 32 out of 33 studies were non-incremental in processing the grammatical productivity whereas one study learned grammar incrementally. The reason to distinguish these two types is to see if an implementation can be extended to incorporate

functional-cognitive insights. The incremental parsing algorithm by Seginer (2007) uses a new link representation for syntactic structure which allows a prefix of an utterance to be parsed before the full utterance has been read.

It should be noted that there could be studies where the unsupervised model that learns significant grammatical patterns could be incrementally trained. Corpus-level incremental training is an important consideration for any scheme of grammar learner. However, we explicitly did not consider it a factor of analysis in this section because its importance is equally relevant for grammar learners of any theoretical persuasion. We wanted to understand what aspects can reveal the influence of one grammatical school or the other. For example, the ADIOS method extracts statistical patterns incrementally from a corpus of sentences. When new sentences are observed in the corpus, the algorithm learns new patterns and updates the directed multigraph. However, to decode the parse of a target sentence, the entire sequence of words of the target sentence is considered non- incrementally. We do not count such studies as processing the grammatical productivity in an incremental fashion.

2.7.4 Output representation

The next aspect of analysis is the output representation of the grammatical structure of a sentence. We identified six types of output representations from the 33 studies. They are:

A. Constituency trees
B. Dependency trees
C. A directed multi-graph
D. Construction slots or templates
E. Transient structure
F. Common cover link set

Let us consider an example sentence and illustrate these output representations. For a sentence ‘The man cut an apple with a knife’, the various output representations after grammar induction are illustrated in Figures 2.3 to 2.7.

A Constituency tree shows the phrase structures that make up the sentence. In this analysis every sentence is broken down into smaller phrases which combine with other such phrases to form a larger phrase until the entire sentence is analysed as well-formed according to the phrase structure rules. This is shown in Figure 2.3.

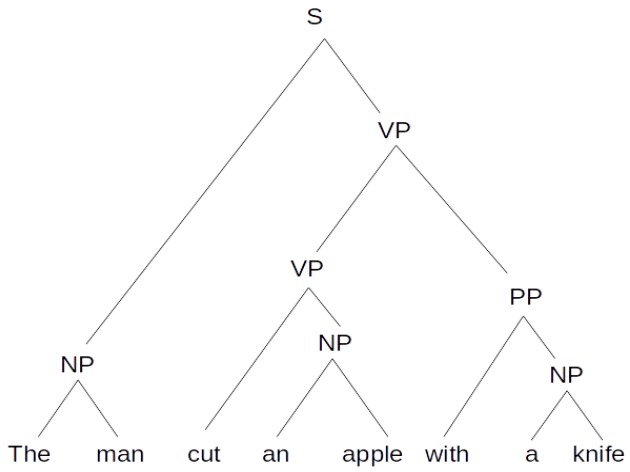


Figure 2.3 Constituency representation

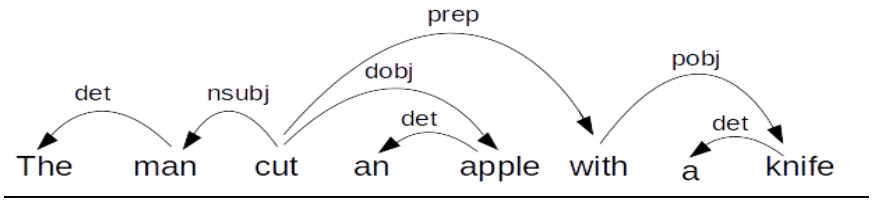


Figure 2.4 Dependency representation

In dependency analysis, a sentence is analysed in terms of the words (constituents) and their relationships. A dependency tree is a directed graph where each node is a word (constituent), child nodes are marked as dependents of parent nodes and the edges between the dependents and heads indicate their syntactic relationship. This is shown in Figure 2.4.

ADIOS implementations learn a directed multigraph where incremental update of usage patterns is done based on structural similarities and statistical information present in the text. In this technique frequent strings of similar structure are treated as significant patterns and these patterns are treated as 'Equivalent Classes' which means members of the same equivalent class are valid alternatives in a usage pattern. ADIOS algorithm repeatedly applies its pattern recognition algorithm on all sentences in a text and outputs one directed multigraph showing both patterns and equivalent classes. A sample directed multigraph with just 3 sentences is shown below in figure 2.5. More details can be obtained from (Solan et al. 2004; Brodsky and Waterfall 2007; Berant et al. 2007; Edelman et al. 2005).

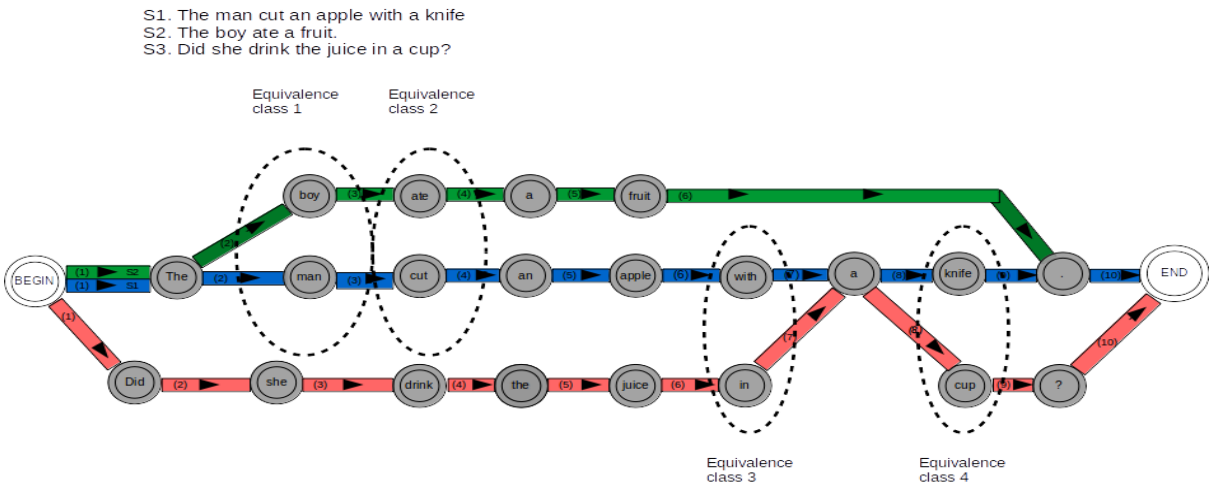


Figure 2.5 An initial directed multigraph for a simple corpus of three sentences

Constructions are productive and schematic form-meaning mappings that differ in their size, level of schematicity and their internal complexity. When it comes to construction grammar, the difficulty is not so much about the representation of constructions as much it is about learning and evaluation of these constructions. One representation that we identified in our review is a simple sequence of construction slots. For the given example, a construction slot sequence could be [NN PHRASE - VB PHRASE - NN PHRASE - PREP PHRASE] or [NP <ANIMATE> – VB <TRANSFER> – NP <ANIMATE> – NP] or [NN – “give” – NP <ANIMATE> – “a piece of” – NP <ANIMATE> – “mind”]. It should be noted that the construction slot can be filled with phrasal heads, semantic information and idioms, lexical information and so on. It is not tied to any particular type of formal linguistic category. These slots can be filled with any type of symbolic assembly with a meaningful form-function mapping. In our review, we found that the implementations adopting this output structure are compatible with functional-cognitive school of thought even though the authors themselves have used syntactic and semantic parse information to demonstrate that construction grammars are indeed learnable. More about the learning method and its relevance for future research is discussed in section 2.8.

In computational linguistics it is common to view both linguistic production and parsing in terms of a chain of consecutive operations over the linguistic structure. This chain of operations is called a transient structure in Fluid Construction Grammar. There was one study in our review which used transient structure as its output representation after parsing. (Marques and Beuls 2016). An example of a transient structure is shown in figure 2.6.

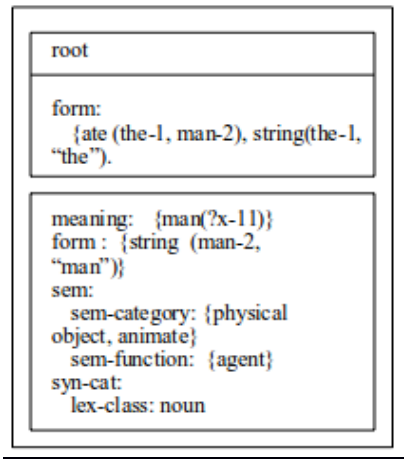


Figure 2.6 Transient structure

Seginer (2007) introduced a representation of syntactic structure that is similar to dependency structure but suitable for incremental parsing. It is based on links between a pair of words and it defines the shortest common cover link sets for a given utterance and its bracketing. An example of the shortest common cover link set for a sentence is given in figure 2.7. Although it looks like a dependency structure, there are two differences. The first difference is in the linking of the noun phrase 'the boy' where the link goes back and forth between the two words, which is not the case in a dependency tree. The second difference is based on the property called adjacency, which makes a connection between 'know' and 'the'. Such a connection does not occur in a standard dependency tree.

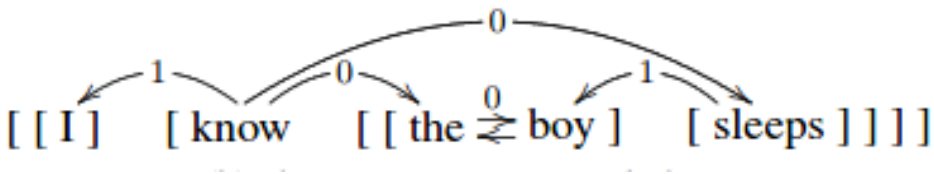


Figure 2.7 Shortest common cover link set

The distribution of the output representations for the 33 studies is shown in figure 2.8 and table 2.2. It can be seen from this distribution that overwhelmingly 25 studies out of 33 represent their outputs in the form of constituency or dependency trees.

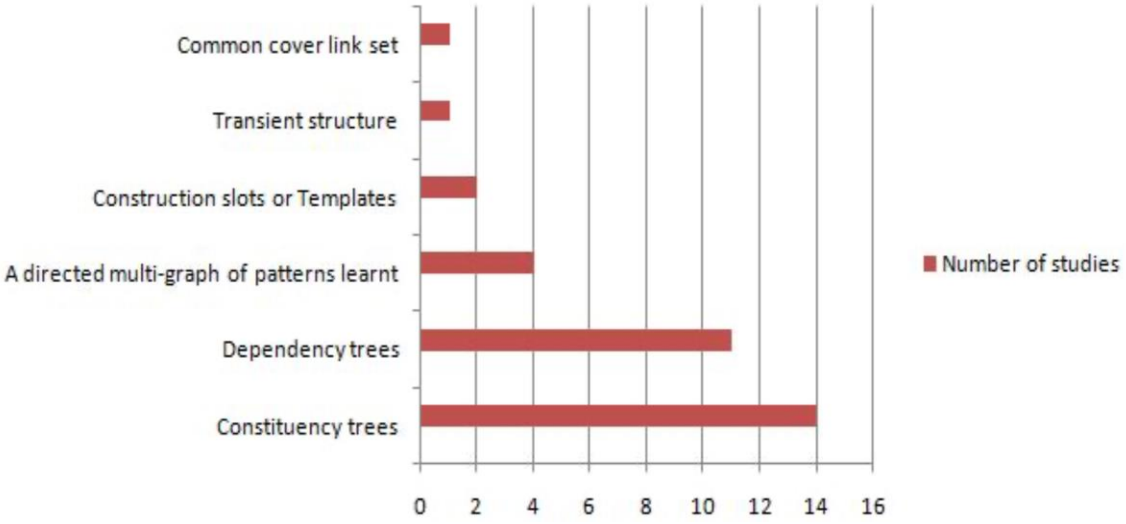


Figure 2.8 Output representations

2.7.5 Evaluation strategies

The following evaluation strategies have been extrapolated from the 33 studies considered:

A. Comparison against the gold standard.
B. Performance of the learner in grammaticality judgement tasks.
C. Comparison of models trained on disjoint corpora with respect to sentence acceptability.
D. Qualitative evaluation.
E. Maximum-coverage, minimum-size, stability measures for a usage-based grammar.
F. Output is compared against an artificial grammar that generated the test corpus.

Comparing the system output against the gold standard is a widely used method for evaluating performance of NLP systems. In the context of formal grammars, this strategy requires a treebank, a parsed corpus in which each sentence is manually annotated with syntactic relations to be used as the gold standard. Here, the treebank annotations need to be compatible with the system output, e.g., if the system learns a directed dependency tree as its output representation, then the gold standard also needs to be annotated with the directed dependencies. Manual annotation involves a variety of activities such defining an annotation schema, writing annotation guidelines, training experts for the annotation task and achieving consensus (Neves and Ševa 2019). Consequently, this time-consuming and labour-intensive process may create a bottleneck in this evaluation strategy when application-specific annotations or data are required. In addition, given the inductive nature of usage-based grammars, the use of a gold standard, which inevitably reflects the theoretical

commitments of human annotators, seems counter-intuitive (Clark and Lappin 2010). Nonetheless, we observed that the majority of practical implementations (25 out of 33) were evaluated against the gold standard treebank. This was facilitated by the fact that most methods considered represented their outputs as either constituency or dependency trees for which the existing treebanks such as Penn Treebank or Prague English Dependency Treebank can be re-used.

Alternatively, a grammar induction system can be evaluated by testing how well it performs in grammaticality judgement tasks. For example, the Goteborg multiple-choice test consists of 100 sentences, each containing an open slot. There would be three choices for each question and one of them should be filled in the open slot for the sentence to make the sentence grammatical. If a system has learned the grammar of a language, the open slot would be filled with the correct option. Solan et al. (2004) and Edelman et al. (2005) subjected their systems to such tasks in the form of multiple-choice questions used in English as Second Language (ESL) classes. The authors demonstrated that as the number of sentences processed by the system increased so did the proportion of multiple-choice questions answer correctly. The performance of the system proposed by Solan et al (2004) was also compared against that of a bigram language model i.e., a language model where probability distribution of every two-word sequence in a text is calculated. The ADIOS model outperformed the bigram model by answering 60% of questions correctly which is equivalent to the average score of 9th grade ESL students. The bigram precision benchmark is 45%. This is an extrinsic evaluation of the grammar induction system by looking at its performance in the context of its application. Unlike the gold standard evaluation, this pragmatic approach does not constrain the system by conformance to the existing theoretical frameworks and in that respect seems to be better aligned with the usage-based nature of grammars considered.

Similarly, in an attempt to remove the constraint of *a priori* imposed grammatical structures, Brodsky and Waterfall (2007) base their evaluation on an assumption that it is highly unlikely for two highly-constrained models trained on disjoint corpora to coincidentally agree on whether a sentence is grammatically acceptable or not. This idea is similar to that of inter-annotator agreement, where, in the absence of ground truth, high reliability implies validity. In other words, if two independently produced outputs agree, then it is considered to be valid. The authors have proposed a precision-testing scheme based on the above observation. We, however, suggest making use of existing inter-annotator agreement measures, which take into account chance agreement.

Whereas all of the above evaluation strategies use quantitative measures of performance, Dunn (2017b) proposes a qualitative assessment of representative examples of constructions from various subsets of corpus in addition to the quantitative evaluation metrics. Representative examples of constructions learnt by the system could be as follows: *[Wh-Determiner] + [Modal] + "be" + [Past-Participle]* is a productive schematic representation generalised from multiple usages in the corpus such as 'that will be provided', 'that can be played', 'which will be represented', 'that should be made'. Qualitatively, one can verify that this schema covers relative clauses with passive verbs that generalises to many complementizers and modal verbs. However, as we can see, the above schema does not cover relative clauses with various tense forms. In this manner many other schematic representations such as *"to" + [Verb] + [Determiner] + [Noun]*, *[Noun] + [Preposition] + [Determiner] + <religion>, [Preposition] + "the" + <location>* can be analysed from various subsets of the corpus to see how good are they close to speaker intuitions, what type of generalisations these schemas represent. The problem with qualitative evaluation is obviously the human effort involved in manually checking the representative construction patterns from various subsets of corpus. Also, ascertaining what constitutes a reasonably good number of representative examples for evaluation poses a problem. The question of the psychological validity of the schemas arises when an individual evaluates the constructions.

Subsequently the same author presented their quantitative evaluation metrics to evaluate the construction learner (Dunn 2017a; 2017b). They proposed 14 multi-unit association measures whose frequency and association strength can be used to quantify a model of constructions. From many potential construction grammars that could be learnt, they used the degree of coverage and the size of grammar to choose an optimal grammar.

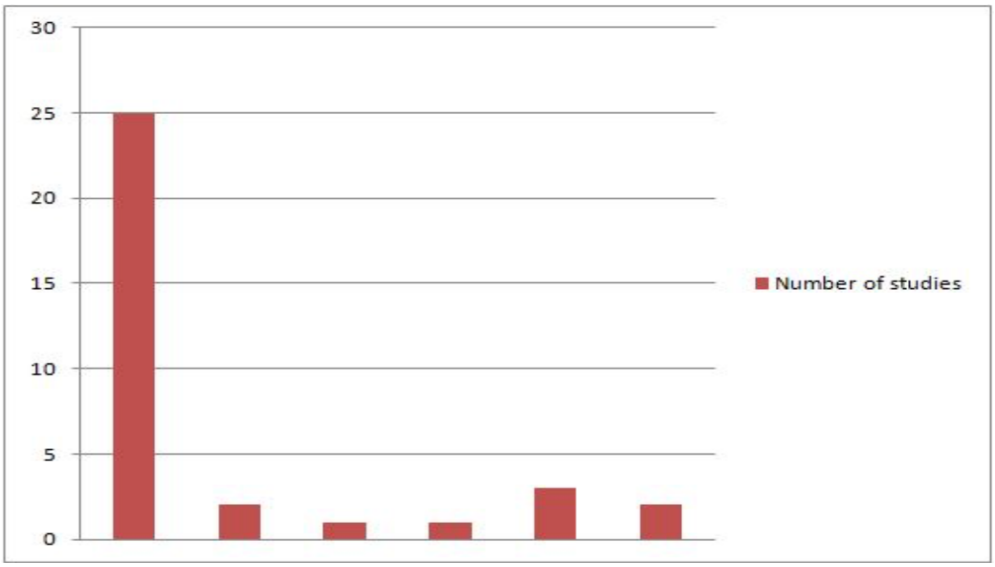
Finally, a Context Free Grammar (CFG) learnt by the system can be compared against the original CFG used to generate the test sentences. Edelman et al. (2005) evaluated their system using this method. A hand-crafted CFG was used to generate a corpus of sentences, a large portion of which is used for training and the other smaller portion is used for testing. The system learns a set of patterns which can be equivalently represented as context-free rules. The rules learnt by the system are compared against the manually defined rules. This approach suffers from a similar drawback as the gold standard approach. The idea of developing a grammar to generate a large gold standard automatically avoids the drawbacks of generating gold standard annotations and can measure the degree to which unsupervised approaches can recreate complex models. However, any such grammar will always be an

approximation and as such may not be representative of a language studied. Even more so, knowing that natural languages are mildly context-sensitive (Joshi 1985), the use of CFGs for this purpose may not be sufficient.

Table 2.3 Factors relevant for evaluation

Property	**Evaluation strategies**					
	E1	**E2**	**E3**	**E4**	**E5**	**E6**
Intrinsic evaluation	✓			✓	✓	✓
Extrinsic evaluation		✓	✓			✓
Manual labour	✓			x		✓
Multiple correct solutions		✓	✓	✓	✓	

Comparison of systems based on their performance should consider the differences in datasets used for evaluation. There are distinctions between evaluations done on artificial datasets produced with hand-crafted CFG rules, on sentences used in ESL multiple choice questions, CHILDES datasets and broad-coverage datasets. For grammar induction this determines how hard the task may be. Table 2.3 shows the factors that are relevant to apply an evaluation strategy and marks which factors are applicable to which methods.



E1 - Comparing against gold standard treebank annotation

E2- Performance of the learner in grammaticality judgement tasks

E3 - Train two highly-constrained models on disjoint corpora and evaluate how they agree or disagree on sentence acceptability

E4 - Qualitative evaluation

E5 - Maximum-coverage, minimum-size, stability measures for a usage-based grammar

E6 - Agreement with CFG grammar that generated the learning data

Figure 2.9 Evaluation strategies

The distribution of the number of papers adopting these different methods of evaluation in their work is shown in figure 2.9.

2.7.6 Features used for learning

A feature in machine learning is defined as some characteristic of the problem studied. In linguistic applications of machine learning, features can be words themselves or their properties, which can be provided *a priori* (e.g., gold standard annotations or lexica), engineered by applying domain knowledge (e.g., named entity formation patterns) or inferred automatically (e.g., stem, embedding, etc.). Different studies used various features to learn the grammatical structure. As the focus of this review is on unsupervised approaches, we are interested primarily in those features that can be extracted from the raw data automatically, possibly by external tools, and subsequently utilised by a learning algorithm. For example, POS tags were frequently provided as features for the grammar inducers before learning (Klein and Manning 2004; Snyder et al. 2009; Bod 2006; Spitkovsky et al. 2010; Zuidema 2006; Reichart and Rappoport 2008; Sangati 2010; Santamaria and Araujo 2010; Gillenwater et al. 2011; Mareček and Žabokrtsk 2012b; Dunn 2017a; Dennis 2005; Dunn 2017b; Bod 2009; Boonkwan and Steedman 2011; Headden et al. 2009; Araujo and Santamaría 2010; Dominguez and Infante-Lopez 2011). Rarely are such features extracted by the induction model itself. For instance, Jin et al. (2018) described a Bayesian Dirichlet model of depth-bounded PCFG induction where the model induces the categories for constituents and tree structures from the raw text provided as input.

Of course, richer linguistic features are likely to result in better grammar induction models. Spitkovsky et al. (2011) use punctuation as a feature to improve the standard Dependency Model with Valence (DMV) and in 2012 the same authors (Spitkovsky et al. 2012) propose that capitalisation cues can improve dependency grammar induction. The general idea behind these implementations is that orthographic cues and their boundaries can help improve the

grammar induction performance. In this way, we have identified 8 types of commonly used features:

A. Parts of speech (POS)
B. Distributional representation of words
C. Head, valence and direction of attachment
D. Word alignments in parallel corpora
E. Chunks
F. Orthographic cues
G. Heuristic rules
H. Construction association measures

Most features are straightforward and are used in various studies irrespective of other aspects of implementation. *POS tags* indicate the grammatical category of the words in a text. For example, the sentence 'The man cut an apple with a knife' can be tagged as follows: 'The/DT man/NN cut/VBD an/DT apple/NN with/IN a/DT knife/NN./.', where the POS tags DT, NN, VBD and IN indicate the grammatical categories *determiner, noun, verb* and *preposition* respectively. These tags are taken into account when dividing a sentence into non-overlapping regions of text called *chunks*, which are typically non-recursive. For example, the same sentence can be chunked as *[The man] [cut] [an apple] [with a knife].*

POS tags were used as features in 18 studies (shown in table 2.4). Ponvert et al. (2011) used chunking based on a probabilistic finite state model such as Hidden Markov Model (HMM). The chunker is cascaded to achieve constituent parsing. Head, valence and direction of attachment are the parameters that are used in generative models of dependency parsing introduced by Klein and Manning (2004) and used in nine other studies. Word alignment refers to the task of extracting translation equivalents of words from sentence-aligned bilingual corpora, i.e., corpora where the same set of sentences in the source language are aligned with the same sentence in a target language. Word alignments are typically used as features in machine translation. In our review, we observed that this feature is used for grammar induction by Snyder et al. (2009).

Distributional representation of words gives a probability distribution of a word occurring in a context when a centre word is given. The distributed representation of words is used in five studies (Klein and Manning 2004; Brooks 2006; Søgaard 2011; Adriaans et al. 2000; Seginer

2007). Orthographic cues such as capitalisation, punctuation is used as features in 2 studies (Spitkovsky et al. 2011, 2012).

Features that we have listed as heuristic rules and construction association measures need explanation. There were four studies (Araujo and Santamaría 2010; Santamaria and Araujo 2010; Boonkwan and Steedman 2011; Marques and Beuls 2016) which used different features: (a) identifying a set of POS tags as separator tags and using these separator tags as features to identify phrase structure boundaries, (b) encoding prior linguistic knowledge as a small number of syntactic prototypes and using them as features (c) use hand-crafted rules to learn artificial construction grammar. We see that these different features can be generalised as heuristics exploited by the authors based on their observations of linguistic patterns. We classified these different implementations as using heuristic rules as features of grammar induction. There were two studies (Dunn 2017a, 2017b) which used 14 different association measures to identify what sequence of construction patterns qualify as actual constructions rather than being potential constructions. These two studies use construction association measures as features for their learning in addition to other features.

It was noticed that POS tags, morphological and orthographic cues are very generic features that are not tied to any particular implementation methodology. Although a fully unsupervised grammar induction uses no syntactic information, POS tags are typically used in most implementations (18 out of 33 studies). Orthographic cues such as capitalisation, punctuations are easier to obtain from a large corpus which can then be used in conjunction with other features to improve the overall accuracy. However orthographic cues obtained from English are not readily transferred to languages such as Thai which do not mark punctuations or word boundaries consistently and even sentence boundaries are not indicated by any discernible cue like full stops. Another observation is that a purely distribution-based approach to grammar induction does not generalise to effectively learn syntax from raw text. Brooks (2006) showed that using a method of attachment to form constituents is more effective than distribution analysis alone. Other features such as valence (number and types of arguments taken by a word linguistically), direction of attachment, word alignments, and chunks are again used to learn constituency and dependency trees. The heuristics such as separator tags and sub-separator tags proposed by Araujo and Santamaría (2010) can be automatically extracted from the corpus and can improve constituent parsing. Table 2.4 shows the list of features and the studies which use those features.

Table 2.4 Features used in various studies

Feature	**Studies which use the feature in their implementation**
POS tags	Araujo and Santamaría (2010); Bod (2006); Boonkwan and Steedman (2011); Dennis (2005); Dominguez and Infante-Lopez (2011); Dunn (2017a, 2017b); Gillenwater et al. (2011); Headden et al. (2009); Klein and Manning (2004); Mareček and Žabokrtský (2012); Marques and Beuls (2016); Reichart and Rappoport (2008); Sangati (2010); Santamaria and Araujo (2010); Snyder et al. (2009); Spitkovsky et al. (2010); Zuidema (2006)
Distributed representation of words	Adriaans et al. (2000); Brooks (2006); Seginer (2007); Klein and Manning (2004); Søgaard (2011)
Head, Valence, direction of attachment	Boonkwan and Steedman (2011); Dominguez and Infante-Lopez (2011); Gillenwater et al. (2011); Headden et al. (2009); Klein and Manning (2004); Mareček and Žabokrtský (2012a, 2012b); Reichart and Rappoport (2008); Sangati (2010); Spitkovsky et al. (2010)
Word alignments from parallel corpus	Snyder et al. (2009)
Chunks or Bracketed structures	Ponvert et al. (2011)
Morphological clusters	Candito and Crabbé (2009)
Orthographic cues	Spitkovsky et al. (2011, 2012)
Heuristic rules	Araujo and Santamaría (2010); Brooks (2006); Marques and Beuls (2016); Santamaria and Araujo (2010)

Construction association measures	Dunn (2017a, 2017b)

2.7.7 Methodologies

Finally, we generalise the methodologies used to support grammar induction into the following categories:

A. Top-down dependency grammar models
B. Similarity or exemplar-based models
C. Distribution-based models
D. Clustering
E. Heuristic approaches
F. Automatic Distillation of Structure (ADIOS)
G. Probabilistic CFG (PCFG)
H. Chunkers and their extensions
I. Data-Oriented Parsing (DOP)
J. Construction grammar induction

Let us first define these categories. Top-down dependency grammar models start with a head, generate a sequence of arguments left and right conditioned on the head, valence and direction of attachment. As already discussed in section 2.7.1, ten studies incorporated generative dependency model or its variation in their implementations (Klein and Manning 2004; Spitkovsky et al. 2010, 2011, 2012; Sangati 2010; Søgaard 2011; Mareček and Žabokrtský 2012a, 2012b; Boonkwan and Steedman 2011; Dominguez and Infante-Lopez 2011). One of the successful baselines for unsupervised parsing is the Dependency Model with Valence (DMV) model of Klein and Manning (2004). In exemplar-based approaches the parse of a sentence is viewed as a set of alignments with exemplars from memory. Approximately nearest neighbour exemplars and their parse are exploited in identifying the parse of the target sentence. Two studies followed this similarity or exemplar-based implementation methodology (Snyder et al. 2009; Dennis 2005).

Distribution-based models use word vectors to derive the parse. There were three studies which employed these in their implementations (Brooks 2006; Seginer 2007; Søgaard 2011). Clustering approaches in general involve dividing the data points into groups where members in each group are more similar to one another than to those in the other group. Natural

Language Processing clustering typically involves grouping of grammatical elements of a sentence into groups such as POS clusters, morph clusters and so on. There are two studies in our review which involved clustering as a part of grammar induction (Reichart and Rappoport 2008; Adriaans et al. 2000).

Heuristic approaches employ practical methods which yield near-optimal solutions to a problem where a perfectly acceptable optimal solution is not available or too difficult to obtain. Examples of heuristic methods are educated guesses, intuitive judgements, rule of thumb etc. In our review, we identified two studies which involve heuristic methods to recognise phrase boundaries to form bracketed structures (Santamaria and Araujo 2010; Araujo and Santamaría 2010). ADIOS is a method which distils structural regularities and contextual cues from raw text, identifies equivalence classes of usage patterns from these structural regularities and represents this learning as a directed multigraph. This multigraph can be equivalently represented as context free rules which represent the grammar learnt by the system. This methodology was used in four studies in our review (Solan et al. 2004; Brodsky and Waterfall 2007; Berant et al. 2007; Edelman et al. 2005).

Probabilistic context free grammars are context-free production rules that generate valid strings in a language. It is a type of formal grammar which is learnt by three studies in our review (Adriaans et al. 2000; Headden et al. 2009; Jin et al. 2018). Chunkers and their extensions begin with learning bracketed structures or shallow parsing and subsequently proceed to use them to induce the full grammar. There was one study in our review that cascades chunkers with HMM to achieve constituent parsing (Ponvert et al. 2011). Next method is Data Oriented Parsing (DOP) which is an interesting way to approach grammar. Usually, statistical methods of grammar induction start with a *predefined* grammar and use the corpus to estimate rule probabilities. In DOP, no prior grammar is assumed but uses corpus fragments to induce grammar. DOP models sentence structures based on the observed frequencies of sentence fragments without imposing any constraint on the size of such fragments. There were four studies which implemented grammar induction bottom-up using the DOP method (Bod 2006,2009; Zuidema 2006; Post and Gildea 2013).

Finally, construction grammar induction is a fully usage-based approach to grammar which learns form-meaning pairings called constructions. Constructions are mappings between linguistic form and linguistic function which can be from any level of generalisation such as at the lexical level, phrase level, semantic level and so on. Learning a construction grammar from raw corpus is difficult because constructions can be of any size and they can be internally

complex. Jonathan Dunn (2017a) demonstrates that construction grammars are learnable and falsifiable by providing a model of learning an optimal construction grammar from a raw corpus. The same author (Dunn 2017b) presents a detailed implementation and evaluation method of the CG implementation. The distribution of these ten methods from our list of 33 implementations is shown in figure 2.10 and table 2.2.

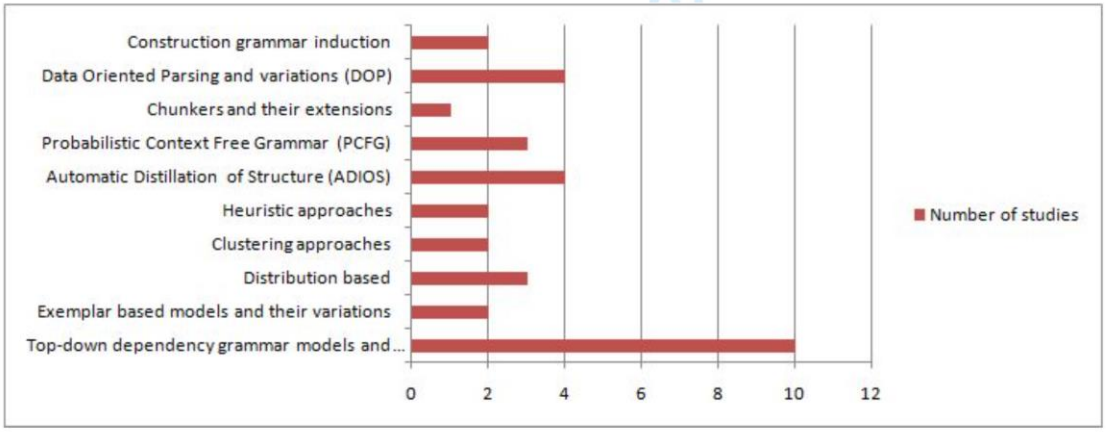


Figure 2.10 Grammar induction methods

Among these methods, we identified that the top-down dependency grammar models and the probabilistic context free grammars are used in multiple studies (13 out of 33) to learn formal dependency and constituency trees respectively. As far as usage-based automatic grammar induction was concerned, we recognised four methodologies as relevant and useful: ADIOS, DOP and Construction Grammar induction. ADIOS is shown to be successful for inducing patterns from short sentences but exhibits limitations in parsing complex sentences. The ADIOS method can be successfully applied once the complex sentences are split into multiple simple ones. DOP works like an analogy-based pattern-matching technique which analyses a sentence by looking up the nearest neighbour constituent structure already stored in memory. The nearest neighbour for a given analysis is defined as the constituency tree derivation which shares the largest number of common nodes. DOP takes a corpus of sentences and allows all possible subtrees to be built from a sentence and chooses an optimal parse by computing the tree which leads to the shortest derivation. DOP is good but it is computationally expensive because it allows all possible subtrees theoretically. Although the subsequent DOP models reduced this computational space significantly in comparison to the general-case DOP, the complexity increases with sentence length. The core idea from DOP that abstract linguistic rules such as movement, agreement, discontiguous phrases etc. can be learnt from recursive tree structures and an analogical match algorithm is powerful.

However, the analogical matching that DOP employs starts with the assumption of constituent tree structures whose internal nodes can be substituted with analogous candidates. How to adapt DOP ideas without assuming constituent tree structures *a priori* is open for future exploration.

Among the studies considered, we noticed that Construction Grammar induction is the only method which demonstrated a fully usage-based, bottom-up approach to grammar induction successfully. The evaluation strategies are also discussed in detail. Because the outputs are not the usual tree structures, but construction slots, evaluation against a gold standard is not possible. The qualitative and quantitative evaluation measures (construction association measures) discussed in these implementations can be adopted for usage-based implementations. Apart from these 33 implementation papers there were seven theoretical and experimental studies, and there were three studies that provided surveys on grammar induction or parser evaluation. We will present our findings from these studies in the next section.

2.8. Findings from non-implementation studies

We will now present the information and insights obtained from three non-implementation papers. D'Ulizia et al. (2011) present a survey of the various techniques of parsing from the literature (Lawrence et al. 2000). Included in the survey are fourteen parser implementations with six computational techniques (Statistical methods, Evolutionary computing techniques, Minimum description length, Heuristic methods, Greedy search, Clustering techniques), two presentation sets (text and informant), three types of information for learning (supervised, semi-supervised, unsupervised), three types of grammar evaluation techniques (looks-good-to-me, compare against treebank, Rebuilding known grammars).

Cramer (2007) experimentally demonstrates the limitations of existing grammar induction algorithms and suggests that an incremental grammar induction is preferable to the conventional methods. A short illustration of such a system is also presented by the authors. Rimell et al. (2009) bring into question the suitability of PARSEVAL measures (the standard metrics for parser performance) as reflecting the parser performance. As an alternative to the exclusive focus on incremental improvements in measures of overall accuracy such as PARSEVAL, the authors suggest a more focused parser evaluation methodology (e.g., construction-focused evaluation) as a way of improving parsing technology. These insights are strikingly significant in the light of the data that we synthesised from the 36 implementations.

Most of these studies evaluated their parser output in terms of the PARSEVAL measures by comparing against gold standard output. In the future, there is more scope to explore better methods and metrics for evaluation of grammar induction systems.

From the other seven theoretical studies included in our literature review, the following insights are obtained.

- Saffran (2001) conducted an experiment to verify if abstract grammatical hierarchies can be learnt using the statistical clues of local predictive dependencies. In an artificial language learning task, the adult participants were exposed to artificial language with no semantic, visual or any other clue except distribution of words and their category. The results support the hypothesis that learners can detect predictive dependencies and use it to acquire simple phrase structures. This suggests that phrase structure boundaries can be induced in an unsupervised way based on predictive dependencies bottom-up with simple categories and statistics.

- Usually discourse information (units of language beyond the level of a sentence e.g., topic, old information, new information, focus etc.) is not taken into consideration while modelling computational grammar induction. Kaiser and Trueswel (2004) show that the processing of non-canonical structures is facilitated by the presence of an appropriate discourse context.

- Productivity in grammar is usually treated in terms of recursion and hierarchy. Frank et al. (2012) suggest that grammar is better modelled sequentially and incrementally. The authors discuss evidence from multiple research fields such as cognitive neuroscience, psycholinguistics, computational models of language acquisition and argue that combining productive grammatical units need not happen hierarchically but a sequential process would suffice.

- Two studies suggested that abstract linguistic properties (subjacency constraints, movement, agreement) can be explained using analogy-based statistical models and constraints on sequential processing (Bod 2007; Ellefson and Christiansen 2000). Rens Bod (2007) showed that learning discontiguous construction patterns, agreement, movement were possible for computational bootstrapping

without a need for special, top-down universal grammar. Ellefson and Christiansen (2000) conducted two types of experiments (artificial language learning experiment and connectionist simulation) to enquire into the nature of subjacency constraints which are usually explained as part of universal grammar. Their results suggest that constraints arising from general cognitive processes, such as sequential learning and processing, are likely to play a larger role in sentence processing than has traditionally been assumed.

- Yang (2011) proposes a statistical test to check if grammar is abstract and productive or lexically-specific and usage-based. Results of the statistical test show that a lexically-specific, usage-based, memory-and-retrieval approach is unsupported. This result is consistent with a productive abstract grammatical system in child speech.

Six studies out of seven from our list support the view that a sequential, statistics-based, bottom-up induction of grammar explains the grammar acquisition and processing by humans. The statistical test by Yang (2011), in contrast, supported a productive, abstract grammatical system governing the child speech rather than a usage-based system. The study shows that a mere usage- based, lexically specific schema could not statistically explain the child speech data. This statistical test however does not resolve the innateness debate in language acquisition. It merely points out that an abstract, productive system should be in place at a very early age in the child's speech. The summary and discussion of all these studies considered in the systematic review are presented in the next section.

2.9. Summary and discussion

Specific findings discussed in the previous section are summarised in figures 2.11 and 2.12. The findings from this review suggest that many aspects of unsupervised induction of usage-based grammars are yet to be fully explored. Only two studies (Dunn 2017a, 2017b) focussed on computational learning of a construction grammar. They used POS tags, dependency relations and semantic labels to learn construction grammar from a large corpus. Though these components can also be learnt in an unsupervised fashion, further research is needed to establish the impact of their accuracy on the overall grammar induced. The properties of construction slots cannot be evaluated properly by comparison against a gold standard as we explained earlier. A practical implementation of usage-based grammar induction should

embed means of evaluating the grammar itself. The two studies mentioned demonstrated the feasibility of unsupervised construction grammar learning as well as its falsifiability and proposed a set of practical evaluation methods. They provide a baseline for further research into fully unsupervised parsing approaches. We will outline the approach of these studies, illustrate what challenges still remain to be addressed and underline the future directions of research.

In Dunn's (2017a, 2017b) implementations, upper-case 'Grammar' refers to the domain-independent model that learns grammatical generalisations from linguistic input. Lower-case 'grammar' refers to a particular inventory of grammatical generalisations learnt for a specific language (i.e., a large corpus of sentences). For example, given a sentence such as 'Bill gave his neighbour a piece of his mind', there are multiple potential linguistic generalisations possible. A few possibilities are shown below but there could be many other possible generalisations.

- **Sentence:** Bill gave his neighbour a piece of his mind
- **Unit-based syntactic generalisation:** [NN – VB – PRN – NN – DT – NN – PREP – PRN – NN]
- **Constituent-level syntactic generalisation:** [NP – VB – NP – NP]
- **Semantically-constrained generalisation:** [NP <ANIMATE> – VB <TRANSFER> – NP <ANIMATE> – NP]
- **Lexically fixed idiomatic usage:** [NN – "give" – NP <ANIMATE> – "a piece of" – NP <ANIMATE> – "mind"]

Similarly, for all the sentences in the corpus, there are multiple potential linguistic generalisations. Of all the possible potential construction units that can generalise the sentence, one is chosen as the actual construction. An inventory of such constructions induced from each sentence in a corpus is the lower case 'grammar' learnt from the corpus. The model which learns these constructions is the upper case 'Grammar'. In the two studies, the authors provide a reproducible model that distinguishes the potential constructions from actual constructions. Their algorithm provides two insights: (a) Constructions are meaningful symbolic units; (b) Co-occurrence and distribution are indicators of meaning. An actual construction differs from all other potential constructions in terms of its being productive (high frequency) and meaningful (presence of significant co-occurrence patterns and association measures).

