Doctoral Thesis

Knowledge Extraction from Simplified Natural Language Text

Hazem Safwat Abdelaal

October 6, 2019

External Examiner	**Supervisors**	**Internal Examiner**
Prof. Laurette Pretorius	Dr. Brian Davis Dr. Manel Zarrouk	Dr. Colm O'Riordan

A thesis submitted in fulfillment of the requirements
for the degree of Doctor of Philosophy

Insight Centre for Data Analytics
College of Engineering and Informatics, National University of Ireland,
Galway

DECLARATION

I declare that this thesis, titled "*Knowledge Extraction from Simplified Natural Language Text*", is composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification.

Galway, October 6, 2019

_____

Hazem Safwat Abdelaal

ABSTRACT

Knowledge base creation and population are an essential formal backbone for a variety of intelligent applications, decision support and expert systems and intelligent search. Although knowledge extraction from unstructured text offers a means of easing the knowledge acquisition process, the ambiguous nature of language tends to impact on accuracy when engaging in more complex semantic analysis.

Controlled Natural Languages (CNLs) are subsets of natural language which are restricted grammatically in order to reduce or eliminate ambiguity for the purposes of machine understanding, or unambiguous human communication within a domain or industry context, such as Simplified English. Moreover, CNLs help engaging non-expert users with no background in knowledge engineering, as these languages offer user-friendly interfaces that are easier to understand and accepted by users. The latter type of *human-oriented* CNL is under-researched despite having found favor in industry over many years.

Rewriting such human-oriented CNL content into a machine-oriented CNL could potentially unlock significant silos of implicit valuable general purpose domain knowledge. In this thesis, we have a developed an approach for a series of corpus based rewriting rules for subsequent knowledge capture. Our work confirms that a substantial amount of human-oriented CNL content can be easily translated into a machine processable CNL for formal knowledge capture with little semantic loss. In addition, we describe a novel dataset which aligns a representative sample of Simplified English Wikipedia sentences with a well known machine-oriented CNL. This linguistic resource is both human-readable and semantically machine interpretable, where it can be used by the community as a gold-standard dataset which can benefit a variety of language processing and knowledge based applications.

ACKNOWLEDGEMENTS

In the name of Allah, The Most Gracious, The Most Merciful

First and foremost, All praise, thanks and glory be to the Almighty Allah for answering my prayers to give me the opportunity, knowledge, strength, patience, and hope to undertake this research study and produce this thesis. Without his kindness and mercy, I would not have been able to complete this endeavor.

I have to take this opportunity to mention that a PhD thesis is far from being only a personal work and its accomplishment requires the help, the support and the collaboration of several people. In the next few lines, I will mention some people names, who have helped and supported me during my PhD journey, without them this thesis would not have been possible.

The first person I would like to thank is my supervisor, Dr. Brian Davis for giving me the opportunity and trust that I can achieve my goal to pursue postgraduate studies at Insight, NUI Galway. I would like to thank him for always being a friend before being a supervisor and a mentor. I will never forget him believing in me, and giving me all the opportunities to be an independent researcher. I have learned a lot from his countless advice, patience, guidance and mentorship in all aspects of research and career.

I would like to thank my co-supervisor Dr. Manel Zarrouk, for her practical advice, help, and research group management, by ensuring that there is a stimulating environment and resources for all aspects of research in the knowledge discovery unit.

Special thanks to my examiners Prof. Laurette, Dr. Colm and the chair Prof. Mathieu for their feedback during viva. Also, I would like to thank other academics from Insight NUI Galway, and in particular Prof. Dietrich Rebholz-Schuhmann, Dr. John Breslin and Dr. Adegboyega Ojo, for their constructive feedback and useful critiques to ensure that this PhD comes true.

In Insight NUI Galway, I have met so many great and helpful people and made friends. I want in particular to thank Islam, Wassim and Ahmed who were the first guys to meet me in Insight and who helped make my

research time as enjoyable as possible. During my time at Insight, they were always there to help when things get complicated in work and in life.

Also, my thanks go to the rest of my friends and colleagues in Insight, who helped make my research time as cheerful and delightful as possible, I will always remember our fruitful discussions in the kitchen chatting about our work, our life, and our future. Special thanks to Omnia for helping me to print and deliver this thesis.

A special thanks to all my friends whom I met in Galway city, and shared with them dreams, very special and nice moments. In particular, I will mention Elkassaby and Wael. They helped me a lot to have patience and overcome my hard times, things would become much harder without their continuous support and help.

I am fortunate and grateful to have the support and prayers from all my friends I have known through my life. I would like to deeply thank all of them. Although, we do not see each other as frequent as before, their existence and their prayers always gave me the motivation, courage and positive energy to continue and pursue my goals. A special thanks to my friends Halawa and Kareem for all their motivational support at all times.

Finally, but perhaps most importantly, I can not find any words to tell my family how much I want to thank them for their infinite love, and support all over the past years.

For my parents, I express my deepest gratitude as I would not step any forward without your prayers and help. As well as, my brother Ramy and sister Riham who gave me all the strength and emotional support when life gets harder.

For my wife Salma and my son Ezzeldin, I can not express how much I am thankful for all your help and support during my PhD journey, although we faced a lot of difficulties, I am proud to mention that our love to each other made it possible to achieve it together as a family.

The research contributions reported by this thesis was supported by Science Foundation Ireland (SFI) under Grant Number SFI/12/RC/2289.

To My Family And Friends

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

1.1 GENERAL INTRODUCTION AND MOTIVATION

The idea of the Semantic Web envisaged extending the current version of the World Wide Web, enriching it with semantics and hence enabling machine processable web content, in effect transforming the "Web of Documents" into a "Web of Data" [BHL01]. It attempted to address the problem that most of the data on the Web is not uniformly defined or linked together. This problem prevents machines from enabling data integration and reusing the data across different applications [VCM08]. The Semantic Web model seeks to enable machines to interact freely with each other across different web resources and perform more sophisticated tasks than the current machines. Although the idea has not achieved its full potential yet, there are plenty of applications in data mining and knowledge discovery which have exploited Semantic Web technologies, and Linked Open Data [RP16][CP15][Pau17]. Examples of these technologies include the Resource Description Framework (RDF) [KC06] model that represents associations and assertions between Web resources, RDF Schema (RDFS) [McB04], and Web Ontology Language (OWL) [MV+04] which are formal vocabularies that define the semantics of Web resources. Most recently semantic technologies have been leveraged to create the Knowledge Graph which is a knowledge based generated by a search engine such as Google to improve its results with information collected from different sources [BEP+08].

However, the Semantic Web model relies on the existence of semantic data, which can be driven in the form of ontologies. We refer to the definition of ontology as mentioned in [Gru93] to be "a formal explicit specification of a shared conceptualisation". The ontology is formal and explicit which means that it should be machine readable, and its concepts should be explicitly defined and shared between a group of users [SBF+98].