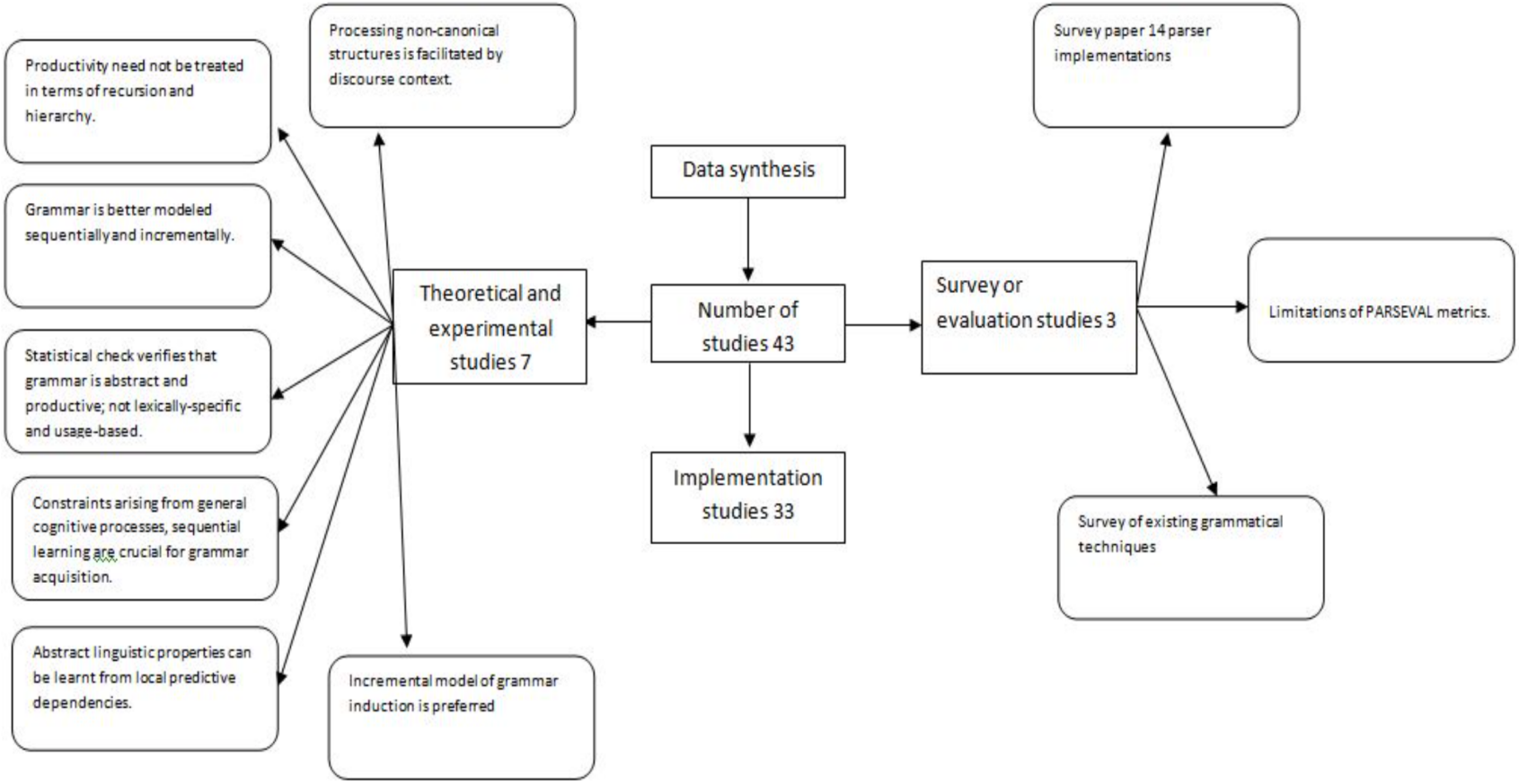


Figure 2.11 Overview of the findings from literature review

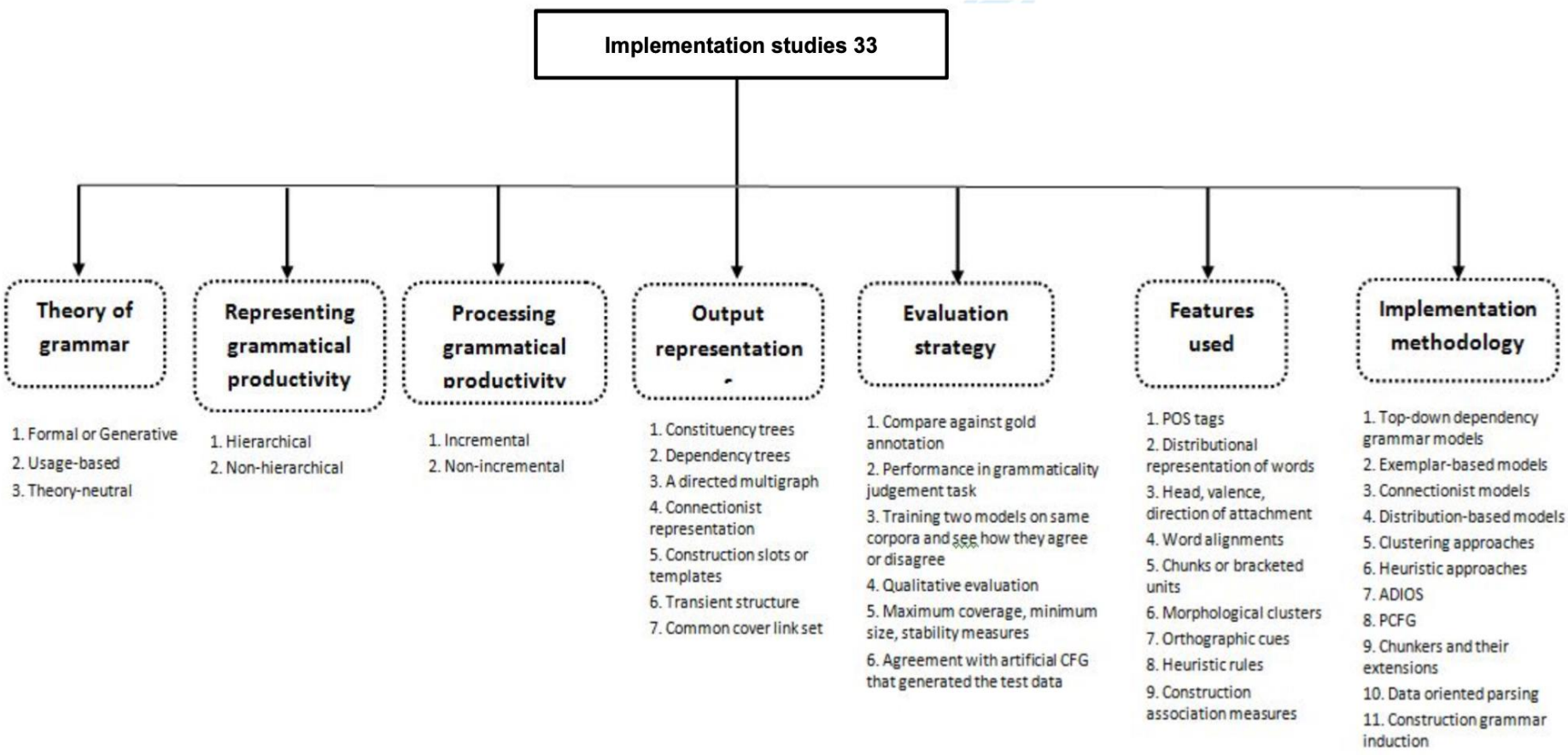


Figure 2.12 Findings from the implementation studies

Two types of thresholds are used to distinguish potential constructions from actual constructions: Frequency threshold and thresholds on vectors of 14 independent association measures. Using the two types of thresholds, the algorithm learns many possible 'grammars' from a given corpus of sentences. From these multiple 'grammars', the algorithm chooses an optimal 'grammar'. An optimal grammar is one which has larger coverage (explains maximum number of linguistic observations) but smaller size (posits as few constructions as possible). The algorithm minimises the objective function of *-log (C^2 /S)* to achieve this effect. In summary, these implementations are focussed on building a computational model that learns an optimal set of mappings between individual sentences and their meaningful linguistic generalisations, with maximum coverage and minimum size.

These two studies relate to our research goal in the following manner: There are two components to 'Grammar' in the sense described earlier: (a) Inventory of construction schemas that generalise the patterns of usage from a corpus of sentences (b) An assembly of these construction schemas to form incrementally larger units. Dunn's studies were interested in the first part with focus on learnability of construction grammars. The second part is a potential research area that is yet to be explored. This involves identifying a method which learns construction patterns inductively and assembles those patterns in an incremental fashion. We would like to explore how an assembly of construction schemas can be achieved in an incremental fashion and use this for unsupervised grammar induction.

The analysis and discussion made so far connect to the research questions in the following manner. The research questions 1 to 4 are answered by various subsections in the data synthesis part discussed earlier. We will now discuss how the different implementations compare with one another in terms of their performance. The implementations discussed in our review differ in their methodology, theoretical influence and various other factors. However, studies with similar evaluation strategies can be compared against one another to identify the range of performance in unsupervised parsing. We also would like to see which evaluation strategies can be adapted for parsing inspired by the functional-cognitive school of thought. The most common evaluation method found in our review was to compare the system output against a gold standard annotation and report the precision, recall and F-measure of the system. There were 25 studies which used this method to evaluate their systems. Not all of them are comparable. One way we tried to compare the systems is if they used similar output representations and then compared their evaluation metrics. There were 14 studies which represented their output as constituency trees and evaluated against the

treebank annotation. 3 studies out of 14 reported their F-measure results by testing their output against treebanks from multiple languages. The best performing system out of these three reports the F-score of 82.9, 67.0 and 47.2 for English, German and Chinese respectively (Bod 2006). It was consistently observed that the systems performed better for English and the lowest scores were for Chinese in the other two systems. There was one system which learnt a subclass of shallow context-free languages. Here the evaluation metrics used was the number of syntactic types and clusters learnt by the system. They reported that 8000 syntactic types and clusters were learned from a dataset of 2000 sentences. It could not be compared with any other system. The remaining 11 studies (out of 14) reported their Unlabelled Precision (UP), Unlabelled Recall (UR) and F-measure for English. They could be directly compared. The lowest accuracy was reported by Brooks (2006) with UP 33.6%, UR 14.1% and F-measure 19.8%. This system used a distributional approach to grammar induction and simple heuristics to reduce the number of candidate constituents using alignment patterns. The results suggest that distributional methods do not generalise enough to learn syntax effectively from raw text, but that attachment methods are more successful.

There were 11 studies which evaluated their dependency output against a treebank. Three of them evaluated their result in multiple languages. Most studies reported their results as Labelled Attachment Scores (LAS) and Unlabelled Attachment Scores (UAS). LAS were generally lower than the UAS in all the studies. The UAS reported in these systems were typically low ranging from 38.3% to 64.5%. The accuracy of these unsupervised parsers in comparison with the state-of-the-art results in supervised parsing is unsurprisingly lower. For instance, Stanford supervised and semi-supervised dependency parsers report accuracy in the range of 95% to 97% (Chen and Manning 2014, Kiperwasser and Goldberg 2016). The other studies in our review have different evaluation methods distinct from each other and their performances cannot be compared directly. The frequency and association measures of co-occurrence used in Dunn (2017a, 2017b) to distinguish potential constructions from actual constructions have the potential to be adapted for future research as well.

From the literature survey, it became clear that most of the existing approaches to unsupervised grammar induction modelled a grammar as a formal, top-down system. Consequently, when such grammars were used to support parsing, their outputs were represented as a constituency tree or dependency tree. In this case, the parsing results can be compared against a treebank. Here, the correct (or acceptable) parsing is determined *a priori* based on existing grammars. However, theoretical studies suggested that an

incremental, sequential, usage-based approach to grammar induction that takes discourse information and local dependencies into consideration can model the same task more effectively. It is as if the theoretical or experimental studies are the antithesis of the computational implementations included in our review. This gap between the two should be addressed to enable further progress in grammar induction and evaluation. The most challenging aspect about a truly bottom-up, usage-based approach to grammar induction is that there are usually multiple ways to generalise usage patterns bottom-up. Learnability and falsifiability of usage-based patterns, evaluation strategies for the output in the absence of a gold standard are the major challenges. There were just two studies that demonstrated that unsupervised construction grammars are learnable from a raw corpus and they can be evaluated for optimality by maximising the degree of coverage and minimising the size of the grammar learnt by the system. It was, however, observed that these implementations do not learn usage-patterns incrementally, but rather choose sequences of construction slots as generalisations for the sequence of words in a sentence. There is scope for research to explore how to assemble usage patterns in an incremental way and how such a system can be evaluated. In the next chapter we will explore ideas from psycholinguistic literature and how ideas from Cognitive Grammar can be useful to develop a viable model of parsing that bridges this gap between the linguistic literature and computational linguistic models.

In this chapter, we described the systematic literature review on the prevalent approaches in unsupervised parsing and presented the results obtained from the review. In the upcoming chapters, we will elaborate our research methodology, introduce theoretical ideas from Cognitive Grammar, Computational Paninian Grammar and psycholinguistic studies. By synthesising the ideas from these three fields, we will identify the functional properties that an unsupervised cognitive parser should have.

Chapter 3 Research Methodology

The main aim of this research project is to investigate whether an unsupervised learning approach can be used to develop a syntactico-semantic parser. The research methodology adopted to realise this aim is described in this chapter. Figure 3.1 shows the key stages of research and how they are linked to the chapters in the thesis.

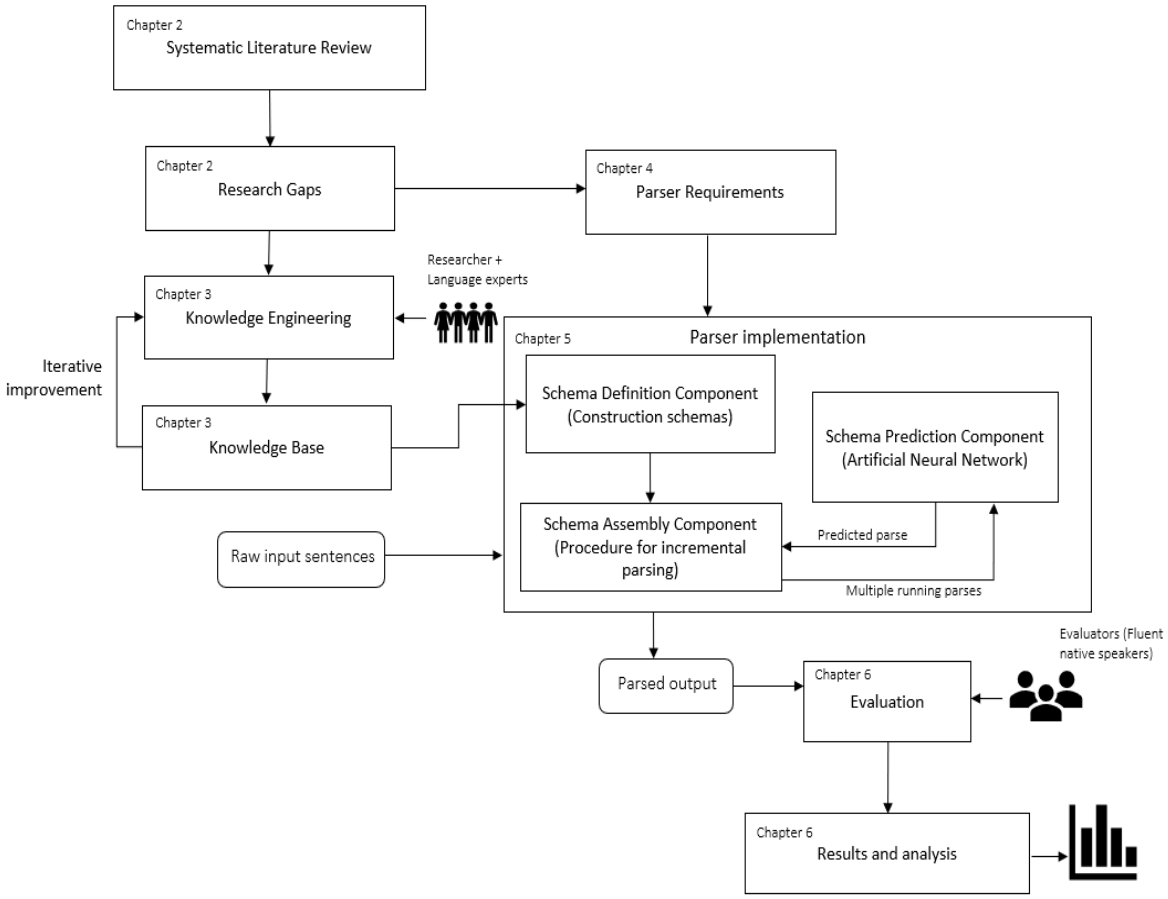


Figure 3.1 Research Methodology

3.1 Systematic literature review

To build this approach upon the existing body of knowledge, we conducted a systematic literature review of unsupervised parsing approaches, which was presented in chapter 2. The literature review revealed research gaps related to the following points:

- Most of the prevalent parsing models learn grammar as a formal system of rules whose outputs are often represented as constituency trees and dependency trees.

- The most common method for parser evaluation was by comparing the output parse against a gold standard.
- The theoretical and experimental studies conducted with human speakers suggest that parser is better modelled as an incremental, sequential system that learns syntax based on linguistic usage.
- Other than the conventions of usage, local dependencies and discourse information were also found to be useful features to parse grammatical structures.
- Learnability and falsifiability of usage-based grammatical structures and identification of suitable evaluation strategies in the absence of a gold standard output are some of the major challenges.

The review concluded that there are gaps between theoretical / experimental studies on parsing and the computational models of parsing in terms of theory of grammar, representation and processing of grammatical productivity, approach to grammar induction, features used, output representation and evaluation strategy. To bridge these gaps, we propose a new parsing methodology that is based on the functional-cognitive school of linguistics and whose characteristics are compatible with the findings from the literature review. The next section describes the process of identifying the requirements for the new parsing model so that it can effectively address the given research gaps.

3.2 Parser requirements

To identify the parser requirements, we studied the theoretical aspects of Cognitive Grammar (Langacker, 2008), Computational Paninian Grammar (Bharati and Sangal 1993; Sangal et al. 1995) and psycholinguistic studies on online sentence processing (MacDonald et al 1994; McRae et al. 1998; Staub, 2015). We chose these three topics to synthesise the specifications of the parser for the following reasons:

- Cognitive Grammar treats meaning as central to grammar and emphasises the role of non-linguistic cognitive abilities in grammatical processing. Construal, construction schemas, profile, autonomy, dependence and grammar as symbolisation are some concepts from CG that are useful to create a knowledge base that connects both syntax and semantics.

- Computational Paninian Grammar is a computational linguistic framework proposed for analysing the syntax of major Indian languages belonging to both Indo-Aryan and

Dravidian language families. The insight from this framework is that sentences can be analysed at a level that lies between syntax and semantics. This framework proposes functional relations between verbs and its arguments (called karaka relations). We generalised this concept to identify the nature of interaction possible between any two parts of a sentence and included this dimension of analysis in our construction schemas.

- The psycholinguistic studies on online sentence processing by humans gave evidence for how humans process parts of sentences and integrate them to get the full parse. We considered these studies to develop parser specifications that closely imitate the strategies used by humans in sentence processing experiments.

The requirements of the parser identified from these three fields of study are:

- A sentence should be parsed incrementally.
- Multiple competing parses should be maintained until one of them succeeds.
- The parse should unify syntax and semantics.
- The parse should be treated as an action not as a product.

Chapter 4 draws ideas from these three studies at length and specifies the parser requirements in more detail.

3.3 Knowledge Engineering

We used the principles of knowledge engineering to inform the development of an unsupervised syntactico-semantic parser. Knowledge engineering is the process of modelling the domain knowledge with the aim of realising the abilities that are comparable to those of a domain expert (Studer et al., 1998). The domain in this project is linguistics and in particular its subdomains of syntax and semantics. In traditional knowledge-based systems, the explicitly encoded knowledge is used to support inference in a top-down approach. In this project, however, knowledge engineering is limited to the specification of a knowledge base of an otherwise unsupervised approach.

In this work, knowledge engineering involved identification of patterns of grammatical constructions in a language and extraction of commonalities across them. By identifying the commonalities across constructions, we were able to create higher levels of organisation called construction schemas. These construction schemas are specified in terms of each other

from specific low-level schemas to abstract high-level schemas. Apart from schema definitions, we also identified how low-level schemas assemble with each other to form higher-level schemas. The knowledge base thus serves both as a repository of grammatical categories (construction schemas) and their rules of arrangement (patterns of assembly of construction schemas). At the heart of our approach is the creation of a knowledge base whose definitions of grammatical categories are interpretable by humans and can facilitate meaningful processing of sentences. The schemas identified and specified in the knowledge base integrate both syntax and semantics so that they can be interpreted by humans, even the non-expert ones.

The specification of knowledge base involved analysing the syntagms and paradigms of constructions that are conventional in a language by consulting the language experts. The language experts listed out various constructions that lie at the intersection of syntax and semantics specific to a given language. Fluent speakers of Welsh and English were involved in this stage to identify the constructions involved in expressing grammatical relationships between parts of sentences. Some of the typical grammatical phenomena considered for knowledge engineering are constructions involving pronouns, noun phrases, noun compounds, arguments and adjuncts of a verb involving noun phrases, prepositional phrases, adverbials, finite and non-finite verb phrases, adjectives as modifiers and predicates, quantifiers and qualifiers, subordinating and coordinating conjunctions, sentence adverbials, clausal connectives, punctuations and relative clauses. Different languages realise these constructions using different syntactic devices and by employing different word orders but commonalities can be observed and extracted as construction schemas. We identified that the constructions mentioned above can be schematised into constructions involving things, processes, statuses, operators and events.

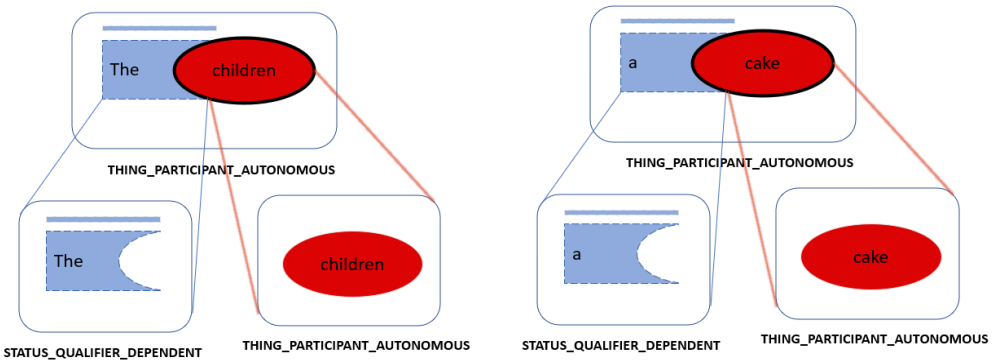


Figure 3.2 Assembly of component schemas to composite schema

For illustration, consider the example sentence from the introduction in chapter 1 ‘The children ate the cake with a spoon’. The phrases ‘The children’ and ‘a spoon’ are schematised as THING_PARTICIPANT_ AUTONOMOUS according to our construction schema definitions. The phrase ‘The children’ is obtained as a result of assembling ‘The’ and ‘children’ which are schematised as STATUS_QUALIFIER_DEPENDENT and THING_ PARTICIPANT_AUTONOMOUS. The same analysis holds good for the phrase ‘a spoon’. The assembly of the words into phrases is shown in figure 3.2.

Chapter 4 explains fully what a construction schema is and proposes a list of basic construction schemas that we identified to be relevant for parsing. These basic construction schemas are listed in tables 4.3 and 4.4 in chapter 4. A more comprehensive list of construction schemas that were specifically identified for English and Welsh as a result of knowledge engineering are listed in tables 5.4, 5.5, 5.6, 5.7 and 5.8 in Chapter 5. Some constraints on how the construction schemas can assemble with each other are also specified as a part of the knowledge base. These patterns of assembly of construction schemas are listed in tables 5.9, 5.10, 5.11, 5.12 and 5.13 of chapter 5.

Since the knowledge base is an approximation of a portion of the domain expertise, we should be able to improve it iteratively as we observe new constructions and new commonalities that can lead to newer ways of schematising the constructions. The knowledge base in its current form can now be repurposed for other languages with few changes such as order of assembly, addition or deletion of one or more schemas, adding or removing more granular schemas and so on. In this way, knowledge engineering results in a knowledge base that is language independent and can be easily adapted to other languages. This was demonstrated by adapting the knowledge base, which was originally developed for English, for Welsh. By observing the grammatical properties of Welsh and how it differs from English, suitable changes were made to the knowledge base. The details of the differences in grammatical structures between Welsh and English are explained in section 5.5 of chapter 5. The list of changes in construction schemas identified for Welsh are listed in table 5.14 of chapter 5.

The construction schemas and their patterns of interaction were specified such that additional grammatical knowledge, that is not explicitly given, is implicitly inferred by matching one or more parts of the specified knowledge base. This inference stage involves a chain of matchings where the inferred knowledge is again matched against the knowledge

base and additional inferences are obtained. How long this inference chain continues depends on the knowledge base and the inference methodology adopted. In our implementation, we match the parts of input sentences with the construction schemas in our knowledge base and obtain all possible assignments of construction schemas to a given word. Sequences of words are thus matched to many possible sequences of construction schemas. Out of these many possibilities, some sequences are rejected as invalid based on the valid patterns of interactions between construction schemas specified in the knowledge base. The remaining valid sequences of construction schemas are assembled to form higher construction schemas based on the interaction patterns in the knowledge base. This process continues as we encounter more and more words in the sentence.

The knowledge base represents a lexical component of the parser. It can match parts of sentences to construction schemas and identify all possible valid assemblies. However, it cannot determine which path of assembly will result in the optimal parse. This will be dealt by the parser itself as part of its computational component.

3.4 Parser implementation

Recall that we derived the following requirements for the parser based on the synthesis of ideas from CG, CPG and psycholinguistic studies on online sentence processing in humans:

- A sentence should be parsed incrementally.
- Multiple competing parses should be maintained until one of them succeeds.
- The parse should unify syntax and semantics.
- The parse should be treated as an action not as a product.

In order to implement human-like reasoning in knowledge-based systems, several approaches have been discussed in the knowledge engineering literature such as rule-based methods, fuzzy methods, connectionist methods and hybrid methods. Out of these various methods, connectionist models represent and process information closer to human-like reasoning because of its ability to exhibit self-organisation in the course of its learning (Kasabov, 1996). Artificial Neural Network (ANN) is a connectionist model that is inspired by biological neurons and their connections in the human brain. The processing element in ANN, called an artificial neuron, is defined by its inputs having input weights. An activation function calculates the activation value of a neuron as a function of its weighted aggregate inputs and possibly the previous state of the network. Given a set of repeated examples of inputs and outputs, an ANN can start with no knowledge and can be trained on the input-output pairs

such that the network learns how to produce desired outputs for the given inputs (Kasabov, 1996, Shanmuganathan, 2016).

ANNs have several important characteristics that make them suitable for representing and processing information similar to that of domain experts especially in the context of a natural language. Firstly, the learning of the ANN is not based on an explicit representation of knowledge. The learning lies in the way the connection weights change while mapping inputs to outputs. The connection weights that are bound to the ANN model is a result of the training step and the state of these connection weights represents the long-term memory of the model. Secondly, ANNs have the capability to generalise the learning so that when a new input vector is given, it produces the best output based on the examples observed during training. Thirdly, ANNs are robust models in the sense that even if some neurons in the network go wrong, the overall system may still perform well. Finally, ANNs are capable of making partial matches in the course of their learning. This means that if the example data used during the training phase do not exactly coincide with the new data in the testing phase, the ANNs are still capable of matching partial similarity between the training and testing data. These characteristics of ANNs related to their learning, generalisation, robustness and partial matching abilities make them excellent candidates for coping with missing or noisy data while still providing an appropriate solution.

We concluded that the ANNs provide a suitable mechanism for selecting the best assembly out of all possible assemblies between any two sequences of words in a sentence. In the context of our project, we use ANNs to train the pairs of construction schemas and their assemblies. For any two sequences of words, the ANN model learns the mapping between all possible sequences of construction schemas and their valid assemblies obtained via knowledge base. The ANN model chooses the sequence of construction schemas that produces the best assembly. The words, their POS tags, their construction schemas are used as features for learning the best possible assembly. As the number of sentences fed to the model increases, the model learns to map the words, POS tags and construction schemas to the best possible schema assembly. By combining the static knowledge base and the ANN model, we propose that the grammatical structure of unseen sentences can be inferred without additional annotation. The details of how the knowledge base is used in conjunction with the ANN model to create a full parser is described in Chapter 5.

The parser is implemented with three components, namely schema definition, schema assembly and schema prediction components. The construction schemas and their patterns of assemblies, which are specified in the knowledge base, form the schema definition component. The ANN model, which is used for selecting the optimal assembly out of many possible assemblies, forms the schema prediction component. Another component, called the schema assembly component, makes use of the other two components and makes several decisions related to parsing including:

- Incremental processing of sentences
- Mapping words to basic construction schemas
- Recognition of different levels of construction schemas from token level to event levels
- Retention of multiple parses in parallel
- Using the knowledge base to discard some of these parses as invalid
- Identifying that there are multiple valid parses as the size of the sentence increases
- Pruning the number of parses so that one of them can be chosen as optimal parse by the ANN
- Specifying the criteria for pruning the number of parses

All these decisions related to parsing are done in the schema assembly component. The implementation details of the three components and how they interact with one another to take the input sentence and produce the parsed output is explained fully in chapter 5.

3.5 Evaluation

The outputs obtained from the parser are not in the form of a constituency or dependency tree. The newly proposed parser splits a sentence into various chunks, assigns construction schemas to those chunks and finally assembles these construction schemas into higher level schemas. There are multiple valid paths of assembling the low-level construction schemas into higher-level assemblies. This very property makes it inherently impossible to use a single analysis as the gold standard to compare against. Therefore, we proposed a few alternative evaluation strategies to rate the performance of the parser and the meaningfulness of the categories labelled by the parser:

1. Quantitative evaluation by comparing the parsed chunks against the constituents in a phrase structure tree
2. Manual evaluation by listing the range of linguistic constructions covered by the parser

3. Evaluation by identifying the number of edits required for a correct assembly
4. Qualitative evaluation based on Likert scales in online surveys

The first approach is to compare the parsed chunks against the constituents found at any level of hierarchy in the phrase structure tree of a given sentence. If the intermediate chunks identified by the parser are structurally valid, they should occur at some level in a constituency tree. The number of matches between the parser chunks and the constituents is quantified and the results are discussed.

The second approach is a manual evaluation where the outputs are analysed subjectively by identifying the range of constructions actually recognised by the parser. Since the knowledge base itself was created by identifying a list of common constructions and their syntagms and paradigms, this evaluation methodology is circular. However, it is still useful as a means to know the coverage of constructions actually parsed by the parser.

The third evaluation strategy involved manually identifying the errors in the parsed outputs and counting the number of edits required in a parse to arrive at a correct assembly. The ratio between the number of edits and the number of assemblies in a parse is measured and reported. If the ratio is small, we may say that the performance of the parser is better.

The final evaluation strategy is to adopt a qualitative approach to evaluation by involving the fluent speakers of English and Welsh language. Since the construction schema labels are defined by unifying syntax and semantics, they must be qualitatively meaningful for a fluent speaker of the language. In order to study how meaningful the parses are to fluent speakers of the language, we conducted a survey to rate the parser output. Specifically, the participants were asked to rate the correctness of the chunk boundaries and their labels on a Likert scale of 5 ranging from *Strongly agree* to *Strongly disagree*. The details of the survey, the instructions and examples provided to the participants, the experimental setup, the study protocol, and ethical review procedure are described in Chapter 6. In the same chapter, we present the results and statistics of the survey, perform error analysis and describe the strengths and limitations of the evaluation methodologies adopted and scope for future improvements.

In this chapter, we presented the research methodology adopted to conduct the various stages of research. The next chapter describes in detail how the parser specifications are identified by studying CG, CPG and psycholinguistic studies on online sentence processing by humans.

Chapter 4 Specification of the parser

In this chapter, we present the ideas from Cognitive Grammar (CG), Computational Paninian Grammar (CPG) and results of psycholinguistic studies to highlight theoretical aspects of grammar and parsing from a functional perspective. Based on the discussion of the ideas from these topics, we synthesise some of the essential properties that a cognitive parser should have. Finally, we conclude this chapter with a short illustration of how a sentence would be analysed by such a cognitive parser in an unsupervised way. The actual algorithm and implementation details will be presented in the next chapter.

4.1 Cognitive Grammar

4.1.1 Construal and meaning

Cognitive Linguistics (CL) assumes that language is not an autonomous faculty but is rather intertwined with other areas of cognition (Croft and Cruse 2004). Cognitive Grammar (CG) is a theory of grammar proposed by Ronald Langacker (2008) whose hypothesis is that grammar is meaningful and emphasis is on cognitive and symbolic explanation to grammatical structures. According to CG, mental operations such as metaphors, schematisation, focus, profiling and scanning, which rely on non-linguistic cognitive abilities, are adapted to perform linguistic processing by humans. These mental abilities, when adapted for linguistic processing, allow humans to conceive the same objective situation in cognitively different ways. As an illustration, consider the sentences.

Sentence 1: 'X is above y'
Sentence 2: 'Y is below X'

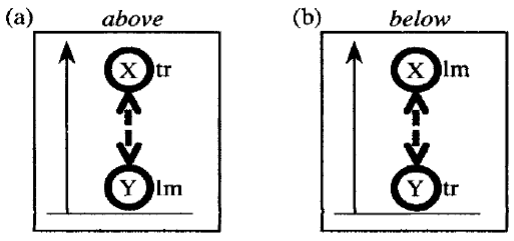


Figure 4.1 Difference in construing 'above' and 'below'

Although both of them describe the same situation, the semantic difference lies in how they are mentally perceived. In the first sentence, X is seen as the entity in primary focus and its positional relation with Y is conceived. Technically, X is called the trajector 'tr' (the entity in primary focus in a relationship) and Y is called the landmark 'lm' (secondary focus) in sentence

1. In the second sentence, Y is in primary focus and its positional relation with X is perceived. Here Y is the trajector 'tr' and X is the landmark 'lm'. Figure 4.1 (a & b) shows the differences in the form of image schemas.

This mental ability to perceive the same situation in different ways is called a *construal* in CG. Within the cognitive linguistic framework, linguistic structures are understood in terms of the construals invoked by them. In the examples given above, trajector and landmark are construals of **prominence** i.e., how prominently an entity is viewed in a relation. Prominence is just one of the dimensions of construal. There are other dimensions such as specificity, focusing, scoping and perspective. We will provide a short description of each dimension of construal in order to give a background of how each one is necessary to understand and analyse linguistic expressions. More detailed understanding of each of these dimensions can be obtained from (Langacker 1987, 2008, 2012).

4.1.2 Dimensions of construal

Prominence: Prominence refers to the asymmetry perceived in terms of relative importance given to some entities than others. For example, the difference between the trajector and landmark in a sentence is a matter of prominence. Another example is the difference between the main clause and subordinate clause in a sentence where the former is conceived as more prominent than the latter.

Specificity: The level of detail or resolution at which we perceive a scene is the specificity of construing it. If a scene is viewed with more resolution or detail, we regard it as more specific. However, if the same scene is viewed at a higher level of resolution with less attention to specific details, it is said to be more schematic. For instance, we can use different expressions such as *he, that guy, the good guy, the guy who came, the guy who came to the hospital yesterday* to refer to a *man*. Although all of them are referring expressions, the difference lies in the level of detail in characterising the referent. The phrase *the guy who came to the hospital yesterday* is more specific while the pronoun *he* is more schematic / less specific.

Focusing: In a given scene, one or more objects could be selected as the conceptual content for our construal. This dimension of construal is called its focus. For example, when a story is narrated, we perceive that the characters and incidents of the story are the background against which the storyline moves forward. In other words, the focusing is on the progression of the story. Again, the foreground vs background distinction is just one aspect of focusing.

There are two other aspects related to focusing namely: **composition** and **scoping**.

- **Composition**: Composition refers to how a linguistic expression is analysable in terms of its components and whether the components are perceived as salient in the overall meaning of the expression. For example, an expression '*costumes and makeu*p' is clearly analysable in terms of its components *costumes<->and<->makeup.* Here each one component is construed as salient for building the total meaning. But even though *makeup* is further analysed as *make<->up*, this compositional meaning is less salient and therefore not construed to be in focus. In this thesis, we use the symbol '<->' to show that the components are assembled to form a composite expression.
- **Scoping:** Scope refers to how we are able to zoom in and out of a scene while construing it. When looking at a laptop screen, the scope is immediate but while looking out of a window the scope is maximal.

Perspective: A perspective is one of the viewpoints that a speaker takes in relation to the scene. For instance, a speaker can assume a subjective viewpoint of the situation described or he / she can take fictional viewpoints where the views of multiple conceptualisers coexist in a single discourse.

In this way, the meaning of a linguistic expression depends not only on its conceptual content but on its construal. According to CG, meaning, and therefore construal, plays a major role in understanding grammatical structures.

4.1.3 Grammar as symbolisation

CG makes the claim that every linguistic structure is symbolic in nature. This means that for every linguistic expression, there is mapping between its phonological/ orthographic form and its meaning. The most obvious symbol is the lexicon where there is an arbitrary mapping between a word and its meaning, e.g., [[book]/[BOOK]]$_{\Sigma}$. Here the subscript sigma represents that it is a symbol. The lower case indicates the orthographic side of the symbol (*book)* and the upper case represents the semantic side of the same symbol *(BOOK)*. According to CG, such symbolic mapping does not stop with the lexical level but rather extends to morphology and grammar as well. Smaller symbols are integrated and assembled into larger symbolic structures progressively to form grammatically complex expressions. Let us illustrate a simple example for how smaller symbols can assemble with each other to form larger symbols. Consider the expression 'Jar lid' which is made up of two words 'Jar' and 'lid'. These two words

are symbols for the concepts JAR and LID. The symbolic mapping between the form and meaning for the two words can be written as [[Jar]/[JAR]] and [[lid]/[LID]] respectively. A JAR is construed as a thing. It is also understood as a physical container which is open at the top as shown in figures 4.2 and 4.3 (Langacker 2008 pp. 163-164).

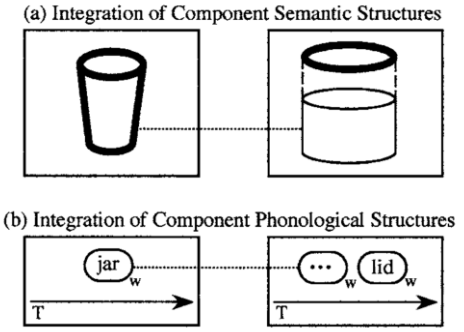


Figure 4.2 Integration of phonological and semantic structures

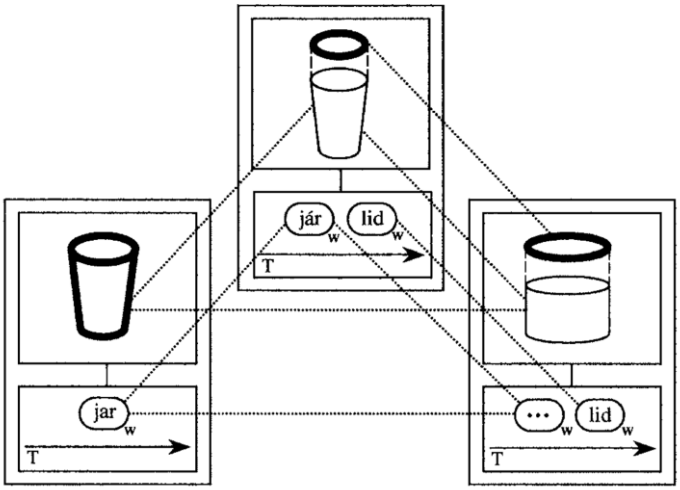


Figure 4.3 The composite assembly after integration

A LID is also construed as a THING with further characterisation as a cover for an opening in the upper side of a container. LID does not specify what is the container but its construal necessarily invokes an unspecified container as shown in figure 4.3. The construal of JAR corresponds with the unspecified container. Because of this correspondence, semantic integration between the two concepts is possible which results in the expression 'Jar lid'. Figures 4.2 and 4.3 show how this integration happens. The heavy lines indicate which part of the construals are relevant for semantic integration, the dotted indicates how they are integrated through semantic correspondences and the resultant construal ('jar lid' shown in figure 20) is an assembly of both 'jar' and 'lid'. The lid portion is highlighted in the final

construal because a ‘jar lid’ is a type of lid, not a type of jar. In this manner, smaller symbols integrate with each other by establishing semantic correspondences and composite symbols are assembled. This process of symbolic assembly repeats indefinitely to result in larger expressions, clauses and sentences. In the next subsection, we introduce the notions of profile, profile determinance, constructions and how commonalities from various constructions can be abstracted and analysed as construction schemas.

4.1.4 Construction schemas

All linguistic expressions are symbolic in nature. This means that everything from the lexicon to grammar lie along the same continuum of assemblies of symbolic structures. The goal of CG is to describe these symbolic assemblies with as much detail as possible. In the previous subsection, two component symbols were integrated into one composite symbol and their assembly was shown as an example. When more and more component symbols are integrated and assembled into larger grammatical structures such as phrases, clauses, sentences and so on, the symbolic assembly will be more complex. The patterns to create such symbolically complex expressions are called *constructions*. Noun compounds created by placing one noun after another (like ‘Jar lid’), dative verb constructions to indicate transfer (like ‘give someone something’), comparative constructions (like ‘the more X the more Y) are some of the examples for constructions. Such construction patterns can be specific or schematic. The commonality that is extracted from various constructions is represented at a higher level of abstraction called a *construction schema*. The process of recognising commonalities between different entities to create higher levels of organisation is called *schematisation*. The opposite of schematisation is *specification*. In our example, the ‘jar’ is specific in its construal but the ‘unspecified container’ construed as a part of the concept LID is schematic. The degree of schematicity (and specificity) varies between different expressions. Referring expressions such as ‘he, the man, the doctor, the cardiologist, the cardiologist in Apollo hospitals’, when referring to the same person, vary in their degree of schematicity. The pronoun ‘he’ is highly schematic but the noun phrase ‘the cardiologist in Apollo hospitals’ is more specific.

A construction schema invoked for understanding an expression plays a role in categorising that expression. If a specific linguistic expression is created in conformance to the schema specifications, we call it an *instance* of the construction schema. An instance can be an elaboration or an extension of the original schema. If the instance fully conforms to the

schema it is said to be an elaboration while if the expression partially conforms to the schema it is said to be an extension. In the 'jar lid' example, the construal of LID has an unspecified container as a schematic entity which is elaborated by JAR. In the expression 'boxing ring' the concept RING is an instance of the original schema of an 'enclosed region in the form of a circle'. Even though the actual boxing ring itself could be rectangular in form, it is conceived as a type of ring. This is an example for extension.

Using these ideas of construals, their symbolic assemblies and construction schemas, CG has attempted to outline how a meaningful description of grammatical categories might look like. The most basic grammatical categories available in any language are noun and verb. According to CG, a noun can be characterised by a construal called *THING* and a verb by a construal called *PROCESS*.

THING: A *THING* is any entity which is a product of grouping and reification. As an example, when we look at the constellations in the sky, even though there could be many individual stars that are small and large in size, we group them together mentally and reify them as one unit and call it, for instance, 'The Great Bear' constellation. Thus, it invokes the *THING* schema which makes it a noun. The *THING* schema is so abstract that many types of linguistic units can be conceived as a result of grouping and reification. 'The bus', 'the man who came from London', 'it', 'earthquake' can all be schematised as *THINGS* by this understanding.

RELATIONSHIP: A *RELATIONSHIP* is the construal of viewing different entities not in isolation but in terms of their overall arrangement or configuration. There are two types of realisations for *RELATIONSHIP* schema. One is the *PROCESS* schema and the other is the *ATEMPORAL RELATIONSHIP* schema. A *PROCESS* is a *RELATIONSHIP* whose configuration is mentally scanned through time sequentially. An *ATEMPORAL RELATIONSHIP*, as the name indicates, invokes the same configuration holistically instead of sequentially scanning every component relationship. Experientially we can conceive the *PROCESS* like watching a movie, like a sequence of images scanned mentally through time. But an *ATEMPORAL RELATIONSHIP* is like a holistic snapshot of different sequences superimposed over one other.

Let us illustrate these three schemas with the following example: the verb 'enter', the preposition 'into' and the noun 'entry' invoke *PROCESS, ATEMPORAL RELATIONSHIP* and *THING* schemas respectively which are shown in figure 4.4. They express the same configuration of two objects involving the following *states* or *component relationships*.

a. Two objects are initially separate from each other
b. Later one object comes close to another object
c. Finally, one object merges within the space of another object.

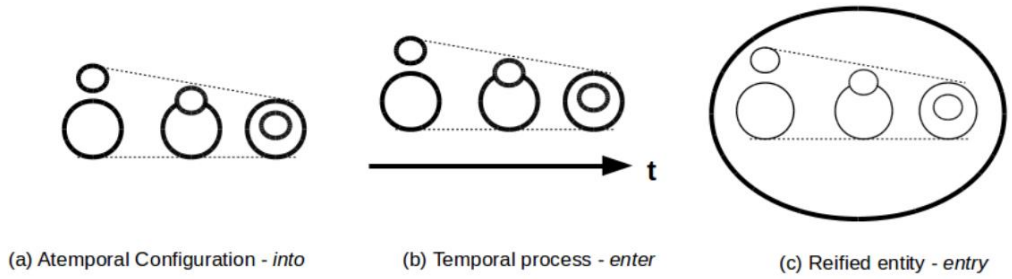


Figure 4.4 Three schemas for the same component relationship

Although the component relationships are the same, the difference between the three parts of speech lie in their construal and the schemas invoked by the construals. In the preposition interpretation, the component relationships are viewed in summary without focussing on individual states distinct from one another (Figure 4.4a), which makes it an instance of *ATEMPORAL RELATIONSHIP*. In the verb interpretation, the component relationships are viewed sequentially through sequential scanning which involves a conceived time or scanning time (Figure 4.4b). This makes it an instance of *PROCESS* schema. In the noun interpretation, the focus is not on the component relationships at all. The components are summarily scanned just like *ATEMPORAL RELATIONSHIP* but the focus is on the reified entity made up of these components (Figure 4.4c). This makes it an instance of *THING* schema.

Profile and profile determinant: The entity designated by a linguistic expression is called its profile. Within an expression's construal, there could be multiple parts out of which one part stands out to designate the meaning of the expression itself. In the case of the adjective 'long', a schematic thing which the adjective modifies is also a part of its construal. But, the *ATEMPORAL RELATIONSHIP* is the portion designated by the adjective and hence it profiles the meaning of the expression. As another example, if we want to understand the concept of 'arc', a circle is imagined and a portion of the circle is construed as an arc. The profile of the word 'arc' is the portion of the circle selected for construing the arc.

The portion of an expression's profile with which we engage in categorisation relationships is called its profile determinant. In the case of an expression 'long road', the profiles of 'long' and 'road' are integrated and the profile of the composite expression is 'road', not 'long' because 'long road' is a type of road. Profile determinant provides a meaningful

characterisation for the concept of 'head' and in the traditional dependency grammar analysis. In the following subsection let us introduce the notions of autonomy and dependence in grammatical analysis.

4.1.5 Autonomy and dependence

Grammatical relations are defined on the basis of semantic factors and correspondences between the component expressions. Any inconsistencies in one or more of these correspondences lead to semantic anomalies. Considering the same example 'jar lid', the construal of 'lid' includes an unspecified container that is open at the top. Establishing correspondence between 'jar' and the schematic 'unspecified container' allows semantic integration of the components 'jar' and 'lid', which makes the composite expression 'jar lid' grammatical. Because such a correspondence cannot be established between the components, the expression 'wall lid' cannot be semantically integrated which makes it ungrammatical.

There are some linguistic structures which by their very nature do not stand alone but take the support of others for their conception (Langacker 2008, pp 199-200). Such component structures which make salient schematic reference to other structures are said to be dependent. If the construal of a linguistic structure does not require another structure for its own manifestation, it is said to be autonomous. The difference between the two can be illustrated by the relation between the preposition and the noun in a prepositional phrase. The very conception of a preposition schematically invokes a reference to a schematic *THING* as a part of its construal. In contrast, conceiving a noun does not require can be done in its own terms without conceiving the noun to be a part of any relationship. Thus, prepositions are by their very nature dependent structures while nouns are autonomous structures. Other examples would be the relation between vowel and consonants in a syllable, the relation between affixes and stem within a word. Here the vowels and stems are autonomously conceived while the consonants and affixes are dependents by their very nature. What is crucial for the distinction is whether conceiving one structure invokes another implicitly or not. This distinction is not based on whether the structures themselves can occur independently. Thus, even though stems might not occur as independent words in a sentence, they are autonomous because conceiving a stem does not require conceiving affixes (even schematically). But conception of affixes necessarily invokes the conception of a stem (at least schematically). In the construal of a dependent structure, the more autonomous substructure

is only schematically specified, which will be *elaborated* (i.e., specified with more fine-grained details) by another component in the sentence. Such a substructure is called an *elaboration site* or *e-site* (Langacker 2008, pp. 198-199). E.g., 'into the hall'. Here the schematic substructure in the construal of 'into' that corresponds to the specific noun phrase 'the hall' is the *e-site.*

Just because a grammatical element is autonomous in an expression, it does not imply that the autonomous entity will be the one which determines the profile of the expression. The profile determinant within an expression may be the autonomous entity or the dependent entity. Continuing with the same example of prepositional phrase (e.g., 'from London'), even though the preposition ('from') is the dependent entity and the noun ('London') is the autonomous entity we find that the profile of 'from London' is determined by the preposition and not by the noun. Thus, prepositions become the heads of prepositional phrases even though the nouns are more autonomous than prepositions in the construal. This kind of organisation of relationships in terms of their autonomy and dependence is a basic feature of language structure. For example, the difference between complements and adjuncts can be understood based on whether the component structure is dependent with respect to the constructional head or otherwise.

4.1.6 Cognitive grammar and parsing

Based on the ideas introduced by CG so far, we understand that grammar is entirely symbolic in nature where smaller expressions integrate and assemble into larger expressions by establishing correspondences between orthography and semantic poles of a symbol (discussed in subsection 4.1.3). This process repeats itself indefinitely to produce symbolically more complex expressions. The patterns for generating composite symbols from component symbols are the construction schemas in a language. Construction schemas are again symbolic in the sense that the mapping between orthography and meaning behind every construction is arbitrary and so it varies from language to language. Let us explain this with two examples:

1. **Idiomatic construction:** 'The more X the more Y' is a construction in English which symbolises the meaning 'as the degree of X increases or decreases, then the degree of Y increases or decreases relatively'. Other languages might construe some other strategy to symbolise this concept. For example, in Tamil, it is symbolised like 'How much how much X, that much that much Y'.

2. **Tense and aspect marking:** Some languages (like English) encode tense explicitly while some other languages (like Chinese) encode only grammatical aspects allowing the tense to be inferred from the discourse context (Lin 2006; Zhang and Xue 2014).
3. **Construal behind prepositions:** The relation between prepositions (or postpositions) and the nouns in the prepositional (or postpostional) phrases is different in different languages because the conventions of how the preposition-noun relations are construed are different in different languages (Taylor 1988; Feist 2008; Zwarts 2017).

This arbitrary mapping between form and function is not just limited to lexicon or some idiosyncratic portions of grammar such as idiomatic phrases. It is true for every level of grammatical analysis such as morphology, phrase level, clause level, sentence level and discourse structures as well because of the lexicon-grammar continuum (Croft 2001; Evans 2012). In CG, grammar is considered to be overt with no hidden or deeper levels of grammatical organisation than what is symbolised by the surface forms. The surface forms themselves embody the ways in which the mapping between form and meaning are symbolised and integrated. Therefore, at every level of analysis, there is arbitrary symbolisation governed by the conventional usage patterns sanctioned by the language. One of the implications of this position is that the grammatical variations observed in different languages are not merely apparent but are actual variations in how the languages conventionally symbolise their semantic content. Every language has a different set of conventions and so no two languages can be described by the same grammar. Despite such variations, grammatical universals can still be sought by empirically studying the construals behind constructions in different languages. Such universals should be formulated in cognitively meaningful ways with sufficient room for allowing variations encountered in reality.

CG offers an image schema based, informal understanding of these concepts. One of the central tenets of CG is that grammatical organisation is grounded on general cognitive abilities and therefore they could be analysed in terms of construals and image schemas. The descriptions provided by CG are illustrative and by no means exhaustive. They have to be verified independently by studying various languages and finding how these ideas are applicable for different language families. Understanding the universals of construction schemas by studying a variety of languages and inducing the common properties behind their construals is one of the larger goals of cognitive linguists. It is also relevant for computational linguistics because if there are universal patterns of symbolic assembly that is common to

multiple languages from different language families, they can be used in natural language processing for learning cognitive parts of speech identification, create annotation treebanks based on universal construction schemas, devise new approaches to cognitively parse a text and so on. Let us summarise some of the concepts introduced so far in the next subsection.

4.1.7 Summary of concepts introduced

Let us define the terminology for the concepts discussed so far.

1. **Construal:** The mental ability to perceive the same objective situation in cognitively different ways.
2. **Symbol:** A mapping between form and meaning. Every linguistic expression is seen as symbolic in CG. e.g., [[lid/LID]]
3. **Profile:** The entity that is designated by an expression is called its profile. In the expression 'lid', the covering on the top and the unspecified container are part of the expression's profile. In the expression 'finger', the hand to which the finger is attached is a part of its profile.
4. **Profile determinant:** A part of an expression's construal that determines its profile is called its profile determinant. In the examples above, it is indicated by heavy lines. In the expression 'jar lid', the lid is the part of the construal that determines the profile i.e., 'jar lid' is a type of lid.
5. **Correspondence:** When the construals of two symbols have portions that designate the same entity, they are said to have correspondence with each other. In our discussion, JAR and the unspecified container in the profile of LID have correspondence.
6. **Component structure:** When correspondences between portions of two expressions are established and they are integrated to form a larger expression, the two expressions that were integrated are called component structures. 'Jar' and 'lid' are component structures in our example.
7. **Composite structure:** When component structures integrate to form a larger expression, the resulting structure is called a composite structure. In our example, 'jar lid' is the composite structure.
8. **Symbolic assembly**: When component structures integrate with each other to form a composite structure, various operations happen:
    a. Semantic correspondences between components are established
    b. Components are integrated into composite expression,

c. Profile determinant of the composite expression is identified

d. A holistic image of how the components are related to the composite structure is formed.

The composite expression is said to be a symbolic assembly of its components.

9. **Schematisation:** The commonality that is extracted from different construals is represented as a higher level of abstraction which is called a *schema*. This process of recognising the commonality to create higher levels of organisation is called *schematisation*. The opposite of schematisation is *specification*. In our example, the 'jar' is a specific thing and the 'unspecified container' is schematic. The degree of schematicity (and specificity) varies between different expressions. Referring expressions such as 'he, the man, the doctor, the cardiologist, the cardiologist in Apollo hospitals' when referring to the same person vary in their degree of schematicity. The pronoun 'he' is highly schematic but the noun phrase 'the cardiologist in Apollo hospitals' is more specific.
10. **Autonomy / Dependence:** Any grammatical entity whose construal can be characterised in its own terms without requiring another entity for its own manifestation is said to be *autonomous*. An entity which necessarily invokes another entity as a part of its construal is said to be a *dependent* structure. E.g., Prepositions are dependent while nouns are autonomous.

Using the ideas introduced so far, CG has attempted to provide a meaningful, functional description of grammatical concepts such as nouns, verbs, adjectives, noun phrases, preposition phrases, perfect, imperfect aspects of a verb and a few other grammatical constructions typically found in English. A detailed discussion on conceptual characterisations of grammatical concepts is provided in (Langacker 1987, 2001). CG has attempted to characterise grammar in terms of meaning, the attempts so far have been restricted to defining basic grammatical classes such as nouns, verbs, adjectives, adverbs, aspects of verb, prepositions, count nouns, mass nouns, grounding systems (such as articles in grammar), quantifiers, qualifiers and so on. However, meaningful concepts necessary for generation / parsing at sentence level in different languages are not directly available from the literature in Cognitive Grammar. The primary concern of literature on CG is at the sentence level: issues such as subject-verb agreement, verb phrases, a great deal of information about the states, actions, processes, whether the action is completed or not, if the verb is construed as active or passive, complements of a verb phrase, adjuncts are laid out in detail to give a cognitive linguistic analysis for familiar grammatical structures. However, what comes beyond the verb

phrases is not entirely clear in a Cognitive Grammar account of sentential analysis. There seems to be a missing link between grammar, meaning and discourse. In the next subsections, we will discuss ideas from Karaka theory, a theory of grammar adopted for Computational Linguistics in Indian languages based on the functional relationships between verbs and nouns. Ideas from Karaka theory complement the ideas from CG and provide room for synthesis of ideas using which we develop our approach to parsing.

4.2 Computational Paninian Grammar

Computational Paninian Grammar (CPG) is a computational linguistic framework proposed for analysing the syntax of major Indian languages belonging to both Indo-Aryan and Dravidian language families (Bharati and Sangal 1993; Sangal et al. 1995; Kesidi et al. 2013; Das et al. 2017). CPG is inspired from Karaka theory that was originally proposed for analysing Sanskrit grammar (Kak 1987; Bhatta 1988). One of the major contributions of Karaka theory is the recognition of the speaker's cognitive viewpoint in conceiving how nouns participate in an action. CPG takes its insight from Karaka theory and proposes a computational framework that analyses sentences at a level that lies between syntax and semantics. The CPG framework labels the functional relations between verbs and its arguments (called karaka relations) and uses these labels in a dependency grammar analysis.

In our work, we analyse the grammatical structure of sentences by unifying syntax and semantics and propose construction schemas based on this unified analysis. Every construction schema label consists of three axes of analysis namely *composition, interaction* and *autonomy.* The *interaction* axis analyses any two parts of a sentence in terms of the functional relations that exist between them and labels them appropriately. This analysis and the labels used therein are extended from the functional relations between nouns and verbs found in the Karaka theory and CPG. For example, a prepositional phrase with the preposition as the head is schematised as a PARTICIPANT along the *interaction* axis. Therefore, we introduce the notions of karaka theory and its contribution to grammatical analysis in sections 4.2.1 and 4.2.2 and motivate the analysis beyond karaka relations in section 4.2.3.

4.2.1 Karaka theory

Karaka theory is a grammatical framework that was originally proposed by Panini for describing the morphosyntactic forms in Sanskrit language two millennia ago (Kak 1987;

Bhatta 1988). Karaka theory is a theory of natural language communication where the connection between nouns and verbs in a sentence are explained through Karaka relations. Karaka relations are syntactico-semantic relations that describe the functional roles played by nouns while participating in the activities denoted by verbs. Karaka relations are neither syntactic relations (formal grammatical relations such as subject, object) nor semantic relations (thematic roles such as agent, patient (Van Valin Jr 1990; Primus 2009) but are functional relations that are motivated by the speaker's viewpoint. According to this theory, the speaker has a cognitive attitude or viewpoint (called *vivaksha)* towards the activity described by the verbs and nouns in a sentence. A sentence is not merely a statement expressing the objective activities but also a representation of the speaker's viewpoint. The constituent of the sentence that is viewed by the speaker as the locus of an activity is called *karta*. Out of all the participants in an action, *karta* is the most independent. We will illustrate what the meaning of *karta* is through the following examples.

1. The boy opened the lock.
2. The key opened the lock.
3. The lock opened.

In formal syntactic analysis, the noun phrases highlighted in all the three sentences are treated as syntactic subjects. In formal semantic analysis, 'the boy' is the *agent*, 'the key' is the *instrument*, 'the lock' is the *theme*. In karaka analysis, all the three noun phrases 'the boy', 'the key' and 'the lock' are analysed as *karta*. The karaka role karta might superficially look like another name for the semantic role agent but it is not the case. As the above examples show, the notion of karta is quite different from semantic agent because both key and lock, which are analysed as karta in sentences 2 and 3, do not have any agency in opening the lock. Karta can be understood in the light of the speaker's viewpoint.

Every verb in a sentence is conceived as an activity complex and every other constituent of the sentence is seen as participating in the activity complex denoted by the verb. Oftentimes the same verb is used to refer to not the main activity itself but to the subparts of the activity complex (Sangal et al. 1995 pp. 61-62). In sentence 1, 'open' refers to a complex of activities such as 'inserting the key, turning the levers, removing the shackle etc'. The locus of this activity is viewed by the speaker as 'the boy', hence the noun phrase is analysed as karta. In sentence 2, the same verb 'open' refers to the subprocess of 'causing the levers to move enabling the shackle to open'. Here 'the key' is the karta because the speaker wishes to emphasise the central role of the key in this subprocess ('It is *this key* that opened the lock!!'*)*.

In sentence 3, the verb 'open' refers to the sub-activity of 'the motion of the latch followed by the loosening of the shackle'. Here the speaker focuses on a sub-action where the unfastening of the shackle is in the speaker's viewpoint. Which key was employed, who inserted the key are zoomed out and they become simply irrelevant at this level of characterisation. That the subprocesses viewed by the speaker are different is evidenced by the choice of different verbs used for sentences 2 and 3 in Indian languages such as Hindi.

There are six karaka relations namely *karta* (locus of the action), *karma* (locus of the result of the action), *karana* (the instrument or the means to achieve the result of the action), *sampradana* (the beneficiary in the action involving transfer), *apadana* (the reference point which remains stationary in activities involving separation), *adhikarana* (the locus of karta or karma i.e., the space or time which supports the activity). The following table shows each karaka relation with examples.

Table 4.1 Karaka relations

Karaka relation	**Functional role**	**Example**
Karta	Locus of action	'The boy opened the lock'
Karma	Locus of result of the action	'The boy opened the lock'
Karana	Instrument or means to achieve the result of the action	'The boy opened the lock with the key'
Sampradana	The beneficiary in an action involving transfer	'The teacher gave a book to the student'
Apadana	The stationary reference point in an action involving separation	'The leaves fell off the branch'
Adhikarana	The locus of karta or karma i.e., the space or time which supports the action	'The leaves fell on the ground'

These six karaka relations look very similar to the semantic roles of agent, patient, instrument, beneficiary, source, location but the crucial difference is that the karaka roles are based on the speaker's viewpoint and not purely semantic. Other than the karaka roles, there could be other participants in the sentence which have non-karaka relations to the main verb. Table 4.2 lists some of the examples of non-karaka relations with the verb.

Table 4.2 Examples of some non-karaka relations

Non-karaka relation	**Function**	**Example**
Taadarthya	A constituent which expresses the purpose of an action	'The boy took the knife to cut the fruit'
Saha sambandhi	A constituent which functions as an associative participant in an action	'The boy went to school with his friends'
Sambandha	A constituent which expresses the function of possession, affinity, ownership. It is a noun-noun relation.	'The teacher's presentation was very good'
Vina	A constituent which expresses non-association in an an action	'The boy went to school without his friends'.

The list of non-karaka relations given above are non-exhaustive. There are many differences between karaka and non-karaka relations. One fundamental difference is that the karaka relations are participant relations where the speaker views a constituent as participating in an activity. The participant is seen as a necessary ingredient playing a particular functional role. Depending on the role played by the participant in an action, it is assigned a karaka role. However, in all non-karaka relations, the constituents are not conceived as participants in the unfolding of the activity. They are not viewed as necessary for conceiving the activity denoted by the verb. For example, conception of the action 'open' (i.e., the action complex of inserting a key, moving the lever etc.) is not possible without supposing the locus of the action 'karta' (the entity who is viewed as enabling the activity i.e., 'the boy') or the locus of the result 'karma' (where the result of the action lies i.e., 'the lock'). The non-karaka relations like purpose, association are not necessary for conceiving the activity itself.

The notions of karaka and non-karaka relations were originally proposed in Panini's grammar for analysing Sanskrit morphosyntax. In Paninian grammar, systematic functional mappings between the noun case declinations (the *vibhakti*) and their participant roles (called *karaka*) were identified and rules were written to generalise their analysis at the karaka level. Wherever systematic mapping between the morphosyntax and its function was not possible, Paninian grammar handled them formally based on their structural or semantic properties. Linguistic structures which could not be analysed functionally were treated formally as exceptions to the general rules, indeclinables and non-karaka semantic categories.

4.2.2 Contribution of Karaka theory for grammatical analysis

Although karaka analysis was proposed for Sanskrit, which is an inflectional language with case declinations, it has since then been adopted for analysis of several other Indian languages. A computational linguistic framework inspired by Paninian grammar called Computational Paninian Grammar (CPG), which was proposed for natural language processing in Indian languages, has been adopted for analysis of English grammar as well (Bharati et al. 1996).

One of the major contributions of Paninian grammar to linguistic analysis is the recognition of the speaker's viewpoint in conceiving how nouns participate in an action (karaka). The karaka level is introduced as an intermediate layer between vibhakti level (an abstraction on noun declensions) and the semantic level (formal meaning). Figure 4.5 shows where the karaka level fits in relation to syntax and semantics (Sangal et al. 1995).

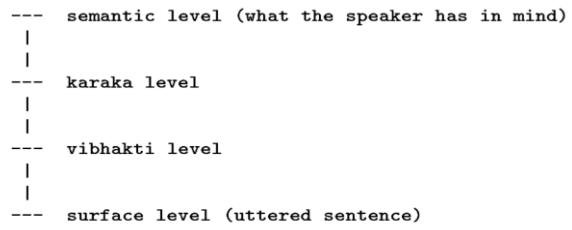


Figure 4.5 Levels in the Paninian model

The surface level is the orthographic level at which a sentence is written. The *vibhakti* level is where a set of words are locally grouped based on their case endings, prepositions or postposition markers (called *vibhakti*) e.g., 'from London', 'towards east', 'in the hospital' are local word groups that are identified based on their vibhaktis. The vibhakti level abstracts away many minor differences between how local word groups are realised in different languages such as prepositional phrases or case declensions or suffix agglutinations. The semantic level is the uppermost level which is the meaning that lies in the speaker's mind. Karaka level is between the syntax and semantics where there is a mapping between the surface form and the meaning based on the speaker's viewpoint. One can imagine several semantic levels between the karaka level and the topmost semantic level with more and more semantic information added at each level.

4.2.3 Beyond karaka relations

The idea of the speaker's viewpoint which is analysed at a level in between syntax and semantics is similar to the idea of construal in CG from section 4.1.1. In fact, *karta* and *karma* karakas are the special cases of *trajector* and *landmark* in CG. *Trajector* of the activity designated by the verb is analysed as *karta* and the *landmark* of the activity designated by the verb is analysed as *karma*. In my earlier research work on syntactic parsing of Tamil, the connection between karaka theory and CG was noticed. It was observed that the functional roles proposed by karaka theory could be generalised further by drawing insights from Cognitive and Construction grammar (Muralidaran and Sharma 2016). Without this generalisation, many constructions in Dravidian languages - such as relative participles, relative pronominal constructions, non-finite verbal inflections on function words, issues of tense and finiteness and other syntactic peculiarities - posed problems in parsing while using the CPG framework. We showed that our generalised framework could explain the syntactic peculiarities observed in Dravidian languages in a cognitively meaningful way by recognising functional viewpoints of the speaker above and beyond the karaka level (Vigneshwaran 2016). Using the insights from construction grammar and CG, we proposed a syntactic annotation scheme to develop a parser for Tamil language. The level of analysis proposed in the work was above the Karaka level but lower than the semantic level. The syntactic annotation scheme proposed for parsing Tamil was induced by observing the morphosyntax of Dravidian languages. However, we hypothesise that the schemas proposed in our earlier work are relatively language-independent and so they can be extended for other languages as well. Let us discuss what the proposed construction schemas were and how they can describe linguistic structures at a level of functional abstraction that is applicable to a wide range of languages.

The basic idea is as follows: a linguistic construction is not viewed in isolation but in a world of discourse mentally conceived by the speaker. When an expression is uttered, it can be analysed as *made up of certain construction schemas* and as if it *expects certain other construction schemas* to model how the speaker's mental discourse builds up. The former axis of analysis is called *composition* and the latter axis is called *interaction.* Composition axis analyses what makes up the content of an expression while interaction axis analyses how the expression is viewed as interacting with other surrounding expressions to build a larger discourse. Recognising the profile and profile determinant (refer to section 4.1.4) of an expression is a way to conceive its composition axis while recognising the elaboration sites (refer to section 4.1.5) is a way to conceive its interaction axis.

Composition axis: When two or more linguistic expressions occur in formally different syntactic environments, yet express similar functional behaviour, there must be a common profile shared by these different expressions. For example, the verb in the expression 'The army crossed the border' and the prepositional phrase in the expression 'The army base was beyond the border' are formally different but they express the same function, the former is the *PROCESS* interpretation of the relationship while the latter is the *STATUS* interpretation of the same relationship. This dimension of analysis via an expression's profile leads us through more and more abstract construction schemas from which seemingly different syntactic categories inherit their functional properties. Ultimately all such schemas inherit their properties from *THING* or *RELATIONSHIP* schema. This axis of analysis that determines what schemas make up the profile of an expression is its composition.

Interaction axis: Every linguistic expression is understood not only in terms of what it is made up of but also in terms of its role in building up the discourse. At any given time, there are two possible construals available at the speaker's disposal for shaping the discourse.

1. The speaker construes that the current expression does not complete the discourse and expects more entities to come up that will lead towards the central / nuclear idea that he / she intends to build. This is called *INCOMPLETE* schema.
2. The speaker construes the current expression as the central / nuclear idea that he intended to convey with no further expectation. This is called *COMPLETE* schema.

For example, the reader reads the phrase 'The boy' at the beginning of a sentence. Naturally the expectation is that there are more words coming up to build the discourse further such as: *what kind of boy? What did he do? What is going to be said about the boy?* and so on. This is an instance of *INCOMPLETE* schema. However, imagine that the same phrase occurs as the title of a book. Now the expectation is not that there are more words coming up to build towards some other nuclear idea. 'The boy' itself conveys the nuclear ideas in this context such as: *this is the book's title, maybe its content is about a boy.* The reader understands this and expects no further linguistic expression to elaborate the phrase. Similarly, if somebody asks the question 'Who wrote the examination?' and someone else replies 'The boy', that itself conveys the nuclear idea in the discourse with no further expectation. These are the instances of *COMPLETE* schema. The COMPLETE and INCOMPLETE are highly schematic and they are trivial. More specific schemas such as CONTINUATIVE, COMBINATIVE are inherited from the INCOMPLETE schema. When an expression expects a THING, it is said to be an instance of COMBINATIVE schema and when it expects a RELATIONSHIP it is said to be in

CONTINUATIVE schema. We will discuss the four types of interactions in discourse based on how they continue and combine with the expected relation or thing.

4.2.4 Basic types of interactions in discourse

At the most abstract level, every expression can be categorised as either a *THING* or a *RELATIONSHIP* along the composition axis. It can either expect a *THING* or *RELATIONSHIP* along the interaction axis. This gives rise to four possible types of interactions:

1. An expression is a *THING* along its *composition* axis and it expects a *THING* along its *interaction* axis - *COMBINATIVE* schema. E.g., Noun compounds, Genitive case, Nouns which are made adjectives, apposition phrases that describe another noun phrase are instances of this schema.
2. An expression is a *THING* along its *composition* axis and it expects a *RELATIONSHIP* along its *interaction* axis - *PARTICIPANT* schema. E.g., Karaka relations and non-karaka relations
3. An expression is a *RELATIONSHIP* along its *composition* axis and it expects a *THING* along its *interaction* axis - *QUALIFIER* schema. E.g., Attribute adjectives, Determiners, Relative clauses modifying a noun phrase, genitive case relationships.
4. An expression is a *RELATIONSHIP* along its composition axis and it expects a *RELATIONSHIP* along its *interaction* axis - *ASSOCIATIVE* schema.

The four types of interactions are shown in figure 4.6. The direction of the arrow indicates the dependent unit in the interaction.

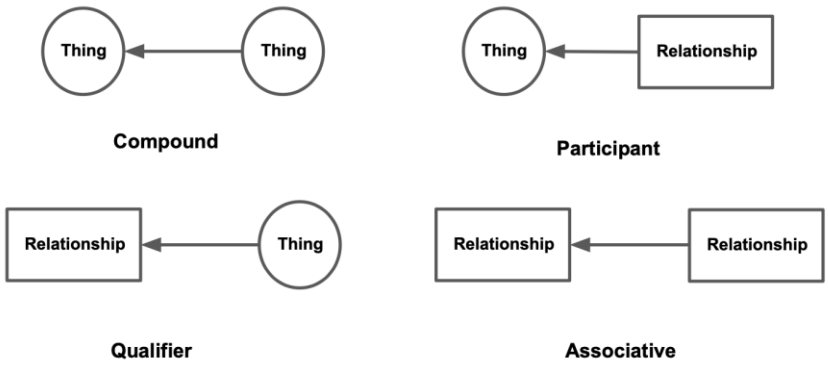


Figure 4.6. Basic types of interactions in the discourse

The basic construction schemas identified along the composition axis are shown in table 4.3.

Table 4.3 Basic construction schemas along the composition axis

Schema	**Definition**	**Examples**	**Formal syntactic categories**
THING	Any entity that is conceived as a product of grouping and reification.	1. The boy 2. The great bear constellation 3. Earthquake	Nouns, Pronouns, Referring expressions, Noun phrases
RELATIONSHIP	Any entity which is not seen in isolation but in the overall arrangement or configuration with respect to other entities	1. Came 2. Long 3. Slowly 4. Running	Verb, Adverb, Adjective, Gerund, Prepositions, Function words
PROCESS	Inherits RELATIONSHIP schema. A type RELATIONSHIP where the states of the relationship are outlined sequentially through time	1. Comes 2. Do 3. has done 4. to try	Verbs and their various inflections
STATUS or ATEMPORAL RELATIONSHIP	Inherits RELATIONSHIP schema. A type RELATIONSHIP where the states of the relationship are outlined instantaneously as a holistic configuration	1. Into 2. Fast 3. Silently 4. Fast 5. Apart	Prepositions, Adjective, Adverb, Quantifiers and other function words
PRONOMINAL	Inherits THING schema. Least specific thing in discourse invokes PRONOMINAL schema.	1. He 2. That guy 3. She who spoke 4. The long one	Pronouns in non-genitive case, Referring expressions, Relative clauses headed by a non-specific thing

More details about the basic schemas along the *composition* axis can be found in section 5.1 and its subsections from chapter 5. *Interaction* axis depends on what entity is expected as a part of an expression's profile to build a world of discourse. The basic schemas along the *interaction* axis are given in Table 4.4.

Table 4.4 Basic construction schemas along the interaction axis

Schema	**Definition**	**Examples**	**Formal syntactic categories**
CONTINUATIVE	Any entity which expects a RELATIONSHIP to build the discourse.	1. The show starts at 2 PM 2. Slowly he dragged himself 3. Although it was Sunday, he was working	Cased nouns and pronouns, adverbs, adverbial clauses
COMBINATIVE	Any entity which expects THING to build the discourse.	1. The boy 2. Long journey 3. Three idiots 4. Many times	Determiners, Attribute adjectives, Numeric adjectives, Relative clauses, Nouns that are a

		5. The boy who lived 6. University rank 7. John, my best friend, spoke	part of a compound, Apposition phrases modifying noun
PARTICIPANT	Inherits CONTINUATIVE schema. A THING which expects a RELATIONSHIP to build the discourse.	1. The boy went to school 2. 2 metres away 3. They are coming	Cased nouns and pronouns
ASSOCIATIVE	Inherits CONTINUATIVE schema. A RELATIONSHIP which expects a RELATIONSHIP to build the discourse.	1. Walked slowly 2. If it is true, it will come out.	Adverbs, Dependent clauses, adverbial participial form of verbs
QUALIFIER	Inherits COMBINATIVE schema. When a RELATIONSHIP expects a THING it is said to instantiate the QUALIFIER schema	1. The boy 2. Long journey 3. Three idiots 4. Many times 5. The boy who lived	Determiners, Attribute adjectives, Numeric adjectives, Relative clauses
COMPOUND	Inherits COMBINATIVE schema. When a THING expects a THING, it is said to instantiate the COMPOUND schema	1. Book shelf 2. Birthday gift shop 3. Pierre Vinken, aged 35	Nouns that are a part of a compound, Appositions modifying the nouns
PRONOMINAL	Inherits THING schema. Any least specific THING in the discourse that functions as a referring expression	1. He, she, it 2. The other one 3. He who must not be named	Pronouns and referring expressions

The basic schemas can be inherited by more specific constructions in a given language to realise more language specific constructions schemas. A detailed list of all construction schemas proposed for parsing Tamil morphosyntax is proposed in (Vigneshwaran 2016). If our hypothesis is true i.e., the construction schemas represent a level of abstraction that is relatively language-independent and so they can be extended for other languages as well, it should be possible to extend the basic schemas to handle the construction patterns in English and Welsh languages. So far, we have discussed ideas from CG, CPG and outlined the basic schemas along the composition and interaction axes. We also suggested that they could be extended for other languages as well. In order to develop an unsupervised cognitive parser, there are two aspects that should be understood: the nature of grammar and the nature of parsing. The nature of grammar is covered by the discussion made so far. In order to understand the nature of parsing we would like to outline the insights obtained from psycholinguistic studies. Combining them both we have been able to develop a parsing strategy which will be discussed in chapter 5.

4.3 Psycholinguistic studies and parsing

In the systematic literature review we found that an incremental, sequential and usage-based approach to grammar induction, that takes discourse information and local dependencies into consideration, can model parsing effectively. There are different questions to consider in the development of such an approach. When we say that a computational model should parse a sentence incrementally, does this analysis happen word by word, at the clause level, or at some other level of analysis? If there are multiple analyses after each increment, should the ambiguities in analyses be retained and if so, how long do we retain them? How long does it take before we integrate the upcoming words in a sentence into the running parse? Studies which discuss how humans process a sentence in real time can provide crucial information about what aspects of natural language parsing a computational model could take into account. There are five aspects of an online parsing model arising from psycholinguistic literature that need to be discussed before we present our approach to parsing. These five aspects are: (a) whether the parsing model is incremental or non-incremental, (b) whether the parsing model handles ambiguities serially or in parallel, (c) whether syntactic processing is separate from semantic processing, (d) the relation between the grammar (as a system) and parser (as a processing tool), and whether there are any set of declarative rules and grammatical representations that form the core of language or whether there are other mechanisms at play, and (e) questions about the costs associated with memory and integration of new elements into the running parse. If we are going to model a parser computationally that is in line with insights from functional theories of grammar and consistent with psycholinguistic elements of parsing, then these aspects of the parsing model should be considered. These will be discussed in turn in the following sections of this chapter.

4.3.1 Psycholinguistic studies

4.3.1.1 Incremental versus non-incremental parsing

The first aspect of a parsing model is whether it is incremental or non-incremental. The systematic literature review on unsupervised approaches to grammar induction revealed that productivity in grammar can be modelled by a sequential process and so a sequential and incremental approach to parsing is desirable. Psycholinguistic experiments reveal that there is not much lag between identifying a word in a sentence and making an attempt to integrate it into the running syntactic or semantic representations (Staub 2015). As we read each word, we try to combine it with the parse representations that we have been building thus far. One

of the clearest pieces of evidence comes from tracking the participant's eye movements while processing the garden path sentences such as the ones shown below (taken from Frazier and Rayner 1982):

1. Since Jay always jogs a mile, this seems like a short distance to him.
2. Since Jay always jogs a mile seems like a short distance to him.

In the second sentence, on seeing the word 'seems', the readers' eye movement stops for a few milliseconds and moves towards the left side of the sentence before they continue to read the sentence and interpret it (Frazier and Rayner 1982). The structural principle of attaching 'a mile' in the object slot of the neighbouring word 'jogs', which works very well for sentence 1, fails in sentence 2. The halt in the eye movement and backtracking the sentence suggests that the readers had already committed to analysing 'a mile' as the object of the verb. On encountering the next word 'seems', this commitment had to be revisited in order to update the parse. This garden path commitment is confirmed by other studies as well (Christianson et al. 2001, Clifton et al. 2007). There are many studies which have investigated how eye movements are influenced by how likely a word can serve to continue the sentence (Rayner et al. 2004; Staub et al. 2007; Warren and McConnell, 2007; Filik, 2008; Cohen and Staub, 2014). These experiments show that unless the reader has already created a model of parse incrementally, the occurrence of implausible words at some position in the sentence could not trigger the halt of eye movement. Within the short span of time, when a word is encountered, the reader not only identifies its lexical category but also has begun to construct a parse representation of the sentence. Another important property of real-time parsing is that not only do the humans fit the incoming words into the running parse incrementally but they also anticipate upcoming portions of the sentence. This aspect makes the parsing mechanism hyper-incremental where the reader actively expects how the sentence will continue based on the prior experience surrounding the running parse. Therefore, the structures which are more likely based on prior experience are more anticipated and the ones that are less likely. Thus, when we develop our approach to parsing, incrementality and the expectation of upcoming structure is an essential property that can improve the performance of the model.

4.3.1.2 Serial versus parallel parsing

The second aspect of a parsing model is whether a human builds a syntactic representation serially or in parallel. In a serial model, the parser maintains only one analysis at any given time and it revises the analysis only if the corresponding parse is shown to be incorrect.

However, in a parallel processing model, a parser maintains multiple possible analyses while incrementally processing the sentence. In a serial parsing model, the first analysis made from the previously observed words is updated only if the upcoming words bear some properties that invalidate the initial analysis. In a parallel parsing model, ambiguous analyses are allowed at every stage and the system will continue to be indecisive until it comes across a disambiguating unit (MacDonald et al 1994; McRae et al. 1998). A hybrid model maintains multiple analyses and the one which finishes first is retained as the viable parse (Traxler et al. 1998; van Gompel et al. 2000, 2001; van Gompel et al. 2005). In parallel constraint-based models many non-structural features influence the parsing decisions (Straub 2015). The parsing constraints can be:

- The number of words seen.
- The number of words integrated into one or more of the running parses
- The number of upcoming structures anticipated
- The construction patterns available in a given language
- The frequency of occurrence of each construction pattern
- The probability of expecting an upcoming structure given the current running parses
- Recent phrases and other local factors.

Multiple ambiguous parses are maintained in parallel and an optimal parse has to be chosen from the running parses within these constraints. In contrast to this, the serial models proposed by van Gompel et al. (2001) maintain that only one of the analyses is retained where there is ambiguity and that the parse is generated based on a lot of non-structural features such as the ones mentioned above. The model decisions are not deterministic but based on probabilities.

4.3.1.3 Syntactic and/or semantic parsing

The third aspect of a parsing model that is debated in psycholinguistic literature is whether syntactic processing is separate from semantic processing. When it comes to online processing by humans, both types of analysis are taking place simultaneously, but there are different views about how they relate to each other. There is some experimental evidence in French, Dutch and German languages using Event-Related Potential method (ERP), which is a peak in electrical brain activity measured by Functional Magnetic Resonance Imaging (fMRI) and Magnetoencephalography (MEG) to understand the mechanisms underlying online

sentence processing. Kim and Osterhout (2005) conducted a study which measures the ERP waveform response on the participants' brains. P600 is a positive-going ERP waveform that is elicited by syntactic phenomena in both reading and listening experiments (Hagoort et al. 1999; Hagoort 2007). N400 is a negative-going deflection that is a part of the brain response to potentially meaningful stimuli such as words, sign language signs, pictures, faces etc (Van Petten et al. 1999; Laszlo et al. 2008; Federmeier and Kutas 2001; Daltrozzo and Schön 2009). When there are violations of syntactic principles in reading experiments, it brings about an increase in the amplitude of the P600 waveform in the readers' brains. However, if the sentences are semantically implausible, it triggered an increase in N400 waveform amplitude on the participants' brains. What happens when there are both syntactic and semantic violations in the same sentence? E.g., '*The door lock was in the eaten'. It was found that when both the semantic and syntactic violations occurred in the sentence, P600 effect occurred (syntactic violation) but no N400 response (semantic violation) in the participants' brain. Other studies conducted on European languages also suggest that the analysis of semantic composition was blocked by syntactic violation (Van Herten et al. 2005; Kuperberg et al. 2007; Stroud and Phillips 2012). However, this interpretation was not supported by experiments conducted on Chinese language (Zhang et al. 2010; Wang et al. 2013) where the presence of both the syntactic and semantic violations caused the N400 effect (indicating semantic violation). The implication is that semantic composition is not blocked by syntactic violation in Chinese language. One of the reasons might be that because Chinese language lacks explicit morphological cues for gender, number and case, the syntactic processing system of Chinese speakers might be different from that of European languages. In summary, this aspect of the parsing model - whether syntactic processing is separate from semantic processing - does not have a definitive answer from the psycholinguistics literature.

4.3.1.4 Grammar and Sentence parsing

The fourth aspect of a parsing model is concerned with the grammar as a system of arranging the words into higher levels of organisation, and its relation to sentence parsing. There are two perspectives regarding their relationship. Language can be perceived either as a product or an action (Clark 1992,1996). Generative grammar theories view language as a product, which implies that the linguistic capacity of human beings is made of a set of rules and representations (Hauser et al. 2002). Within the language-as-a-product outlook, the representations of the grammar are treated as context independent (Gregoromichelaki et al. 2013). Related assumptions are that the linguistic information has hierarchical organisation

with distinct levels of representation such as phonological, morphological, syntactic, discourse, pragmatic and so on. These levels are ordered and the representation from one level invokes another level. Moreover, it is also considered that the speaker has got a definite propositional content that they want to convey and the role of the listener is to grasp this proposition (Hauser et al. 2002).

The alternative view is that language is not a product but rather an action that speakers and listeners engage in. What is shared between them is not their joint intentionality or an abstract system of procedural knowledge but a set of processing mechanisms and practices. In an emergent approach to syntax (O'Grady 2010), arguments are given for the idea that syntactic properties of a language emerge from and are shaped by non-linguistic factors, such as memory costs, ordering constraints etc., which are much simpler and more basic than the emergent grammar. The study also presents a case for unifying the theory of sentence processing with the theory of syntax. Apart from this study there is evidence from Japanese and Korean languages, which reflect a huge gap between the theory of syntax and how the processing of sentences actually happens online. These languages are rigidly verb final where incremental processing of sentences is demonstrated unambiguously, underspecified tree structures are created in order to build the parse, case markers and particles are treated as cues to mark the phrases and other structural boundaries (Inoue and Fodor 1995; Kamide and Mitchell 1999; Miyamoto 2002; Ferreira and Yoshita 2003; Aoshima et al. 2004). The evidence from these studies shows that syntax closely reflects the sentence processing. Finally, in an emergent approach to syntax, efficiency seems to be the driving force of the computational system supporting the human linguistic faculty. Therefore, it is proposed that there are only two components to a parsing model: a lexicon (declarative memory) plus a computational system supported by a working memory (O'Grady 2010). The core idea is that there is no Universal Grammar. A linear processor that is driven by efficiency and a declarative memory that has the properties of a lexical category are sufficient to model the language processing.

4.3.1.5 Memory and integration costs

The final aspect related to the parsing model pertains to the memory limitations and the cost of integrating the upcoming words into the running parse. The relevant questions are:

- Is the syntactic working memory specific to the language faculty or shared by other cognitive faculties as well?

- What are the cognitive mechanisms involved in maintaining the unintegrated syntactic information within the working memory in an ongoing processing of a sentence?

It has been shown that the short-term memory (STM) that is involved in language processing is shared by musical syntax (Patel 2012), arithmetic sequences (Kennedy 2001; Fedorenko et al. 2007) and in motor movements used with a particular expressive purpose (Fiebach & Schubotz 2006). The language processing system takes sequences of structurally dependent elements and integrates them, retains syntactic information for a sustained period, processes new information and anticipates upcoming structures. STM is constantly being used in these activities with two associated costs - the memory cost and the integration cost. The memory cost is concerned with the extent of computational resources needed to store the partial parses in an ongoing parse. The integration cost is concerned with the extent of resources required to integrate the upcoming words into the partial parses currently stored in memory. The number of structural predictions made so far and the number of new referrents entering the discourse are the factors that govern these costs. When the integration of words happens over long distances, too many syntactic predictions may need to be stored in the working memory, which increases these costs. The question is whether this increase in memory costs proceeds linearly or hierarchically. In other words, if the syntactic integration has to happen over a long distance, does the memory cost increase due to the number of intervening words or due to the syntactic complexity of the elements in between? Evidence from wh-movements in German (Fiebach et al. 2002) showed that the longer the filler-gap distance between the trace and its antecedents, the more computational load in processing the sentence. A study by Felser et al. (2003) determined that the integration cost was influenced by the type of linguistic dependency. For example, wh-movement and topicalisation have different costs for integration. However, the memory cost was found to depend on the complexity of the elements in the intervening gap. Adding more words from the same syntactic level did not increase the memory load.

4.3.2 Functional requirements of a parser model

In the introduction chapter 1, we mentioned that our goal is to develop a parsing strategy by exploiting the cognitive nature of grammar and by closely replicating the strategies used by humans in online sentence processing tasks. Based on the results of the systematic literature review of unsupervised approaches to parsing and the ideas presented from CG, CPG and psycholinguistic studies, we have identified the following functional requirements in a parser

model. The actual details of parser implementation are elaborated in chapter 5. The functional properties that a parser model should have are:

- The model should parse the sentence incrementally. This is in line with the results of the systematic literature review (section 2.8) where it was found that an incremental, online model has a broad linguistic coverage and is able to fit child data from a statistical study (Frank et al. 2012). As each new word is encountered, the integration of the words with the ongoing parse should begin. All possible structural predictions should be updated with every upcoming word and the updated prediction should be stored in the working memory. If the upcoming word cannot be integrated into any of the predictions of the running parse, then it should be added to the sequence of unintegrated words in the working memory, until more context becomes available to enable integration. When new words are added to the sequence, if the parser is able to integrate the sequence into an ongoing parse, the content of the working memory is updated.
- The parser model should follow a hybrid approach where multiple ongoing parses are allowed to compete with one another and the one that finishes first is selected. The reason for this choice is that we want to take the advantage of both serial and parallel processing models described in subsubsection 4.3.1.2. In order to limit the huge number of possible structural predictions, it was decided that the competition amongst different ongoing parses shall continue for a span of text (e.g., a span of two verb phrases). We call this the *retain span*. After the retain span, the best parse is selected and the working memory is updated to contain only the optimal parse. For a sentence composed of several retain spans, one optimal parse is chosen out of each and all the foregoing structural predictions are based on the choice made in the previous retain spans. If there are long distance dependencies beyond the retain span, the choices from the previous retain spans should be rolled back starting from the latest span. In any case, if the retain span and the criteria to select the optimal parse are chosen well, the need for revisiting the parse might not arise in most of the sentences except for some tricky garden path sentences.
- Syntax will be treated as a mapping between the surface form (the words) and function (meaningfully motivated). The basic idea is that every word can be mapped to some meaningful cognitive construal and the words can assemble into larger structures based on the semantic constraints imposed by the construal of each

component word. The resultant composite structures are again meaningful and so they invoke new cognitive construals and their semantic constraints in turn determine how other units can assemble with them to form further larger structures and so on. In this view, the grammar is entirely symbolic because every expression from lexicon to final parse are meaningful mappings between forms and functions. This is inspired from the traditions of Cognitive Grammar, Construction Grammar and Computational Paninian Grammar. The implementation details are discussed in sections 5.1, 5.2 and 5.3.

- The grammar will closely reflect the way in which a sentence is parsed. This is in line with the language-as-action perspective discussed in section 4.3.1.4. The parser model will have two components: a lexical component and a computation component with a memory model. The lexical component defines a finite number of *composition* and *interaction* patterns as construction schemas. In the computational component, as and when a new word is read, it is identified as an instance of one or more construction schemas based on the definitions in the lexical component, and added to the ongoing parse. Ambiguous mappings between the ongoing parts of sentences and their construction schemas are allowed for up to a span of text e.g., until two verb phrases are encountered. After this span, the number of possible construction schemas assigned to a word are reduced based on its *interaction* with the neighbouring words. Finally, one schema is chosen by a neural network.
- The parse will not be represented using a constituency or dependency tree because we chose to model a language as an action not as a product. As long as the words could be assembled incrementally and valid schemas can be identified at every step, we treat that as a valid parse. This approach is predicated on the idea that since the definitions of construction schemas in the lexical component are motivated by meaningful cognitive construals, any sequence of assembly that is validated by this component should inherently be meaningful and, therefore, could be a valid parse. Whether it is a structure that should finally be retained or not is determined by the neural network, which learns the transition probabilities of construction patterns.

In this chapter, we have introduced ideas from Cognitive Grammar, Karaka theory and Computational Paninian Grammar and psycholinguistic studies and identified the functional properties a cognitive parser should have. In the next chapter, we will describe the actual parser implementation, evaluation and results.

Chapter 5 Implementation of full parser

In chapter 2, the systematic literature review and its results were presented. After the discussion of research methodology in chapter 3, the synthesis of ideas from functional theories of grammar and psycholinguistic aspects of parsing were examined in detail in chapter 4. On the basis of these discussions, we identified the functional requirements of an unsupervised cognitive parser at the end of chapter 4. In this chapter, we describe how a full parser that demonstrates these functional properties was implemented. The prototype algorithm and its implementation were published in Statistical Language and Speech Processing (Muralidaran et al. 2020).

The functional requirements of the proposed parser, which were elaborated in chapter 4, can be summarised as follows.