Currently, machines have yet to research the required level of intelligence envisaged for the Semantic Web especially through the Natural Language

Processing (NLP) domain, to increase machine intelligence for understanding human language and increase the accuracy of specific sub-tasks such as semantic search, machine translation and question answering [DKP+09; BCS+07; FBC+10; NP10; SKW07]. These tasks require that knowledge should be captured and stored in a machine-readable format, which is the aim of knowledge base population and construction.

Since, knowledge bases are constructed by extracting information from natural language text, hence it is crucial to use the natural language processing and understanding approaches. These approaches transform the text representation from being unstructured and ambiguous, into a structured and unambiguous representation that enables machine reading and understandability. Formal data extraction and representation (also known as ontology authoring/development) is one of the core challenges for building the Semantic Web. This is because the process can be a barrier for non-expert users and small organizations who are trying to adopt semantic technologies [Dav13].

1.2 PROBLEM OVERVIEW AND REQUIREMENTS

Knowledge extraction from text is still a problem that is not fully solved [PDG12]. There are a variety of advantages that makes solving this problem crucial in the community. Most of these advantages lie within the number of applications that can benefit from the extracted knowledge. These applications include but are not limited to automatically generating quality linked data and ontologies from text, and generating machine-readable content to support Semantic Web Technologies. Furthermore, the demand for the automatic generation and bootstrapping of quality linked data has increased with the advent of projects such as DBpedia [BLK+09] . Although there are a lot of benefits, there are also many challenges with respect to the current approaches for ontology learning and population from text [WLB12]. Most of the existing tools focus on helping the user drafting ontologies by identifying some key elements in the textual resources, without taking into consideration the user background and expertise. According to [DGP+13], these tools have some limitations in automatically producing quality linked data for the Semantic Web due to the following conditions :

- A training phase is required which is time consuming and expensive.

- Further refinement is required to construct the output in a logical form.
- The output of these tools does not overlap with the expressivity and good design practices of a formal ontology language, such as the Web Ontology Language (OWL).
- It does not facilitate linking the output with existing predefined linked data vocabularies.

One common approach to address this problem is the Ontology Learning and Population (OL&P) approach. Several tools from the literature are using this approach to automatically learn ontologies from textual resources such as Text2Onto [CV05], SPRAT [MFP09], and LODIFIER [APR12]. These tools are based on different NLP techniques such as named entity recognition, ontology-based information extraction, relation extraction, and lexico-syntactic patterns to extract entities and relations from text. Although these tools helped in drafting and engineering ontologies, they usually require that the user has an ontology engineering background. This requirement is not easy to be fulfilled not only for non-technical users, but also for domain specialists such as clinical experts, business professionals, and linguists, etc. This makes it a big challenge as it requires a training phase for these professionals by knowledge engineers, in order to be familiar with the Semantic Web concepts and formalisms, which is a time and resources consuming process. On the other hand, collaboration between various users from knowledge engineers to domain specialists to link various conceptualizations of the domain is a challenge, as there is no common ground for communication due to different areas of expertise. These problems challenge the researchers to develop an approach that facilitates ontology authoring for non-technical users.

So based on the discussed considerations, we can conclude that it is very crucial to represent the output in a logical form such as OWL. Hence, in [DGP+13] the authors defined certain requirements that have to be met in order to enable robust knowledge extraction systems as follows:

- **R1:** Eliminate the need for training of non expert users for ontology engineering.
- **R2:** Map natural language to OWL representations.
- **R3:** Represent complex relations in text.

- **R4:** Eliminate the need for large-size domain-specific textual resources
- **R5:** Reduce the need for high computational resources.

1.3 PROPOSED APPROACH

With respect to the presented challenges and requirements, researchers have introduced alternative approaches for knowledge creation and management. One of these approaches is to use natural language interfaces such as as Controlled Natural Languages (CNLs). A CNL is a subset of natural language that is less ambiguous or unambiguous for both human consumption and machine processing, which can lower the comprehension barrier for non-expert users, while simultaneously allowing domain-experts to create more consistent and less ambiguous content in a user-friendly manner [SD14]. In addition, CNLs can help users from different backgrounds and with different levels of expertise to understand confusing or complex sentences, since the controlled language used to write the content is restricted, thus making it more formal and unambiguous.

As shown from Figure 1, CNLs can be grouped into two broad categories as follows [Huj98]:

- *Machine-oriented CNLs*, unambiguous and user friendly language for non experts to engineer formal knowledge-bases. It allows for automated reasoning to infer implicit knowledge, that follows from the explicit statements and to check for consistency. They are used in various applications such as knowledge representation and ontology construction, software and hardware specifications, and medical regulations. An example of a machine-oriented CNL is Attempto Controlled English (ACE) [FKK08a] which is based on First-Order Logic (FOL) [Smu12].
- *Human-oriented CNLs*, a subcategory of CNLs which offers a middle ground of reduced ambiguity for language processing but less restricted than a machine-oriented CNL [Huj98]. Their origins are motivated for the purposes of improving the quality of machine translation for technical documentation [NM00], and unambiguous communication between humans in a domain context such as air traffic control

[Pooo6]. An example of a human-oriented CNL is Simple English used to author Simple Wikipedia[1].

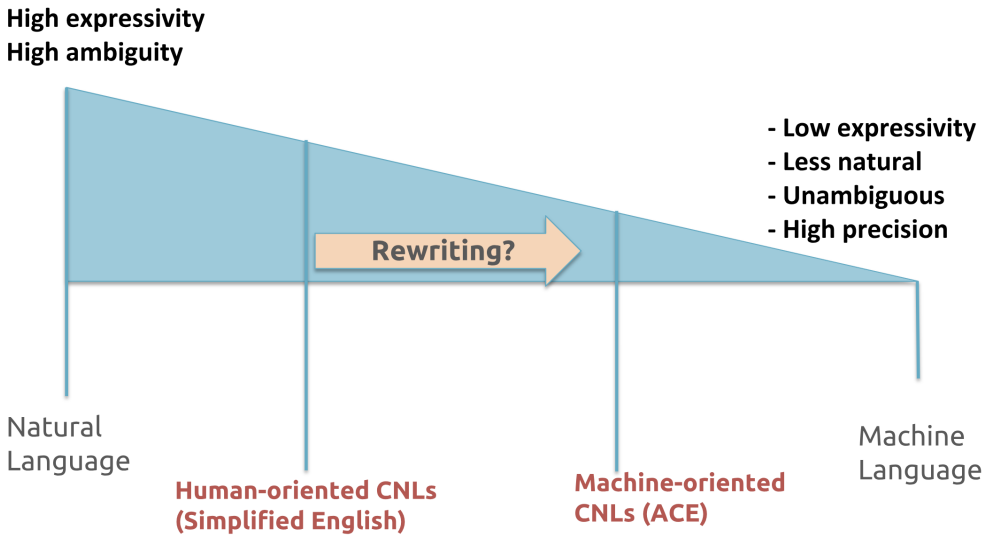


Figure 1: Ambiguity and expressivity of the controlled natural languages.