1. Parsing model is incremental.
2. Sentence processing model is a hybrid of both serial and parallel processing of the sentence parts.
3. Syntax is analysed in terms of meaningful assembly of constructions.
4. There are only two components to the parser: lexical component and the computational component.
5. Grammar is an action, not a product. Therefore, the output is not a syntactic representation such as constituency or dependency tree. Grammar is inherent in the process of assembling the construction schemas of larger syntactic units out of the schemas of the smaller expressions.

Our parser implementation has three components: Schema definition component, Schema assembly component and Schema prediction component. Out of the three, the schema assembly component is central to the parser that interacts with both schema definition and schema assembly components to process the input sentence into running parses across the various stages of parsing. The parsing process is divided into three stages namely: the pre-processing stage, the processing stage and the predict and feedback stage. In each of the three stages one or more of the three components of the parser are used to perform the actions pertaining to that stage. Figure 5.1 shows the system overview that represents how the three components are connected to the three stages. The three components of the parser are shown in the figure with dotted lines around them and the three stages are shown to be

connected with each component to perform various actions. In this chapter, we present the implementation details of each of the three components and how they are used in various stages of parsing. An explanation of the three components and their connection to the three stages of the parser implementation is given below.

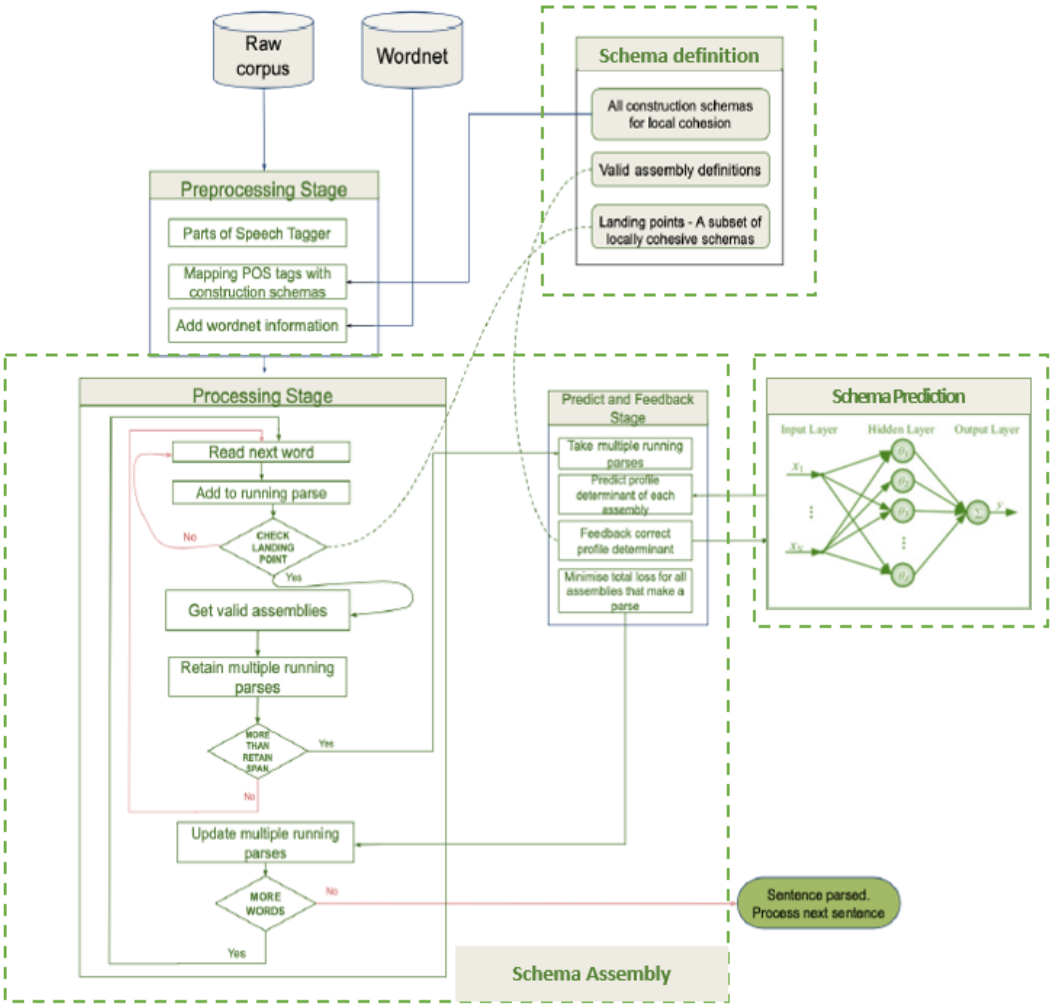


Figure 5.1 System overview

Given a corpus of raw sentences, the preprocessing stage involves POS tagging followed by mapping of tokens and POS tags to token level construction schemas. The preprocessing stage may optionally use additional semantic information from an external resource like WordNet to enrich the raw text before further processing.

The processing stage reads the tokens, POS tags, wordnet synsets and token level construction schemas and assigns all possible analysis for assemblies of token level schemas into higher level schemas. The assemblies are validated to discard the invalid ones and multiple valid paths of assembly are retained. These multiple assembly paths are treated as

multiple running parses. They will be retained in the processing stage as long as the number of words processed do not exceed a *retain span*. A retain span is a span of text until which multiple parses are allowed to compete after which one of the parses is chosen as the optimal parse.

The third stage is the predict and feedback stage where the tokens, POS tags, semantic information from synsets and multiple running parses are taken as inputs and one of the running parses is chosen as the optimal parse. The optimal parse is returned to the processing stage again to update the list of running parses. As the name indicates, this stage predicts the most likely parse and when errors are made, the correct assembly head is fed back to the same prediction component in order to improve the prediction subsequently.

The three components namely schema definition, schema assembly and schema prediction components are used in the three stages to perform the various steps in parsing. The schema definition component is a repository of definitions of construction schemas stored in the form of recursive rules. Each schema is defined in terms of other schemas until the token level schemas are reached. Their definitions are very similar to the definitions of context-free grammar rules except that the definitions are based on the unification of syntax and semantics and therefore interpretable by humans. This component serves as the grammar knowledge base that will be used for bootstrapping higher level grammatical analysis in terms of the valid patterns of assembly.

The schema assembly component is a procedural component that contains the algorithm for incremental cognitive parsing. It is this algorithm that actually implements the decisions to be taken by the processing stage that we discussed earlier. The algorithm in the schema assembly component interacts with the schema definition component to get all possible assemblies from the sequence of input tokens. It also connects with the schema prediction component to get the optimal parse after every *retain span*. This component finishes its processing after all the words in the sentence are assembled and a complete parse is obtained.

The third component i.e., the schema prediction component is a statistical learning component that learns patterns of assembly of construction schemas obtained from the schema assembly component. As more and more sentences are processed and many patterns of assemblies are fed into this component, it learns to predict the most likely assembly that should be chosen given multiple sequences. We used Artificial Neural Networks to implement the schema prediction component. The reasons for choosing ANNs to implement this

component and the details of implementation are given in section 5.3 and its subsections.

The outputs obtained from the parser are not in the form of a constituency or dependency tree. The parser that we developed divides the sentence into various chunks and assigns construction schemas to those chunks and finally assembles these construction schemas into higher level schemas. There are multiple valid paths of assembling the low-level construction schemas into higher level assemblies. Because of this reason, it becomes inherently impossible to assign one of the analyses as the gold standard to compare against. Let us reconsider the example sentence 'The children ate the cake with a spoon' to understand the difficulty in having a gold standard. The sentence may be chunked and assembled in two valid ways shown below in tables 5.1 and 5.2. All the intermediate running parses that were considered and then discarded are not shown here.

Table 5.1 Assembly 1

Chunk	Schema
The children	THING_PARTICIPANT_AUTONOMOUS
ate	PROCESS_CLOSED_AUTONOMOUS
the cake	THING_PARTICIPANT_AUTONOMOUS
with a spoon	STATUS_PARTICIPANT_AUTONOMOUS
The children ate the cake	EVENT_CLOSED_AUTONOMOUS
The children ate the cake with a spoon	EVENT_CLOSED_AUTONOMOUS

Table 5.2 Assembly 2

Chunk	Schema
The children	THING_PARTICIPANT_AUTONOMOUS
ate	PROCESS_CLOSED_AUTONOMOUS
The children ate	PROCESS_CLOSED_AUTONOMOUS
the cake	THING_PARTICIPANT_AUTONOMOUS
with a spoon	STATUS_PARTICIPANT_AUTONOMOUS
The children ate the cake with a spoon	EVENT_CLOSED_AUTONOMOUS

The noun phrases and prepositional phrases in the example sentence are analysed as THING_PARTICIPANT_ AUTONOMOUS and STATUS_PARTICIPANT_AUTONOMOUS schemas respectively. The finite verb is analysed as PROCESS_CLOSED_AUTONOMOUS schema. The output construction schema labels are meant to reflect both the syntactic and semantic properties of the corresponding chunk. Each label has three dimensions of analysis each separated by an underscore. Each of these labels are defined in the schema definition component.

In the upcoming section 5.1 and all its subsections schema definition component is explained in detail. This includes the definitions of different types of schemas and how they are motivated by syntactic and semantic factors. Apart from the schema definitions, valid patterns of interactions between the proposed schemas are also defined here. In section 5.2 and all its subsections, the cognitive parsing algorithm and the various steps related to the processing stage of the parser are elaborated. In section 5.3 and all its subsections, the schema prediction component and how it is implemented with ANNs is presented. The justification for choosing neural networks and the detailed architecture of the network is also given in the subsections. Finally, the reason for using wordnet synsets in the parsing model is explained and the chapter concludes with section 5.5 that outlines how the proposed approach can be adapted for Welsh language.

5.1. Schema definition component

A dictionary of construction schemas is defined in this component. Each construction schema is an abstraction of the morphosyntactic patterns observed in a given language (English and Welsh). Formally, different syntactic environments which have similar functional motivations are analysed as instances of the same schema. These schemas are meaningful templates that define some semantic conceptualisation. Since the construction schemas are functionally defined, more specific definitions or more schematic definitions can be composed from one or more already existing schema definitions and an expression can be analysed as an instance of one or more schemas.

In chapter 4, we introduced the notions of *composition*, *interaction* and *autonomy*. In order to use these notions for the task of parsing, we propose that every linguistic expression can be analysed along its *composition, interaction* and *autonomy* axes. This is because every word, phrase or any other part of the sentence profiles something, expects something else to continue the discourse and can be seen as autonomous or dependent. Accordingly:

- Words, phrases or any other part of a sentence is analogous to one of the construction schemas given in column 1 of table 5.3 (their *composition*),
- The way these expressions create discourse expectations and assemble with other words, phrases or parts of the sentence is analogous to one of the construction schemas in column 2 of table 5.3 (their *interaction*)
- The composition of the expression can be construed independently or its construal inherently depends on other expressions (their *autonomy*).

Table 5.3 Composition, Interaction and Autonomy axes

Composition axis	Interaction axis	Autonomy axis
THING PRONOMINAL RELATIONSHIP PROCESS STATUS OPERATOR EVENT	CONTINUATIVE COMBINATIVE CLOSED QUALIFIER PARTICIPANT DESCRIPTIVE	AUTONOMOUS DEPENDENT JOIN

The construction schemas mentioned in table 5.3 are the most fundamental schemas along the three axes. More specific schemas in the language will be defined based on these fundamental schemas. Out of these fundamental schemas, the following schemas were already introduced in Chapter 4: *THING, PRONOMINAL, RELATIONSHIP, PROCESS, STATUS, CONTINUATIVE, COMBINATIVE, CLOSED, QUALIFIER, PARTICIPANT, AUTONOMOUS* and *DEPENDENT* (sections 4.2.3 and 4.2.4). Other fundamental schemas namely *OPERATOR, EVENT, DESCRIPTIVE, JOIN* will be introduced now.

OPERATOR is a schema that breaks the current parse so that the portion to the left of the operator is treated as one unit and the upcoming words will be treated as another unit. *OPERATOR* schema is invoked in various environments such as conjunctions, disjunctions, listing of items in a sequence, pauses and breaks in a sentence introduced by punctuations. *EVENT* schema is invoked when a *PROCESS* schema is integrated with all its arguments and modifiers and then assembled into the next level of organisation. *EVENT* schema comes into play at the clausal level of linguistic analysis. *DESCRIPTIVE* schema is a type of *ASSOCIATIVE* schema (when a *RELATIONSHIP* expects a *RELATIONSHIP*, it is said to be in *ASSOCIATIVE* schema) where the first entity is a *STATUS (i.e., ATEMPORAL RELATIONSHIP)* and the expected entity can be either PROCESS or STATUS. So far, we have introduced the list of fundamental schemas along the three axes.

More specific construction schemas, which are extensions of the fundamental schemas along the three axes, were identified by us as a part of the schema definition component of the parser. These schemas were identified based on two factors:

- The level of grammatical organisation at which a construction occurs
- How self-similar many different constructions are to one another across different levels

Levels of grammatical organisation: There are three levels of grammatical organisation within which all the schemas defined in the schema definition component can be fitted in: token level schemas, process level schemas and event level schemas. Token level schema definitions are relevant at the level of each word in a sentence. In our implementation, this refers to the schemas which map to the formal POS tags. Process level schemas are relevant before verbs and their arguments are integrated into one large assembly. Event level schemas become relevant after the verbs and their arguments are integrated into an assembly and more larger units are ready to be parsed.

Self-similar constructions across different levels: Different constructions which are formally different and which may occur at different levels of grammatical organisation can be analogous to each other based on some functional property. For example, '<u>your</u> friends', '<u>good</u> friends' and 'friends <u>whom we met in the party</u>' are formally different syntactic units but they are all in COMBINATIVE schema because they are entities that modify a THING (i.e., friends). More specifically they are in QUALIFIER schema because they are RELATIONSHIPS that modify a THING. Further specifically, 'your' and 'good' are in *STATUS_QUALIFIER* schema and 'whom we met in the party' is in *PROCESS_QUALIFIER* schema because the former expressions profile atemporal relationships while the latter profiles a process.

By observing syntactic usage patterns in the three levels of grammatical organisation and by noticing which ones are similar to each other, we have arrived at a list of construction schemas arranged into five categories: THINGS, PROCESSES, STATUSES, OPERATORS and EVENTS. The list of schemas belonging to each of these categories are shown in tables 5.4 to 5.8. We will explain how to understand these schema definitions meaningfully.

5.1.1 Schemas defining THINGs

Table 5.4 contains all the schemas that are categorised as *THINGS*. The construal of *THING* is so generic that any grammatical unit that can be seen as a product of grouping and reification

can be construed as a *THING*.

Table 5.4 Construction schemas - THINGs

CONSTRUCTION SCHEMA	**IS ANALOGOUS TO**	**EXAMPLES**
THING_PARTICIPANT	1. NOUN_CONTINUATIVE 2. STATUS_CLOSED-AUTONOMOUS 3. NUM_CONTINUATIVE 4. DET_CONTINUATIVE 5. THING_PARTICIPANT<->STATUS_CLOSED-DEPENDENT	1. John went 2. Blessed are the meek, It is a given that… 3. One went out and two came in 4. That is good 5. The guards around were vigilant
PRON_PARTICIPANT	1. PRON_CONTINUATIVE	1. They walked
EVENT_PARTICIPANT	1. (THING_PARTICIPANT)<->EVENT_QUALIFIER	1. The company which has its headquarter in London
THING_COMBINATIVE	1. NOUN_COMBINATIVE 2. PRON_COMBINATIVE	1. Language Processing workshop happened 2. Your book is with me
THING_CLOSED-AUTONOMOUS	1. PROCESS_CONTINUATIVE-AUTONOMOUS 2. STATUS_CLOSED-AUTONOMOUS 3. EVENT_CLOSED-DEPENDENT	1. He asked me to come 2. ..refrained from doing 3. ...saw someone coming 4. Speaking to the reporters he said .. 5. It is a given that.. 6. Blessed are the meek .. 7. Much has gone into the research 8. That the man came from Paris is a news to me

But, more specific construction schemas, which are useful for making parse decisions, can be defined as extensions of the basic THING schema. We recognised such schemas and listed them in table 5.4. THINGs can be schematised from constructions at all the three levels - token, process and event levels. The notation used in these tables are: The symbol '_' splits the schema into *composition* and *interaction* axis. The symbol '-' splits the schema into the *composition_interaction* axis group and the *autonomy* axis if the autonomy axis is relevant for conceiving the schema. The symbol '<->' indicates that two schemas are integrated into one unit. Any schema which is enclosed by left and right brackets (e.g., **(THING_PARTICIPANT)**<->STATUS_CLOSED-DEPENDENT in the table) indicates that the definition includes all other schemas which are even partially analogous to it. In this specific example, the definition includes not only *THING_PARTICPIANT* but also just THINGs or PARTICIPANTs. These same notations are used in all other tables in the schema definition component.

The construction schemas mentioned in column 1 of the table are said to be analogous to the schemas in column 2. Any expression which can be analysed as one of the schemas is column 2, can be analysed as the schema in column 1. We say column 1 is composed of schemas from column 2. Schemas in column 1 are *composed* schemas and the ones in column 2 are *composing* schemas. We will illustrate how *composing* schemas and *composed* schemas are related with the following examples.

Illustration 1: Let us consider the expression 'Friends came'. The first word is a noun and it is perceived as a part of the activity 'came' which is a RELATIONSHIP. At the token level, the word 'friends' is in NOUN_CONTINUATIVE schema because it is conceived to expect a RELATIONSHIP to build the discourse. NOUN_CONTINUATIVE can in turn be analysed as THING_PARTICIPANT because of the definition in row 1 and column 1 in table 5.4. Now the *composing* schema NOUN_CONTINUATIVE is a token level schema and therefore it cannot be composed from anything else. There are cases where the level of the composing schemas are above the token level. In that case, they will be further composed of other schemas analogous to them. Illustration 2 shows that.

Illustration 2: Consider the expression 'That the man came from Paris is news to me'. The underlined portion is in EVENT_CLOSED_DEPENDENT schema that composes a THING_PARTICIPANT according to the definition in table 5.4. EVENT_CLOSED_DEPENDENT itself is defined in table 5.8 in subsection 5.1.5 because it is an EVENT level schema. It is composed of other schemas *EVENT_CONTINUATIVE-DEPENDANT<->EVENT_CLOSED-AUTONOMOUS*. These two schemas *EVENT_CONTINUATIVE-DEPENDANT* and *EVENT_CLOSED-AUTONOMOUS* are in turn composed of other schemas and the composition of schemas continues until the token level schemas are reached.

The idea behind the definitions is that one schema is composed of other analogous schemas, these other schemas are in turn composed of what is analogous to them and so on until the token level schemas are reached. This idea is very similar to the recursive definition of phrase structure rules made of non-terminals and terminals. In our case, the event level and process level schemas are similar to non-terminals and the token level schemas are similar to terminals. The difference lies in the fact that these definitions are inherently meaningful and not merely formal.

Token level THINGS: At the token level, demonstrative pronouns, nouns, pronouns, numbers, adjectives can function as THINGS. The examples of such instances are given in column 3 of

table 5.4. The schemas representing token level constructions that function as THINGS are: *NOUN_CONTINUATIVE, NUM_CONTINUATIVE, DET_CONTINUATIVE, PRON_CONTINUATIVE, NOUN_COMBINATIVE, PRON_COMBINATIVE*.

Process level THINGS: At the process level, noun phrases, adjective phrases, noun headed relative clauses with preposition, gerunds, infinitive form of verbs can function as THINGS. The schemas corresponding to these syntactic units are: STATUS_CLOSED-AUTONOMOUS, THING_PARTICIPANT, PRON_PARTICIPANT, THING_PARTICIPANT<->STATUS_CLOSED-DEPENDENT, THING_COMBINATIVE, THING_CLOSED-AUTONOMOUS and PROCESS_CONTINUATIVE-AUTONOMOUS.

Event level THINGS: At the event level, noun phrases which are modified by relative clauses, reported clauses can function as THINGS. The schema corresponding to these syntactic units is EVENT_PARTICIPANT.

5.1.2 Schemas defining PROCESSes

Table 5.5 Construction schemas - PROCESSes

CONSTRUCTION SCHEMA	IS ANALOGOUS TO	EXAMPLES
PROCESS_CONTINUATIVE-DEPENDENT	1. VB 2. VBD 3. VBZ	1. do visit, be interested 2. did happen, was moving 3. Has happened, is watching
PROCESS_CONTINUATIVE--AUTONOMOUS	1. TO<->VB 2. VBG	1. Wanted to speak, to be 2. Tried escaping, was watching
PROCESS_CLOSED--DEPENDENT	1. MD	1. Will come, might open
PROCESS_CLOSED--AUTONOMOUS	1. VB, VBZ, VBD 2. PROCESS_CONTINUATIVE-DEPENDENT<->PROCESS_CONTINUATIVE--AUTONOMOUS 3. PROCESS_CLOSED-AUTONOMOUS<->PRT 4. PROCESSED_CLOSED-AUTONOMOUS<->PROCESS_CONTINUATIVE-AUTONOMOUS 5. PROCESSED_CLOSED-AUTONOMOUS<->STATUS_CLOSED-	1. Go home, He comes, it happened 2. Has happened, is watching, did happen, was moving 3. Give up, go away, work out, give in, ended up 4. Come running, wanted to speak, Tried escaping, continues to rain 5. Remains unopened, left unscathed, seems interesting, have been, 6. Will come, might open, would have been done

	6. DEPENDENT PROCESS_CLOSED-DEPENDENT<->PROCESS_CLOSED-AUTONOMOUS	

Table 5.5 contains schemas that are defined as PROCESSES. They are composed in terms of analogous schemas as explained in the previous subsection.

Token level PROCESSES: At the token level dummy auxiliary verbs, auxiliary verbs showing aspects, modal verbs, base form of verbs function as PROCESSES. The schemas recognised from these syntactic units are: PROCESS_CONTINUATIVE-DEPENDENT, PROCESS_CLOSED-DEPENDENT.

Process level PROCESSES: At the process level, finite verbs with tense, aspects and modality integrated with the verb phrase, prepositional verb phrases, finite verbs modifying gerund or infinitive forms of verbs, verbs describing a resulting state are examples of constructions that can function as PROCESSES. The schemas identified from these syntactic units are: PROCESS_CONTINUATIVE-AUTONOMOUS, PROCESS_CLOSED-AUTONOMOUS and other schemas that they are composed of.

Event level PROCESSES: There are no event level schemas used in the definitions of PROCESS in table 5.5.

5.1.3 Schemas defining STATUSes

Table 5.6 Construction schemas - STATUS / ATEMPORAL RELATIONSHIPs

CONSTRUCTION SCHEMA	IS ANALOGOUS TO	EXAMPLES
STATUS_PARTICIPANT	1. STATUS_CONTINUATIVE-DEPENDENT<->THING_PARTICIPANT 2. STATUS_CONTINUATIVE-DEPENDENT<->PRON_PARTICIPANT 3. STATUS_CONTINUATIVE-DEPENDENT<->THING_CLOSED-AUTONOMOUS	1. In the school, from London, around the area 2. By them, to her 3. dedicated to helping others 4. among the poor and elderly 5. for the initiated 6. Both the programming languages are similar in that they do not allow state changes and side-effects
STATUS_DESCRIPTIVE	1. ADV	1. Slowly, the door opened 2. Yet it was not true

STATUS_CONTINUATIVE-DEPENDENT	1. ADP	1. Into the room 2. From the lecture
STATUS_QUALIFIER	1. ADJ 2. VBN 3. NUM 4. ADP	1. Good boy, thin layer 2. Broken legs, unspoken agreement 3. Three musketeers, Five years 4. The coach to London, Men in black
STATUS_COMBINATIVE	1. PRP$ 2. DET	1. Your book 2. The town
STATUS_CLOSED-AUTONOMOUS	1. VBN 2. ADV 3. ADJ	1. The known and the unknown 2. Much has been discussed 3. The good, bad and ugly
STATUS_CLOSED-DEPENDENT	1. ADJ 2. VBN	1. Turned pale, is open, seems clear 2. Remains broken, left no stone unturned

List of schemas categorised as STATUS / ATEMPORAL RELATIONSHIPs are provided in table 5.6.

Token level STATUSES: At the *token level* parts of speech related to adverbs, prepositions, adjectives, determiners, genitive case of pronouns, stative verbs, past participial form of verbs, numerical qualifiers, quantifiers can function as ATEMPORAL RELATIONSHIPS. The schemas identified from these syntactic units are: STATUS_DESCRIPTIVE, STATUS_CONTINUATIVE-DEPENDENT, STATUS_QUALIFIER, STATUS_COMBINATIVE, STATUS_CLOSED-AUTONOMOUS and STATUS_CLOSED-DEPENDENT.

Process level STATUSES: At the *process level,* preposition phrases and to+gerund constructions, phrases indicating purpose and cause can function as STATUS/ ATEMPORAL RELATIONSHIPS. The schemas identified from these syntactic units are: STATUS_PARTICIPANT composed from STATUS_CONTINUATIVE-DEPENDENT<->THING_PARTICIPANT and STATUS_CONTINUATIVE- DEPENDENT<->PRON_PARTICIPANT.

Event level STATUSES: At the *event level*, complementiser that-clauses that form a part of a prepositional phrase can function as STATUS / ATEMPORAL RELATIONSHIPS. This is because the complementiser that-clause is a construction in English that can be analysed as THING_CLOSED-AUTONOMOUS which can in turn integrate with prepositions to profile ATEMPORAL RELATIONSHIPS. An example is given in table 5.8. The schema identified from this syntactic unit is: STATUS_PARTICIPANT.

5.1.4 Schemas defining OPERATORs

Table 5.7 Construction schemas - OPERATORs

OPERATOR_DESCRIPTIVE	CONJ	1. But, Yet as clause adverbs 2. Chapter number, bullet point numbers
OPERATOR_JOIN	CONJ, .	1. And, but and many coordinating conjunctions 2. Listing of items in a sequence marked by comma, hyphen which pauses the parse and continues the parse from another hyphen
OPERATOR_CLOSED	.	1. Sentence full stop, question mark, a colon before at the end of a statement

Operator schemas shown in table 5.7 occur at all the three levels.

5.1.5 Schemas defining EVENTs

Table 5.8 Construction schemas - EVENTs

CONSTRUCTION SCHEMA	IS ANALOGOUS TO	EXAMPLES
EVENT_CONTINUATIVE-AUTONOMOUS	1. EVENT_CONTINUATIVE-DEPENDENT<->EVENT_CLOSED-AUTONOMOUS 2. EVENT_CONTINUATIVE-DEPENDENT<->STATUS_CLOSED-DEPENDENT 3. EVENT_CONTINUATIVE-DEPENDENT<->THING_CLOSED-AUTONOMOUS	1. While I was studying in college.. 2. Speaking to the reporters, he said….. 3. He asked me to come with my friends 4. If possible, could you please… 5. When cornered, they had no other option except …
EVENT_CONTINUATIVE-DEPENDENT	1. ADP 2. ADV 3. DET	1. While I was studying in college.. 2. Wanted to know if it was possible 3. I heard that it happened
EVENT_COMBINATIVE-DEPENDENT	1. DET 2. PRON 3. ADV	1. The man who came from .. 2. The incident that happened … 3. The idea, which was proposed by..

EVENT_QUALIFIER	1. EVENT_COMBINATIVE-DEPENDENT<->PROCESS_CLOSED-AUTONOMOUS 2. EVENT_COMBINATIVE-DEPENDENT<->EVENT_CLOSED--AUTONOMOUS	1. The man who came from Paris 2. The incident that happened.. 3. The incident that happened last night...
EVENT_CLOSED-AUTONOMOUS	1. (PARTICIPANT)<->PROCESS_CLOSED-AUTONOMOUS 2. (DESCRIPTIVE)<->PROCESS_CLOSED-AUTONOMOUS	1. The tourists visited the museum 2. Inside the room were two tables on which… 3. Lightly the cat moved into ...
EVENT_CLOSED-DEPENDENT	1. EVENT_CONTINUATIVE--DEPENDENT<->EVENT_CLOSED-AUTONOMOUS	1. Whether it happened.. 2. Heard that the man came from Paris

The list of schemas that are categorised as EVENTS are defined in table 5.8.

Token level EVENTS: At the token level, subordinating conjunctions, complementisers, gerunds, relative pronouns, quotative particles require an event level analysis. The schemas identified from these units are: EVENT_CONTINUATIVE-DEPENDENT, EVENT_COMBINATIVE-DEPENDENT.

Process level EVENTS: There are no process level definitions for EVENTS in table 5.8.

Event level EVENTS: At the event level, subordinate clauses, complement clauses, gerunds with their arguments, quotative clauses, relative clauses require an event level analysis. The schemas identified from these units are: EVENT_CONTINUATIVE-AUTONOMOUS, EVENT_QUALIFIER, EVENT_CLOSED-AUTONOMOUS, EVENT_CLOSED-DEPENDENT.

From tables 5.4 to 5.8, we have listed all the five types of schemas along the three levels of grammatical organisation. As mentioned before, the composition of these schemas happens across levels and therefore the same schema can instantiate very different grammatical structures when composed differently. The illustrations are given below:

Illustration 1: Past participial form a verb (*VBN*), an adjective (*ADJ*) and an adverb (ADV) are the formal syntactic categories that can be analysed as *STATUS_CLOSED-AUTONOMOUS* schema because they are temporal relationships, they can be construed

autonomously and they can complete the flow of discourse when treated as a THING e.g., 'Blessed are the meek for they shall inherit the earth'. Here the expression 'the meek' refers to 'the meek people' and so it is construed as a THING. Such an interpretation is possible because the expression instantiates *STATUS_CLOSED-AUTONOMOUS* schema which in turn can be reanalysed as a *THING_PARTICIPANT* (from definitions in table 5.4). The same *THING_PARTICIPANT* can however be composed not out of STATUS_CLOSED-AUTONOMOUS but out of a noun functioning as a *karta* or *karma* discussed in the previous chapter.

Illustration 2: One of the construals of the formal part of speech adjective (*ADJ*) is its function as a *STATUS_QUALIFIER* because it is an atemporal relationship (*STATUS*) which expects a THING. It is further understood as an instance of QUALIFIER schema as well as *STATUS* schema (Refer tables 10 and 12). However, the same adjective can be analysed as *STATUS_CLOSED-DEPENDENT* when the token is construed as a part of a predicate adjective. In the predicate adjective construction (e.g., 'is good'), the adjective that is analysed as *STATUS_CLOSED-DEPENDENT* can in turn be analysed as a part of PROCESS_CLOSED-AUTONOMOUS (i.e., 'is'). These two schemas interact with each other and assemble into PROCESS_CLOSED-AUTONOMOUS schema (based on the *interaction* patterns to be introduced in the next section 5.1.6).

Illustration 3: Finite Verbs (*VBZ*, *VBD*) are analysed as *PROCESS_CLOSED-AUTONOMOUS* schema. *PROCESS* - because its internal states can be mentally scanned temporally as a *PROCESS*; *CLOSED* - because finite verbs can potentially complete the discourse, *AUTONOMOUS* - because the internal states of a verb's construal can occur independently of other parts of the sentence; The verb in PROCESS schema is analysed as EVENT at the next level of analysis. Arguments of the verb, adverbs and other modifiers in the finite verb phrase can be conceived internal to the EVENT schema. Thus, when all these arguments are integrated with the verb, it becomes the EVENT_CLOSED-AUTONOMOUS schema.

From these illustrations, we see that different types of expressions can be analysed as instances of the same schema based on how the intermediate schemas are composed. Moreover, the same expression can be conceived as instantiating different schemas based on the usage pattern within which it is construed. So far, we have discussed how schemas are composed meaningfully. Apart from composition, the Interaction between any two schemas also forms a part of the schema definition component. These schemas are about the *interaction* axis of constructions. A list of all such interactions is given in table 5.9. The

question mark '?' in this table indicates that it can be any schema involved in the interaction.

5.1.6 Interactions between schemas

Table 5.9 Interactions between schemas

Schema interaction	Assembly head
QUALIFIER<->THING	THING
CONTINUATIVE<->PROCESS	PROCESS
THING<->PROCESS	PROCESS
THING<->STATUS_PARTICIPANT	THING
?<->OPERATOR_JOIN<->?	?
PROCESS_CLOSED<->.	PROCESS
?<->OPERATOR	?
PROCESS<->THING	PROCESS

Illustrations for the interaction patterns are given below:

- In English, when we observe an adjective followed by a noun, the relationship profiled by the adjective is qualified on the noun. The adjective is in *STATUS_QUALIFIER* schema and the noun is a type of *THING* schema (e.g., *THING_PARTICIPANT*, *THING_CLOSED-AUTONOMOUS* etc.). The profile determinant of the resulting noun phrase is a THING and not a QUALIFIER. Thus, the interaction pattern is *QUALIFIER<->THING: THING.* In the Welsh language this definition is modified as *THING<->QUALIFIER: THING* because the adjective typically occurs after the noun in Welsh.
- When a prepositional phrase / a noun phrase is a part of a verb phrase, the preposition / noun phrases are in *STATUS_PARTICIPANT*/ *NOUN_PARTICIPANT* schema and the verb is a type of PROCESS schema. The interaction results in the integration of the PARTICIPANTs and the PROCESS. The integration of PARTICIPANTs and the PROCESS is a type of PROCESS. Finally, this fully integrated PROCESS plus its PARTICIPANTs profiles an EVENT schema. These functional properties are the basis of the definition of EVENT_CLOSED-AUTONOMOUS schema resulting from the following interactions:

- *'EVENT_CLOSED-AUTONOMOUS' : ['(PARTICIPANT)<->PROCESS_CLOSED-AUTONOMOUS', '(DESCRIPTIVE)<->PROCESS_CLOSED-AUTONOMOUS']*

Just like the illustrations given above, each and every schema definition given in tables 5.4 to 5.9 can be understood based on the composition and interaction properties of each part of the definition. Among the composition definitions in tables 5.4 to 5.8, some schemas are dependent and some are autonomous. The dependent schemas by their very nature depend upon some head schemas as a part of their definition. This dependent-head relation is also a part of the schema definition component which is shown in table 5.10.

5.1.7 Dependent and head schemas

Some linguistic structures are inherently dependent on more autonomous structures to characterise their profile. We introduced the notions of autonomy and dependence in section 4.1.5. In the schema definition component, there are some schemas that are dependent on more autonomous schemas to fulfil the expectations along the interaction axis. Such definitions are given in table 5.10. For example, the word 'election' in the expression 'election results' is said to be in THING_COMBINATIVE schema because it is construed as a part of a compound noun. By definition, this implies that its construal depends on some other schematic THING (in this example the schematic THING is elaborated by 'results'). Thus THING_COMBINATIVE is a dependent schema whose head is a THING. Similar explanations can be given for other schemas in table 5.10.

Table 5.10 Dependent schemas and their head schemas

Dependent schema	**Head schema**
THING_COMBINATIVE	THING
PROCESS_CLOSED-DEPENDENT	PROCESS
STATUS_CONTINUATIVE-DEPENDANT	THING
STATUS_COMBINATIVE	THING_PARTICIPANT
STATUS_CLOSED-DEPENDENT	PROCESS_CLOSED-AUTONOMOUS, THING_PARTICIPANT
EVENT_CONTINUATIVE-DEPENDENT	EVENT_CLOSED-AUTONOMOUS, STATUS_CLOSED-DEPENDENT, THING_CLOSED-AUTONOMOUS
EVENT_COMBINATIVE-DEPENDENT	PROCESS_CLOSED-AUTONOMOUS,

	EVENT_CLOSED-AUTONOMOUS
EVENT_CLOSED-DEPENDENT	EVENT_CLOSED-AUTONOMOUS

5.1.8 Miscellaneous definitions

Most schema names in our tables are autological i.e., their names themselves indicate the functional properties represented by the schema. For example, EVENT_CLOSED-AUTONOMOUS itself means that the schema is at the event level, it can potentially close the discourse with no further expectation and can be construed independently of other parts of the sentence. But there are some schemas which are not so e.g., EVENT_PARTICIPANT can function as a THING, which is why it occurs in table 5.4. However, since the name itself is not autological we list their profiles explicitly as miscellaneous schemas in tables 5.11, 5.12 and 5.13 so that they can be used later in the schema assembly component.

Table 5.11 Miscellaneous schemas - Composition

Schema name	**Analogous to**
1. PRON 2. EVENT_PARTICIPANT	THING
1. PROCESS, STATUS 2. EVENT 3. OPERATOR_CLOSED	RELATION

Table 5.12 Miscellaneous schemas - Interaction

Schema name	**Its expectation**
1. PARTICIPANT 2. DESCRIPTIVE	CONTINUATIVE
1. QUALIFIER	COMBINATIVE

Table 5.13 Miscellaneous schemas - Autonomy

Schema name	**Its expectation**
1. PARTICIPANT 2. CLOSED 3. DESCRIPTIVE	AUTONOMOUS
1. COMBINATIVE 2. QUALIFIER	DEPENDENT

With this, the schema definitions are concluded. The next component 'Schema assembly component' takes these definitions (five categories of *composition* schemas, the *interaction* schemas, *dependent-head* schemas and the *miscellaneous* schemas) and uses them to integrate parts of a sentence into whole.

5.2. Schema assembly component

```
input : corpus, schema_definitions, assembly_definitions
output: Every sentence is assembled into one CLOSED schema
STEP 1 - PREPROCESSING STAGE;
for sentence in corpus do
    pos_sequence←POS-TAGGER(sentence);
    for current_pos in pos_sequence do
        list_cx_schemas←ALL-POSSIBLE-SCHEMAS(current_pos) FROM schema_definitions;
    end
    cx_schema_list_sequence ← Sentence stored as a sequence of list of all schemas for each word;
end
retain_span ← 0 ;
STEP 2 - PROCESSING STAGE;
for (pos_tag, list_schema) in (pos_sequence, list_cx_schemas) do
    running_parse ← ADD-TO-RUNNING-PARSE (pos_tag, list_schema);
    if CHECK-LANDING-POINT (list_schema) then
        retain_span ← retain_span +1;
        GET-VALID-ASSEMBLIES (running_parse, assembly_definitions);
        multiple_running_parses ← RETAIN-MULTIPLE-RUNNING-PARSES (running_parse);
        if retain_span > 2 then
            STEP 3 - PREDICTION AND FEEDBACK STAGE;
            for possible_parse in multiple_running_parses do
                for assembly_pattern in possible_parse do
                    assembly_pattern is of the form schema1 <=>schema2 =schema3;
                    head_predicted ← PREDICT-ASSEMBLY-HEAD-ANN (schema1 <=>schema2);
                    LEARN-ASSEMBLY-ANN(schema1<=>schema2=schema3);
                end
                - Collect the total error for each assembly prediction inside every possible parse.
            end
            - Choose the best parse as the one whose total error is minimum - Remove all other possible
              parses in multiple_running_parses and update it with the best parse chosen
        end
    end
end
```

Algorithm 5.1 Procedure for incremental cognitive parsing

The definitions from the previous component are used by the schema assembly component. This component interfaces with both schema definition component and schema prediction component and assembles the tokens into chunks, chunks into clauses and so on until the entire sentence is parsed. The procedure for incremental cognitive parsing performed by this component is shown in Algorithm 5.1. The algorithm treats every part of a sentence as made up of one or more construction schemas. When a new word in a sentence is read, all possible construction schemas for the word are assigned based on the definitions mentioned in section 5.1. If there are multiple schema assignments possible for the previous word in a sentence and if valid interactions are possible between the schemas of previous words and the current ones, then we have two possibilities: (a) we can either integrate them into one assembly to get a new schema right away (serial decision) (b) or add them to a list of unintegrated sequence of schemas until further context is available (parallel decision). In the serial decision, one parse is retained at any given time but in parallel decision multiple possible parses are

retained. In our implementation, both the possibilities are treated as valid parse decisions. This decision is in accordance with the functional requirement that the parsing model should be a hybrid of both serial and parallel processing.

As we read a sentence word by word, the parser creates a *running parse,* which is a list containing all possible serial and parallel decisions for the words encountered so far. With every new word read, the number of serial and parallel decisions increases and the number of possible parses becomes complex. The multiple possible parses are retained until a span of two verb phrases i.e., until two PROCESS_CLOSED schemas are encountered and their dependents are integrated to form two EVENT_CLOSED schemas. Let us call this span of the sentence the 'retain span' of multiple parses. After the retain span, the parse decisions will be pruned so that only the most likely path of assembly of schemas (i.e., the most likely parse) is chosen. All the other possibilities are discarded. Pruning decisions are based on the most likely predictions given by the schema prediction component (to be described in the upcoming section 5.3).

The schema assembly component carries out two major tasks. Firstly, it assigns schema analysis to the raw words, allows all possible assemblies between them and prunes them to update the running parse after every retain span. Secondly, it recognises the three different levels of grammatical organisation from the schema definition component and uses them in making parse decisions: token level, process level and event level. The token level, process level and event level schema definitions from tables 5.4 to 5.8 were discussed in section 5.1. Let us give a small illustration of how these definitions will be used by the schema assembly component.

5.2.1 Assign all possible analyses

On reading a word and its POS tag, the parser assigns all possible analyses for the word depending on its definition. For example, on seeing the word 'slow' we construe an ATEMPORAL RELATIONSHIP/ STATUS of 'being slow' that is attributed to an unknown THING. The focus is on the ATEMPORAL RELATIONSHIP/ STATUS and the THING slot is out of focus. We say that 'slow' instantiates ATEMPORAL RELATIONSHIP/ STATUS schema along its composition axis. This is shown in figure 5.2[4] . The unknown THING functions as an elaboration

[4] Conventions of image schema used in CG are more detailed and specific than what is used here. The figures used here do not share those conventions. This is just our own rough sketch for the sake of illustration.

site, which means that when a noun, e.g., 'transition', is encountered, it semantically corresponds to this site. The STATUS ('being slow') qualifies its attribute on the THING ('transition'). We say that 'slow' instantiates QUALIFIER schema along its interaction axis. This again is shown in figure 5.2.

Figure 5.2 Composition and interaction axes of an expression

How do we determine whether the expression 'slow' is construed as AUTONOMOUS or DEPENDENT along the autonomy axis? In certain expressions like 'the' it is clear that its construal necessarily depends on a THING and so it always instantiates a DEPENDENT schema along the autonomy axis. However, 'slow' can be conceived either autonomously or dependently. When used in attribute adjective constructions (e.g., 'slow transition'), the focus is on the STATUS slot and the unknown THING slot serves as a site of elaboration. So, in an attribute adjective construction, the adjective is construed as DEPENDENT. However, in predicate adjective constructions (e.g., 'the transition is slow'), the focus is both on the STATUS as well as unknown THING. Thus, as a predicate adjective, it is construed as AUTONOMOUS.

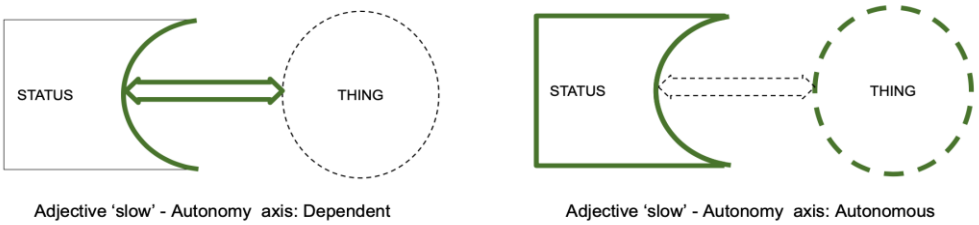


Figure 5.3 Autonomy axis of an expression

The construals of attribute and predicate adjectives and the difference in their autonomy is shown in figure 5.3. Based on the discussion above we can say that the adjective in the phrase 'slow transition' instantiates STATUS_QUALIFIER-DEPENDENT schema but in the expression 'the transition is slow' the same adjective has STATUS_QUALIFIER-AUTONOMOUS schema. In this way, every word has multiple possible analyses depending on how it can be

composed from token level schemas and how it can integrate with its neighbouring words. The parser retains these multiple possible schemas for every token and allows many serial and parallel decisions for integrating them until two EVENT_CLOSED schemas are encountered.

What do we mean by multiple serial and parallel decisions until two EVENT_CLOSED schemas are encountered? The token 'book' with the POS tag *noun* has two possible analyses: THING_CONTINUATIVE or THING_COMBINATIVE. It is analysed as THING_CONTINUATIVE if it is construed as a part of an action (e.g., 'The *book* fell …'). Alternatively, it can be mapped to THING_COMBINATIVE when it is construed as a part of a larger THING (e.g., *'book* shops'). Depending on the next token and its schema, one of the analyses can be discarded or retained. If the next token is 'fell' with the POS tag VBD, there are two possible schemas for this token: PROCESS_CONTINUATIVE-DEPENDENT or PROCESS_CLOSED-AUTONOMOUS. The two words can be integrated into a larger expression 'book fell' because the interaction between THING_CONTINUATIVE and PROCESS is valid. The profile of the resulting expression will be a PROCESS. Once this integration is made, the possibility of analysing the first word as THING_COMBINATIVE should be discarded. This is the serial decision. The parallel decision will be to retain both the analyses for the first word and the second word and wait for more context.

Alternatively, if the first word 'book' has the POS tag *verb*, it can be mapped to 'PROCESS' schema because it is construed as a relation unfolding through time (e.g., *book* a ticket ….) along its composition axis. Along the interaction axis, the token 'book' can be interpreted in 'CONTINUATIVE' or 'CLOSED' schema based on its neighbouring words (e.g., 'Please *book* the ticket' - 'CLOSED', 'to *book* the ticket' - 'CONTINUATIVE'). In both uses, the expression is construed as AUTONOMOUS. In this manner, the parser retains multiple possible analyses for each word and integrates them immediately if possible (serial decision) or allows all possibilities until more context (parallel decision).

5.2.2 Integration of expressions across the levels of grammatical organisation

From the concepts introduced in section 4.2.3, every expression at any syntactic level - such as words, phrases and clauses - can be analysed along three axes: composition, interaction and autonomy and their construction schema can be identified. From section 5.1, there are seven basic construals along composition axis: THING, RELATIONSHIP, PRONOMINAL,

PROCESS, STATUS, OPERATOR, EVENT, six basic construals along interaction axis: CONTINUATIVE, COMBINATIVE, PARTICIPANT, DESCRIPTIVE, QUALIFIER, CLOSED and three basic schemas along autonomy axis: AUTONOMOUS, DEPENDENT and JOIN. In section 5.1, we also explained how the schemas were defined for composition at three levels of grammatical organisation: token, process and event levels. We have a finite number of construction schema definitions, finite number of interaction schemas and finite number of levels of grammatical organisation in our schema definition component but a wide variety of syntactic phenomena should be analysed in terms of these finite number of schemas. In order to do that, the parser allows composition and integration of expressions across different levels of grammatical organisation.

Consider auxiliary verb constructions such as 'has come' (POS tags: VBZ VBN), 'would have given' (MD VB VBN), 'are doing' (VB VBG), 'must have been going on' (MD VB VBN VBG PRT). All these different expressions are recognised to be instances of *PROCESS CLOSED-AUTONOMOUS* schema. One path of integration for the expression 'has come' is: 'has come' < *VBZ VBN* < *PROCESS CLOSED-AUTONOMOUS<->STATUS CLOSED-DEPENDENT* < *PROCESS CLOSED-AUTONOMOUS*. In this notation, '<' means that the expression on the left side of this symbol is reanalysed as an instance of the right side expression. '<->' indicates that two schemas are integrated into one new schema. The path of integration is a ***parse*** of the expression. Let us see how this path of integration is recognised from the words. The sequence of tokens 'has come' is analogous to the VBZ *VBN* sequence because of the POS tags of the words. VBZ is analogous to PROCESS_CONTINUATIVE-DEPENDENT and PROCESS_CLOSED-AUTONOMOUS from table 5.5. VBN is analogous to STATUS_QUALIFIER, STATUS_CLOSED-AUTONOMOUS and STATUS_CLOSED-DEPENDENT from table 5.6. Therefore, there are four ways of interaction possible between these two words. Out of these, the interactions that will lead to successful integration of neighbouring words are treated as valid parses and added to the list of running parses. Those that do not lead to integration of neighbouring words are treated as invalid. For example, one of the possible ways to integrate the sequence *VBZ VBN* into one larger assembly is *PROCESS_CLOSED-AUTONOMOUS<->STATUS_QUALIFIER.* This path of assembly is subsequently treated as invalid, because there is no definition for this type of integration in any of the tables in the schema definition component. The two available options for valid integration are *PROCESS_CONTINUATIVE-DEPENDENT<->PROCESS_CLOSED-AUTONOMOUS* and *PROCESS_CLOSED- AUTONOMOUS<->STATUS_CLOSED-DEPENDENT*. If the former choice is made, the subsequent path of integration will be *PROCESS_CONTINUATIVE-DEPENDENT<->PROCESS_CLOSED-*

AUTONOMOUS < PROCESS_CLOSED-AUTONOMOUS according to table 5.5. In this way, the ***parse 1*** is obtained from the words and POS tags. If the other valid option is chosen by the parser, the subsequent path of integration will be *PROCESS_CLOSED-AUTONOMOUS<->STATUS_CLOSED-DEPENDENT < PROCESS_CLOSED-AUTONOMOUS.* This is ***parse 2*** obtained from the same sequence of words. In this particular example both the valid paths of integration can be considered as meaningful parses. The ***parse 1*** interpretation is that the first word 'has' is an auxiliary verb (dependent) on the main verb (autonomous). The ***parse 2*** interpretation is that the first word 'has' is the main verb which completes the discourse (autonomous and closed) and the word 2 'come' is a status verb that depends on it (dependent). Both of them are meaningful in the following sense: in the parse 1 interpretation, the word 'has' is grammaticalised as an auxiliary verb but in parse 2 interpretation, the word 'has' still retains its lexical meaning before this grammaticalisation which then interacts with the past participial form of verb in status interpretation.

In this way, the parser allows the integration of schemas across different levels i.e., between token level and event, between two event levels, between two process levels, between token level and process level and so on as long as they are valid according to the definitions in the tables 5.4 to 5.13. Because the schema definitions themselves are carefully proposed in terms of their meaning, many invalid assemblies are rejected in subsequent steps of integration of schemas. Rejection of some types of invalid assemblies and approval of all valid assemblies is an inherent feature of the schema definition. So, the other components have only to focus on the other types of invalid assemblies that are not inherent to the schema definitions. If there are multiple valid assemblies permitted by the schema definition component, one of them will be more likely in the given context of neighbouring words and their schemas. In order to choose the most likely parse we use the schema prediction component.

5.3 Schema prediction component

The schema prediction is the third component of the parser that chooses one optimal path of schema integration out of all valid assemblies allowed by the definition and assembly components. In the previous subsection 5.2.2, we showed that the same expression 'has come' can be parsed in two different ways because of the schema definitions and because the assembly component allows integration across different levels as long as it is valid. In many cases, the number of possible paths of integration may be many and all of them can be valid purely in terms of their composition and interaction but in real-time parsing, only some

of the paths are favoured by the humans than the other ones. The garden path sentences discussed in section 4.3.1 is an example for this. In order to resemble this psycholinguistic reality and also to prune the number of serial and parallel decisions in parsing, we used the schema prediction component which chooses one parse as the most likely parse after every **landing point**. We define the landing point as any part of the sentence at which the addition of more possible parses are halted and one optimal path of integration of the words seen so far is chosen. An instance of the landing point (introduced earlier in sections 4.3.2 and 5.2) was a span of two verb phrases (which we called the retain span) after which multiple paths of integration are pruned and one parse out of all possible parses is chosen as the most likely. Other than the verb phrases, there are other components of the sentence that can indicate the point of pruning the multiple parses. We noted down such parts of the sentence as landing points. A list of landing points noted down were: ['PROCESS', 'CLOSED', 'EVENT', 'THING', 'QUALIFIER', 'JOIN', 'COMBINATIVE']. We varied the number of landing points and qualitatively checked the parse decisions made in each case and fixed the number of landing points to be 2.

After every landing point, we prune the number of possible parses and then the schema prediction component is used to find out the most likely parse. By pruning, we mean that wherever serial decisions were immediately available but parallel decisions for parse were retained, we discard the parallel paths and retain only the serial decisions. This prunes the number of possible parses significantly. After this step, we send the remaining number of possible parses to the schema prediction component. For example, if a landing point is reached after seeing three words and there are ten possible paths for parsing let us say that three of those paths are invalid because there is no schema definition supporting such an integration. Out of the remaining seven possibilities, let us say two of them are parallel decisions which are discarded at the pruning step. Now we have five possible valid parses out of which one has to be chosen.

How do we choose which of the five possible paths should be considered a valid parse? In every possible path, two words are integrated first into a larger expression. Then the integrated words become the neighbour to the third word and so on. At each step of integration there are two neighbours whose schemas try to integrate with schemas of the second neighbour. The schema prediction component is used by the schema assembly component to decide which of these possible interactions is more likely.

Given one current schema and a list of possible upcoming schemas, the schema prediction

component returns one amongst the latter to be the best candidate for interaction with the first schema. As a concrete simple example, for the expression 'College friends meetup', we have following schemas for each token: ['THING_COMBINATIVE', 'THING_CONTINUATIVE'], ['THING_COMBINATIVE', 'THING_CONTINUATIVE'] and ['THING_COMBINATIVE', 'THING_CONTINUATIVE']. Final valid possible paths of assemblies are

1. 'College friends': *THING_COMBINATIVE* < 'College friends meetup': *THING_CONTINUATIVE*
2. 'College friends': *THING_CONTINUATIVE* < 'College friends meetup' : *THING_CONTINUATIVE*
3. 'College friends': *THING_COMBINATIVE* < 'College friends meetup' : *THING_COMBINATIVE*

Out of these three, the first one is the most likely parse if the next word after this expression is a verb e.g., 'College friends meetup happened'. The second parse is valid schematically but less likely for the given words. The third parse is more likely if the next word is another noun so that the whole expression becomes a noun phrase such as 'college friends meetup invitation'. The schema prediction component gives probability of assembly in each case and the most likely path of integration is chosen as the optimal parse after every landing point.

5.3.1 Schema prediction and Artificial Neural Networks

We used Artificial Neural Networks (ANNs) to implement the schema prediction component (Yegnanarayana 2009). In this section, we will give a short introduction to ANN and our motivation for using them in schema prediction. ANNs are inspired by neural networks in the human brain and their ability to learn through examples (Jain et al. 1996). ANNs are composed of multiple nodes or neurons that take input data to perform simple operations on the data and pass these results to other nodes. The output from each node is called the node activation. The nodes are connected to one another with each connection associated with a weight which determines the node activation given an input. The nodes are arranged into interconnected layers with each node capable of making simple decisions and propagating it to other nodes. Each node learns to associate the input data to output activation by altering its weight value. An activation function determines whether a node should be activated or not by calculating weighted sum of the connected nodes and also adds bias to it. This function performs non-linear transformation of the inputs by which the network learns to define complex relationships between input and outputs. By defining complex relationships between

the input data and output activations of multiple nodes and by altering their link weights, ANNs are able to learn to recognise complex patterns in the data. Some of the common activation functions are sigmoid function, tanh function, Rectified Linear Unit (RELU) and softmax function (Sharma et al. 2017).

The neurons in the feedforward neural network are organised into input, hidden and output layers. There must always be one input layer and one output layer in the network. In the input layer, the data is fed to the neural network with the aim of making predictions. Typically, the input data is a set of real values with each value fed into a neuron in the input layer. This layer takes the inputs, performs calculations on the input through its neurons and transmits the results to the next layer. The output layer, which is responsible for producing the final result, takes the values transformed by the previous layers and computes the output. Between the input and output layers there can be zero or more layers that are private to the neural network and not accessible to the outer world. These are the hidden layers of the network. Figure 5.4 shows a feedforward neural network with an input layer, two hidden layers and an output layer. The hidden layers and the output layer are shown to use tanh and softmax activation functions respectively.

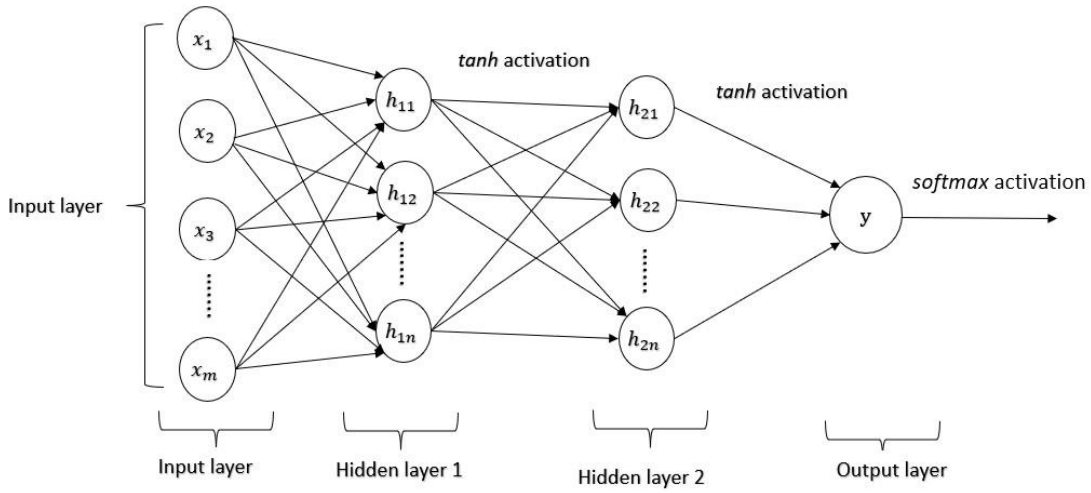


Figure 5.4 Schematic diagram of a feedforward neural network

ANNs are useful for a variety of pattern recognition tasks. As human beings we learn the right and wrong ways to do a task based on the feedback we receive on our performance. Neural-networks learn what is right and wrong by a feedback process called back-propagation. Given a training set of inputs for which the correct output is known, the network learns to compare its output with the correct output it was meant to produce and adjusts the network weights by propagating the difference between the two outputs through hidden layers into the input

neurons backwards. After repeating this process for several input-output pairs, the backpropagation reduces the difference between the network output and the intended output. The network finally learns the correct output for the given input.

In our proposed parser, the schema prediction component takes the sequence of words that are reduced to a sequence of schemas by the previous two components. The number of valid ways in which these sequences of schemas assemble with each other is constrained by the schema definitions and the landing points. After pruning, we have a list of possible paths of assembling them, each one of them a valid parse according to the schema definition component. Each parse available after pruning is a unique path to assemble the sequence of schemas identified by the previous two components. The goal of the schema prediction component is to choose the best among these unique paths by selecting the most likely path of integration. Each unique path involves repeated integration of neighbouring schemas into one larger schema.

Two tasks are performed by the schema prediction component to choose the best parse:

A. Given the current schema and a list of possible assemblies with its neighbour, it chooses the assembly that reduces the network error i.e., the most likely assembly in the neighbourhood is predicted
B. To learn all valid assemblies from every possible parse got from the previous two components.

The pruned parses showing many possible assemblies are used as the training data for the neural network. The basic idea is to train the network to learn the likelihood of assembly between two neighbouring schemas. Locally, the most likely assembly is chosen out of all possible assemblies in a neighbourhood. Globally, the parse that reduces the total loss in schema predictions along its path is chosen as the best parse. By choosing the most likely assemblies locally and by minimising the total prediction loss of the network, we get a globally valid parse. We have so far presented the components of the parser, algorithm and system overview. Now we will discuss the implementation details such as the library used, how the input is formatted and the parameters of the system.

5.3.2 ANN architecture for schema prediction

We used Dynet to implement the schema prediction component. Dynet is a neural network library developed by Carnegie Mellon University and many others (Neubig et al. 2017). Dynet

implements the neural network models based on the dynamic declaration of network structure. The basic template for implementing neural network models with Dynet involves the following steps:

1. Creating a Dynet Model.
2. Adding the necessary Parameters and LookupParameters to the model. A trainer object is created and associated with the Model.
3. For each input example:
    a. A new ComputationGraph is created. An Expression that represents the desired computation is added to this ComputationGraph.
    b. The result of the computation is fed forward using value() or npvalue() functions.
    c. During training, an expression that represents the loss function is calculated and used for back-propagation.
    d. The Trainer object is used to update the Model parameters after backpropagation.

A feedforward neural network with two hidden layers was implemented with the DyNet toolkit. The toolkit provides many optimisers to tune the Model parameters such as SimpleSGDTrainder, CyclicalSGDTrainer, AdagradTrainer and so on. We initialised a simple stochastic gradient trainer (SGD) with a learning rate 0.1. Each training sample is of the form (*Inputs*, *Output*) where the *Inputs* are the embeddings of two construction schemas that make up a chunk and the actual chunk tokens. *Output* refers to the embedding of the construction schema predicted for the chunk tokens. For example, the chunk '61 years' is formed by assembling the tokens '61' and 'years'. The schemas corresponding to the two tokens are STATUS_QUALIFIER and THING_PARTICIPANT. In this example, the phrase '61 years' and the assembly 'STATUS_QUALIFIER <==>THING_PARTICIPANT' are treated as *inputs* and the resulting schema 'THING_PARTICIPANT' is the *output.* The code snippets showing the initialisation of the neural network parameters and the calculation of the score of a chunk assembly are given in Appendix B.

The Dynet embedding matrix is of the dimension *nchunks x EMB_SIZE. nchunks* is the number of chunks that are available for training, EMB_SIZE is the size of the embedding layer and HID_SIZE is the size of the hidden layer. The EMB_SIZE and HID_SIZE are initialised as 128. The number of hidden layers N=2. We then create a graph that will calculate the score vector for the training instance. The word embedding *W_emb* and the construction schema embedding *Cons_sch_emb*

are concatenated to form the *Inputs* part of the training and the resulting construction schema becomes the *Output*. The calculation of the score of the chunk assembly is performed is shown in Appendix B. We used *tanh* activation function for the hidden layer and *softmax* function in the output layer. The chunk and construction schema embeddings are concatenated and affine transformation is performed before applying the *tanh* activation function in the hidden layer. The score for the assembly of chunks is then used to calculate the loss i.e., the negative log likelihood of the output after taking the softmax over the scores. After performing backward pass and accumulating the gradients we update the training parameters.

The basic idea is that the assembling words, POS tags and the construction schemas are initialised as model parameters and are updated during training. The assembly head of the construction schemas is predicted as the output. Given two expressions next to each other and their possible construction schemas, the network predicts the assembly head. In each case, the rules of assembly in the schema definition component would predict a list of many possible assemblies. Every such possible assembly given by the schema definition is used as the feedback for the network prediction and the loss function is back-propagated. The network prediction for which the loss function is minimum is chosen as the most likely assembly for the given two schemas. Because the schema definitions are functional and the same assemblies occur in different contexts with different neighbourhoods, we run this schema prediction component for thousands of constructions from thousands of sentences. The more the number of times the network sees two schemas and their assembly to be valid in different environments, the stronger the association between the two becomes.

5.4 Advantage of Wordnet Ontology in schema prediction

The words, POS tags, all possible schemas and assemblies allowed by the schema definitions and the neighbourhood are the constraints within which the network learns to choose the most likely assembly. On the one hand, the POS tags provide low-level syntactic information. On the other hand, schemas have been defined to reflect the high-level semantics. Both types of information are combined into assemblies, which reflect the dichotomy between syntax and semantics. However, better balance can be struck between the two extremes by grouping synonymous words together. For example, consider the noun 'dad'. It has many synonyms, including 'dada', 'daddy', 'pa', 'papa', 'pappa' and 'pop'. When each of them is considered independently, any machine learning algorithm would need much more data to infer that they indeed follow the same pattern of use and, therefore, play the same role in schemas and

assemblies. However, if they are grouped together explicitly and treated as synonyms, i.e., referring to the same semantic concept, then their occurrences in a corpus can be aggregated, thus allowing the machine learning algorithm to learn the higher-level patterns faster and reduce overfitting to a particular surface form and instead focus on its referent.

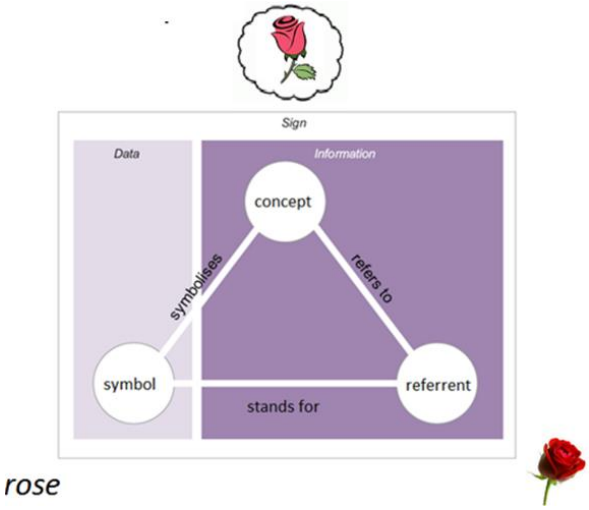


Figure 5.5 Meaning triangle

Let us explain this point using a simple model of semantics called the meaning triangle (Wright, 2003; Amaglobeli, 2012). In this triangle, a sign is broken into three parts: symbol, concept and referent. Figure 5.5 provides an example in which the sign is the word 'rose', the concept is a generic notion of a rose and the referent is a particular instance of that concept. This is perhaps best illustrated by a quote from Romeo and Juliet:

> What's in a name? That which we call a rose
> By any other name would smell as sweet;

A natural language offers a plethora of synonyms to signify the exact same concept as our example of the word 'dad' mentioned above illustrates. By treating these words as synonyms, the parser effectively processes them as a single concept, therefore, operating at a concept level, which bridges the gap between the POS tags as the low-level syntax and the schema level as the high-level semantics. We pointed out how synonyms can help embed lower-level semantics into the parser and facilitate the learning of the way in which POS tags can be combined into schemas and how schemas can then be combined into assemblies. A practical question that remains is how to identify synonyms. As a practical method of synonym identification to embed lower-level semantic information into the parser, we used Wordnet.

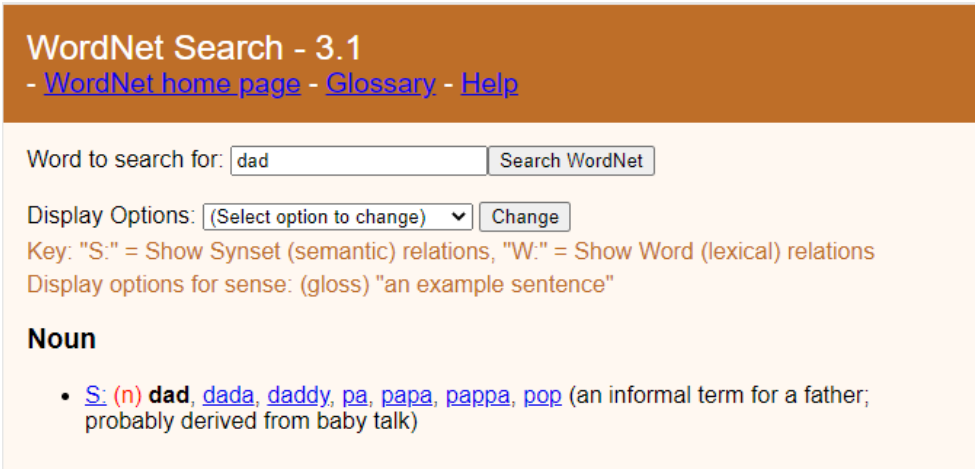


Figure 5.6 Wordnet search

Wordnet is a large lexical database that groups words into sets of cognitive synonyms called synsets. Each synset represents a distinct concept that is linked with other synonymous words sharing the same synset. For example, the synonymy between 'shut' and 'close' is established by two synsets shared by the two words 'close.v.01' and 'close.v.02'. Since WordNet is freely available and readily usable, we added the synset information along with words and POS tags to facilitate learning of POS tags into schemas and schemas into assemblies. In Welsh, such a project started only recently as a satellite of the CorCenCC project and became publicly available for researchers[5].

Even though the POS tags and the schema definitions themselves can model the grammar, by using the additional semantic / ontological information provided by resources like Wordnet (Miller 1995; Loper and Bird 2002), we will be able to improve the performance of the parser. We used NLTK's wordnet interface for getting the synsets that matched with the POS tag of the word in the corpus. Some words have one synset while others may have several synsets. When there were multiple synsents for words, we chose the first synset and added it as the parameter to the network along with word and POS tags.

5.5 Adapting the schema definition component for Welsh language

The schema definition component introduced in section 5.1, is implemented as a dictionary and a list. The definitions given in tables 5.4 to 5.13 were based on our observations of English

[5] https://github.com/CorCenCC/wncy

syntax. The parser is primarily developed for English and we have attempted to adapt the scheme definitions to handle the peculiarities of Welsh syntax. The following are some of the grammatical properties of the Welsh language which makes it different from English. There could be more differences but our attempt here is just to demonstrate how the implementation can be adapted for the Welsh language as well.

1. The word order of the Welsh language is Verb-subject-object. The usual order of occurrence of syntactic units in a Welsh sentence is: preverbal particle, verb, subject, direct object, indirect object, preposition phrases, adverbs. For example, 'The cat killed the mouse' (Subject-Verb-Object) in Welsh is 'Lladdodd y gath y llygoden' (Verb-Subject-Object).
2. Typically adjectives and determiners come after the nouns that they modify. E.g., 'good boy' (adjective-noun) becomes 'bachgen da' (noun-adjective).
3. Focussed constituents are moved to the beginning of the sentence for emphasis. E.g., 'i Lundain yr wyt ti yn mynd' (Literally: To London I am going) to focus on 'to London'. Unlike English, most constituents in a Welsh sentence can be moved to the front.
4. Genitive relations are expressed in Welsh by apposition. E.g., 'llyfr ffrind' (Literally: 'book friend'. Meaning: 'friend's book').
5. Use of possessive adjectives along with verbal nouns to express passive voice, static passive, prepositional phrases that modify a noun. In English, these are expressed with past participle forms of verbs. E.g., *Passive voice:* 'The mouse was killed' is expressed as 'Cafodd y llygoden ei lladd' (Literally: 'The mouse got its killing'), *Stative passive:*'The chair is broken' is expressed as 'Y mae'r gadair wedi torri' (Literally: 'The chair is after its breaking'), *Prepositional phrase attributed on noun:* 'an opened letter' is expressed as 'llythyr wedi'i agor' (Literally: 'The letter after its opening').
6. The use of the word 'dim' / 'ddim' in various constructions such that it can be treated as an argument of a verb, an adverb / pseudo-argument, a quantifier, a pseudo-quantifier, a sentence final adverbial, focus-negating / constituent negation 'dim' (Willis 2006).
7. Welsh distinguishes direct and indirect relative clauses and therefore it uses different relative pronouns in these two types of clauses. When the relativised element is the subject of its clause or the direct object of an inflected verb, then relative pronouns are used but in all other cases a complementiser is used (Roberts 2005).

Even though there could be many other formal differences, we are only focusing on what

changes should be made to the parser's schema definition component so that the same implementation can be reused for Welsh language. In order to identify the required changes, we noted down the syntactic differences between the two languages and mapped which of them are purely formal and which of them can be handled functionally. By separating the purely formal aspects of a language from its functional aspects, we were able to adapt the schema definitions for Welsh language without huge changes. For instance, the fact that Welsh recognises two different types of relative clauses and uses different syntactic elements (relative pronoun and complementisers) to encode them cannot be predicted in any functional manner. Therefore, relative pronouns and complementisers are mapped to the token level schemas as they cannot be composed functionally from other schemas. However, the way these syntactic elements are used to construct relative clauses in Welsh can be mapped to process level and event level *QUALIFIER* schemas. In table 5.14, the list of such changes is mentioned.

Table 5.14 Changes to schema definition component - Welsh

English schema	Welsh schema	Example
Interaction pattern QUALIFIER<->THING : THING	Interaction pattern THING<->QUALIFIER : THING	'Wythnos yma' (This week), 'dilema sylfaenol' (fundamental dilemma)
PROCESS_CLOSED-AUTONOMOUS VB, VBZ, VBD PROCESS_CONTINUATIVE-DEPENDANT<->VB PROCESS_CONTINUATIVE-DEPENDANT<->VBZ PROCESS_CONTINUATIVE-DEPENDANT<->VBD PROCESS_CLOSED-AUTONOMOUS<->PRT PROCESS_CLOSED-AUTONOMOUS<->PROCESS_CONTINUATIVE-AUTONOMOUS PROCESS_CLOSED-AUTONOMOUS<->STATUS_CLOSED-DEPENDANT	PROCESS_CLOSED-AUTONOMOUS VBF PRT<->PROCESS_CLOSED-AUTONOMOUS PROCESS_CLOSED-AUTONOMOUS<->STATUS_CLOSED-AUTONOMOUS PROCESS_CLOSED-DEPENDANT<->PROCESS_CLOSED-AUTONOMOUS	'Roedd ef' (Was) 'Gawn ni' (We get) 'Yr wyf i' (I am) 'Yr wyf i yn meddwl' (I think, I am thinking) 'Nid ydynt' (They are not)

PROCESS_CLOSED-DEPENDANT<->PROCESS_CLOSED-AUTONOMOUS		
PROCESS_CONTINUATIVE-AUTONOMOUS TO<->VB, VBG	PROCESS_CONTINUATIVE-AUTONOMOUS VB	Mwynhau (enjoying, to enjoy, enjoy)
STATUS_DESCRIPTIVE ADV	STATUS_DESCRIPTIVE ADV, NEGY	Fwyfwy (more and more, increasingly) Ddim (not)
STATUS_CLOSED-AUTONOMOUS VBN, ADV, ADJ	STATUS_CLOSED-AUTONOMOUS YNPRED<->ADJ VBADJ<->ADJ YNPRED<->THING_PARTICIPANT VBADJ<->THING_PARTICIPANT YNPRED<->PROCESS_CONTINUATIVE-AUTONOMOUS CYSY5TAIR<->ADJ	Yn tynnu (Pulls, Draws) Yn yr ail (in the second) Yr un melfedaidd, esmwyth (the velvety, smooth one) Yn dda (well) Cyn gynted (as soon as) wedi ei wysgo (exhausted)
EVENT_CONTINUATIVE-AUTONOMOUS EVENT_CONTINUATIVE-DEPENDANT<->EVENT_CLOSED-AUTONOMOUS EVENT_CONTINUATIVE-DEPENDANT<->STATUS_CLOSED-DEPENDANT EVENT_CONTINUATIVE-DEPENDANT<->THING_CLOSED-AUTONOMOUS	EVENT_CONTINUATIVE-AUTONOMOUS EVENT_CONTINUATIVE-DEPENDANT<->EVENT_CLOSED-AUTONOMOUS EVENT_CONTINUATIVE-DEPENDANT<->STATUS_CLOSED-DEPENDANT EVENT_CONTINUATIVE-DEPENDANT<->THING_CLOSED-AUTONOMOUS EVENT_CLOSED-AUTONOMOUS	os nad awn ni (If we don’t go) a yw’n wir Tra ei fod yn wir Wrth siarad â'r gohebwyr
EVENT_CONTINUATIVE-DEPENDANT ADP, ADV, DET	EVENT_CONTINUATIVE-DEPENDANT CONJ, VBF	os nad awn ni adref (If we do not go home) pan fydd hyn yn digwydd (When this happens)..
EVENT_COMBINATIVE-DEPENDANT DET, PRON, ADJ	EVENT_COMBINATIVE-DEPENDANT VBF, RELPRON, PVPART	anodd i ni erbyn heddiw ydyw dychmygu 'r a gymerai 'r saint ym y bobl (It is difficult for us to imagine today what the saints

		took from the people). Y llyfr y clywsom amdano (The book that we heard about) y prif broblemau yw cost a y prosesau sy 'n ymwneud a+ 'r (The main problems are the cost and the processes involved).
EVENT_CLOSED-AUTONOMOUS (PARTICIPANT)<->PROCESS_CLOSED-AUTONOMOUS (DESCRIPTIVE)<->PROCESS_CLOSED-AUTONOMOUS	EVENT_CLOSED-AUTONOMOUS (PARTICIPANT)<->PROCESS_CLOSED-AUTONOMOUS EVENT_CLOSED-CONTINUATIVE (DESCRIPTIVE)<->PROCESS_CLOSED-AUTONOMOUS	Y mae 'r bws in sefyll yn yr orsaf fysiau (The bus is standing in the bus station)

The list of differences mentioned above are based on our personal observations of how the same function is realised differently in the two languages. Typologically, both English and Welsh are right branching languages where the modifiers come after the head in a sentence. The occurrence of modifiers after the head is more prevalent in Welsh where many adjectives, numeric adjectives, determiners, subjects and objects of the sentences occur after their respective heads. Adverbs are expressed as adjectives in atemporal relationship with verbs. The English adverb 'well' is expressed as 'yn dda' which can be construed as 'being in a good atemporal relationship with a process'. This construction is analogous to how finite verb phrases are realised in Welsh e.g., 'John is speaking' is expressed as 'Mae ef yn siarad' (Literally: 'Is John in-speaking' where the verb phrase is expressed as imperfect processes (yn+<verb>) assembled with with finite, perfect *atemporal relationships* (i.e., mae, bydd, gall etc.). Tenses may also be expressed morphologically ('aeth' - 'went', 'meddai' - 'told'). In some cases, the modalities are expressed as experience for the subject ('rhaid i mi' - Literally: 'necessity for me'). Perfect aspect is expressed as imperfect process in adverbial relation with finite, perfect *atemporal relationships* ('wedi dod' - Literally: 'Is john after coming', meaning 'John has come', 'efallai fod John wedi dod' - Literally 'Possibly is John after coming' meaning 'Joh may have come'). Wherever constructions have similar construals, they are expected to exhibit similar functional behaviour and therefore we group them under same or similar schemas in our definitions.

Our ability to adapt the schema definition component for Welsh has been limited by our lack of intimate knowledge of Welsh language. Linguists and native speakers of Welsh language with better understanding about the range of constructions available in Welsh and their functional behaviour can take up this approach and identify what syntactic constructions are functionally analogous to each other to create better definitions in the schema definition component.

In this chapter we have presented the schema definition, schema assembly and schema prediction components. The adaptation of the schema definition component for Welsh language and the implementation of schema prediction component using ANN was discussed. An algorithm that uses these three components for incremental cognitive parsing and the overview of the system was also shown. This chapter focussed on the design of the system and its implementation details. In the next chapter, we will present the experiments and evaluation of the proposed parser.

Chapter 6 Evaluation

In this chapter, we describe the experiments that were conducted to evaluate the parser's performance. Most commonly, parsers are evaluated against a gold standard, which has been manually annotated by the experts. The major evaluation bottleneck for a radically different parsing approach is the lack of an appropriate gold standard. In section 6.1, we describe the inherent limitations of gold standard evaluation in detail. In section 6.2, we present some examples from the parser outputs to demonstrate why multiple valid parses are possible in our approach. This makes defining a gold standard inherently impossible. Because of these issues, we propose four different ways of evaluation to rate the parser performance and to ascertain the meaningfulness of the chunks identified and their labels:

1. Repurposing existing parse treebanks to enable quantitative evaluation of the chunking performance against the constituents in a phrase structure tree
2. Manual evaluation by listing the range of linguistic constructions covered by the parser
3. Identifying the number of edits required to obtain a correct assembly
4. Qualitative expert evaluation based on Likert scales in online surveys

The English and Welsh corpora selected for the experiments in the first three evaluations are described in detail in section 6.3. The fourth evaluation strategy was performed at the final stage of research and therefore a different corpus was used for Welsh language alone. These four evaluation methodologies and their results are described in sections 6.4, 6.5, 6.6 and 6.7 respectively. Finally, we conclude the chapter by discussing the strengths and limitations of our evaluation strategies and scope for future improvements.

6.1 Limitations of gold standard evaluation

From the systematic literature review described in chapter 2, we find that the most common evaluation strategy used in state-of-the-art parsers is comparison against a gold standard. This method is suitable only when annotations are available. This approach is problematic for a number of reasons. Out of around 7000 languages in the world (Abney and Bird, 2010), only a small number of languages (20 languages) are considered high-resourced languages (Baumann and Pierrehumbert 2014). When it comes to parsing, natural language resources like syntactic treebank are available only for major languages like English, Chinese, German,

Portuguese, etc. Apart from this, there is a more fundamental problem with gold standard evaluation. Although papers on parsing may report higher accuracy in terms of precision, recall and F-measure against gold standard, it need not translate into higher impact in NLP applications as such (Poibeau and Messiant 2008). Tools like parsers are developed so that they can be used as building blocks for real-world NLP applications such as text summarisation, machine translation, question-answer systems and so on. It is implicitly assumed that the higher the parser accuracy when evaluated against the gold standard, the more suitable it will be for the real-world applications.

However, there are many studies which reveal that there is a weak, sometimes negative, correlation between the improvement in gold standard accuracy and the usefulness of the tool in real NLP applications. For example, Miyao et al (2009) report that with a 1% absolute improvement in the parser accuracy, there is approximately 0.25% performance improvement in an application that extracts protein-protein-interaction. Diego and Molla (2003) report their findings from comparing the intrinsic and extrinsic evaluations of parsing systems. They point out that intrinsic evaluations are of very limited value in the context of an answer extraction system. Kilgarriff et al. (2014) show that there is no correlation between gold standard evaluation of a parser and the performance of the collocation extraction system where the parser was used. Katz-Brown et al. (2011) reported that there is negative correlation between machine translation performance and the English parsers that were used in the translation system. The higher the accuracy of the translation application the lower the accuracy of the gold standard evaluation. Kovar et al. (2016) discuss the problems in the evaluation methods used in NLP and argue that comparing against a gold tree structure is not meaningful in the context of NLP applications. This evaluation only measures the ability of the parser to imitate the predefined syntactic annotation scheme laid out in annotation manuals based on some linguistic theory. What is learned by the parser is heavily dependent on the gold annotation decisions which might not reflect the language intuition that are needed in the actual applications. In addition to these complexities, there are also NLP tasks where the conception of a gold standard is not truly possible, especially with high inter-annotator agreement. For example, terminology extraction, topic detection, text summarisation are some tasks where gold standards are difficult to define or used only for specific tasks, thus cannot be generalised. For example, a gold standard corpus created for extracting terminology in the biomedical domain (Kim et al., 2003) skews the results when used for a general terminology extraction application.

Thus, the inherent limitations of evaluation against a gold standard are:

- The training model runs the risk of overfitting the gold standard
- The problem of inter-annotator agreement
- Weak correlation of the intrinsic evaluation with extrinsic evaluations

6.2 Unsuitability of gold standard evaluation for the proposed parser

The outputs of our parser do not represent constituency or dependency trees. The parser matches the tokens from raw text to basic construction schemas and assembles them into higher level construction schemas based on patterns of assembly. The construction schemas of the chunks can be assembled into higher level schemas via multiple valid paths. Therefore, defining a gold standard annotation for the outputs of the proposed parser is inherently impossible. An example sentence and one valid path of assembly is shown in tables 6.1, 6.2 and 6.3.

Table 6.1 Tokens in an input sentence

The	federal	government	suspended	sales	of	U.S	savings	bonds	because
Congress	has	not	lifted	the	ceiling	on	government	debt	.

Table 6.2 POS Tags of tokens in an input sentence

DT	JJ	NN	VBD	NNS	IN	NNP	NNS	NNS	IN
NNP	VBZ	RB	VBN	DT	NN	IN	NN	NN	.

Table 6.3 One of the possible valid parses of the input sentence

Word	**Construction schema**
The	STATUS_QUALIFIER_DEPENDENT
federal	THING_COMBINATIVE_DEPENDENT
government	THING_PARTICIPANT_AUTONOMOUS

The federal government	THING_PARTICIPANT_AUTONOMOUS
suspended	PROCESS_CLOSED_AUTONOMOUS
sales	THING_PARTICIPANT_AUTONOMOUS
The federal government suspended sales	EVENT_CLOSED_AUTONOMOUS
of	STATUS_COMBINATIVE_DEPENDENT
U.S	THING_COMBINATIVE_AUTONOMOUS
savings	THING_COMBINATIVE_AUTONOMOUS
bonds	THING_PARTICIPANT_AUTONOMOUS
U.S savings bonds	THING_PARTICIPANT_AUTONOMOUS
of U.S savings bonds	STATUS_COMBINATIVE_AUTONOMOUS
sales of U.S savings bonds	THING_PARTICIPANT_AUTONOMOUS
The federal government suspended the sales of U.S savings bonds	EVENT_CLOSED_AUTONOMOUS
because	EVENT_CONTINUATIVE_DEPENDENT
Congress	THING_PARTICIPANT_AUTONOMOUS
has	PROCESS_CONTINUATIVE_DEPENDENT
not	STATUS_ASSOCIATIVE_DEPENDENT
lifted	PROCESS_CLOSED_AUTONOMOUS
has not lifted	PROCESS_CLOSED_AUTONOMOUS
Congress has not lifted	EVENT_CLOSED_AUTONOMOUS
because Congress has not lifted	EVENT_CONTINUATIVE_AUTONOMOUS
the	STATUS_QUALIFIER_DEPENDENT
ceiling	THING_PARTICIPANT_AUTONOMOUS
on	STATUS_QUALIFIER_DEPENDENT
government	THING_COMBINATIVE_AUTONOMOUS
debt	THING_PARTICIPANT_AUTONOMOUS
.	OPERATOR_CLOSED_AUTONOMOUS
government debt	THING_PARTICIPANT_AUTONOMOUS

on government debt	STATUS_QUALIFIER_AUTONOMOUS
the ceiling on government debt .	THING_PARTICIPANT_AUTONOMOUS
because Congress has not lifted the ceiling on government debt.	EVENT_CONTINUATIVE_AUTONOMOUS
The federal government suspended the sales of U.S savings bonds because Congress has not lifted the ceiling on government debt .	EVENT_CLOSED_AUTONOMOUS

In the parse shown above, each expression is mapped to a construction schema starting from individual tokens to phrase chunks, clausal chunks and even the entire sentence. There are multiple valid ways to group and assemble the words into larger schemas. Therefore, there are multiple valid paths to build a correct parse. Mapping them to a single gold standard annotation is therefore not possible. In this section, we present examples from parser outputs to explain this point. Consider the following sentence from the Wall Street Journal corpus used as input for the English parser.

Table 6.4 Two valid paths of assembly to arrive at the same construction schema

Sentence		‘['Newsweek', ',', 'trying', 'to', 'keep', 'pace', 'with', 'rival', 'Time', 'magazine', ',', 'announced', 'new', 'advertising', 'rates', 'for', '1990', 'and', 'said', 'it', 'will', 'introduce', 'a', 'new', 'incentive', 'plan', 'for', 'advertisers', '.', '']
A chunk in the sentence		[‘trying to keep pace with rival Time magazine..’]
Construction schema for the chunk		EVENT_CONTINUATIVE_AUTONOMOUS
Possible paths of assemblies (words and schemas)	Path 1	1. [‘trying’]<->[‘to keep pace’]<->[‘with rival Time magazine’] [PROCESS_CONTINUATIVE-AUTONOMOUS]<->[PROCESS_CONTINUATIVE- AUTONOMOUS]<->[STATUS_PARTICIPANT]
		2. [‘trying’]<->[‘to keep pace with rival Time magazine’] [PROCESS_CONTINUATIVE-AUTONOMOUS]<->[PROCESS_CONTINUATIVE-AUTONOMOUS]
		3. [‘trying’]<->[‘to keep pace with rival Time magazine’] [PROCESS_CONTINUATIVE-AUTONOMOUS]<->[EVENT_CONTINUATIVE-AUTONOMOUS]

		4. [‘trying to keep pace with rival Time magazine’] [EVENT_CONTINUATIVE-AUTONOMOUS]
	Path 2	1. [‘trying’]<->[‘to keep pace’]<->[‘with rival Time magazine’] [PROCESS_CONTINUATIVE-AUTONOMOUS]<->[PROCESS_CONTINUATIVE- AUTONOMOUS]<->[STATUS_PARTICIPANT]
		2. ['trying to keep pace']<->['with rival Time magazine'] [PROCESS_CONTINUATIVE-AUTONOMOUS]<->[PROCESS_CONTINUATIVE-AUTONOMOUS]
		3. ['trying to keep pace']<->['with rival Time magazine'] [PROCESS_CONTINUATIVE-AUTONOMOUS]<->[EVENT_CONTINUATIVE-AUTONOMOUS]
		4. [‘trying to keep pace with rival Time magazine’] [EVENT_CONTINUATIVE-AUTONOMOUS]

In table 6.4, the construction schema of the gerund phrase is schematised as EVENT_CONTINUATIVE-AUTONOMOUS. However, this construction schema is assembled from lower-level construction schemas along two different paths of assembly. Both the paths are valid according to the schema definitions and patterns of schema assembly defined in the schema definition component. In this manner, a single construction schema can be assembled in multiple valid ways. Therefore, annotating a sentence as an assembly of construction schemas and comparing the parser output against the gold standard is not feasible. In order to evaluate whether what is learned by the system is even meaningful or valid, we performed evaluation using four approaches. The English and Welsh corpora selected for our experiment purposes using the first three evaluation methods are described in the next section.

6.3 Corpus selection

For the first three experiments we used a total of 3,912 sentences from a freely available portion of Wall Street Journal Corpus (WSJ) for English and 3,000 sentences from a portion of Cronfeg Electroneg o Cymraeg corpus (CEG) for Welsh (Ellis et al. 2001). WSJ corpus is a component of the Penn Treebank (Marcus et al. 1993) and it is available for free as a part of Python Natural Language Toolkit (NLTK) (Bird 2006). The corpus contains sentences of

different length and it is a resource that is useful for studying different constructions. The corpus has articles that are POS tagged and annotated for their formal syntactic structure. We use only the raw tokens and the POS information from the corpus and use it for the parser algorithm and ignore the syntactic annotations. The POS tagset of Penn treebank is shown in table 6.5.