Although different applications that are based on knowledge extraction and knowledge base population could benefit from the idea of extracting and representing silos of knowledge from human-oriented CNLs by rewriting them into human-oriented CNLs, it has not been explored yet in the literature as shown in Figure 1. So, in this thesis we argue that rewriting a human-oriented CNL (Simplified English) into a machine processable CNL, will allow us to circumvent the barrier with respect to, uptake of machine processable CNLs by users working outside the knowledge engineering context. The notion of rewriting unstructured text into a machine processable CNL is very challenging as it is similar to the process of text simplification. However, this might be possible to apply on human-oriented CNLs (simplified text) as they represent that middle ground between both. Simplified text such as Simple Wikipedia, means that the text is written using style-guides[2] to avoid complex and ambiguous syntax. Rewriting all or most of a human-oriented CNL content into a machine-oriented CNL, could unlock significant rich silos of implicit general purpose formal knowledge. Moreover, the rewriting approach could be generalized by induction in different domains and industries using human-oriented CNLs such as Boeing Simplified English [WH96], and ASD Simplified Technical English[3],

1 https://simple.wikipedia.org/wiki/
2 https://simple.wikipedia.org/wiki/Wikipedia:How_to_write_Simple_English_pages
3 http://www.asd-ste100.org/

and other CNLs used for technical writing [Pooo6] using similar writing style-guides, since most of these CNLs have common properties as they are based on Basic English [Kuh14]. In this thesis we refer to the CNLs properties based on the study performed by O'Brien in [OBr03], defining common properties between different CNLs as the shared common rules between the different CNLs. These rules can be categorized based on their functionality into three categories as follows:

- *Lexical*, rules that affect word selection and their semantics such as abbreviation, prefix, numbering and anaphoric references.
- *Syntactic*, rules that affect the syntax of the text such as subject-verb agreement, noun clusters, tenses and voices.
- *Textual*, rules in this category are subdivided into two classes known as; i) Text structure rules, that are affecting the information load, such as sentence length, word counting, and ii) Pragmatic rules, that are influencing the reader response, such as avoid writing idioms and slang.

So, our motivation for choosing the approach of rewriting human-oriented CNLs (Simplified English) into a machine processable CNL (ACE)[FS96] to tackle the knowledge extraction problem is based on the following:

- There is a vast amount of content created using human-oriented CNLs in various domains such as internal corporate technical documentation e.g. Caterpillar, GM, aerospace e.g. Boeing, information technology e.g. IBM English, Avaya, Alcatel, and online encyclopedias e.g. Simple Wikipedia.
- No computational linguistic analysis i.e corpus based evidence to support O'Brien's CNL properties, which requires a linguistic analysis to confirm empirically if CNLs (human and machine) share O'Brien's properties.
- Empirically, the overlap between the properties of human-oriented CNLs and the properties of machine-oriented CNLs is unknown. So, we need to investigate the linguistic similarity between O'Brien's properties and unrestricted (free) NL text.

- We choose ACE as it is a widely known CNL that is mainly designed for knowledge representation. ACE texts are both human readable and machine processable that can be unambiguously mapped into different formal language such as Discourse Representation structures (DRS) [FKKo8b]. DRS is a variant of first order logic and its output of ACE texts can be bidirectionally translated into Web Ontology Language (OWL) [KFo6a]. Furthermore, we choose ACE for testing our rewriting system as it provides open access to its tools and resources such as the ACE parsing engine - APE[4], which we used for our system validation and for generating the DRS and OWL representations.

1.4 MAIN RESEARCH QUESTIONS

In order to bridge the gap between knowledge authoring and non-expert users based on the limitations in the current systems in the literature. We developed an approach that can satisfy the mentioned requirements for enabling robust knowledge extraction systems. This approach is based on four research questions as follows:

- **RQ1:** What is the overlap[5] between the properties of Simple Wikipedia (human-oriented CNL) and the CNL properties[6]?
- **RQ2:** What is the overlap between the properties of unrestricted text (e.g. Wikipedia) and the common CNL properties?
- **RQ3:** What is the feasibility[7] of rewriting human-oriented CNL (Simplified text) into machine processable text (e.g. ACE CNL)?
- **RQ4:** What is the effect of rewriting with respect to semantic loss?

4 https://github.com/Attempto/APE
5 The frequency of existence of two or more similar properties.
6 The common properties described earlier that can be found in most of the CNLs, defined by O'Brien in a research study [OBro3].
7 Refers to the coverage of the rewriting system in terms of the percentage of sentences that could be successfully rewritten.

1.5 HYPOTHESES

Based the aforementioned research questions, we generated the following core hypotheses:

1.5.0.1 *Corpus Analysis Hypothesis*

We developed a corpus analysis experiment to explore the CNLs properties defined in [OBr03], between both the Simplified Text (Simple Wikipedia abstracts) versus Unstructured Text (Parallel Wikipedia abstracts).

Hypothesis 1 (H1). *Lexical, syntactic and textual properties derived from the shared common rules between the different CNLs in O'Brien's analysis have higher statistical presence in Simplified text than unstructured text due to the recommended style guides imposed on authors to simplify the text.*

1.5.0.2 *System Approach Hypothesis*

Since, there are not any data sets in the literature that map between a natural language sentence and its parallel CNL equivalent, we could not train a machine learning model to automatically rewrite human-oriented CNLs into a machine readable format via translations into machine processable CNL. Therefore, our approach is mainly based on rewriting rules, and for that we explored the lexico-syntactic patterns in the sentence structures through their Part-of-Speech (POS) tags structures. A POS tag structure is the POS tag sequence of a given sentence, for example the POS tag structure of the sentence *hello world, good morning* is *NN NN JJ NN*. The aim of capturing the extracted patterns is to derive a methodology for developing the rewriting rules by grouping similar POS tags structures.

Hypothesis 2 (H2). *The shared common rules between the different CNLs in O'Brien's analysis can be used to extract corpus based syntactic patterns in the sentence structures through their POS tags structures to rewrite a human-oriented CNL sentence into a machine-oriented CNL sentence, such that the newly rewritten machine-oriented CNL sentence preserves the semantics of the original source human-oriented CNL sentence.*

1.6 RESEARCH METHODOLOGIES

In order to answer the mentioned research questions we developed a research methodology as follows:

- Comprehensive literature survey of all the machine-oriented CNLs and human-oriented CNLs, with their respective tools.
- Comparative analysis & evaluation for selecting the appropriate CNL for different tasks.
- Computational Corpus linguistic analysis with respect to common CNLs properties in both Simplified Text versus unrestricted(free) text.
- Creation of a gold-standard dataset of Simplified English (human-oriented CNL) sentences aligned with their equivalent machine-oriented CNL sentences.
- Development of the rewriting approach to automatically extract knowledge from Simplified English using rewriting rules.
- Evaluation for the percentage of extracted formal knowledge, and the semantic loss after rewriting.

1.7 THESIS CONTRIBUTIONS

With respect to the research questions addressed in Section 1.4. The next subsections present the main contributions of this thesis which are divided in **three** parts. The **first** part, describes our review and analysis of all the Semantic Web related CNLs in the literature. The **second** part, describes our contribution towards creating a gold-standard dataset which aligns a representative sample of Simplified English Wikipedia sentences with a well known machine-oriented CNL. Finally, the **third** part describes our work on developing and evaluating an approach for knowledge extraction, using a rewriting system to transform Simplified English sentences into a machine-oriented controlled language.

1.7.1 Analysis of Semantic Web related CNLs

With respect to CNLs Analysis for the Semantic Web, this thesis makes the following contributions:

- A comparative analysis between all the Semantic Web Related CNLs and their respective tools.
- A comparative analysis of evaluations performed for Semantic Web Related CNLs and corresponding tools.
- Analytic conclusions and recommendations for selecting the appropriate CNL for different tasks.

Based on the results of comparing all the Semantic Web related CNLs in the literature, we refer to ACE as the selected machine-oriented CNL for developing our rewriting approach.

1.7.2 CNLs Gold-Standard Data-set

This thesis makes the following contributions on human-oriented CNL corpus analysis and providing CNLs Gold-standard dataset:

- The results of computational linguistic analysis between the Simplified Text (Simple Wikipedia abstracts) and O'Brien's CNL properties. **(RQ1)**
- The results of computational linguistic analysis between unrestricted text (Parallel Wikipedia abstracts) and O'Brien's CNL properties. **(RQ2)**
- A gold-standard dataset of Simplified English (human-oriented CNL) sentences aligned with their equivalent machine-oriented CNL sentences, that is both human-readable and semantically machine interpretable. **(RQ3)**

1.7.3 On Knowledge Extraction

This thesis makes the following contributions on knowledge extraction:

- A novel approach to automatically extract knowledge from Simplified English using rewriting rules. **(RQ3)**
- Evaluation for the percentage of extracted formal knowledge in the form of OWL triples, the semantic loss of rewriting a human-oriented CNL to a machine-oriented CNL. **(RQ4)**

1.8 THESIS STRUCTURE

The rest of this thesis is organized as follows:

- *Chapter 2- Background and State of the Art:* This chapter provides an overview of all the required background technologies underlying this thesis and analyses some related work from the literature. In addition, it motivates the problem by discussing the limitations of the current knowledge extraction systems.
- *Chapter 3- Analysis of Human & Machine Oriented Controlled Natural Languages:* This chapter provides a review and analysis of the literature review for all the Semantic Web related CNLs, together with providing our analytic conclusions for different aspects of the current and future trends with respect to CNLs.
- *Chapter 4- A Gold-Standard Dataset for Human to Machine Oriented CNLs:* This chapter focuses on research questions **RQ1**, **RQ2** and its associated hypothesis H1. It explains the corpus analysis experiments, and results.
- *Chapter 5- From Simplified Text to Controlled Natural Language for Knowledge Extraction:* This chapter focuses on research questions **RQ3** and **RQ4** and its associated hypothesis H2. It presents the approach, the developed architecture, the algorithms used, and the results of the respective evaluations.
- *Chapter 6- Conclusions and Future Work:* This chapter provides conclusions for the thesis, and discusses the potential use cases along with future work.

1.9 ASSOCIATED PUBLICATIONS

Various aspects of our contributions in this thesis have been published in relevant venues as follows:

- **Safwat, Hazem**, and Brian Davis. "A brief state of the art of CNLs for ontology authoring". In International Workshop on Controlled Natural Language, pp. 190-200. Springer, Cham, 2014.
- **Safwat, Hazem**, and Brian Davis. "CNLs for the semantic web: a state of the art". journal of Language Resources and Evaluation 51, no. 1 (2017): 191-220.
- **Safwat, Hazem**, Zarrouk, Manel and Davis, Brian. "From Simplified Text to Knowledge Representation using Controlled Natural Language". International Journal of Computational Linguistics and Applications. In Proceedings of 18th International Conference on Intelligent Text Processing and Computational Linguistics (CICLING 2017).
- **Safwat, Hazem**, Manel Zarrouk and Brian Davis. "Rewriting Simplified Text into a Controlled Natural Language". In International Workshop on Controlled Natural Language CNL (2018). Frontiers in Artificial Intelligence and Applications. pp. 85 - 91.
- **Safwat, Hazem**, Brian Davis, and Manel Zarrouk. "Engineering an Aligned Gold-Standard Corpus of Human to Machine Oriented Controlled Natural Language". In 2018 IEEE/WIC/ACM International Conference on Web Intelligence (WI), pp. 421-427. IEEE, 2018.
- **Safwat, Hazem**, Normunds Gruzitis, Ramona Enache, and Brian Davis. "Embedded controlled language to facilitate information extraction from eGov policies". In Proceedings of the 17th International Conference on Information Integration and Web-based Applications & Services, p. 71. ACM, 2015.
- **Safwat, Hazem**, Normunds Gruzitis, Brian Davis, and Ramona Enache. "Extracting semantic knowledge from unstructured text using embedded controlled language". In 2016 IEEE Tenth International Conference on Semantic Computing (ICSC), pp. 87-90. IEEE, 2016.

2 BACKGROUND AND STATE OF THE ART

2.1 INTRODUCTION

This Chapter provides an overview of the required background technologies underlying this thesis, together with some related work from the literature. Section 2.2 describes Tim Berners Lee vision for the Semantic Web, and the migration from the current Hypertext Web to "Web of data". The Semantic Web related technologies are also introduced; this includes a discussion about ontologies and the Resource Description Framework (RDF) model, which is a set of *subject-predicate-object* triples. Although the Semantic Web model is very promising, it faces several challenges that makes its adoption hard due to lack of annotated data, and hence no interesting applications can be built. On the other hand, the research community are ahead of this adoption by presenting initiatives such as Semantic Web Services (OWL-S[1]) that would help in building interesting applications. Other movements in the last decade include, the Linked (open) data initiative such as the public datasets (e.g. DBpedia[2]), reusable vocabularies (schemas[3]), and user-friendly frameworks and languages (e.g. SPARQL[4], RDFa[5], JSON-LD[6]). Later, the technology giants such as Google, Yahoo, and Microsoft started to utilize metadata to support web markup for building applications (e.g. Google receipe search[7]). Other companies that adopted the use of Semantic

1 https://www.w3.org/Submission/OWL-S/
2 https://wiki.dbpedia.org/
3 www.schema.org
4 https://www.w3.org/TR/rdf-sparql-query/
5 https://rdfa.info/
6 https://json-ld.org/
7 https://developers.google.com/search/docs/data-types/recipe

Web technologies to build their applications include Facebook Open Graph[8], Twitter Cards[9], Apple Web Content markup[10], and Pinterest Rich Pins[11].