Table 6.5 POS tagset used in Penn Treebank

POS tag name	**POS tag description**
CC	Coordinating conjunction
CD	Cardinal number
DT	Determiner
EX	Existential there
FW	Foreign word
IN	Preposition or subordinating conjunction
JJ	Adjective
JJR	Adjective, comparative
JJS	Adjective, superlative
LS	List item marker
MD	Modal
NN	Noun, singular or mass
NNS	Noun, plural
NNP	Proper noun, singular
NNPS	Proper noun, plural
PDT	Predeterminer
POS	Possessive ending
PRP	Personal pronoun
PRP$	Possessive pronoun
RB	Adverb
RBR	Adverb, comparative

RBS	Adverb, superlative
RP	Particle
SYM	Symbol
TO	to
UH	Interjection
VB	Verb, base form
VBD	Verb, past tense
VBG	Verb, gerund or present participle
VBN	Verb, past participle
VBP	Verb, non-3rd person singular present
VBZ	Verb, 3rd person singular present
WDT	Wh-determiner
WP	Wh-pronoun
WP$	Possessive wh-pronoun
WRB	Wh-adverb

A sample annotation from the corpus is shown in figure 6.1.

```
( (S
    (NP-SBJ
      (NP (NNP Pierre) (NNP Vinken) )
      (, ,)
      (ADJP
        (NP (CD 61) (NNS years) )
        (JJ old) )
      (, ,) )
    (VP (MD will)
      (VP (VB join)
        (NP (DT the) (NN board) )
        (PP-CLR (IN as)
          (NP (DT a) (JJ nonexecutive) (NN director) ))
        (NP-TMP (NNP Nov.) (CD 29) )))
    (. .) ))
```

Figure 6.1 Sample sentence from the WSJ corpus

There are many different conventions for annotating the POS tags of the words in a corpus. The WSJ corpus uses the Penn tagset made of 36 POS tags which are given in table 6.5. Since these tags are too granular for our schema definitions, we mapped each POS tag in the treebank to a smaller number of tags that are less granular. For example, JJ (adjective), JJR (comparative adjective), JJS (superlative adjective) are different types of adjectives

recognised in the Penn tagset. We map these three tags to a single tag ADJ which is definitely less detailed but captures enough information to be mapped to the token level schemas in our schema definition component. The list of POS tags that are finally used in the schema definition component are: NOUN, NUM, DET, PRON, TO, VB, VBG, MD, VBZ, VBD, ADV, ADP, ADJ, PRP$, CONJ. The POS tags from the annotated WSJ corpus are mapped to one of the above POS tags and used for experiments and evaluation. For our evaluation purposes, we chose sentences which had more than three words but less than 20 words. This is to avoid sentences which are mere interjections or replies to some questions (less than three words) but also to avoid a large continuous discourse (more than 20 words). A sample sentence from the text file used for our experiments and its token level schema assignments is shown below:

Table 6.6 Sample sentence from the corpus

Words	['Pierre', 'Vinken', ',', '61', 'years', 'old', ',', 'will', 'join', 'the', 'board', 'as', 'a', 'nonexecutive', 'director', 'Nov.', '29', '.', '']
POS tags	['NOUN', 'NOUN', '.', 'NUM', 'NOUN', 'ADJ', '.', 'MD', 'VB', 'DET', 'NOUN', 'ADP', 'DET', 'ADJ', 'NOUN', 'NOUN', 'NUM', '.', '']
Possible token-level schemas	[['THING_COMBINATIVE', 'THING_PARTICIPANT'], ['THING_COMBINATIVE', 'THING_PARTICIPANT'], ['OPERATOR_JOIN', 'OPERATOR_CLOSED'], ['STATUS_QUALIFIER', 'THING_PARTICIPANT'], ['THING_COMBINATIVE', 'THING_PARTICIPANT'], ['STATUS_CLOSED-AUTONOMOUS', 'STATUS_QUALIFIER', 'STATUS_CLOSED-DEPENDANT'], ['OPERATOR_JOIN', 'OPERATOR_CLOSED'], ['PROCESS_CLOSED-DEPENDANT'], ['STATUS_CLOSED-AUTONOMOUS', 'STATUS_QUALIFIER', 'PROCESS_CLOSED-AUTONOMOUS', 'STATUS_CLOSED-DEPENDANT', 'PROCESS_CONTINUATIVE-AUTONOMOUS'], ['EVENT_COMBINATIVE-DEPENDANT', 'STATUS_COMBINATIVE', 'EVENT_CONTINUATIVE-DEPENDANT', 'THING_PARTICIPANT'], ['THING_COMBINATIVE', 'THING_PARTICIPANT'], ['STATUS_QUALIFIER', 'STATUS_CONTINUATIVE-DEPENDANT', 'EVENT_CONTINUATIVE-DEPENDANT'], ['EVENT_COMBINATIVE-DEPENDANT', 'STATUS_COMBINATIVE', 'EVENT_CONTINUATIVE-DEPENDANT', 'THING_PARTICIPANT'], ['STATUS_CLOSED-AUTONOMOUS', 'STATUS_QUALIFIER', 'STATUS_CLOSED-DEPENDANT'], ['THING_COMBINATIVE', 'THING_PARTICIPANT'], ['THING_COMBINATIVE', 'THING_PARTICIPANT'], ['STATUS_QUALIFIER', 'THING_PARTICIPANT'],

	['OPERATOR_JOIN', 'OPERATOR_CLOSED'], ['STATUS_CLOSED-AUTONOMOUS', 'STATUS_QUALIFIER', 'EVENT_QUALIFIER', 'EVENT_COMBINATIVE-DEPENDANT', 'PROCESS_CLOSED-DEPENDANT', 'THING_COMBINATIVE', 'PROCESS_CLOSED-AUTONOMOUS', 'EVENT_PARTICIPANT', 'EVENT_CLOSED-DEPENDANT', 'STATUS_CONTINUATIVE-DEPENDANT', 'STATUS_PARTICIPANT', 'OPERATOR_JOIN', 'STATUS_CLOSED-DEPENDANT', 'STATUS_COMBINATIVE', 'EVENT_CONTINUATIVE-AUTONOMOUS', 'PRON_PARTICIPANT', 'OPERATOR_DESCRIPTIVE', 'EVENT_CONTINUATIVE-DEPENDANT', 'STATUS_DESCRIPTIVE', 'EVENT_CLOSED-AUTONOMOUS', 'THING_CLOSED-AUTONOMOUS', 'PROCESS_CONTINUATIVE-AUTONOMOUS', 'OPERATOR_CLOSED', 'THING_PARTICIPANT']]

The CEG corpus that is used for Welsh experiments has 1 million words of written Welsh prose selected from a wide range of text types in modern written Welsh. We chose the CEG corpus for Welsh language experiments because we started working on this when the CorCenCC corpus (Knight 2020) was not fully ready and because the initial results were published using this corpus. The sentences which had more than three words but less than 20 words were chosen for our experiments just like in English. The CEG corpus contains the following information annotated for each sentence: token, lemma, POS tag, document number, sentence number, token number. We used the token and POS tag for our experiments. The POS tagset was more granular like in English, which we mapped to less detailed tags in order to be used in the schema definition component. All the original POS tags available from the CEG corpus are given in table 6.7. They are reduced to one of the following tags and used in the schema definition component: NOUN, ADP, VBF, PRON, CONJ, DET, YNPRED, NOUN, EXCL, ADV, ADJ, VBADJ, VB, PVPART, OPER, NEGPART, NUM, CARDIN, PRT, RELPRON, NEGY and CYSY5TAIR.

Table 6.7 Mapping the Welsh POS tags to less granular tags

Original tag	**New tag**		**Original tag**	**New tag**
np	NOUN		foreign	NOUN
prep	ADP		NegPart	NEGPART
vbf	VBF		number	NUM
pron	PRON		alphnum	NUM

conj	CONJ		DemPron	PRON
DefArt	DET		part	PRT
YnPred	YNPRED		punct	OPERATOR
npl	NOUN		excl	EXCL
cardm	CARDIN		adv	ADV
RelPron	RELPRON		nperson	NOUN
CompAdj	ADJ		letter	NOUN
adj	ADJ		npm	NOUN
vbadj	VBADJ		other	NOUN
vb	VB		npf	NOUN
nf	NOUN		negy1	NEGY
cardf	CARDIN		SupAdj	ADJ
card	CARDIN		cprep	ADP
nm/f	NOUN		cysy5tair	CYSY5TAIR
nplace	NOUN		EqAdj	ADJ
pronm	PRON		pronf	PRON
adjf	ADJ		PvPart	PVPART
adjpl	ADJ		conj	CONJ
negyd	NEGY		abbr	ABBR

Two sample sentences with annotation are shown below.

Table 6.8. Sample sentences with POS mappings

Sentence No.	Token	POS tag	Mapped POS tag	Lemma
1	doedd	vbf	VBF	bod:62
1	yr	DefArt	DET	yr
1	iaith	nf	NOUN	iaith
1	ddim	negy1	NEGY	dim

1	yn	vbadj	VBADJ	yn
1	gwybod	vb	VB	gwybod
1	.	punct	CONJ	.
1	.	punct	CONJ	.
2	doedd	vbf	VBF	bod:62
2	cefn	npf	NOUN	cefn
2	ddim	negy1	NEGY	dim
2	yn	vbadj	VBADJ	yn
2	gwybod	vb	VB	gwybod
2	chwaith	adv	ADV	chwaith
2	.	punct	CONJ	.

We chose a subset of the WSJ corpus and CEG corpus where the sentence lengths were more than 5 but less than 20. Table 6.9 shows the summary of the datasets used in the experiments for English and Welsh.

Table 6.9 Summary of the datasets used

Data	**English (WSJ)**	**Welsh (CEG)**
Number of sentences	3912	3000
Average sentence length	13.65	12.78
Maximum sentence length	19	19
Minimum sentence length	6	6

Out of the total number of sentences, the final 50 sentences were considered for evaluation and the remaining sentences were all used for training the schema prediction.

6.4 Quantitative evaluation by comparing chunks against the constituents in phrase structure tree

In this approach, the parse output is not an explicit representation such as constituency or dependency tree. The correctness of the parse is implicit in the schema definitions and the

order in which the words are assembled progressively from token level schemas into event level schemas. As long as each step of assembly is valid and larger spans of text are successfully mapped to one or more of the valid schemas in the schema definition component, the parsing is considered valid. The entire sentence would be analysed as some type of CLOSED schema finally. The parsed output does not follow traditional representation such as constituency or dependency tree, but the sequence in which locally cohesive schemas were assembled into self-similar schemas at higher levels of organisation. Therefore, it was not possible to reuse existing treebanks as a gold standard directly. Since these definitions are manually defined by us, how do we measure the validity of the patterns recognised by the parser? One naive approach that we considered was to evaluate the intermediate assemblies in the parse against the constituents in the constituency tree. We assumed that if the assembly heads predicted by the system are correct, the text within the span must match the constituents in the standard phrase structure tree. The idea is that if the parse chosen by the network is correct, the words and the schemas involved in all the assemblies that make up a parse should be a valid constituent in the constituency tree. The matching constituent should not be the leaf nodes in the tree because those are just words. When all the assembly heads of the network's parse match at some constituency level in a constituency, we consider it to be correct and otherwise incorrect. We used the constituency representation of 50 sentences in English and manually annotated the constituency representation of 50 sentences in Welsh. The accuracy of predicting assembly heads all of which have a match in the constituency tree are shown in table 6.10. Adding synset information to the model parameters along with POS tags and the construction schemas improves the accuracy of getting a correct parse.

Table 6.10 Evaluation results

Language	Accuracy (Constituencies matched / Total number of constituencies)	Condition
English	83.46%	Without synsets
Welsh	76.51%	Without synsets
English	87.19%	With synsets
Welsh	78.85%	With synsets

This is a baseline result for parsing by applying the theoretical insights from Cognitive Grammar. These evaluation results were obtained for 50 test sentences. Also, the evaluation criterion of comparing the assembly heads with constituents allows for ungrammatical

structures such as inverted constituencies (e.g., ‘from Japan the man’ instead of ‘the man from Japan’) to be evaluated as valid outputs. To overcome the above limitations, we used two other strategies for parser evaluation.

6.5 Manual evaluation by listing the range of constructions covered by the parser

One simple way to evaluate the grammatical coverage of the parser is to do a manual survey of a list of linguistic constructions covered by the system and not covered by the system. This evaluation procedure does not involve any corpus resource. Moreover, by looking at how the parser chunks the words, what schemas are assigned and how these chunks are assembled together to form more complex expressions, we shall be able to identify the list of constructions covered by the parser. This method is at least useful in the sense of guiding and monitoring how good the development of the cognitive parsing strategy has been. With that goal in mind, we attempted to collect a list of linguistic constructions covered by the parser. The list in table 6.11 was prepared by manually going through the schemas and assemblies recognised by the parser on the 50 test sentences in English (out of 3,912 sentences from WSJ corpus) and 50 test sentences in Welsh (out of 3,000 sentences from CEG corpus).

Table 6.11 Construction patterns covered by the English parser

Type of grammatical construction	**Construction schemas involved**	**Examples from test sentences recognised by the parser**
Nouns, Noun phrases, pronouns as arguments of a verb	THING_PARTICIPANT PRON_PARTICIPANT THING_CLOSED-AUTONOMOUS	‘the substance’ ‘an old story’ ‘surviving workers’ ‘Going for a walk everyday’ 'The average seven-day compound yield of the 400 taxable funds'
Preposition phrases as arguments of verb	STATUS_PARTICIPANT	‘In 1956’ ‘to the problem’
Part of a named entity, Part of a noun compound, Genitive case expressions	THING_COMBINATIVE	‘Consolidate Gold Fields PLC’ ‘of cancer deaths’ ‘Its’
Relative clauses	EVENT_QUALIFIER	‘symptoms that show up’ ‘Researchers who studied the workers’

Noun phrases modified by relative clauses	EVENT_PARTICIPANT	'Those of us who study' 'factory that made paper'
Verb phrase in gerund form, To + verb phrase	PROCESS_CONTINUATIVE-AUTONOMOUS THING_CLOSED-AUTONOMOUS	'.. currently waiving management fees..' 'Hasn't any authority to issue new debt obligations' 'To prove to 125 corporate decision makers ...'
Copula with predicate nominals, predicate adjectives	PROCESS_CONTINUATIVE-DEPENDANT STATUS_CLOSED-DEPENDANT THING_PARTICIPANT PROCESS_CLOSED-AUTONOMOUS	'Mr. Vinken is chairman of Elsevier N.V.' 'Imports were for $50.38 billion' 'Crocidolite is unusually resilient'
Auxiliary and modal verb constructions	PROCESS_CLOSED-DEPENDANT PROCESS_CONTINUATIVE-DEPENDANT PROCESS_CLOSED-AUTONOMOUS	'will support' 'is going' 'have died'
Passive form of verb, Stative verbs connected to adjectives / past participle form of verbs	STATUS_CLOSED-DEPENDANT PROCESS_CLOSED-AUTONOMOUS	'Is owned' 'are classified' 'Will be outlawed' 'Lie low' 'Remains unchallenged'
Adverbs, adjuncts	STATUS_PARTICIPANT STATUS_DESCRIPTIVE	'Into the campus' 'Particularly', 'more importantly'
Numerical adjectives Attribute adjectives Appositive phrases Genitive case Past participle noun modifiers	STATUS_QUALIFIER STATUS_COMBINATIVE THING_COMBINATIVE PRON_COMBINATIVE	'imported material' 'Clouds of blue dust' 'asbestos-related diseases' '35 years', 'these events', 'our work' 'Dreyfus World-wide Dollar, the top-yielding fund'
Coordinating conjunctions, punctuations, Sentence adverbials	OPERATOR_DESCRIPTIVE, OPERATOR_JOIN, OPERATOR_CLOSED	'But you have to recognise ..' '... managers can vary maturities and go after higher rates…' 'Champagne, desert followed ...'
Subordinate clauses	EVENT_CONTINUATIVE-AUTONOMOUS	'While imports increased sharply..' 'If congress doesn't act..'
Main clause, Sentence, Nuclear part of a	EVENT_CLOSED-AUTONOMOUS	'The treasury said ..

discourse or a dialogue		'Not this year.' 'Yes, Speaking.'
Attributive clauses	EVENT_CLOSED-DEPENDANT THING_CLOSED-AUTONOMOUS	'The idea is to prove that it's a good place for a company to expand' 'That the news was not reported is surprising'.

These are some of the example construction patterns that could be covered by the parser based on the outputs from the 50 test sentences. While manually going through the output we could notice many instances of ungrammatical patterns recognised as valid construction schemas. Listed in table 6.12 are some of those instances.

Table 6.12 Examples of parse errors in English

Wrong assembly of words	Construction pattern identified	Our comment
['with the substance , 28'] Sentence context: … worked with the substance, 28 have died.	STATUS_PARTICIPANT	Treated '28' as an apposition modifying 'the substance' and grouped them together as one unit. Correct parse should have grouped ['with the substance'] as one chunk and assemble '28' with upcoming words.
['that these events took place'] Sentence context: … that these events took place 35 years ago.	EVENT_CONTINUATIVE-DEPENDANT	Treats the attributed clause just like any other subordinate clause. Should recognised this group as EVENT_CLOSED-DEPENDANT
['Despite recent declines in yields , investors'] Sentence context: Despite recent declines in yields, investors continue to pour cash...	STATUS_PARTICIPANT	This schema is a valid interpretation if we assume that the decline is in both yields as well as investors. But it is wrong in the context of the sentence. The whole sentence finally could not be parsed fully because of this interpretation. After processing all the words, the parser had a series of unassembled schemas. If we had implemented a parse rectification/edit component that revisits earlier parse decisions this can be handled.

['And the city decided to treat its guests more like royalty or rock'] Sentence context: And the city decided to treat its guests more like royalty or rock stars than factory owners …	EVENT_CLOSED_AUTONOMOUS	Technically this is not incorrect. By changing the number of landing points and retain span in the parser settings, this kind of grouping can be avoided.
['The next morning , with a'] Sentence context: The next morning, with a police escort, busloads of executives and their wives raced …	THING_PARTICIPANT	The preposition and article should not have been assembled together with the noun phrase.

If the advantage of this method is that it does not need any corpus resource, its disadvantage is that such a listing does not give any precise information about the coverage of linguistic data. This is because much of the complexity of human languages arises from a range of interactions between its core constructions and its peripheral constructions whose boundaries are unclear.

For example, given a grammar, one might claim that *relative clause* is a type of construction covered by the grammar and a few examples could indeed be covered by it but it might not be able to deal with relative clauses which involve resumptive pronouns e.g., 'the opportunity which the investor thought **he** cannot afford to miss'. These types of constructions are usually missed because they are thought of as peripheral or marginal. The other disadvantage is that identifying the coverage of constructions in this manner is inherently subjective with no objective measure of the regularities in language captured by the parser. Having such a metric would be preferable because then the parser can be applied to different types of corpus and cross-corpus comparison and evaluation of the parser can be done.

6.6. Evaluation by identifying the number of edits required for a correct assembly

We did not choose a gold standard evaluation because there are multiple valid paths of assembly. This means there are many gold standards to the parser output. So instead of evaluating the output as either correct or incorrect and measuring the accuracy of the parser,

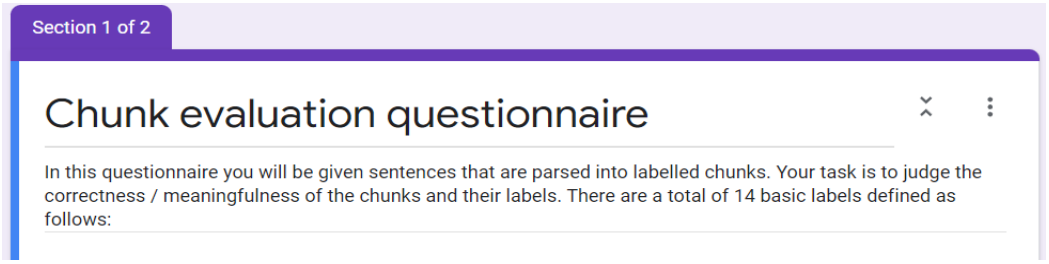
the idea is to measure how much correct or incorrect the parser analysis is. This can be done by measuring how many edits are required along the path of assembly to produce a valid parse. Every parse can be seen as repeated assembly of constructions and the path in which the words are assembled incrementally. We say that a parser is wrong when it groups words in such a way that there is no valid way to assemble the current schemas subsequently and we end up with an unassembled, disjoint sequence of schemas. The parser's decisions are also wrong if it results in a sequence of assembly that is not one of the many gold standards. If one or more decisions along the path of assembly can be edited and it results in subsequent valid parse then the performance of the parser can be measured in terms of how many edits are required before a valid assembly is formed. Let us consider that a sentence is made of *a* number of assemblies. If it requires *e* number of edits to arrive at a correct sequence of assembly, then the ratio *e/a* for every sentence gives an idea of how good or bad the parser is for the given sentence. For the same 50 English sentences and 50 Welsh sentences used in the previous experiments (in section 6.2), we manually calculated the 'e/a' ratio for each sentence and found the mean value for 50 sentences. If the average ratio is more, then it means that the decisions made by the parser in intermediate steps are more incorrect and more edits were required and the parser performance is bad. The average *e/a* ratio for English and Welsh sentences were 0.32 and 0.56 respectively.

6.7 Qualitative evaluation

In the context of evaluating the validity and meaningfulness of the chunks produced by the parser, we adopted a qualitative approach where the fluent speakers of English and Welsh languages were involved in online evaluation surveys. The idea behind this evaluation strategy is that since the construction schema definitions were identified by unifying syntax and semantics, they should be meaningful to the speakers of the language. In order to quantify the meaningfulness of chunks, which is essentially qualitative, a scientifically accepted and validated method is needed. Likert scale is one of the most fundamental and frequently used tools in educational and social sciences research that measures the attitude of the participants in a study (Joshi et al. 2015). A Likert scale is a set of statements given to the participants who are asked to show their level of agreement or disagreement to the given statements on a metric scale.

As a part of the qualitative evaluation of the chunks identified by our parser, we prepared a questionnaire where the participants were given sentences that are parsed into labelled

chunks. The task of the participants was to judge the correctness / meaningfulness of the chunks and their labels. The survey provided the instructions, which covered a total of 14 chunk labels, their definitions and examples (see figures 6.2 and 6.3 for examples). Each question corresponds to a sentence. The answers were elicited for each labelled chunk as analysed by our parser. Each chunk was associated with its own five-point Likert scale *(from strongly disagree to strongly agree)*. The average number of chunks in each sentence was 9.9 (with standard deviation of 1.1) for English and 9.2 (with standard deviation of 1.7) for Welsh. We prepared a total of 200 sentences in English and Welsh, respectively. There was a total of 1,982 chunks in English and 1,839 chunks in Welsh.



L1. Labels based on the content of a chunk

Label	Anything you can imagine as...	Examples
THING	An object	The boy The great bear Earthquake
PROCESS	An activity	came has done will be going to try
STATUS	An arrangement or a configuration	into slowly apart fast much
PRONOMINAL	A referring expression	He That guy She who spoke The long one He who must not be named
EVENT	An activity complex with all its participants and other relevant details	A clause like - If you are not really enthusiastic Participial gerund phrases like – Speaking to the reporters he said... A reported clause like – I heard that he went to Scotland Event connectives like – that he went to Scotland, when he came, while it is true
OPERATOR	A connector of multiple similar parts	Comma – Tom, Dick, Harry Clausal conjunction – I went to the canteen and found no one inside

L2. Labels based on the content of the chunk as well as its expected relations to other chunks

Label	What a chunk label is and what it expects to complete its sentential meaning	Examples
PARTICIPANT	Chunk – THING Expects – PROCESS or STATUS	The show starts at 2 PM 2 metres away
ASSOCIATIVE	Chunk – PROCESS or STATUS Expects – PROCESS or STATUS	was going has been happening went slowly extremely slow
QUALIFIER	Chunk – PROCESS or STATUS Expects – THING	The tempest A long journey Three idiots Many times The boy who lived He who must not be named
COMPOUND	Chunk – THING Expects – THING	Book shelf National Health Service Magnus Carlsen, the grandmaster
CLOSED	Chunk – THING, PROCESS, STATUS, EVENT Expects – No further expectation. Complete meaning is conveyed	Title of a book – complete in itself e.g., Harry Potter and the Philosopher's Stone A full sentence – I came home A finite clause – Although the film was good, it was not a commercial success
JOIN	An operator establishes join relation between any two chunks	Comma – Tom, Dick, Harry Clausal conjunction – I went to the canteen and found no one inside

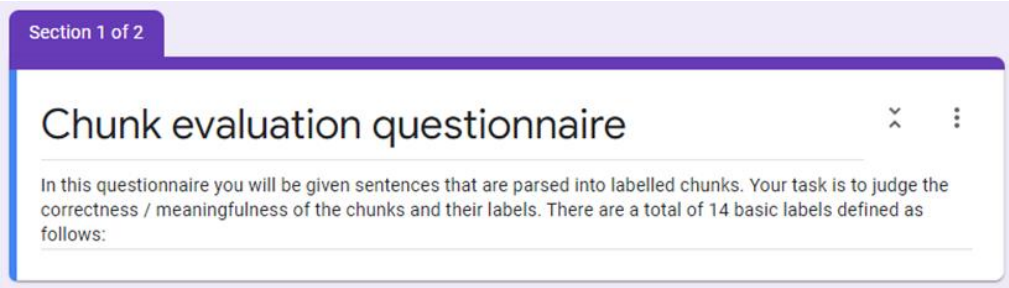
L3. Labels based on the autonomy of a chunk, i.e., whether it can be viewed as a stand-alone unit or not

Label	Chunk can be imagined as a stand-alone expression or not	Examples
AUTONOMOUS	Stand-alone expression – can be imagined on its own and not as a part of another expression Chunks – THING, PROCESS, EVENT, PRONOMINAL	London He went This is long Requirement
DEPENDENT	Depends on other stand-alone expressions – imagined as a part of another expression Chunks – PROCESS, STATUS	Into the castle Out of sight Will have done To walk

Figure 6.2 Participant instructions page for English

In the earlier three experiments described up to section 6.5, we used the CEG corpus sentences for Welsh and a portion of the WSJ corpus for English. For the sake of qualitative evaluation by fluent speakers, we parsed the sentences of CorCenCC corpus (Corpws Cenedlaethol Cymraeg Cyfoes). CorCenCC is a large-scale, open-source corpus of contemporary Welsh language containing over 11 million words of spoken, written and electronic data collected from various sources (Knight et al., 2017; 2020). In the earlier experiments we could not use CorCenCC corpus because the corpus development was taking place in parallel to this research. For English questionnaires, we used the sentences from the

same WSJ corpus just like previous experiments. Ethical approval for the involvement of human participants was granted by the School of Computer Science and Informatics' Research Ethic Committee (SREC) (Refer to Appendix C). We distributed an open call online. The participants were fluent speakers of the language that they were going to rate (either Welsh or English). Once the participants expressed their willingness to take part, we sent them the participant information sheet and obtained a consent from them. The participant information sheet contained information about the scope of the project, the purpose of the survey, information on what the participation would involve, clarification about participant freedom to withdraw at any stage, the possible benefits and risks in taking part in the project.



L1. Labels based on the content of a chunk

Label	Anything you can imagine as...	Examples
THING	An object	Y bachgen Yr arth fawr Daeargryn
PROCESS	An activity	daeth wedi gwneud bydd yn mynd i geisio
STATUS	An arrangement or a configuration	i mewn yn araf ar wahân cyflym llawer
PRONOMINAL	A referring expression	Ef y bachgen hwnnw hi a siaradodd yr un hir Yr hwn ni raid ei enwi
EVENT	An activity complex with all its participants and other relevant details	A clause like - os nad ydych yn wirioneddol frwdfrydig Participial gerund phrases like – Wrth siarad â'r gohebwyr ddoe, dywedodd y gweinidog... A reported clause like – Clywais ei fod wedi mynd i'r Alban Event connectives like - fod wedi mynd i'r Alban, pan ddaeth fy ffrind, tra ei fod yn wir
OPERATOR	A connector of multiple similar parts	Comma – Tom, Dick, Harry Clausal conjunction – Es i i'r ffreutur a gweld nad oedd neb y tu mewn

L2. Labels based on the content of the chunk as well as its expected relations to other chunks

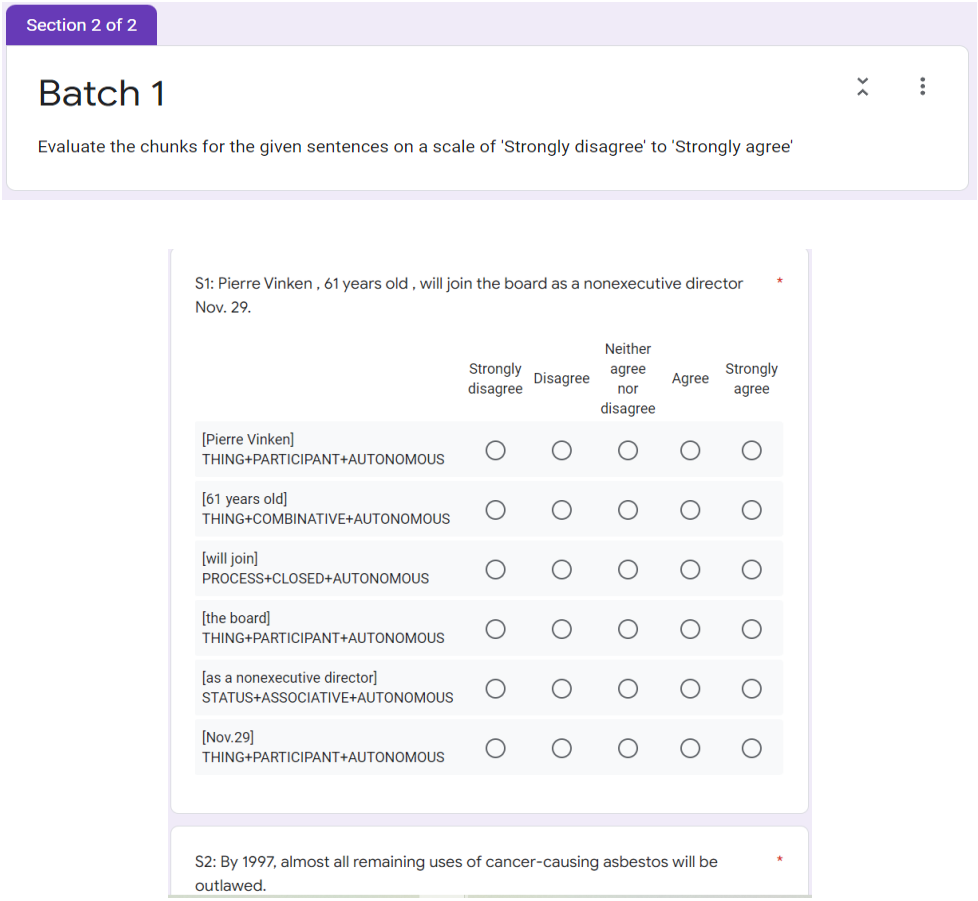
Label	What a chunk label is and what it expects to complete its sentential meaning	Examples
PARTICIPANT	Chunk – THING Expects – PROCESS or STATUS	Mae'r sioe yn dechrau am 2pm 2 fetr i ffwrdd
ASSOCIATIVE	Chunk – PROCESS or STATUS Expects – PROCESS or STATUS	ceisio mynd i mewn wedi bod yn digwydd aeth yn araf poenus dros ben
QUALIFIER	Chunk – PROCESS or STATUS Expects – THING	Y dymestl Taith hir tri mysgedwr lawer gwaith y bachgen oedd yn byw y dyn a ddaeth o Lundain Yr hwn ni raid ei enwi
COMPOUND	Chunk – THING Expects – THING	silff lyfrau gwasanaeth iechyd gwladol Magnus Carlsen, y grandfeistr gwyddbwyll
CLOSED	Chunk – THING, PROCESS, STATUS, EVENT Expects – No further expectation. Complete meaning is conveyed	Title of a book – complete in itself e.g., Harri Potter a maen yr Athronydd A full sentence – Es i i'r ysgol A finite clause – Er bod y ffilm yn dda, nid oedd yn llwyddiant masnachol
JOIN	An operator establishes join relation between any two chunks	Comma – Tom_Dick_Harry Clausal conjunction – Es i i'r ffreutur a gweld nad oedd neb y tu mewn

L3. Labels based on the autonomy of a chunk, i.e., whether it can be viewed as a stand-alone unit or not

Label	Chunk can be imagined as a stand-alone expression or not	Examples
AUTONOMOUS	Stand-alone expression – can be imagined on its own and not as a part of another expression Chunks – THING, PROCESS, EVENT, PRONOMINAL	Llundain aeth mae hyn yn hir gofyniad
DEPENDENT	Depends on other stand-alone expressions – imagined as a part of another expression Chunks – PROCESS, STATUS	i mewn i'r castell o flaen y drych wedi gwneud i fynd

Figure 6.3 Participant instruction page for Welsh

Specifically, we mentioned clearly that the participation was voluntary, that there was no need for the participants to explain their reasons for not taking part in the study and that they were free to withdraw their consent and participation at any time they wanted. No personal details were collected from participants except their email addresses to identify their submissions. The questionnaires were prepared using online forms and distributed to the participants (refer to figures 6.4 and 6.5). The contents of the instruction page were sent to the participants to refer back to the labels and examples while rating the sentences.



Section 2 of 2

Batch 1

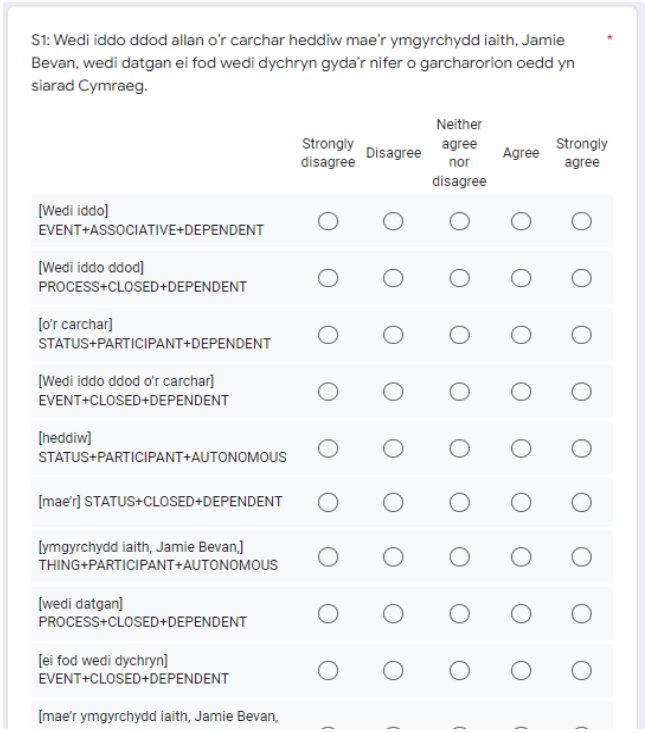
Evaluate the chunks for the given sentences on a scale of 'Strongly disagree' to 'Strongly agree'

S1: Pierre Vinken , 61 years old , will join the board as a nonexecutive director Nov. 29. *

	Strongly disagree	Disagree	Neither agree nor disagree	Agree	Strongly agree
[Pierre Vinken] THING+PARTICIPANT+AUTONOMOUS	○	○	○	○	○
[61 years old] THING+COMBINATIVE+AUTONOMOUS	○	○	○	○	○
[will join] PROCESS+CLOSED+AUTONOMOUS	○	○	○	○	○
[the board] THING+PARTICIPANT+AUTONOMOUS	○	○	○	○	○
[as a nonexecutive director] STATUS+ASSOCIATIVE+AUTONOMOUS	○	○	○	○	○
[Nov.29] THING+PARTICIPANT+AUTONOMOUS	○	○	○	○	○

S2: By 1997, almost all remaining uses of cancer-causing asbestos will be outlawed. *

Figure 6.4 Evaluation page for participants in English



S1: Wedi iddo ddod allan o'r carchar heddiw mae'r ymgyrchydd iaith, Jamie Bevan, wedi datgan ei fod wedi dychryn gyda'r nifer o garcharorion oedd yn siarad Cymraeg. *

	Strongly disagree	Disagree	Neither agree nor disagree	Agree	Strongly agree
[Wedi iddo] EVENT+ASSOCIATIVE+DEPENDENT	○	○	○	○	○
[Wedi iddo ddod] PROCESS+CLOSED+DEPENDENT	○	○	○	○	○
[o'r carchar] STATUS+PARTICIPANT+DEPENDENT	○	○	○	○	○
[Wedi iddo ddod o'r carchar] EVENT+CLOSED+DEPENDENT	○	○	○	○	○
[heddiw] STATUS+PARTICIPANT+AUTONOMOUS	○	○	○	○	○
[mae'r] STATUS+CLOSED+DEPENDENT	○	○	○	○	○
[ymgyrchydd iaith, Jamie Bevan,] THING+PARTICIPANT+AUTONOMOUS	○	○	○	○	○
[wedi datgan] PROCESS+CLOSED+DEPENDENT	○	○	○	○	○
[ei fod wedi dychryn] EVENT+CLOSED+DEPENDENT	○	○	○	○	○
[mae'r ymgyrchydd iaith, Jamie Bevan,					

Figure 6.5 Evaluation page for participants in Welsh

There was a total of 8 participants in English and 7 participants in Welsh who took part in the qualitative evaluation. Each participant was given an option to rate as many sentences as they wanted. The participants submitted their ratings independently of one another. We elicited a total of 4,768 evaluations for English and 4,431 evaluations for Welsh chunks. Their distribution is shown in figures 6.6 and 6.7.

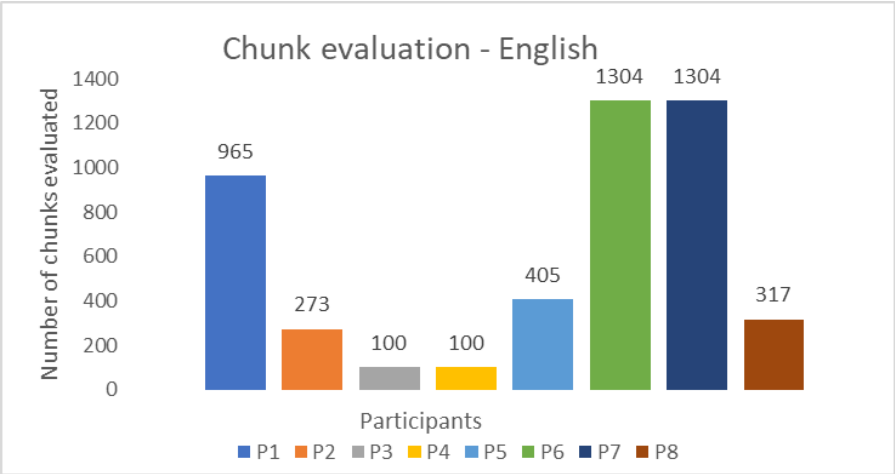


Figure 6.6 Distribution of number of chunks evaluated for English

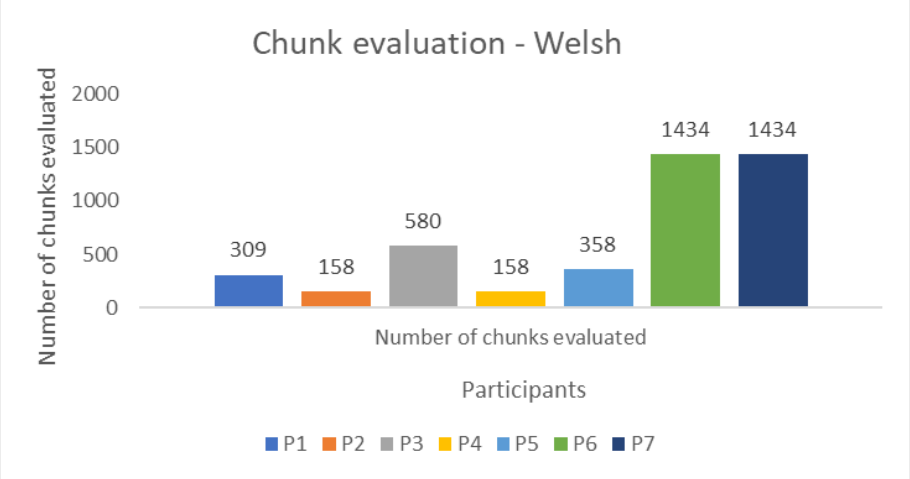


Figure 6.7 Distribution of number of chunks evaluated for Welsh

6.7.1 Statistical analysis

In statistics, validity refers to the extent to which the results represent what is supposed to be measured. In the absence of a gold standard, quantifying the validity of the parser output in terms of parser accuracy becomes impossible. Therefore, we measured the reliability of the parser outputs instead. Reliability refers to the consistency of the measure where the

results can be reproduced under the same conditions. When multiple evaluators rate the same sentence chunks under the same conditions, the inter-rater reliability can be assessed using a range of statistics. We chose two such statistics:

- Pairwise Cohen's Kappa coefficient (κ)
- Krippendorf's alpha (α)

Cohen's kappa coefficient (Cohen, 1960) is a statistical measure that quantifies the reliability of two evaluators rating the same quantity and how frequently they are in agreement with their judgements. Given N items that have to be rated in C categories, Cohen's kappa measures the agreement between any two raters to classify the N items into C categories. In our experiment, we had in total 1,982 labelled chunks in English and 1,839 labelled chunks in Welsh that were rated into one of the 5 classes: *Strongly agree, Agree, Neither agree nor disagree, Disagree, Strongly disagree.* The evaluator ratings were compared and their frequencies and proportions of agreement were measured and a kappa score was calculated. It is possible that a chance agreement occurs when two evaluators make guesses about their rating due to uncertainty. This can influence the reliability of the agreement measure. The advantage of kappa score is that it accounts for chance agreement of ratings between the participants. Cohen's kappa coefficient treats all disagreements equally, which is not suitable when the annotation categories are ordered as is the case with a Likert scale. In this case, it is preferable to use weighted kappa coefficient, which accounts for the degree of disagreement (Cohen, 1968).

Not all participants were expected to annotate all chunks from 200 sentences. By definition, inter-annotator agreement could not be calculated for chunks that had less than two annotations. Similarly, pairwise Kappa score cannot be calculated immediately when more than two annotations were available for a single chunk. In this scenario we need a metric that measures the overall agreement instead of pairwise agreement between participants. In order to deal with the problems of missing annotations for some chunks and multiple annotations for some other chunks, we need an agreement metric that is not affected by missing data, able to accommodate varying sample sizes and number of evaluators. Krippendorff's alpha (α) Krippendorff, 1970; 2011) is a statistical metric that is suitable in this context because it can handle missing data and supports categorical, ordinal, interval and ratio type data as well. We present our results in terms of both pairwise Cohen's kappa scores and Krippendorff's alpha score.

Wherever chunks are rated by at least two participants, we compare the ratings of each pair of participants to calculate their weighted kappa score as well as the agreement of their frequencies and proportions. The kappa scores were calculated using quadratic weighting. This is because the difference between the ratings 'Strongly agree' and 'Agree' should not be weighted similar to the difference between 'Agree' and 'Neither agree nor disagree'. The pairs of participants and the number of chunks commonly annotated by them are shown in figures 6.8 and 6.9 for English and Welsh respectively.