Section 2.3, provides the required knowledge about the main NLP tasks that are commonly used to build different Semantic Web applications. In Section 2.4 we discuss briefly the most popular knowledge extraction tools for the Semantic Web. In Section 2.5, we present some projects that utilize CNLs to build their applications. Furthermore, we discuss in Section 2.6 the main metrics used in the literature to evaluate these approaches. Finally, Section 2.8 offers conclusions.

2.2 THE SEMANTIC WEB

In order to improve the quality of the web, Tim Berners-Lee founded the World Wide Web Consortium (W3C) in 1994. The idea of the Semantic Web was proposed in 1998 as part of this conference, when they found that the current version of the web is full of unstructured documents, and machines can not process or read all these documents and hence humans has to do a lot of tasks manually [Ber+98]. The Semantic Web endeavours to extend the current Web, by introducing a single schema to populate a global database with structured information that can be queried. This process will transform the human readable "Web of Documents" into the "Web of Data"[12] by enriching information with well defined meaning, which is machine processable. As shown from Figure 2, the process of building the Semantic Web is heavily dependent on the existence of two fundamental technologies:

1. Resource Description Framework (RDF), the main data interchange language that models the description of the resources on the Semantic Web.

2. Web Ontology Language (OWL), the main representational vocabulary which describe the domain of interest, in the form of additional metadata attached to the Semantic Web resources.

8 http://ogp.me/
9 https://developer.twitter.com/en/docs/tweets/optimize-with-cards
10 apple.com/library/archive/documentation/General/Conceptual/AppSearch/WebContent.html
11 https://developers.pinterest.com/docs/rich-pins/overview/
12 http://www.w3.org/2001/sw/

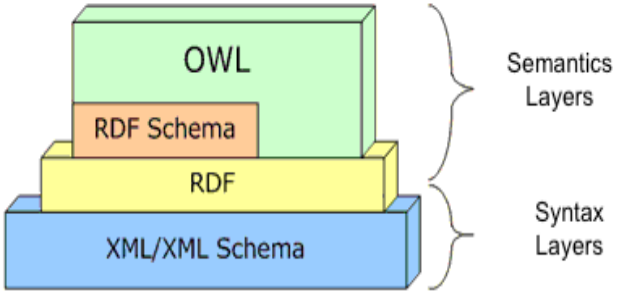


Figure 2: A snapshot of the Semantic Web Architecture.

2.2.1 Resource Description Framework (RDF)

RDF is defined to be the main standardised mechanism that should be used to describe the Semantic Web resources and information [CK04]. RDF statements are written in the form of metadata which are machine understandable. [MMM+04]. RDF statements that accompany the resources are usually structured in the form of three elements aka *triples*, these triples are divided into *subject*, *predicate*, and *object* expressions.

The resource is an object with identity that ranges from a webpage to real world objects such as a person. Pre-defined resources can be named and represented in the form of a URI, however, unidentified resources are unnamed nodes that are usually represented as blank nodes[13]. In Figure 3 we can see the RDF triple representation of an example sentence *Michael studies PhD*, where the subject and the object describe the resource and should be represented in the form of either uniform resource identifiers (URIs) or blank nodes. The predicate describes the relationship between the subject and the object with specific constraints that could not be represented by unnamed nodes, and must be represented in the form of URIs. The object can be specified by a simple string (literal).

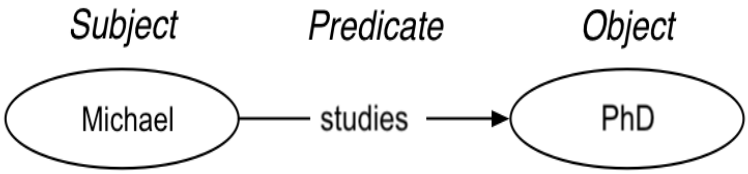


Figure 3: RDF Triple representation of an example sentence.

13 http://www.w3.org/TR/2014/REC-rdf11-concepts-20140225/#section-blank-nodes

In addition, RDF models complex statements with additional knowledge by using either *reification* or named graphs. As shown in Figure 4, reification can be used to model context or to add annotations by treating the additional knowledge as a resource and assign it a new URI, which might lead to shortcomings in the RDF model as it introduces indirectness.

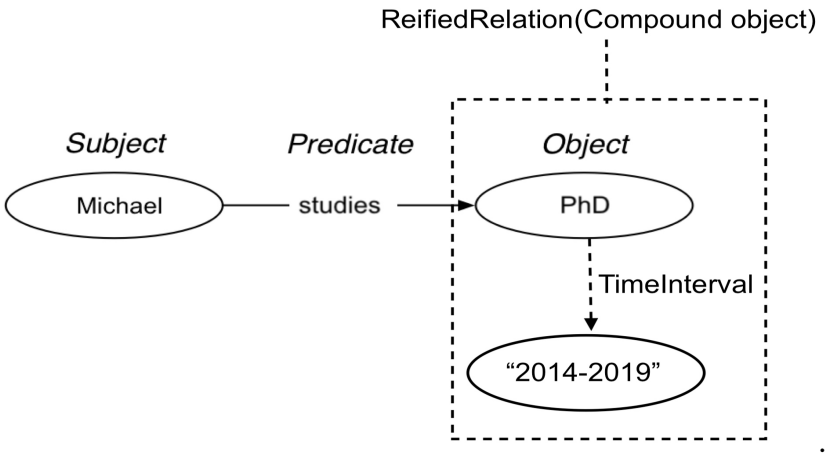


Figure 4: An example demonstrating RDF reification, where two triples combined into a single triple.

However, named graph overcome these shortcomings by assigning one URI into a collection of RDF statements to model additional context. As shown in Listing 2.1, RDF data can be serialised either as XML form [Swa04] or as non XML form called Terse RDF Triple Language (Turtle) [BB].

```
@prefix example: <http://example.com/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .

example:Michael rdf:type foaf:Person;
foaf:studies example:PhD.
```

Listing 2.1: An Example of an RDF statement represented in Turtle

While there are multiple languages for querying RDF such as RDF Data Query Language (RDQL) [Sea04], and RDF Query Language (RQL) [KAC+02], the W3C most recommended querying language of RDF is called SPARQL [PS+06]. SPARQL can be considered as “a set of specifications that provide languages and protocols to query and manipulate RDF content on the Web or in an RDF store” [HS13].

2.2.2 Resource Description Framework Schema (RDFS)

The Resource Description Framework Schema (RDFS) extends RDF by adding a standardized structure that should be used to describe the information about the Web resources, and it can be considered as the main description language of the RDF vocabulary [Bri04]. Although the RDFS structure is used to guide how domain classes and properties should be structured, it does not provide terminologies to describe the classes and properties of the Web resources in the form of metadata to make it machine understandable. RDFS concepts are based on four key terms, which are *rdfs:subClassOf, rdfs:subPropertyOf, rdfs:domain, rdfs:range*. In Figure 5 we can see an example of the RDFS representation using the *rdfs:subClassOf* concept.

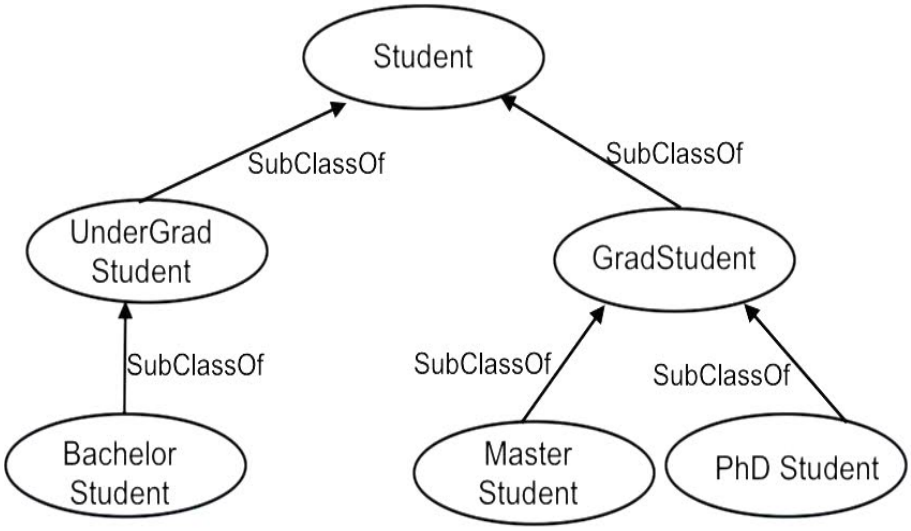


Figure 5: RDFS representation example.

2.2.3 Ontologies and Web Ontology Language (OWL)

Ontologies support the Semantic Web need for a representational vocabulary to represent terms, and the relationship between these terms across any domain of interest. This is a requirement to make the web resources not only understandable by humans, but also machine processable [KBM08]. In [SBF+98], the authors refer to an ontology to be a formal shared conceptualization in the form of concepts with constrains, that has to be accepted between some group. In Figure 6, we can see an example of an OWL ontology, and how the different ontology components are linked together.

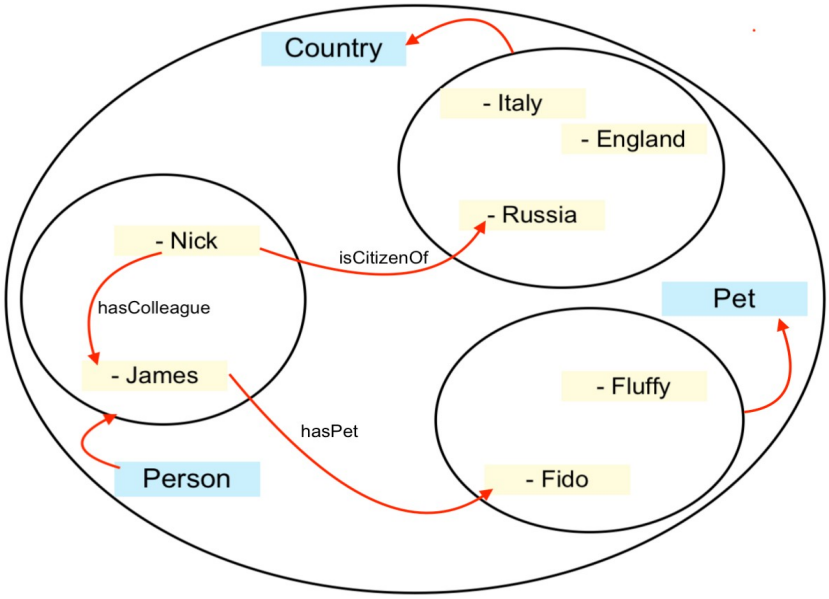


Figure 6: Example of an OWL ontology components.

There are different types of ontologies such as: Upper level ontologies, Domain ontologies, Task ontologies, Domain task ontologies, Method ontologies and Application ontologies [GFC06]. Each one of these ontolgies define the needed vocabulary for a certain task. The detailed description of each type is out of the scope of this thesis.

The Web Ontology Language (OWL) is an extension to RDFS to cover its limitations when it comes to different domains. OWL has more expressive semantics for different domains and become a W3C recommendation as a new ontology language with more coverage [MV+04]. As shown in Figure 7, the wider coverage comes from the OWL different levels of expressivity divided into three sublanguages as follows [AFV05]:

- OWL Lite – meant for users who need a classification hierarchy to express axioms with simple constraints.
- OWL DL – meant for users who need the full expressiveness, while keeping the good computational properties in terms of decidability.
- OWL Full – provides users with maximum expressiveness while not preserving the computational guarantees.

Generally, OWL introduced new language vocabulary to the Semantic Web, and although many of them were introduced by different communities, only few vocabularies have been adopted. One of the most widely known vocabularies is called Friend of A Friend (FOAF) [BM10], which is used to

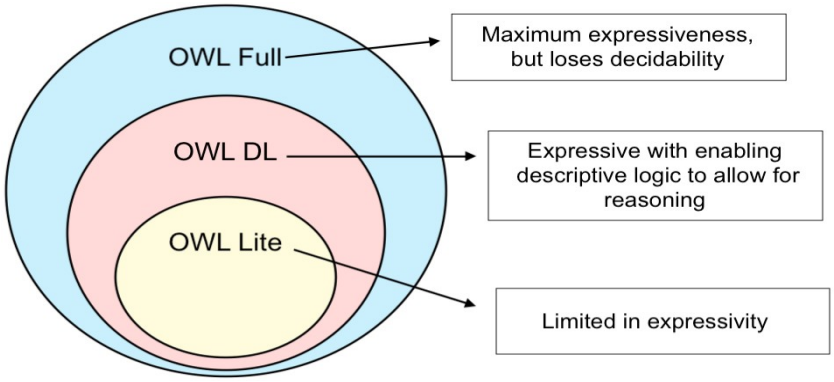


Figure 7: OWL sublanguages and their expressivity levels.

describe people and organizations. Since, it is out of the scope of this thesis to go into these vocabularies, we refer the readers to [Smi04].

Although OWL has been successful, it has some problems that needs to be addressed such as expressivity limitations in complex domains, missing an appropriate set of built-in data types, and difficulty in syntax parsing which derived the need to design the OWL APIs [GHM+08]. Thus, OWL2 [14] came as an extension for OWL, as it offers an extra syntactic layer, more properties and constructors, extended support for data types, simple meta modeling and extended annotations.