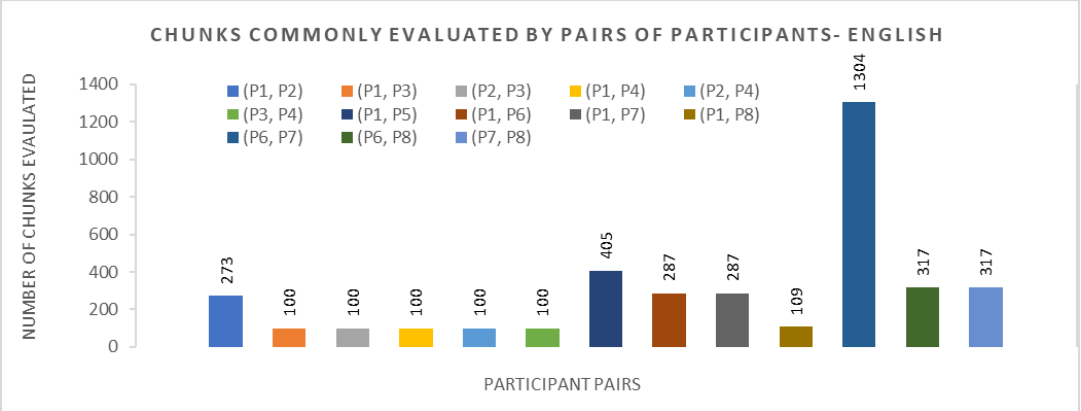


Figure 6.8 Number of chunks commonly annotated by pairs of participants - English

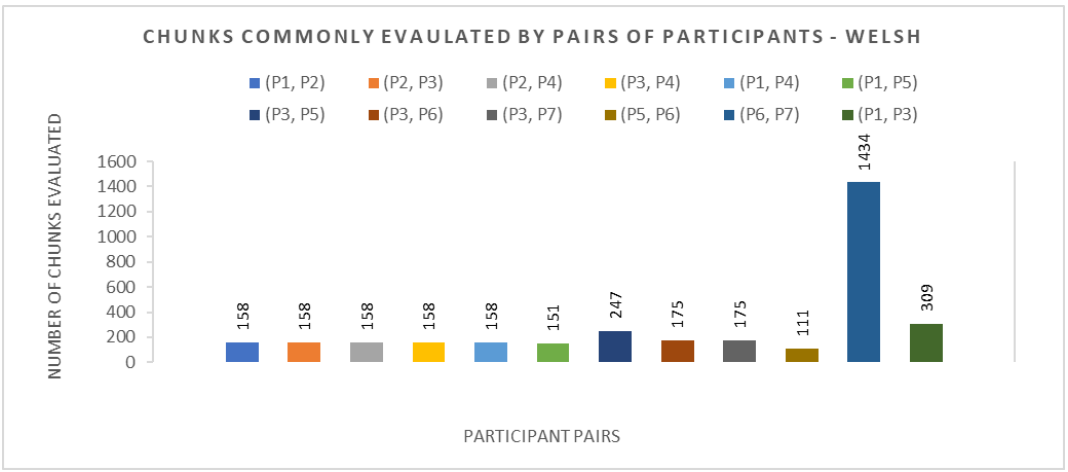


Figure 6.9 Number of chunks commonly annotated by pairs of participants - Welsh

From figures 6.8 and 6.9, we see that there are 13 participant pairs in English and 12 participant pairs in Welsh who had chunks common in their evaluations. The pairwise kappa scores calculated for each of these pairs in English and Welsh are shown in figures 6.10 and 6.11. In English questionnaires, there were 8 pairs of participants whose kappa scores ranged from 0.3084 to 0.5011 showing that the level of agreement between each of these pairs of

participants ranged from fair to moderate. Further 4 pairs showed slight to fair agreements ranging from 0.2086 to 0.2852. Only 1 pair showed none to slight agreement with the kappa score 0.1679. In Welsh, there was substantial agreement between 2 pairs of participants, moderate agreement between 5 pairs, slight to fair agreements between 3 pairs of participants, and no agreement between two pairs of participants (participant pairs P3-P6 and P3-P7). The kappa scores for both these pairs are plotted as -0.01 in figure 6.11 to make them visible.

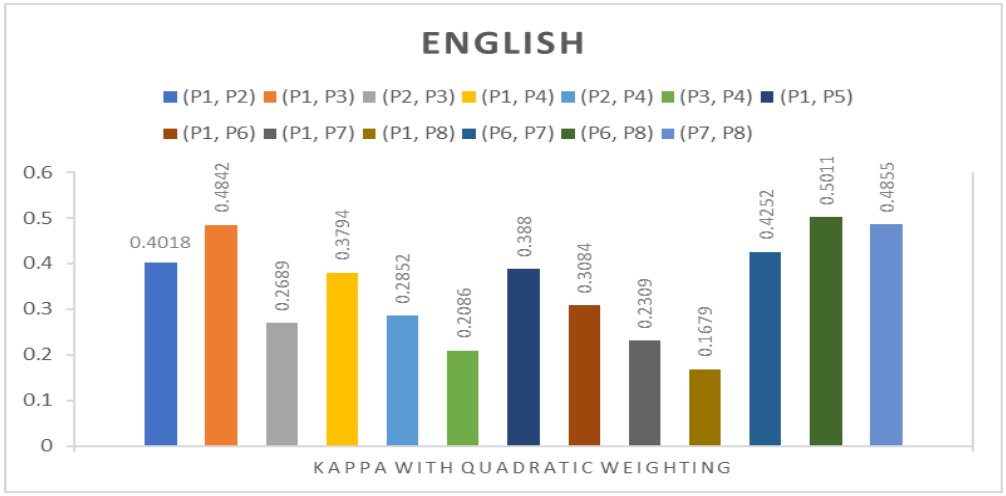


Figure 6.10 Kappa scores with quadratic weighting for 13 pairs of English participants

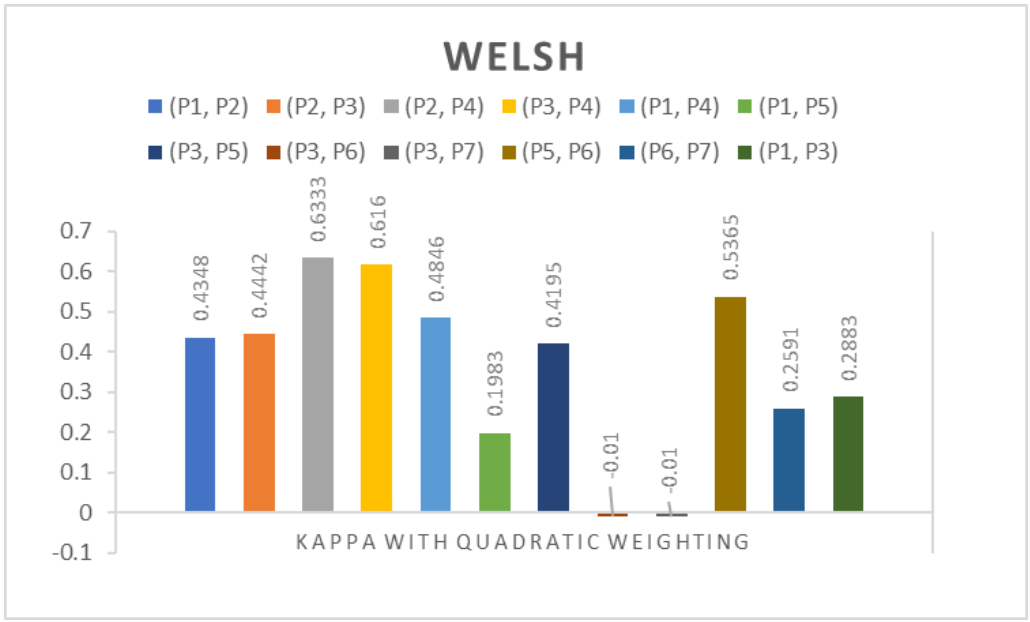


Figure 6.11 Kappa scores with quadratic weighting for 12 pairs of Welsh participants

The frequencies of agreement observed and the proportions of agreement observed are

compared against the frequencies and proportions that are expected by chance. The results of the comparisons for English and Welsh are plotted in figures 6.12, 6.13 and 6.14, 6.15 respectively. It was observed that both the frequencies and proportions of agreement between each pair of participants was more than those expected by chance.

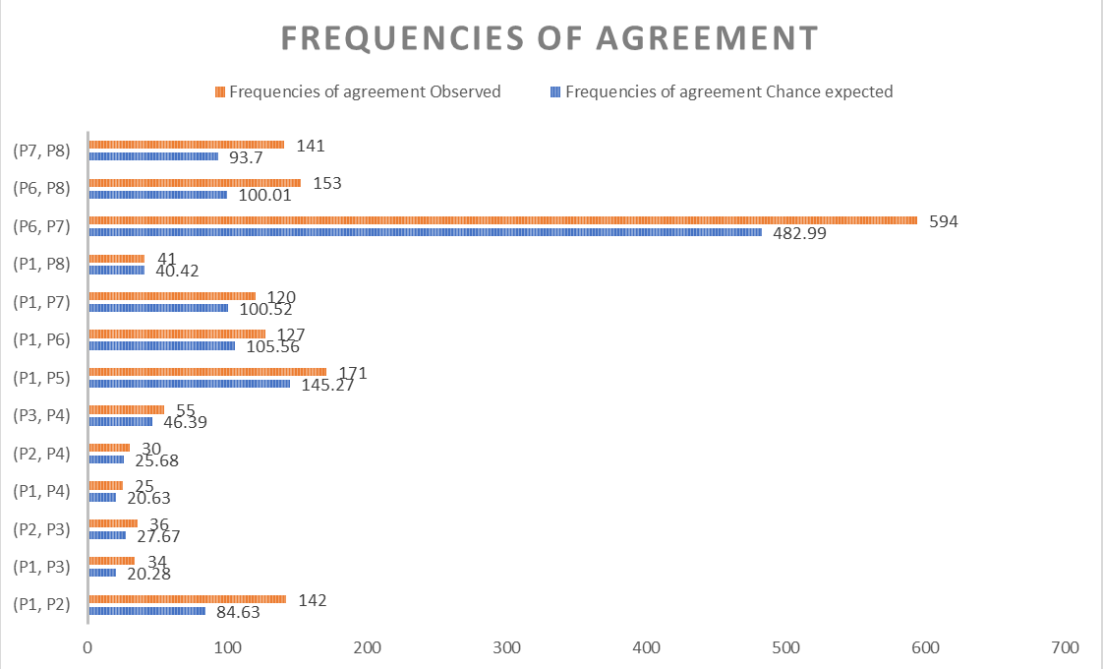


Figure 6.12 Frequencies of agreement compared against chance expected - English

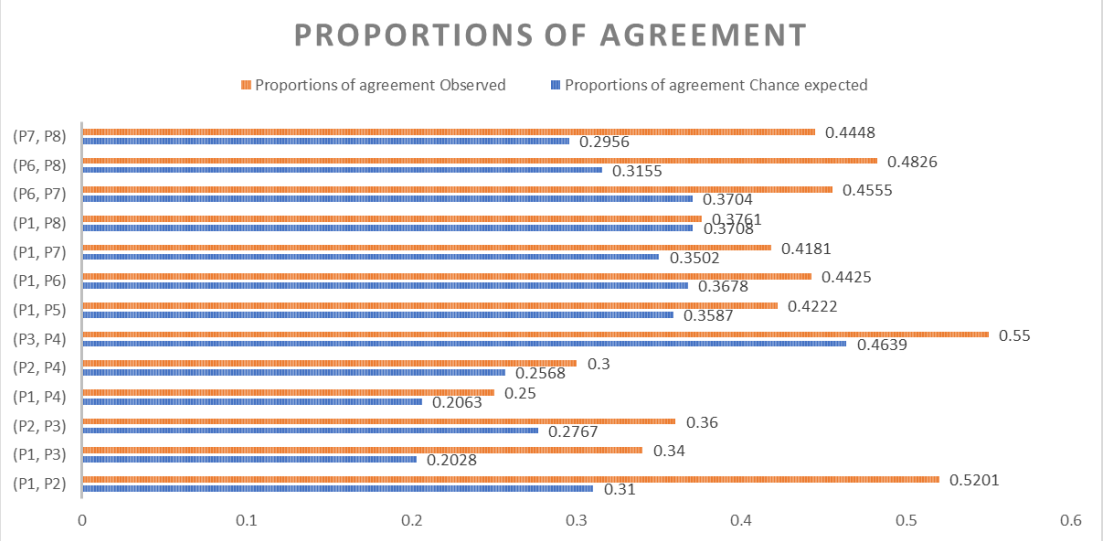


Figure 6.13 Proportions of agreement compared against chance expected - English

For Welsh, we observed that the participant pairs (P3, P6) and (P3, P7) for whom the kappa score could not be calculated, the frequencies and proportions of agreement were less than

those expected by chance.

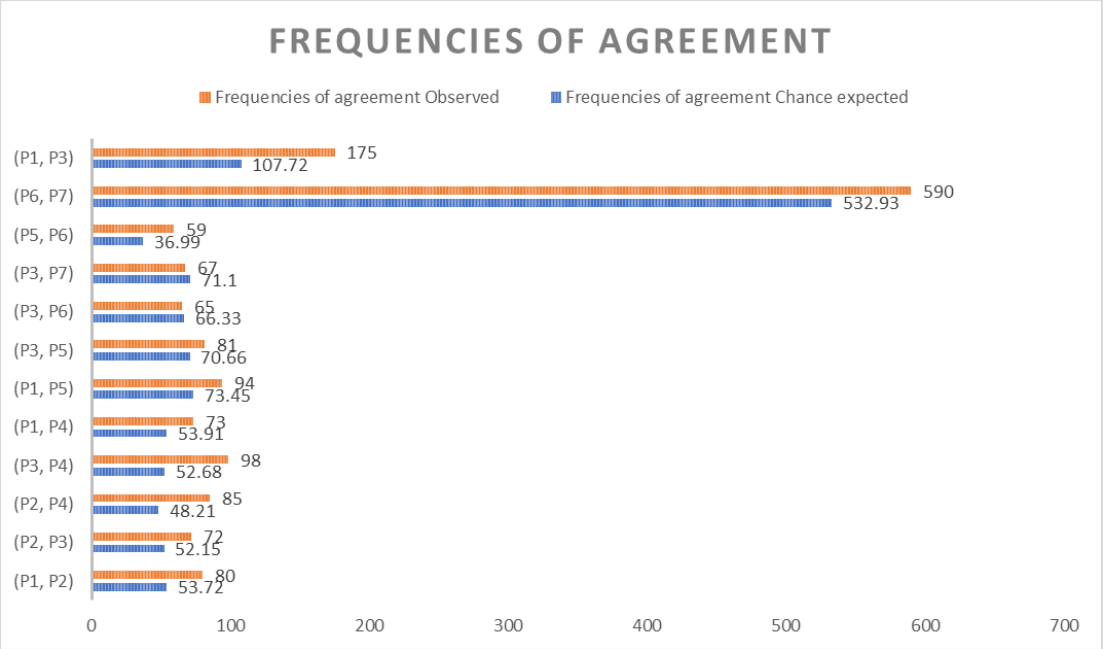


Figure 6.14 Frequencies of agreement compared against chance expected - Welsh

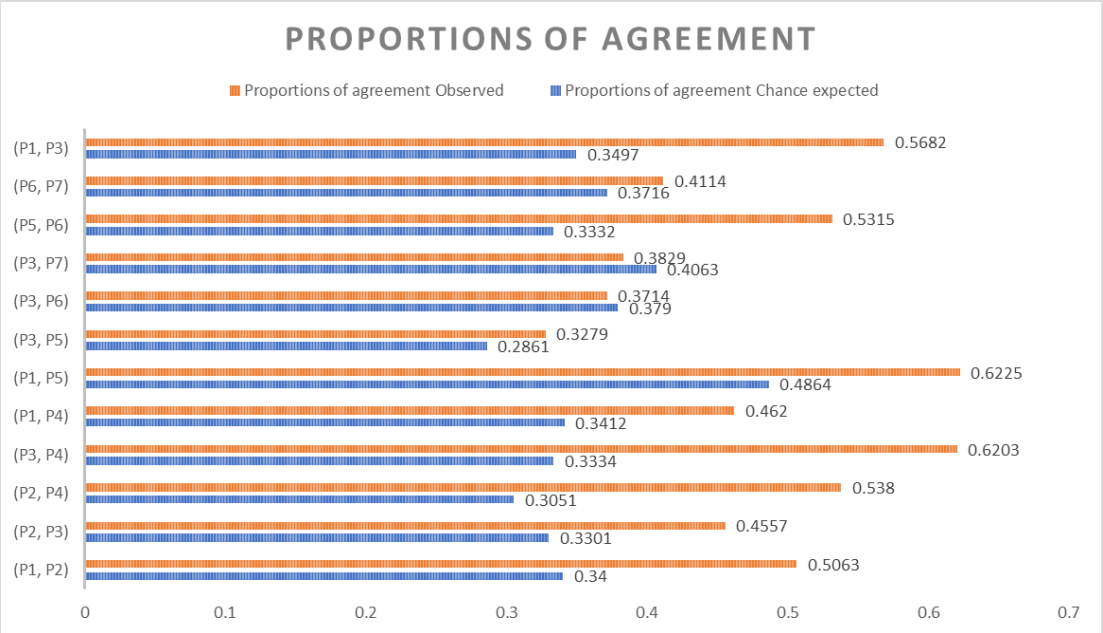


Figure 6.15 Proportions of agreement compared against chance expected - Welsh

In order to deal with varying chunk sizes and number of evaluators involved, we measured the agreement using Krippendorff's alpha score. The alpha score for 8 English participants involving 35 cases calculated for ordinal data was **0.78**. For 7 Welsh participants involving 35 cases calculated for ordinal data was **0.89**. We calculated Krippendorff's alpha score for

ordinal data because the Likert scale is ordinal. Krippendorff's alpha score ranges from -1 to 1. A score of 1 indicates perfect agreement, 0 indicates no agreement beyond chance and negative values indicate systematic disagreement between the participants. The alpha score between 0.67 to 0.8 indicates tentative reliability while anything greater than 0.8 indicates high reliability. The scores obtained for English and Welsh reveal that there is a tentative to high reliability in agreement between the participants for the respective languages.

6.7.2 Disagreement analysis

To gain further insight into the performance of the parser, we analysed the chunks and the labels with which the participants disagreed and recognised that the following factors were relevant:

- Overgeneralisation of construction schemas in the parser output
- Misunderstanding of the labels in the instructions
- Ambiguity in the level of analysis chosen to schematise the chunk

Chunks whose labels were overgeneralisations when a more specific label could have been chosen were marked by some participants as 'Disagree' or 'Strongly disagree'. For instance, pronouns and referring expressions involving predicate adjectives are best analysed as PRONOMINAL schema instead of a more general THING schema. This is one of the common mistakes that is found in the output chunk labels which some participants disagreed with. The conclusion here is that the given parse is still valid albeit suboptimal.

The second type of disagreement came with the differences in the participants' understanding of the proposed construction schemas. Since no explicit training was given to the participants on the schema labels, these disagreements provide insights on the interpretability of the labels. Based on informal definitions and a few example phrases in the instruction page, the participants developed their own understanding of the chunk labels and tried to evaluate the outputs. Some participants contacted us through email and wanted to clarify if it is okay to consistently disagree with the outputs based on one or two parts of the chunk label. For instance, one of the participants mentioned that they disagreed with all the chunks that were labelled by the parser as PROCESS_CLOSED_DEPENDENT. In their understanding, a DEPENDENT chunk by definition cannot be CLOSED. They also provided a short explanation of their understanding by giving some examples. The fact that the participants are able to give consistent, meaningful interpretations to the chunk labels is an

indication that the unified syntactic-semantic labels used by the parser are interpretable by humans. The participant feedback may therefore be used to update the knowledge base iteratively.

The third type of disagreement occurred with chunk labels that can have two valid levels of analysis at the same time. For example, a finite verb with its arguments can be analysed at the PROCESS level and EVENT level. Some participants disagreed with EVENT level interpretations before the entire clause was parsed. We also noticed that assigning EVENT level labels to the clausal connectives had more disagreements from the participants. Again, this provides important feedback that can be used to improve the related parser's components.

6.8 Limitations and future work

By unifying syntax and semantics, we proposed construction schema definitions using knowledge engineering and developed a parser that analyses sentences in terms of assembly of construction schemas. Four evaluation methodologies were adopted to measure the validity of the outputs produced by the parser in the absence of a gold standard. The evaluation section also demonstrated the reliability and interpretability of the labelled parsed chunks using statistical analysis of the ratings provided by fluent speakers on a Likert scale of 5.

To reuse the proposed ideas for a new language, creating the knowledge base and developing a schema definition component is very crucial. In order to create the definitions in the schema definition component, language experts were involved who provided their inputs on how constructions that occur in different formal syntactic environments are functionally analogous to each other. A wide range of grammatical constructions in the language were schematised with the same label based on how they are functionally analogous to each other. The involvement of experts in creating the knowledge base is one of the potential challenges in adapting the proposed idea for other languages. However, as we demonstrated with English and Welsh, the schema definitions can be created for one language and can be adapted to another language with relatively few changes. The effort involved in creating the knowledge base is also not as expensive as that of annotated treebanks.

Although we established the reliability of the parser output labels through a qualitative evaluation task, a more thorough evaluation is needed to establish the utility of the parser

outputs in other NLP tasks. The utility of the assembly of construction schemas produced by the parser would be best demonstrated in extrinsic NLP tasks such as question-answering, grammar correction, machine translation and so on. As this is a more comprehensive task, we are planning to conduct an extrinsic evaluation of our parser as a future work.

Chapter 7 Conclusion

In this work, we have developed a novel approach to parsing based on ideas from usage-based, functional, cognitive theories of grammar and implemented unsupervised, cognitive parsers for English and Welsh languages. The research was motivated by the necessity to develop a parser for Welsh language, which is a minoritised, low-resourced language. Because most of the world's languages do not have large treebanks annotated with syntactic information as is the case for Welsh language, it was necessary to develop a suitable unsupervised approach to parse raw sentences. In this context, we identified the following research question that is central to our work: Can an unsupervised learning approach be used to develop a syntactico-semantic parser? We hypothesised that the answer is affirmative and set four research objectives explicitly to verify the hypothesis. The research objectives that we started with and the results that were obtained on completing the research are as follows:

RO 1*: To utilise the existing body of knowledge and identify research gaps in the field of unsupervised parsing. Specifically, a systematic literature review is to be conducted to survey the existing approaches to unsupervised learning of parsing rules in terms of their theoretical underpinnings, practical implementations and evaluation.*

In order to meet this research objective, we conducted a systematic literature review on unsupervised approaches to parsing. From the literature review, we identified that although unsupervised parsing methodologies have been around for nearly two decades, they exhibited two problems: (a) Their performance was significantly lower than their supervised counterparts. (b) They treated parsing as the problem of modelling the probabilistic patterns of a language generated by simple, hidden statistical models of syntactic structure. The improvements in the performance of unsupervised parsers depended on how sophisticated their inference methods were in estimating the probabilities of the grammatical structures in a language. We also noticed that most of these approaches were influenced by the formal syntactic theoretical ideas, often representing their outputs in the form of dependency tree or constituency tree, and evaluated the parser performance by comparing against the gold standard annotation.

The literature review also revealed that there were significant gaps between the theoretical schools of grammar (such as functional, cognitive, usage-based theories of grammar) and experimental studies on human online parsing (reading, eye-tracking and other

psycholinguistic studies) on the one hand and the statistical and computational aspects of parsing on the other. We identified that this gap should be addressed to enable further advancements in computational approaches to parsing. This leads to the second research objective.

RO2: *To ground an unsupervised learning approach to developing a syntactico-semantic parser into linguistic theory in an effort to bridge the gap between the functional-cognitive schools of grammar and the computational models of language processing. Based on this investigation we aim to synthesise some of the essential properties that the syntactico-semantic parser should have.*

This objective was attained by synthesising ideas from Cognitive Grammar, Construction Grammar and Karaka theory and based on them we defined a set of construction schemas for English (and similarly for Welsh). These construction schemas were identified by noticing the morphosyntactic patterns in a given language and by recognising meaningful templates of semantic construals behind the morphosyntactic patterns. Linguistic experts were involved during knowledge engineering in order to inform the development of construction schema labels. Each schema was defined in terms of how it could be composed from other self-similar schemas. Using these schema definitions, we were able to learn the mapping between surface forms and functional usage patterns.

RO3: *To use the findings from RO1 and RO2 to design and implement a practical approach to unsupervised parsing that takes the advantages of prevalent parsing approaches in computational linguistics and NLP while introducing novelty based on the lessons learnt from the psycholinguistic properties of online sentence parsing by humans and the functional-cognitive theories.*

We recognised that only two parts were sufficient to develop a syntactico-semantic parser - a lexical component and a statistical component. The lexical component was implemented as dictionary cum rules. The dictionary part is realised in our system as the schema definition component while the rule part was realised as the schema assembly component that utilises these definitions to map portions of raw text to all possible construction schemas. The schema definition component was developed based on the knowledge base identified from knowledge engineering. In the schema assembly component, we proposed an algorithm that The statistical component was implemented as a feedforward neural network with backpropagation to predict the most likely assembly given two component schemas. Using

this methodology, we created a prototype parser that could parse raw sentences by assembling them into successively higher units of organisation incrementally.

RO4: *To evaluate the effectiveness of the approach proposed in RO3 and see how it generalises across structurally different languages. The two languages spoken in Wales, Welsh and English, fit this requirement. Given the unsupervised nature of the proposed approach, a novel evaluation framework will be designed to address the lack of ground truth.*

Evaluation posed a major problem in our approach as there was no single gold standard to compare against. There are many correct ways to parse a sentence depending on how the intermediate assemblies were composed. We conducted four evaluation experiments to validate the sequence of construction schema assemblies and their labels. First, in an attempt to quantitatively evaluate the parser outputs, we compared the intermediate assembly heads against the constituents in the constituency tree. We measured the accuracy of the parser as the number of assembly heads exactly matching with any level in the constituent tree except the leaf nodes. Realising the weakness of this approach, we chose manual evaluation of the coverage of the grammar as the second approach to evaluation. In this approach, we reported the coverage of the grammar by manually listing the types of constructions recognised by the parser and listed some of the errors made by the system. In the third approach, we measured how good or bad the parse is by calculating the ratio of the number of assembly edits needed in a parse to the total number of assemblies recognised by the parser.

In all these three approaches, we were not able to directly establish the validity of the parser output since there is no gold standard to compare against. When validity cannot be established in terms of parser accuracy, we proposed to evaluate the reliability of the parser outputs instead. By quantifying the consistency of the agreement between various human evaluators for the same outputs, we assessed the inter-rater reliability and reported them using weighted Cohen's kappa and Krippendorff's alpha scores. The best way to evaluate the parser is to use it extrinsically in another application and see if it is really useful or not. This was outside the scope of this particular project but it provided direction for future research in this area.

The novel research contribution of this work is that it provides a framework for parsing the skeletal structure of a sentence by recognising parts of sentences as instances of construction schemas. The proposed parser uses an unsupervised approach that can be adapted for parsing other languages, including minority languages that have very few resources, because

the implementation does not require syntactic treebanks for training. The path of assembling smaller schemas into larger schemas produced better results when additional semantic information from wordnet was used.

More specific contributions of this project in terms of outputs produced are fivefold. The first contribution is that a systematic literature review was conducted and research gaps between existing computational approaches to unsupervised parsing and the theoretical as well as experimental studies on parsing were identified. Secondly, construction schemas were proposed through knowledge engineering by involving language experts and a schema definition component was developed. The third contribution is the synthesis of ideas from Cognitive Grammar, Computational Paninian Grammar and psycholinguistic studies on online sentence processing in humans to identify the parser specifications. The fourth contribution is the implementation of a parsing model that bridges the research gaps identified from literature review and incorporates the parser specifications identified in the previous stage. Finally, we proposed suitable evaluation methodologies in the absence of a gold standard and presented the results obtained.

In this research, we have taken the first step towards exploring an unsupervised approach for parser development by unifying syntax and semantics that is incremental, bottom-up and usage-based. We hope that more research will follow in the direction of overcoming the limitations of this work and expanding on it.

Appendix A

S. No.	Title	Findings	Reference
1.	Corpus-Based Induction of Syntactic Structure: Models of Dependency and Constituency	**Theory:** Formal grammar **Representing productivity:** Hierarchical **Processing productivity:** Non-incremental **Output:** Dependency tree **Evaluation strategy:** Comparison against gold standard **Features:** Tokens**,** Head, valency, direction of attachment, word distribution clusters **Implementation methodology:** Top-down dependency grammar model	Klein and Manning (2004)
2.	Unsupervised multilingual grammar induction	**Theory:** Formal grammar **Representing productivity:** Hierarchical **Processing productivity:** Non-incremental **Output:** Constituency tree **Evaluation strategy:** Comparison against gold standard **Features:** POS tags, word alignments **Implementation methodology:** Similarity or exemplar-based models	Snyder et al. (2009)
3.	Unsupervised context sensitive language acquisition from a large corpus	**Theory:** Theory-neutral **Representing productivity:** Hierarchical **Processing productivity:** Non-incremental **Output:** Directed multigraph **Evaluation strategy:** Grammaticality judgement task **Features:** Tokens **Implementation methodology:** Automatic Distillation of Structures (ADIOS)	Solan et al. (2004)
4.	An All-Subtrees Approach to Unsupervised Parsing	**Theory:** Theory-neutral **Representing productivity:** Hierarchical **Processing productivity:** Non-incremental **Output:** Constituency tree **Evaluation strategy:** Comparison against gold standard **Features:** Tokens, POS tags **Implementation methodology:** Data Oriented Parsing (DOP)	Bod (2006)
5.	From Baby Steps to Leapfrog: How "Less is More" in Unsupervised Dependency Parsing	**Theory:** Formal grammar **Representing productivity:** Hierarchical **Processing productivity:** Non-incremental **Output:** Dependency tree **Evaluation strategy:** Comparison against gold standard **Features:** Tokens, POS tags, Head, Direction of attachment, Valence **Implementation methodology:** Top-down dependency grammar model	Spitkovsky et al. (2010)
6.	Unsupervised Grammar Induction by Distribution and Attachment	**Theory:** Formal grammar **Representing productivity:** Hierarchical **Processing productivity:** Non-incremental **Output:** Constituency tree **Evaluation strategy:** Comparison against gold standard **Features:** Heuristics, distributed representation of words **Implementation methodology:** Distribution-based model	Brooks (2006)
7.	What are the Productive Units of Natural Language Grammar? A DOP Approach	**Theory:** Theory-neutral **Representing productivity:** Hierarchical **Processing productivity:** Non-incremental	Zuidema (2006)

	to the Automatic Identification of Constructions	**Output:** Constituency tree **Evaluation strategy:** Comparison against gold standard **Features:** Tokens, POS tags **Implementation methodology:** Data Oriented Parsing (DOP)	
8.	Simple Unsupervised Grammar Induction from Raw Text with Cascaded Finite State Models	**Theory:** Formal grammar **Representing productivity:** Hierarchical **Processing productivity:** Non-incremental **Output:** Constituency tree **Evaluation strategy:** Comparison against gold standard **Features:** Tokens **Implementation methodology:** Chunker and its extension	Ponvert et al. (2011)
9.	Characterizing Motherese: On the Computational Structure of Child-Directed Language	**Theory:** Theory-neutral **Representing productivity:** Hierarchical **Processing productivity:** Non-incremental **Output:** Directed multigraph **Evaluation strategy:** Comparison of models trained on disjoint corpora with respect to sentence acceptability. **Features:** Tokens **Implementation methodology:** Automatic Distillation of Structures (ADIOS)	Brodsky and Waterfall (2007)
10.	Unsupervised induction of labeled parse trees by clustering with syntactic features	**Theory:** Formal grammar **Representing productivity:** Hierarchical **Processing productivity:** Non-incremental **Output:** Constituency tree **Evaluation strategy:** Comparison against gold standard **Features:** POS, chunks **Implementation methodology:** Chunk-based approach, Clustering approach	Reichart and Rappoport (2008)
11.	A survey of grammatical inference methods for natural language learning	**Findings:** 14 studies, six computational techniques(Statistical methods, Evolutionary computing techniques, Minimum description length, Heuristic methods, Greedy search, Clustering techniques), two presentation sets (text and informant), three types of information for learning (supervised, unsupervised, semi-supervised), three types of evaluation methods (Looks-good-to-me, compare against treebank, rebuilding known grammars).	D'Ulizia et al. (2011)
12.	A probabilistic generative model for an intermediate constituency-dependency representation	**Theory:** Formal grammar **Representing productivity:** Hierarchical **Processing productivity:** Non-Incremental **Output:** Dependency tree **Evaluation strategy:** Comparison against gold standard **Features:** Tokens, POS tags, Head, direction of attachment, valence **Implementation methodology:** Top-down dependency grammar model	Sangati (2010)
13.	Identifying Patterns for Unsupervised Grammar Induction	**Theory:** Formal grammar **Representing productivity:** Hierarchical **Processing productivity:** Non-Incremental **Output:** Constituency tree **Evaluation strategy:** Comparison against gold standard **Features:** Tokens, POS tags, Heuristics **Implementation methodology:** Heuristics	Santamaria and Araujo (2010)
14.	From ranked words to dependency trees: two-stage unsupervised non-projective dependency parsing	**Theory:** Formal grammar **Representing productivity:** Hierarchical **Processing productivity:** Non-Incremental **Output:** Dependency tree	Søgaard (2011)

		Evaluation strategy: Comparison against gold standard **Features:** Tokens, Distributed representation of words **Implementation methodology:** Distribution-based model	
15.	Posterior Sparsity in Unsupervised Dependency Parsing	**Theory:** Formal grammar **Representing productivity:** Hierarchical **Processing productivity:** Non-Incremental **Output:** Dependency tree **Evaluation strategy:** Comparison against gold standard **Features:** Tokens, POS tags, Head, direction of attachment, valence **Implementation methodology:** Top-down dependency grammar model	Gillenwater et al. (2011)
16.	Capitalization Cues Improve Dependency Grammar Induction	**Theory:** Formal grammar **Representing productivity:** Hierarchical **Processing productivity:** Non-Incremental **Output:** Dependency tree **Evaluation strategy:** Comparison against gold standard **Features:** Tokens, Orthographic cues, head, valence, direction of attachment **Implementation methodology:** Top-down dependency grammar model	Spitkovsky et al. (2012)
17.	Exploiting Reducibility in Unsupervised Dependency Parsing	**Theory:** Formal grammar **Representing productivity:** Hierarchical **Processing productivity:** Non-Incremental **Output:** Dependency tree **Evaluation strategy:** Comparison against gold standard **Features:** Tokens, POS tags, head, valence, direction of attachment **Implementation methodology:** Top-down dependency grammar model	Mareček and Žabokrtsk (2012b)
18.	Unsupervised Dependency Parsing using Reducibility and Fertility features	**Theory:** Formal grammar **Representing productivity:** Hierarchical **Processing productivity:** Non-Incremental **Output:** Dependency tree **Evaluation strategy:** Comparison against gold standard **Features:** Tokens, head, valence, direction of attachment **Implementation methodology:** Top-down dependency grammar model	Mareček and Žabokrtský (2012a)
19.	Learnability and falsifiability of Construction Grammars	**Theory:** Usage-based grammar **Representing productivity:** Non-hierarchical **Processing productivity:** Non-Incremental **Output:** Construction slots **Evaluation strategy:** Maximum-coverage, minimum-size, stability measures for a usage-based grammar. **Features:** Tokens, POS tags, Construction association measures **Implementation methodology:** Construction grammar induction	Dunn (2017a)
20.	Boosting Unsupervised Grammar Induction by Splitting Complex Sentences on Function Words	**Theory:** Theory-neutral **Representing productivity:** Hierarchical **Processing productivity:** Non-Incremental **Output:** Directed multigraph **Evaluation strategy:** Comparison against gold standard, Output is compared against an artificial grammar that generated the test corpus. **Features:** Tokens **Implementation methodology:** Automatic Distillation of Structures (ADIOS)	Berant et al. (2007)

21.	Evaluation Strategies for Computational Construction Grammars	**Theory:** Usage-based **Representing productivity:** Non-hierarchical **Processing productivity:** Non-Incremental **Output:** Transient structure **Evaluation strategy:** Comparison against gold standard. **Features:** POS, Heuristics **Implementation methodology:** Construction grammar induction	Marques and Beuls (2016)
22.	Learning Syntactic Constructions from Raw Corpora	**Theory:** Theory-neutral **Representing productivity:** Hierarchical **Processing productivity:** Non-Incremental **Output:** Directed multigraph **Evaluation strategy:** Comparison against gold standard, Output is compared against an artificial grammar that generated the test corpus. **Features:** Tokens **Implementation methodology:** Automatic Distillation of Structures (ADIOS)	Edelman et al. (2005)
23.	Punctuation: Making a Point in Unsupervised Dependency Parsing	**Theory:** Formal grammar **Representing productivity:** Hierarchical **Processing productivity:** Non-Incremental **Output:** Dependency tree **Evaluation strategy:** Comparison against gold standard. **Features:** Tokens, orthographic cues **Implementation methodology:** Top-down dependency grammar model	Spitkovsky et al. (2011)
24.	An Exemplar-based Approach to Unsupervised Parsing	**Theory:** Theory-neutral **Representing productivity:** Hierarchical **Processing productivity:** Non-Incremental **Output:** Constituency tree **Evaluation strategy:** Comparison against gold standard. **Features:** Tokens, POS tags **Implementation methodology:** Similarity or exemplar-based model	Dennis (2005)
25.	Bayesian Tree Substitution Grammars as a Usage-based approach	**Theory:** Theory-neutral **Representing productivity:** Hierarchical **Processing productivity:** Non-Incremental **Output:** Constituency tree **Evaluation strategy:** Comparison against gold standard. **Features:** Tokens, POS tags **Implementation methodology:** Data Oriented Parsing	Post and Gildea (2013)
26.	Computational learning of construction grammars	**Theory:** Usage-based **Representing productivity:** Non-hierarchical **Processing productivity:** Non-Incremental **Output:** Construction slots **Evaluation strategy:** Qualitative, Maximum coverage and minimum size of grammar **Features:** Tokens, POS tags, semantic tags, construction association measures **Implementation methodology:** Construction grammar learning	Dunn (2017b)
27.	Unsupervised Grammar Induction with Depth-bounded PCFG	**Theory:** Formal grammar or generative grammar **Representing productivity:** Hierarchical **Processing productivity:** Non-Incremental **Output:** Constituency tree **Evaluation strategy:** Comparison against gold standard. **Features:** Tokens **Implementation methodology:** Probabilistic Context Free Grammar (PCFG)	Jin et al. (2018)

28.	From Exemplar to Grammar: A Probabilistic Analogy-Based Model of Language Learning	**Theory:** Theory-neutral **Representing productivity:** Hierarchical **Processing productivity:** Non-Incremental **Output:** Constituency tree **Evaluation strategy:** Comparison against gold standard. **Features:** Tokens, POS tags **Implementation methodology:** Data Oriented Parsing (DOP)	Bod (2009)
29.	Grammar Induction from Text Using Small Syntactic Prototypes	**Theory:** Formal or generative grammar **Representing productivity:** Hierarchical **Processing productivity:** Non-Incremental **Output:** Dependency tree **Evaluation strategy:** Comparison against gold standard. **Features:** Tokens, POS tags, head, valence, direction of attachment **Implementation methodology:** Top-down dependency grammar model	Boonkwan and Steedman (2011)
30.	Improving Unsupervised Dependency Parsing with Richer Contexts and Smoothing	**Theory:** Formal or generative grammar **Representing productivity:** Hierarchical **Processing productivity:** Non-Incremental **Output:** Dependency tree **Evaluation strategy:** Comparison against gold standard. **Features:** Tokens, POS tags, head, valence, direction of attachment **Implementation methodology:** Top-down dependency grammar model	Headden et al. (2009)
31.	Limitations of Current Grammar Induction Algorithms	**Findings:** Survey paper, Different grammar induction algorithms such as EMILE, ADIOS, ABL are discussed. Tested on Eindhoven corpus. Two experiments were conducted on various Grammar induction (GI) approaches. The results from the experiments show that the current grammar induction systems like EMILE, ADIOS, ABL have severe shortcomings in deriving meaningful structure from language as complicated as Eindhoven corpus. An incremental grammar induction strategy is suggested as preferable and a short illustration of how the system should be is presented at the end. ,	Cramer (2007)
32.	Towards High Speed Grammar Induction on Large Text Corpora	**Theory:** Formal or generative grammar **Representing productivity:** Hierarchical **Processing productivity:** Non-Incremental **Output:** Constituency tree **Evaluation strategy:** Comparison against gold standard. **Features:** Distributional representation of words **Implementation methodology:** Chunker and it extension, Probabilistic context-free grammar (PCFG)	Adriaans et al. (2000)
33.	Unbounded Dependency Recovery for Parser Evaluation	**Findings:** Parser evaluation paper, five different parsers compared. The five parsers are: C&C, Enju, DCU, Rasp and Stanford. The suitability of PARSEVAL metrics as a measure of the performance of parsers is called into question in the paper. By pointing out that the parser performance on recovering unbounded dependencies are bad, the study motivates the necessity of better parser evaluation metrics.	Rimell et al. (2009)
34.	Evolving Natural Language Grammars without Supervision	**Theory:** Formal or generative grammar **Representing productivity:** Hierarchical **Processing productivity:** Incremental **Output:** Constituency tree	Araujo and Santamaría (2010)

		Evaluation strategy: Comparison against gold standard. **Features:** POS, Heuristic rules **Implementation methodology:** Heuristic approach	
35.	Unsupervised Induction of Dependency Structures Using Probabilistic Bilexical Grammars	**Theory:** Formal or generative grammar **Representing productivity:** Hierarchical **Processing productivity:** Non-incremental **Output:** Dependency tree **Evaluation strategy:** Comparison against gold standard. **Features:** Tokens, POS, head, valence, direction of attachment, **Implementation methodology:** Top-down dependency grammar model	Dominguez and Infante-Lopez (2011)
36.	Fast Unsupervised Incremental Parsing	**Theory:** Formal or generative grammar **Representing productivity:** Hierarchical **Processing productivity:** Incremental **Output:** Shortest common cover link sets **Evaluation strategy:** Comparison against gold standard. **Features:** Words, Distribution **Implementation methodology:** Distributed-based	Seginer (2007)
37.	Use of predictive dependencies in language learning	**Goal:** To verify experimentally if abstract grammatical hierarchies can be learnt using the statistical cues of local predictive dependencies **Method:** Artificial language learning task by humans. **Experiment:** Adult participants exposed to sentences from artificial language. They had no semantic,visual or any other clue except the distribution of words and their category. **Conditions:** Groups: Intentional group, Incidental group, Control group **Tests:** Rule test, fragment test **Result:** Experimental groups (intentional and incidental) significantly outperformed control group on both the tests under various conditions **Insights:** Predictive dependencies on the naturally occurring text can lead to simple phrase structure acquisition. We can exploit this for unsupervised learning. **Summary:** The results support the hypothesis that learners can detect predictive dependencies in the service of acquiring simple phrase structure.	Saffran (2001)
38.	The role of discourse context in the processing of a flexible word-order language	**Goal:** To find out if the processing difficulty associated with non-canonical word order in a language is reduced in an appropriate discourse context. **Method:** Conducting experiments on speakers of Finnish by presenting them with non-canonical word order sentences to see the effect. **Experiment:** Two experiments – 1. Self paced reading task 2. Eye tracking while hearing the spoken description of scenes **Conditions:** 40 Finnish speakers, given 20 sentences in four patterns. Eye-tracking during listening **Result:** Reading time in first experiment and eye movements in second experiment support the idea that people use word order patterns to predict upcoming referents on the basis of discourse status. **Insights:** Discourse is not taken into consideration while modelling computational grammar induction. The study provides the theoretical motivation for such an approach. **Summary:** The processing of non-canonical structures is facilitated by the presence of an appropriate discourse context.	Kaiser and Trueswell (2004)

39.	A Linguistic Investigation into Unsupervised DOP	**Goal:** Study if a computational bootstrapping model of language can explain the abstract linguistic properties of natural languages without assuming a universal grammar **Method:** An unsupervised DOP model which computes the most probable tree from among the shortest derivations of sentences. Use this model to explain both rule-based and exemplar-based properties of a natural language. **Result:** Learning discontiguous construction patterns, agreement, movement were all shown to be possible for computational bootstrapping without a need for special, top down abstract grammar **Summary:** Recursive tree structure and an analogical matching algorithm can help us learn various syntactic phenomena that usually appeal to an abstract, universal grammar that generates sentences top down.	Bod (2007)
40.	A Statistical Test for Grammar	**Goal:** To see if the children acquire language through a productive grammatical system (typically expressed as generative grammar) or usage based schematic patterns. **Method:** A statistical test is proposed to check if grammar is abstract and productive or lexically-specific and usage-based. **Experiment:** Through case studies on the children's speech data, a measure of overlap between theoretical prediction (productive grammar vs lexical-specific usage based) is measured. **Conditions:** Productivity, Usage-based **Result:** Results of the statistical test show that a lexically-specific, usage-based, memory-and-retrieval approach is unsupported. The results are consistent with a productive abstract grammatical system in child speech. **Summary:** These results do not resolve the innateness debate in language acquisition: they only point to the very early availability of an abstract and productive grammar.	Yang (2011)
41.	How hierarchical is language use?	**Goal:** It is assumed that hierarchical phrase structure rules play an important role in natural language processing? Is it true? Can sequential model predict statistical regularities in language? **Method:** Comparison of emerging ideas in various neurophysiology, behaviour and computational studies suggest that sequential sentential structure has considerable explanatory power. **Task:** Discussion of evidences from multiple research fields: Cognitive neuroscience, psycholinguistics, Computational models of language acquisition **Insights:** Combining productive grammatical units need not happen hierarchically but a sequential process would suffice. **Summary:** Such a model of language processing has implications for the fields of linguistics, ethology, psychology and computer science / natural language processing.	Frank et al. (2012)
42.	Rich Syntax from a Raw Corpus: Unsupervised Does It	**Goal:** Compare the theoretical and computational properties of ADIOS system to some recent works in computational linguistics and in grammar theory **Method:** Discussing the computational principles behind the ADIOS model, by comparing it to select approaches from computational and formal linguistics. **Results:** Principles behind ADIOS: (a) Pattern significance is learnt probabilistically (b) Patterns are context sensitive (c) Patterns are hierarchical **Insights:** Comparison of ADIOS with grammar and computational theories is done. Similar to Construction grammar	Edelman et al. (2003)

		in its general philosophy of linguistic representations and Tree adjoining grammar in its computational capacity **Summary:** Evaluation strategy for a purely empirical , usage-based approach to grammar induction should be found.	
43.	Subjacency Constraints without Universal Grammar: Evidence from Artificial Language Learning and Connectionist Modeling	**Goal:** To experimentally verify if subjacency constraints which usually appeal to universal grammar can be acquired through limitations on sequential learning. **Method:** Two types of experiments were conducted to enquire about the nature of subjacency constraints. Artificial language learning experiments and connectionist simulations using the same data were also made. **Experiment:** Two experiments – 1. Created two artificial languages, natural (NAT) and unnatural (UNNAT). Each artificial language consisted of a set of letter strings, each letter representing a specific grammatical class. Subjects were randomly assigned to one of three conditions (NAT, UNNAT, and CONTROL). NAT and UNNAT were trained using the natural and unnatural languages, respectively. The CONTROL group completed only the test session. During training, individual letter strings were presented briefly on a computer. After each presentation, participants were prompted to enter the letter string using the keyboard. Training consisted of 2 blocks of the 30 items, presented randomly. During the test session, participants decided if the test items were created by the same (grammatical) or different (ungrammatical) rules as the training items. 2. Simple Recurrent Networks (SRNs) were used to show the connectionist simulations of the same human data discussed earlier in Experiment 1. **Conditions:** 1. Sixty undergraduates were recruited from an introductory psychology class at Southern Illinois University for the Experiment 1 2. Standard feed-forward neural networks equipped with an extra layer of context units. **Result:** An overall t-test indicated that NAT (59%) learned the language significantly better than UNNAT (54%). This result indicates that the UNNAT was more difficult to learn than the NAT. **Insights:** The results therefore corroborate the hypothesis that constraints on the learning and processing of sequential structure can explain why subjacency violations tend to be avoided: they were weeded out because they made the sequential structure of language too difficult to learn. **Summary:** The results suggest that constraints arising from general cognitive processes, such as sequential learning and processing, are likely to play a larger role in sentence processing than has traditionally been assumed. This means that what we observe today as linguistic universals may be stable states that have emerged through an extended process of linguistic evolution.	Ellefson and Christiansen (2000)

Appendix B

Dynet parameter initialisation

```
model = dy.Model()
trainer = dy.SimpleSGDTrainer(model, learning_rate=0.1)
W_emb = model.add_lookup_parameters((nchunks, EMB_SIZE))        # Chunk embeddings
Cons_sch_emb = model.add_lookup_parameters((nchunks, EMB_SIZE)) # Construction schemas embeddings
Cons_sch_h_p = model.add_parameters((HID_SIZE, N*EMB_SIZE))     # Hidden layer weights
b_h_p = model.add_parameters((HID_SIZE))                        # Hidden layer bias
Cons_sch_sm_p = model.add_parameters((nchunks, HID_SIZE))       # Softmax weights
b_sm_p = model.add_parameters((nchunks))                        # Softmax bias
```

Calculate score vector for the chunk

```
def calc_score_of_chunk(A, B):
    print("Len of inputs ",len(A),len(B))
    emb1 = dy.esum([W_emb[w2i[a]] for a in A])
    emb2 = dy.esum([Cons_sch_emb[cons2i[b]] for b in B])
    inputs = dy.concatenate([emb1,emb2])
    Cons_sch_h = dy.parameter(Cons_sch_h_p)
    b_h = dy.parameter(b_h_p)
    h = dy.tanh(dy.affine_transform([b_h, Cons_sch_h, inputs]))
    Cons_sch_sm = dy.parameter(Cons_sch_sm_p)
    b_sm = dy.parameter(b_sm_p)
    return dy.softmax(dy.affine_transform([b_sm, Cons_sch_sm, h]))


def update_training(words,chunk_parts,head):
    print('To Train: ', words, chunk_parts, head)
    output = calc_score_of_chunk(words,chunk_parts)
    print("Length of Score: ",len(output.value()))
    my_loss = dy.pickneglogsoftmax(output, cons2i[head])
    my_loss.backward()
    trainer.update()
    return
```

Appendix C

PARTICIPANT CONSENT FORM

Title of research project: **Automatic Grammar Induction from Free Text Using Insights from Cognitive Grammar – Survey for rating the output of a parser**

SREC reference and committee: **COMSC/Ethics/2022/046**

Name of Chief/Principal Investigator: **Vigneshwaran M, PhD Research Scholar**

	Please initial box
I confirm that I have read the information sheet dated 07/05/2022 version 2.0 for the above research project.	
I confirm that I have understood the information sheet dated 07/05/2022 version 2.0 for the above research project and that I have had the opportunity to ask questions and that these have been answered satisfactorily.	
I understand that my participation is voluntary and I am free to withdraw at any time without giving a reason and without any adverse consequences (e.g.,, to medical care or legal rights, if relevant). I understand that if I withdraw, information about me that has already been obtained may be kept by Cardiff University.	