2.3 KNOWLEDGE AND LANGUAGE PROCESSING FOR THE SEMANTIC WEB

Natural Language Processing (NLP) is the automatic processing by machines to the text written in human languages. As seen from the previous section, the concept of the Semantic Web is mainly based on web resources that are described using URIs. Since, the resources in the Semantic web can be in the form of 1) entities, such as "Dog", 2) concepts, such as "Animal" or 3) relations, which describe the connection between each other such as "belongs to". For instance, NLP techniques range from text pre-processing tasks such as segmentation of sentences and tokenization of words, to complex tasks such as using NLP for semantic annotation, Named Entity Recognition and Classification (NERC), Relation Extraction (RE), and Event extraction.

14 OWL 2 Web Ontology Language: http://www.w3.org/TR/owl2-overview/

Recent years have seen an increasing deluge of heterogeneous unstructured text on the Web. This text which is coming from large collections of documents from different sources such as web pages, normative texts, research papers, articles, mailing lists and e-books is rapidly growing. This noticeable growth comes from the development of Web technologies. While semantic technologies and NLP techniques can serve an important role in mapping and linking of text to formal knowledge for efficient retrieval and management, unambiguous processing of natural language, in particular, the ability to capture complex knowledge statements is far from being solved. Information Extraction (IE) systems can serve an important role in extracting knowledge from text by identifying references to named entities, and recognize certain relationships between these entities. However, in order to extract more complex insights from the text, other NLP approaches are needed as well. One of the key approaches is producing quality linked data and ontologies for the Semantic Web from text, which can enhance the accuracy of the question answering, and machine reading applications. Knowledge and language processing can be combined to produce a pipeline that is capable of extracting structured knowledge, to generate quality triples for the Semantic Web through two main tasks [CRA16], 1) linguistic processing tasks, and 2) knowledge extraction tasks. The linguistic tasks are mainly used for the pre-processing of the text, and the knowledge extraction tasks can be used for processing the text into a form that can be used to extract the triples without losing semantics.

This section does not provide a detailed review of all the NLP and knowledge extraction techniques, rather it gives an overview for the approaches that we used through the thesis that are commonly used for serving Semantic Web applications.

2.3.1 Linguistic Processing Tasks

Early NLP approaches were limited to rule-based techniques, because the computational power were not sufficient enough to use machine learning or statistical methods. Later, as the computational power barrier started to break and rule-based systems were unable to resolve complex NLP problems, the NLP community widely adopted the use of statistical models in particular fields such as Part-of-Speech tagging (POS). Most recently, machine learning and neural networks become more popular in some specific

domains, however, they still have limitations in terms of the availability of sufficient training data-sets for different domains and tasks.

Each NLP approach has its own advantages and disadvantages, and both approaches can be combined for particular tasks. The two approaches can be described as follows [MBA16]:

Rule-Based approaches – are based on hand-coded rules that are mostly written by NLP specialists, as they require prior knowledge about the domain of the problem, the grammar, and syntax of the natural language. Although writing and changing the rules can be time consuming, it is much easier to understand and debug the results than in the machine learning approaches.

Machine Learning approaches – the adaptation of these approaches does not require a background in linguistics as they use statistics and linear algebra equations for complex computations. However, the challenge of having or annotating sufficient data-sets for each domain is sometimes a problem, especially after the rise of the deep learning approaches that require a huge amount of training data. Moreover, a re-annotation is needed when changes happen, and the debugging of errors in these approaches is a complex task.

2.3.1.1 *The Pre-processing Pipeline*

In Figure 8, we can see that the pre-processing components are typically the first tasks in the NLP pipeline, and it is usually used for the purposes of preparing and cleaning the text before sending it to the main NLP approaches. Text cleansing usually require splitting text into tokens (tokenization) to remove punctuation, extra spaces, text between brackets, unwanted text..etc. On the other hand, preparing the text might require splitting sentences into segments (segmentation), and assign a category to each token in the sentence (POS tagging).

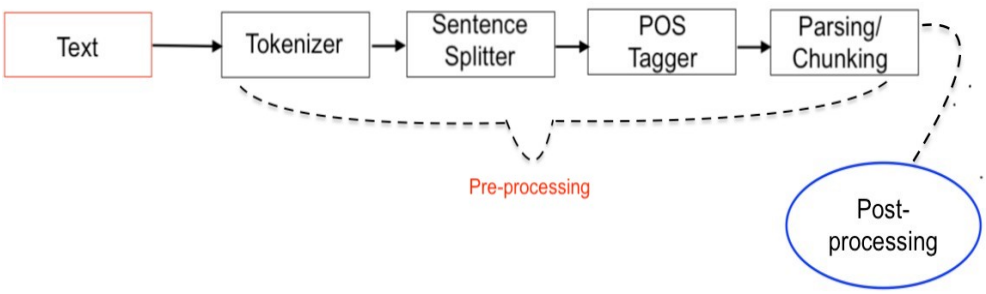


Figure 8: A typical pre-processing NLP pipeline.

2.3.1.2 *NLP Toolkits*

In [MBA16] the authors mentioned different toolkits that can be used for building NLP pipelines; in this section we will discuss four of the most popular toolkits as follows:

NLTK [BL04] – a python open-source NLP toolkit. The toolkit is widely used for building both low-level NLP pipelines for pre-processing text, and high-level NLP pipelines that are based on rules or machine learning techniques. We used NLTK in developing our approach.

GATE [CMB+02] – an open-source java-based NLP toolkit with a user interface. It has a variety of built-in tools and parsers for linguistic processing and another predefined pipelines such as ANNIE, which is a rule-based information extraction pipeline. The toolkit supports the integration of different components from another resources (e.g machine learning components) using a plugin technique.

Stanford CoreNLP [MSB+14] – similar to GATE to be an open-source java-based toolkit with mostly rule-based components, but varies in being an API based without an interface. The toolkit is intended to be used by users from different technical backgrounds via the command line, without the need to be a linguist. The tool is widely adopted due to its simplicity and accuracy of the results.

OpenNLP[15] – another open-source java-based NLP toolkit, but with mostly machine learning-based components. The toolkit supports the most common NLP tasks and can run via the command line.

2.3.1.3 *Tokenization*

Tokenization is an essential NLP task that is typically used for splitting a sentence into smaller units (tokens) using white spaces between them. As shown in Figure 9, the tokens refer to units such as words, numbers or symbols that are separated with white space. The tokenization process is crucial as most of the subsequent components in the NLP pipeline can not process the whole text without being tokenized. Hence, a low accuracy tokenizer can affect the results of the whole NLP pipeline. Tokenizers might include some extra features such as showing the type of the token, estimating the token length, and normalization of words and numbers e.g. date and time normalization.

15 http://opennlp.apache.org

Mostly all NLP toolkits has a tokenization component. NLTK has various tokenizers written in python and mostly built on regular expressions.