I understand that data collected during the research project may be looked at by individuals from Cardiff University or from regulatory authorities, where it is relevant to my taking part in the research project. I give permission for these individuals to have access to my data.	
I consent to the processing of my personal information (Name and Email ID) for the purposes explained to me. I understand that such information will be held in accordance with all applicable data protection legislation and in strict confidence, unless disclosure is required by law or professional obligation.	
I understand who will have access to personal information provided, how the data will be stored and what will happen to the data at the end of the research project.	
I understand that after the research project, anonymised data may be made publicly available via a data repository and may be used for purposes not related to this research project. I understand that it will not be possible to identify me from this data that is seen and used by other researchers, for ethically approved research projects, on the understanding that confidentiality will be maintained.	
I understand how the findings and results of the research project will be written up and published.	
I agree to take part in this research project.	

_____ _____ _____

_____

Name of participant (print) Date Signature

_____ _____ _____

_____

Name of person taking consent Date Signature

(print)

_____

Role of person taking consent

(print)

THANK YOU FOR PARTICIPATING IN OUR RESEARCH

YOU WILL BE GIVEN A COPY OF THIS CONSENT FORM TO KEEP

Recruitment Poster

School of Computer Science and Informatics Cardiff University

PARTICIPANTS NEEDED FOR

RATING THE OUTPUTS OF A PARSER

We are looking for fluent Welsh speakers to take part in a survey being conducted as a part of a PhD research project.

If you volunteer to be in this study, your participation will consist of ***rating the outputs of a parser.*** The survey will contain Welsh sentences parsed into labelled chunks. Your task will be to rate the correctness / meaningfulness of the chunks and their labels. An explanation of the relevant concepts will be provided in the instructions page of the survey

Your participation would involve **filling the survey** which will take ***not more than 45 minutes*** of your time.

In appreciation of your time, the first 20 participants will get a £15 Love2Shop voucher each upon completion of the survey.

For more information about this study, please contact:

Vigneshwaran Muralidaran (PhD Scholar)

E-mail: ***MuralidaranV@cardiff.ac.uk***

This study has been reviewed by, and received ethics clearance

through School Research Ethics Committee (SREC), School of Computer Science and Informatics, Cardiff University

Ethical Approval

Research project title: Automatic Grammar Induction from Free Text Using Insights from Cognitive Grammar – Survey for rating the output of a parser

SREC reference: COMSC/Ethics/2022/046

The SCHOOL OF COMPUTER SCIENCE & INFORMATICS RESEARCH ETHICS COMMITTEE ('Committee') reviewed the above application at the meeting held electronically on 11/5/2022.

Ethical Opinion: FAVOURABLE

The Committee gave a favourable ethical opinion of the above application on the basis described in the application form, protocol and supporting documentation.

Additional approvals:

This letter provides an ethical opinion only. You must not start your research project until all appropriate approvals are in place. This is not a matter for the committee to advise on. For student projects, you should contact your supervisor. For staff projects/supervisors, contact the School Research Office comsc-research@cardiff.ac.uk.

Amendments:

Any substantial amendments to documents previously reviewed by the Committee must be submitted to the Committee via comsc-ethics@cardiff.ac.uk for consideration and cannot be implemented until the Committee has confirmed it is satisfied with the proposed amendments.

You are permitted to implement non-substantial amendments to the documents previously reviewed by the Committee but you must provide a copy of any updated documents to the Committee via comsc-ethics@cardiff.ac.uk for its records.

Monitoring requirements:

The Committee must be informed of any unexpected ethical issues or unexpected adverse events that arise during the research project. This notification should be made via email to us. The Committee must be informed when your research project has ended. This notification

should be made to comsc-ethics@cardiff.ac.uk within 3 months of research project completion.

Complaints/Appeals:

If you are dissatisfied with the decision made by the Committee, please contact the school's Ethics Officer, Dr Katarzyna Stawarz in the first instance to discuss your complaint. If this discussion does not resolve the issue, you are entitled to refer the matter to the Head of School for further consideration. The Head of School may refer the matter to the Open Research Integrity and Ethics Committee (ORIEC), where this is appropriate. Please be advised that ORIEC will not normally interfere with a decision of the Committee and is concerned only with the general principles of natural justice, reasonableness and fairness of the decision.

Please use the Committee reference number (SREC reference) on all future correspondence.

The Committee reminds you that it is your responsibility to conduct your research project to the highest ethical standards and to keep all ethical issues arising from your research project under regular review.

You are expected to comply with Cardiff University's policies, procedures and guidance at all times, including, but not limited to, its Policy on the Ethical Conduct of Research involving Human Participants, Human Material or Human Data and our Research Integrity and Governance Code of Practice.

Bibliography

Abney, S. and Bird, S., 2010. *The human language project: building a Universal Corpus of the world's languages.*

Adriaans P., Trautwein M. and Vervoort M. (2000). *Towards high speed grammar induction on large text corpora.* In International Conference on Current Trends in Theory and Practice of Computer Science. Berlin, Heidelberg: Springer, pp. 173–186.

Amaglobeli, G., 2012. *Semantic triangle and linguistic sign.* Journal in Humanities, pp.37-40.

Aoshima, S., Phillips, C. and Weinberg, A., 2004. *Processing filler-gap dependencies in a head-final language.* Journal of memory and language, 51(1), pp.23-54.

Araujo L. and Santamaría J. (2010). *Evolving natural language grammars without supervision. In Evolutionary Computation (CEC)*, 2010 IEEE Congress on (pp. 1–8). IEEE.

Austin, J.L., 1975. *How to do things with words* (Vol. 88). Oxford university press.

Bates E. and McWhinney B. (1982). *Functionalist approaches to grammar.*

Baumann, P. and Pierrehumbert, J.B., 2014, May. *Using Resource-Rich Languages to Improve Morphological Analysis of Under-Resourced Languages*. In LREC (pp. 3355-3359).

Berant J., Gross Y., Mussel M., Sandbank B., Ruppin E. and Edelman S. (2007). *Boosting unsupervised grammar induction by splitting complex sentences on function words. In Proceedings of the Boston University Conference on Language Development.*

Bharati, A. and Sangal, R., 1993, June. *Parsing free word order languages in the Paninian framework.* In 31st Annual Meeting of the Association for Computational Linguistics (pp. 105-111).

Bharati, A., Bhatia, M., Chaitanya, V. and Sangal, R., 1996. *Paninian grammar framework applied to english. Department of Computer Science and Engineering, Indian Institute of Technology, Kanpur.*

Bhatta, V.P., 1988. *THEORY OF KARAKA*. Bulletin of the Deccan College Research Institute, 47, pp.15-22.

Bird, S., 2006, July. *NLTK: the natural language toolkit.* In Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions (pp. 69-72).

Bloom L., Hood L. and Lightbown P. (1974). *Imitation in language development: if, when, and why.* Cognitive Psychology 6, 380–420.

Bloomfield L. (1962). *Language*. 1933. Holt, New York.

Bod R. (2006). *An all-subtrees approach to unsupervised parsing.* In Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics. Association for Computational Linguistics, pp. 865–872.

Bod R. (2007). *A linguistic investigation into unsupervised DOP.* In Proceedings of the Workshop on Cognitive Aspects of Computational Language Acquisition. Association for Computational Linguistics, pp. 1–8.

Bod R. (2009). *From exemplar to grammar: a probabilistic analogy-based model of language learning*. Cognitive Science 33, 752–793.

Boonkwan P. and Steedman M. (2011). *Grammar induction from text using small syntactic prototypes.* In Proceedings of 5th International Joint Conference on Natural Language Processing, pp. 438–446.

Brin S. and Page L. (1998). *The anatomy of a large-scale hypertextual web search engine.* Computer Networks and ISDN Systems 30, 107–117.

Briscoe T. and Waegner N. (1992). *Robust stochastic parsing using the inside-outside algorithm*. In Proc. of the AAAI Workshop on Probabilistic-Based Natural Language Processing Techniques, pp. 39–52.

Brodsky P. and Waterfall H. (2007). *Characterizing motherese: on the computational structure of child-directed language*. In Proceedings of the Annual Meeting of the Cognitive Science Society, Vol. 29.

Brooks D.J. (2006). *Unsupervised grammar induction by distribution and attachment*. In Proceedings of the Tenth Conference on Computational Natural Language Learning. Association for Computational Linguistics, pp. 117–124.

Cao, Z., Wei, F., Dong, L., Li, S. and Zhou, M., 2015, February. *Ranking with recursive neural networks and its application to multi-document summarization*. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 29, No. 1).

Chen D. and Christopher M. (2014). *A fast and accurate dependency parser using neural networks.* Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP).

Chomsky N. (1957). *Syntactic Structures.* The Hague: Mouton.

Chomsky N. (1965). *Aspects of the Theory of Syntax*. Cambridge, MA: MIT Press.

Chomsky N. (1968). *Remarks on Nominalization*. Linguistics Club, Indiana University.

Chomsky N. (2014). *Aspects of the Theory of Syntax*, vol. 11. MIT Press.

Christianson, K., Hollingworth, A., Halliwell, J.F. and Ferreira, F., 2001. *Thematic roles assigned along the garden path linger.* Cognitive psychology, 42(4), pp.368-407.

Clark A. and Lappin S. (2010). *Linguistic Nativism and the Poverty of the Stimulus*. John Wiley & Sons.

Clifton Jr, C., Staub, A. and Rayner, K., 2007. *Eye movements in reading words and sentences.* Eye movements, pp.341-371.

Cocos A., Masino A., Qian T., Pavlick E. and Callison-Burch C. (2015). *Effectively crowdsourcing radiology report annotations.* In Proceedings of the Sixth International Workshop on Health Text Mining and Information Analysis, pp. 109–114.

Cohen, A.L. and Staub, A., 2014. *Online processing of novel noun–noun compounds: Eye movement evidence.* Quarterly Journal of Experimental Psychology, 67(1), pp.147-165.

Cohen J. *A coefficient of agreement for nominal scales.* Educ Psychol Meas. 1960;20(1):37–46

Cohen J. *Weighted kappa: nominal scale agreement provision for scaled disagreement or partial credit.* Psychol Bull. 1968;70(4):213–20

Cole, P. and Morgan, J.L., 1977. *Syntax and semantics.* Volume 3: Speech acts.

Covington, M.A., 1990. *A dependency parser for variable-word-order languages.* University of

Georgia

Cramer B. (2007). *Limitations of current grammar induction algorithms.* In Proceedings of the 45th Annual Meeting of the ACL: Student Research Workshop. Association for Computational Linguistics, pp. 43–48.

Croft, W., 2001. *Radical construction grammar: Syntactic theory in typological perspective.* Oxford University Press on Demand.

Cunliffe, D., Vlachidis, A., Williams, D. and Tudhope, D., 2022. *Natural language processing for under-resourced languages: Developing a Welsh natural language toolkit.* Computer Speech & Language, 72, p.101311.

Dalrymple M. (2001). *Lexical Functional Grammar*. Brill.

Daltrozzo, J. and Schön, D., 2009. *Conceptual processing in music as revealed by N400 effects on words and musical targets.* Journal of cognitive neuroscience, 21(10), pp.1882-1892.

Daneš, F., 1987. *On Prague School functionalism in linguistics.* In Functionalism in linguistics (p. 3). John Benjamins.

Das, A., Halder, T. and Saha, D., 2017, May. *Automatic extraction of Bengali root verbs using Paninian grammar.* In 2017 2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT) (pp. 953-956). IEEE.

Debusmann, R., 2000. *An introduction to dependency grammar*. Hausarbeit fur das Hauptseminar Dependenzgrammatik SoSe, 99, pp.1-16.

Dennis S.J. (2005). *An exemplar-based approach to unsupervised parsing.*

Di Caro, L. and Grella, M., 2013. *Sentiment analysis via dependency parsing. Computer Standards & Interfaces*, 35(5), pp.442-453.

Dik S. (1987). *Some principles of functional grammar*. Functionalism in Linguistics 20, 81.

Dik S. (1991). *Functional grammar*. Linguistic theory and grammatical description, pp. 247–274.

Dominguez M.A. and Infante-Lopez G. (2011). *Unsupervised induction of dependency structures using Probabilistic Bilexical Grammars.* In Natural Language Processing and

Knowledge Engineering (NLP-KE), 2011 7th International Conference on (pp. 314–318). IEEE.

Dunn J. (2017a). *Learnability and falsifiability of construction grammars*. Proceedings of the Linguistic Society of America 2, 1. Dunn J. (2017b). Computational learning of construction grammars. Language and Cognition 9, 254–292.

D'Ulizia A., Ferri F. and Grifoni P. (2011). *A survey of grammatical inference methods for natural language learning.* Artificial Intelligence Review 36, 1–27.

Edelman S., Solan Z., Horn D. and Ruppin E. (2003). *Rich syntax from a raw corpus: unsupervised does it.* In NIPS-2003 Workshop on Syntax, Semantics and Statistics.

Edelman S., Solan Z., Horn D. and Ruppin E. (2005). *Learning syntactic constructions from raw corpora.* In 29th Boston University Conference on Language Development.

Ellefson M.R. and Christiansen M.H. (2000). *Subjacency constraints without universal grammar: Evidence from artificial language learning and connectionist modeling*. In Proceedings of the Annual Meeting of the Cognitive Science Society, vol.22, No. 22.

Ellis, N. C., O'Dochartaigh, C., Hicks, W., Morgan, M., & Laporte, N. (2001). *Cronfa Electroneg o Gymraeg (CEG): A 1 million word lexical database and frequency count for Welsh.* [On-line]

Evans V. (2006). *Cognitive Linguistics*. Edinburgh University Press.

Evans, V., 2012. *Cognitive linguistics*. Wiley Interdisciplinary Reviews: Cognitive Science, 3(2), pp.129-141.

Falk Y. (2011). *Lexical-Functional Grammar.* Oxford University Press.

Fauconnier G. (1994). *Mental Spaces: Aspects of Meaning Construction in Natural Language.* Cambridge University Press.

Federmeier, K.D. and Kutas, M., 2001. *Meaning and modality: Influences of context, semantic memory organization, and perceptual predictability on picture processing.* Journal of Experimental Psychology: Learning, Memory, and Cognition, 27(1), p.202.

Fedorenko, E., Gibson, E. and Rohde, D., 2007. *The nature of working memory in linguistic, arithmetic and spatial integration processes.* Journal of Memory and Language, 56(2), pp.246-269.

Feist, M.I., 2008. *Space between languages.* Cognitive science, 32(7), pp.1177-1199.

Ferreira, V.S. and Yoshita, H., 2003. *Given-new ordering effects on the production of scrambled sentences in Japanese.* Journal of psycholinguistic research, 32(6), pp.669-692.

Fiebach, C.J. and Schubotz, R.I., 2006. *Dynamic anticipatory processing of hierarchical sequential events: a common role for Broca's area and ventral premotor cortex across domains?*. Cortex, 42(4), pp.499-502.

Fiebach, C.J., Schlesewsky, M. and Friederici, A.D., 2002. *Separating syntactic memory costs and syntactic integration costs during parsing: The processing of German WH-questions.* Journal of Memory and Language, 47(2), pp.250-272.

Filik, R., 2008. *Contextual override of pragmatic anomalies: Evidence from eye movements.* Cognition, 106(2), pp.1038-1046.

Frank S.L., Bod R. and Christiansen M.H. (2012). *How hierarchical is language use?* Proceedings of the Royal Society of London B: Biological Sciences, p.rspb20121741.

Frazier, L. and Rayner, K., 1982. *Making and correcting errors during sentence comprehension: Eye movements in the analysis of structurally ambiguous sentences.* Cognitive psychology, 14(2), pp.178-210.

Galley, M. and Manning, C.D., 2009, August. *Quadratic-time dependency parsing for machine translation.* In Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP (pp. 773-781).

Gamallo, P., Garcia, M. and Fernández-Lanza, S., 2012, April. *Dependency-based open information extraction.* In Proceedings of the joint workshop on unsupervised and semi-supervised learning in NLP (pp. 10-18).

Gazdar G., Klein E., Pullum G.K. and Sag I.A. (1985). *Generalized Phrase Structure Grammar.* Harvard University Press.

Gillenwater J., Ganchev K., Pereira F. and Taskar B. (2011). *Posterior sparsity in unsupervised dependency parsing.* Journal of Machine Learning Research 12, 455–490.

Givón T. (1983). *Topic Continuity in Discourse: A Quantitative Cross-Language Study*, vol. 3.

John Benjamins Publishing.

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

Goldberg, A.E., 1992. *The inherent semantics of argument structure: The case of the English ditransitive construction.*

Gregoromichelaki, E., Kempson, R., Howes, C. and Eshghi, A., 2013. *On making syntax dynamic. Alignment in communication: Towards a new theory of communication*, pp.57-86.

Guo, J., Che, W., Wang, H. and Liu, T., 2016, December. *A universal framework for inductive transfer parsing across multi-typed treebanks*. In Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers (pp. 12-22).

Guo, Y. and Stylios, G., 2005. *An intelligent summarization system based on cognitive psychology.* Information Sciences, 174(1-2), pp.1-36.

Gupta, A., Akula, A., Malladi, D., Kukkadapu, P., Ainavolu, V. and Sangal, R., 2012, November. *A novel approach towards building a portable nlidb system using the computational paninian grammar framework.* In 2012 International Conference on Asian Language Processing (pp. 93-96). IEEE.

Hagoort, P., 2007. 11 *The memory, unification, and control (MUC) model of language.* Automaticity and control in language processing, 1, p.243.

Hagoort, P., Brown, C.M. and Osterhout, L., 1999. *The neurocognition of syntactic processing.* The neurocognition of language, pp.273-316.

Hajic J., Hajicová E., Panevová J., Sgall P., Bojar O., Cinková S., Fucíková E., Mikulová M., Pajas P., Popelka J. and Semecký J. (2012). *Announcing Prague Czech-English Dependency Treebank 2.0.* In LREC, pp. 3153–3160.

Hampe B. and Grady J.E. (eds.) (2005). *From Perception to Meaning: Image Schemas in Cognitive Linguistics*, vol. 29. Walter de Gruyter.

Harrison C., Nuttall L., Stockwell P. and Yuan W. (2014). *Introduction: cognitive grammar in literature. In Cognitive Grammar in Literature.* John Benjamins, pp. 1–16.

Harrison M.A. (1978). *Introduction to Formal Language Theory.* Addison-Wesley Longman Publishing Co., Inc.

Hauser, M.D., Chomsky, N. and Fitch, W.T., 2002. *The faculty of language: what is it, who has it, and how did it evolve?*. Science, 298(5598), pp.1569-1579.

Headden III W.P., Johnson M. and McClosky D. (2009). *Improving unsupervised dependency parsing with richer contexts and smoothing. In Proceedings of Human Language Technologies:* The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics. Association for Computational Linguistics, pp. 101–109. In Proceedings of the 22nd International Conference on Computational Linguistics, vol. 1. Association for Computational

Hengeveld, K. and Mackenzie, J.L., 2006. *Functional discourse grammar.* Encyclopedia of language and linguistics, 4, pp.668-676.

Inoue, A. and Fodor, J.D., 1995. *Information-paced parsing of Japanese.* Japanese sentence processing, pp.9-63.

Jackendoff R. (1977). X syntax: *A study of phrase structure.* Linguistic Inquiry Monographs 4. Cambridge, Mass., (2),pp. 1–249.

Jain, A.K., Mao, J. and Mohiuddin, K.M., 1996. *Artificial neural networks: A tutorial*. Computer, 29(3), pp.31-44.

Jensen K.E. (2014). *Performance and competence in usage-based construction grammar.* In Multidisciplinary Perspectives on Linguistic Competences, pp. 157–188.

Jijkoun, V., Mur, J. and de Rijke, M., 2004. *Information extraction for question answering: Improving recall through syntactic patterns.* In COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics (pp. 1284-1290).

Jin L., Doshi-Velez F., Miller T., Schuler W. and Schwartz L. (2018). *Unsupervised Grammar Induction with Depth-bounded PCFG.* arXiv preprint arXiv:1802.08545.

Jones K.S. (2007). *Computational linguistics: what about the linguistics?* Computational Linguistics 33, 437–441.\

Joshi A.K. (1985). *Tree adjoining grammars: How much context-sensitivity is required to*

provide reasonable structural descriptions? In Dowty D.R., Karttunen L. & Zwicky A.M. (eds), Natural language parsing, Cambridge University Press, pp. 206–250.

Joshi, A., Kale, S., Chandel, S. and Pal, D.K., 2015. *Likert scale: Explored and explained. British journal of applied science & technology,* 7(4), p.396.

Kaiser E. and Trueswell J.C. (2004). *The role of discourse context in the processing of a flexible word-order language*. Cognition 94, 113–147.

Kak, S.C., 1987. *The Paninian approach to natural language processing.* International Journal of Approximate Reasoning, 1(1), pp.117-130.

Kamide, Y. and Mitchell, D.C., 1999. *Incremental pre-head attachment in Japanese parsing.* Language and cognitive processes, 14(5-6), pp.631-662.

Kasabov, N.K., 1996. *Foundations of neural networks, fuzzy systems, and knowledge engineering.* Marcel Alencar.

Katz-Brown, J., Petrov, S., McDonald, R., Och, F.J., Talbot, D., Ichikawa, H., Seno, M. and Kazawa, H., 2011, July. *Training a parser for machine translation reordering.* In Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (pp. 183-192).

Kennedy, M., 2001. *The domain specificity of the resources required for sentence processing* (Doctoral dissertation, University of British Columbia).

Kesidi, S.R., Kosaraju, P., Vijay, M. and Husain, S., 2013. *CONSTRAINTBASED HYBRID DEPENDENCY PARSER FOR TELUGU* (Doctoral dissertation, Ph. D. thesis, International Institute of Information Technology Hyderabad, India).

Kilgarriff, A., Rychlý, P., Jakubicek, M., Kovár, V., Baisa, V. and Kocincová, L., 2014, May. *Extrinsic Corpus Evaluation with a Collocation Dictionary Task*. In LREC (pp. 545-552).

Kim, J.D., Ohta, T., Tateisi, Y. and Tsujii, J.I., 2003. *GENIA corpus—a semantically annotated corpus for bio-textmining*. Bioinformatics, 19(suppl_1), pp.i180-i182.

Kiperwasser E. and Yoav G. (2016) *Simple and accurate dependency parsing using bidirectional LSTM feature representations.* Transactions of the Association for Czomputational Linguistics 4, 313–327.

Kitchenham B. and Charters S. (2007). *Guidelines for performing systematic literature reviews in software engineering.*

Klein D. and Manning C.D. (2004). *Corpus-based induction of syntactic structure: models of dependency and constituency.* In Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics. Association for Computational Linguistics, p. 478.

Knight, D., 2020. *CorCenCC: Corpws Cenedlaethol Cymraeg Cyfoes–the National Corpus of Contemporary Welsh.* Oxford Text Archive Core Collection.

Knight, D., Fitzpatrick, T., Morris, S., Evas, J., Rayson, P., Spasić, I., Stonelake, M., Thomas, E.M., Neale, S., Needs, J. and Piao, S., 2017. *Creating CorCenCC (Corpws Cenedlaethol Cymraeg Cyfoes-The National Corpus of Contemporary Welsh)*.

Knight, D., Loizides, F., Neale, S., Anthony, L. and Spasić, I., 2020. *Developing computational infrastructure for the CorCenCC corpus: The National Corpus of Contemporary Welsh.* Language Resources and Evaluation, pp.1-28.

Kovár, V., Jakubıcek, M. and Horák, A., 2016. *On Evaluation of Natural Language Processing Tasks.* In Proceedings of the 8th International Conference on Agents and Artificial Intelligence (pp. 540-545).

Krippendorff, K., 1970. *Estimating the reliability, systematic error and random error of interval data.* Educational and Psychological Measurement, 30(1), pp.61-70.

Krippendorff, K., 2011. *Computing Krippendorff's alpha-reliability.*

Król-Markefka A. (2014). *Between usage-based and meaningfully-motivated grammatical rules: a psycholinguistic basis of applied cognitive grammar.* Studia Linguistica Universitatis Iagellonicae Cracoviensis 131, 43.

Kuperberg, G.R., Kreher, D.A., Sitnikova, T., Caplan, D.N. and Holcomb, P.J., 2007. *The role of animacy and thematic relationships in processing active English sentences: Evidence from event-related potentials.* Brain and language, 100(3), pp.223-237.

Lakoff G. and Johnson M. (1980). *Conceptual metaphor in everyday language.* The Journal of Philosophy 77, 453–486. Lakoff G. (1988). Cognitive semantics. Meaning and Mental Representations 119, 154.

Langacker R.W. (1987). *Foundations of Cognitive Grammar: Theoretical Prerequisites*, vol. 1. Stanford university press.

Langacker R.W. (2008). *Cognitive Grammar: A Basic Introduction*. Oxford University Press.

Langacker R.W. (2009). *Investigations in Cognitive Grammar*, vol. 42. Walter de Gruyter.

Langacker, R.W. and Langacker, R., 2008. *Cognitive grammar: A basic introduction.* OUP USA.

Langacker, R.W., 1987. *Foundations of cognitive grammar: Theoretical prerequisites* (Vol. 1). Stanford university press.

Langacker, R.W., 2001. *Discourse in cognitive grammar.* Cognitive linguistics, 12(2), pp.143-188.

Langacker, R.W., 2012. *Essentials of cognitive grammar.* Oxford University Press.

Laszlo, S. and Federmeier, K.D., 2008. *Minding the PS, queues, and PXQs: Uniformity of semantic processing across multiple stimulus types.* Psychophysiology, 45(3), pp.458-466.

Lawrence S., Giles C.L. and Fong S. (2000). *Natural language grammatical inference with recurrent neural networks.* IEEE Transactions on Knowledge and Data Engineering 12, 126–140.

Leech G.N. (1993). *Statistically-Driven Computer Grammars of English: The IBM/Lancaster Approach* (No. 8). Rodopi.

Levine R.D. and Meurers W.D. (2006). *Head-driven phrase structure grammar.* Encyclopedia of Language and Linguistics 5, 237–252.

Li, Y., Zhou, X., Sun, Y. and Zhang, H., 2016. *Design and implementation of Weibo sentiment analysis based on LDA and dependency parsing.* China Communications, 13(11), pp.91-105.

Li, Z., Callison-Burch, C., Dyer, C., Khudanpur, S., Schwartz, L., Thornton, W., Weese, J. and Zaidan, O., 2009, March. *Joshua: An open source toolkit for parsing-based machine translation.* In Proceedings of the Fourth Workshop on Statistical Machine Translation (pp. 135-139).

Lin, J.W., 2006. *Time in a language without tense: The case of Chinese.* Journal of semantics, 23(1), pp.1-53.

Linguistics, pp. 721–728.

Loper, E. and Bird, S., 2002. *Nltk: The natural language toolkit.* arXiv preprint cs/0205028.

Maier, W., 2006, July. *Annotation schemes and their influence on parsing results.* In Proceedings of the COLING/ACL 2006 Student Research Workshop (pp. 19-24).

Marcus, M., Santorini, B. and Marcinkiewicz, M.A., 1993. *Building a large annotated corpus of English: The Penn Treebank.*

Marecˇek D. and Žabokrtský Z. (2012a). *Unsupervised dependency parsing using reducibility and fertility features.* In Proceedings of the NAACL-HLT Workshop on the Induction of Linguistic Structure. Association for Computational Linguistics, pp. 84–89.

Marecˇek D. and Žabokrtský Z. (2012b). *Exploiting reducibility in unsupervised dependency parsing.* In Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning. Association for Computational Linguistics, pp. 297–307.

Marques T. and Beuls K. (2016). *Evaluation strategies for computational construction grammars.* In Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, pp. 1137–1146.

Matthiessen C.M. and Halliday M.A.K. (2009). *Systemic functional grammar: a first step into the theory.*

McHugh, M.L., 2012. *Interrater reliability: the kappa statistic.* Biochemia medica, 22(3), pp.276-282.

Miller, G.A., 1995. *WordNet: a lexical database for English*. Communications of the ACM, 38(11), pp.39-41.

Miyamoto, E.T., 2002. *Case markers as clause boundary inducers in Japanese.* Journal of psycholinguistic research, 31(4), pp.307-347.

Miyao, Y., Sagae, K., Sætre, R., Matsuzaki, T. and Tsujii, J.I., 2009. *Evaluating contributions of natural language parsers to protein–protein interaction extraction.* Bioinformatics, 25(3), pp.394-400.

Mollá, D. and Hutchinson, B., 2003, April. *Intrinsic versus extrinsic evaluations of parsing systems.* In Proceedings of the EACL 2003 Workshop on Evaluation Initiatives in Natural Language Processing: are evaluation methods, metrics and resources reusable? (pp. 43-50).

Moshier M. (1988). *Extensions to unification grammar for the description of programming languages.*

Muralidaran, V., Palmer, G., Arman, L., O'Hare, K., Knight, D. and Spasic, I., 2021. *A practical implementation of a porter stemmer for Welsh.*

Muralidaran, V. and Sharma, D.M., 2016, April. *Construction grammar based annotation framework for parsing Tamil.* In International Conference on Intelligent Text Processing and Computational Linguistics (pp. 378-396). Springer, Cham.

Muralidaran, V., Spasić, I. and Knight, D., 2020, October. *A Cognitive Approach to Parsing with Neural Networks.* In International Conference on Statistical Language and Speech Processing (pp. 71-84). Springer, Cham.

Muralidaran, V., Spasić, I. and Knight, D., 2020. *A systematic review of unsupervised approaches to grammar induction.* Natural Language Engineering, pp.1-43.

Nasukawa, T. and Yi, J., 2003, October. *Sentiment analysis: Capturing favorability using natural language processing.* In Proceedings of the 2nd international conference on Knowledge capture (pp. 70-77).

Neale, S., Donnelly, K., Watkins, G. and Knight, D., 2018, May. *Leveraging lexical resources and constraint grammar for rule-based part-of-speech tagging in Welsh.* In Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018).

Neubig, G., Dyer, C., Goldberg, Y., Matthews, A., Ammar, W., Anastasopoulos, A., Ballesteros, M., Chiang, D., Clothiaux, D., Cohn, T. and Duh, K., 2017. *Dynet: The dynamic neural network toolkit.* arXiv preprint arXiv:1701.03980.

Neves M. and Ševa J. (2019). *An extensive review of tools for manual annotation of documents.* Briefings in Bioinformatics.

Nichols J. (1984). *Functional theories of grammar.* Annual Review of Anthropology 13, 97–117.

O'Grady, W., 2010. *An emergentist approach to syntax.* The Oxford handbook of linguistic analysis, pp.257-83.

Paillet J.P. (1973). *Computational linguistics and linguistic theory.* In Proceedings of the 5th Conference on Computational Linguistics, vol. 2. Association for Computational Linguistics, pp. 357–366.

Pao, Y.H. and Sobajic, D.J., 1991. *Neural networks and knowledge engineering*. IEEE transactions on knowledge and data engineering, 3(2), pp.185-192.

Patel, A.D., 2012. *Language, music, and the brain: a resource-sharing framework.* Language and music as cognitive systems, pp.204-223.

Petticrew M. 2001. *Systematic reviews from astronomy to zoology: myths and misconceptions.* British Medical Journal 322, 98–101.

Piao, S.S., Rayson, P.E., Knight, D. and Watkins, G., 2018. *Towards a Welsh semantic annotation system.*

Poibeau, T. and Messiant, C., 2008. *Do we still need gold standard for evaluation?.*

Pollard C. and Sag I.A. (1994). *Head-Driven Phrase Structure Grammar.* University of Chicago Press.

Ponvert E., Baldridge J. and Erk K. (2011). *Simple unsupervised grammar induction from raw text with cascaded finite state models.* In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, vol. 1. Association for Computational Linguistics, pp. 1077–1086.

Porter, M.F., 2001. *Snowball: A language for stemming algorithms.*

Post M. and Gildea D. (2013). *Bayesian tree substitution grammars as a usage-based approach.* Language and Speech 56, 291–308.

Primus, B., 2009. *Case, grammatical relations, and semantic roles.* In The Oxford Handbook of Case.

Prys, D., Jones, D., Prys, G., Watkins, G., Cooper, S., Roberts, J.C., Butcher, P., Farhat, L., Teahan, W. and Prys, M., 2021. *Language and Technology in Wales*: Volume I.

Radford A. (1981). *Transformational Syntax: A Student's Guide to Chomsky's Extended Standard Theory.* Cambridge University Press.

Rayner, K., Warren, T., Juhasz, B.J. and Liversedge, S.P., 2004. *The effect of plausibility on eye movements in reading.* Journal of Experimental Psychology: Learning, Memory, and Cognition, 30(6), p.1290.

Reichart R. and Rappoport A. (2008). *Unsupervised induction of labeled parse trees by clustering with syntactic features.*

Rimell L., Clark S. and Steedman M. (2009). *Unbounded dependency recovery for parser evaluation.* In Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, vol. 2. Association for Computational Linguistics, pp. 813–821.

Roberts, I.G., 2005. *Principles and parameters in a VSO language: a case study in Welsh.* Oxford University Press on Demand.

Roche E. and Schabes Y. (eds) (1997). *Finite-State Language Processing.* MIT press.

Saffran J.R. (2001). *The use of predictive dependencies in language learning.* Journal of Memory and Language 44, 493–515. Sangati F. (2010). A probabilistic generative model for an intermediate constituency-dependency representation. In Proceedings of the ACL 2010 Student Research Workshop. Association for Computational Linguistics, pp. 19–24.

SANGAL, R., Chaitanya, V. and Bharati, A., 1995. *Natural language processing: a Paninian perspective.* PHI Learning Pvt. Ltd.

Santamaria J. and Araujo L. (2010). *Identifying patterns for unsupervised grammar induction.* In Proceedings of the Fourteenth Conference on Computational Natural Language Learning. Association for Computational Linguistics, pp. 38–45.

Schabes Y., Roth M. and Osborne R. (1993). *Parsing the Wall Street Journal with the inside-outside algorithm.* In Proceedings of the sixth conference on European chapter of the Association for Computational Linguistics. Association for Computational Linguistics, pp. 341–347.

Seginer Y. (2007). *Fast unsupervised incremental parsing.* In Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics, pp. 384–391.

Shanmuganathan, S., 2016. *Artificial neural network modelling: An introduction.* In Artificial neural network modelling (pp. 1-14). Springer, Cham.

Sharma, S., Sharma, S. and Athaiya, A., 2017. Activation functions in neural networks. towards data science, 6(12), pp.310-316.

Silveira, N., Dozat, T., De Marneffe, M.C., Bowman, S., Connor, M., Bauer, J. and Manning, C.D., 2014, May. *A gold standard dependency corpus for English.* In Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14) (pp. 2897-2904).

Skinner B.F. (2014). *Verbal Behavior.* BF Skinner Foundation.

Skut, W., Krenn, B., Brants, T. and Uszkoreit, H., 1997. *An annotation scheme for free word order languages.* arXiv preprint cmp-lg/9702004.

Snyder B., Naseem T. and Barzilay R. (2009). *Unsupervised multilingual grammar induction.* In Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP, vol. 1. Association for Computational Linguistics, pp. 73–81.

Solan Z., Horn D., Ruppin E. and Edelman S. (2004). *Unsupervised context sensitive language acquisition from a large corpus.* In Advances in Neural Information Processing Systems, pp. 961–968.

Spitkovsky V.I., Alshawi H. and Jurafsky D. (2010). *From baby steps to leapfrog: How less is more in unsupervised dependency parsing.* In Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics. Association for Computational Linguistics, 751–759.

Spitkovsky V.I., Alshawi H. and Jurafsky D. (2011). *Punctuation: Making a point in unsupervised dependency parsing.* In Proceedings of the Fifteenth Conference on Computational Natural Language Learning. Association for Computational Linguistics, pp. 19–28.

Spitkovsky V.I., Alshawi H. and Jurafsky D. (2012). *Capitalization cues improve dependency grammar induction.* In Proceedings of the NAACL-HLT Workshop on the Induction of Linguistic Structure. Association for Computational Linguistics, pp. 16–22.

Staub, A., 2015. Reading sentences: *Syntactic parsing and semantic interpretation.* The Oxford handbook of reading, pp.202-216.

Staub, A., Rayner, K., Pollatsek, A., Hyönä, J. and Majewski, H., 2007. *The time course of plausibility effects on eye movements in reading: Evidence from noun-noun compounds.* Journal of Experimental Psychology: Learning, Memory, and Cognition, 33(6), p.1162.

Structures, S., 1957. The Hague: Mouton. Janua Linguarum

Stroud, C. and Phillips, C., 2012. *Examining the evidence for an independent semantic analyzer: An ERP study in Spanish.* Brain and language, 120(2), pp.108-126.

Studer, R., Benjamins, V.R. and Fensel, D., 1998. *Knowledge engineering: principles and methods.* Data & knowledge engineering, 25(1-2), pp.161-197.

Søgaard A. (2011). *From ranked words to dependency trees: two-stage unsupervised non-projective dependency parsing.* In Proceedings of TextGraphs-6: Graph-based Methods for Natural Language Processing. Association for Computational Linguistics, pp. 60–68.

Talmy, L., 1975, September. *Figure and ground in complex sentences.* In Annual meeting of the berkeley linguistics society (Vol. 1, pp. 419-430).

Taylor A., Marcus M. and Santorini B. (2003). *The Penn treebank: an overview.* In Treebanks. Dordrecht: Springer, pp. 5–22. Yang C. (2011). A statistical test for grammar. In Proceedings of the 2nd workshop on Cognitive Modeling and Computational Linguistics. Association for Computational Linguistics, pp. 30–38.

Taylor, J.R., 1988. *Contrasting prepositional categories: English and Italian.* Topics in cognitive linguistics, 50, pp.299-326.

Telljohann, H., Hinrichs, E., Kübler, S. and Kübler, R., 2004. *The TüBa-D/Z treebank: Annotating German with a context-free backbone.* In Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC 2004).

Van Gompel, R.P., Pickering, M.J. and Traxler, M.J., 2001. *Reanalysis in sentence processing: Evidence against current constraint-based and two-stage models.* Journal of Memory and Language, 45(2), pp.225-258.

Van Herten, M., Kolk, H.H. and Chwilla, D.J., 2005. *An ERP study of P600 effects elicited by semantic anomalies.* Cognitive brain research, 22(2), pp.241-255.

Van Petten, C., Coulson, S., Rubin, S., Plante, E. and Parks, M., 1999. *Time course of word*

identification and semantic integration in spoken language. Journal of Experimental Psychology: Learning, Memory, and Cognition, 25(2), p.394.

Van Valin Jr, R.D., 1990. *Semantic Roles and Grammatical Relations.*

Vigneshwaran, M., 2016. *Construction grammar approach for Tamil dependency parsing* (Masters thesis, International Institute of Information Technology Hyderabad).

Wang, S., Mo, D., Xiang, M., Xu, R. and Chen, H.C., 2013. *The time course of semantic and syntactic processing in reading Chinese: Evidence from ERPs.* Language and cognitive processes, 28(4), pp.577-596.

Warren, T. and McConnell, K., 2007. *Investigating effects of selectional restriction violations and plausibility violation severity on eye-movements in reading.* Psychonomic bulletin & review, 14(4), pp.770-775.

Willis, D., 2006. *Negation in Middle Welsh.* Studia Celtica, 40(1), pp.63-88.

Wright, S.E., 2003. *From the semiotic triangle to the semantic web.* Journal of the International Institute for Terminology Research, 14, pp.111-135.

Yegnanarayana, B., 2009. *Artificial neural networks.* PHI Learning Pvt. Ltd..

Zhang, Y. and Xue, N., 2014, October. *Automatic inference of the tense of chinese events using implicit linguistic information.* In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP) (pp. 1902-1911).

Zhang, Y., Yu, J. and Boland, J.E., 2010. *Semantics does not need a processing license from syntax in reading Chinese.* Journal of Experimental Psychology: Learning, Memory, and Cognition, 36(3), p.765.

Zuidema W. (2006). *What are the productive units of natural language grammar?: a DOP approach to the automatic identification of constructions.* In Proceedings of the Tenth Conference on Computational Natural Language Learning. Association for Computational Linguistics, pp. 29–36.

Zwarts, J., 2017. Spatial semantics: *Modelling the meaning of prepositions.* Language and linguistics compass, 11(5), p.e12241.

Östman J.O. and Fried M. (eds) (2005). *Construction Grammars: Cognitive Grounding and Theoretical Extensions*, vol. 3. John Benjamins Publishing.