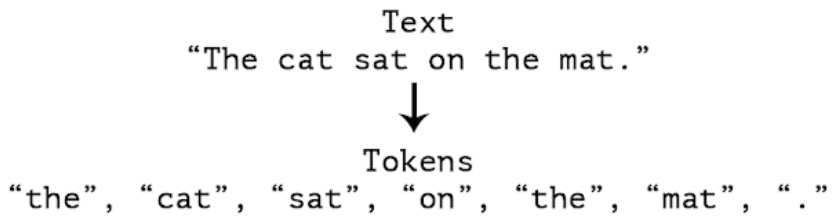


Figure 9: An example output of the tokenization process.

2.3.1.4 *Segmentation*

Sentence segmentation aka. sentence splitting is a common NLP task, usually used in standard NLP pipelines. In Figure 10, we can see that segmentation is used to split the text into separate sentences. The task is based on rules that detect punctuation such as full stops, question marks and determine if they indicate the end of a sentence. Some of these rules use a list of abbreviations such as to distinguish a full stop intended for an abbreviation, from full stops to end a sentence. Moreover, some splitters have rules implemented to identify and handle different sentences structures such as bullet lists, titles and addresses which can cause many errors in the segmentation task. The accuracy of sentence splitters can be improved by training them on different types of text such as wiki text and hashtags..etc.

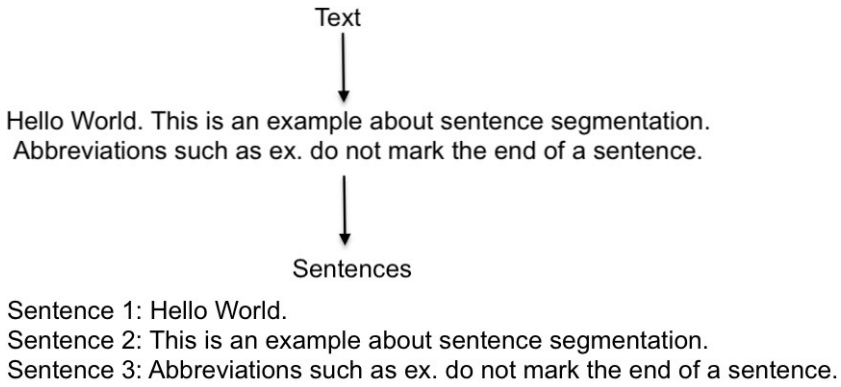


Figure 10: An example output of the segmentation process.

The sentence splitter in NLTK uses a language independent approach called Punkt [KS06] that can be trained on the text of the target language. In contrary to most of the sentence splitters, the approach does not use predefined abbreviation lists to define real sentence boundaries. The ap-

proach builds a statistical model using an unsupervised algorithm to detect abbreviations, collocations, and words that start sentences.

2.3.1.5 *POS Tagging*

Part-of-Speech (POS) tagging is an NLP module that reads text and assigns a part of speech to each word such as noun, verb, adjective, etc. Practically, taggers use more fine-grained POS tags to differentiate for instance between different verb tenses, or different types of numbers (singular/plural). In the English language, there are some POS tag sets that are commonly used such as Penn Treebank (PTB) [MMS93] and the Brown corpus [Fra79] tag set. Although some POS tagging approaches use rule based techniques, the most common approaches rely on machine learning e.g. Conditional Random Fields (CRFs), and probabilistic models such as Hidden Markov models (HMMs). Due to the ambiguity in natural languages, rule based techniques tend to be less accurate at assigning POS tags, as they do not take into account the context in which each word appears. For instance, one word can have different POS tags based on its context and definition (e.g. answer is a Noun in "What is the answer", but in the sentence "Answer this question", it is a verb).

NLTK has some of the most widely used taggers implemented in python that uses the PTB tagset [TMS03]. In Figure 11, we can see an example output from the default NLTK tagger called the PerceptronTagger[16], which can be called using the *pos_tag(tokens)* function. The PerceptronTagger is a pretrained machine learning model, that predicts the correct tag based on a given set of weighted features. The model is trained and tested on the Wall Street Journal corpus, and the weight adjustments are averaged over a number of iterations.

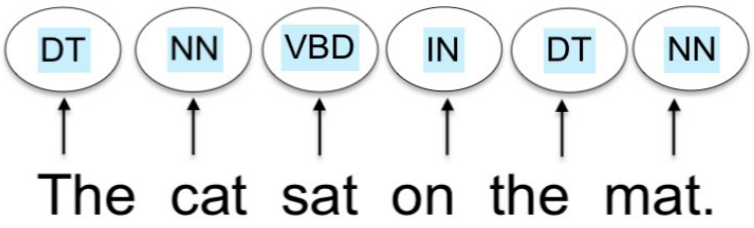


Figure 11: An example output of the POS tagging process using the PTB tagset[17].

16 https://explosion.ai/blog/part-of-speech-pos-tagger-in-python
17 *DT: Determiner, NN: Noun, singular or mass, VBD: Verb, past tense, IN: Preposition*

NLTK has also another widely used tagger called the Brill tagger [Bri92]. The Brill tagger is based on a set of transformation rules that are learned by training a supervised machine learning model to minimize the error. Although, the accuracy of these taggers is very high on generic text such as news articles as it is similar to the text on which the taggers have been trained on, the accuracy drops if the taggers process different types of text such as social media tweets. Generating unreliable POS tags is problematic, since the rest of the NLP pipeline modules rely on it to produce good results for different applications such as relation extraction, named entity recognition, and ontology learning.

2.3.1.6 *Syntactic Parsing*

Syntactic parsing is an NLP method for recognizing the sentences and discover the syntactic relationships between words in each sentence based on the language underlying grammar in the form of syntactic structures. Typically, these syntactic structures could be generated using one of the following approaches:

- Statistical Parsing, starts with a treebank where all sentences are syntactically annotated (parsed sentences), and the frequency of each parse tree is estimated. These parse trees are then used to extract a group of corresponding grammar rules with associated probabilities to define the relative frequency of each rule, which derives the most probable parse in the space against all candidate parses using deduction, such as the Stanford [KM03] statistical parser. As shown in Figure 12 we present the parse tree of the sentence *"The cat sat on the mat"*.

- Dependency Parsing, is a different modern way of parsing, where instead of identifying the relations between words, the dependency parsing captures the dependency between words. The parser works by predicting a sequence of transitions until it approaches the final configuration, these transitions are learned from gold sequences of transitions in a treebank, which are then used to train a multi-class classifier. One of the most widely used dependency parsers is the Stanford dependency parser[18][KM03] and the dependency broad-coverage parser Minipar [Lin03]. Other techniques include the use of

18 https://nlp.stanford.edu/software/lex-parser.shtml

neural networks instead of traditional classifiers such as the Neural Network Dependency Parser [CM14], where words, their POS tags and labels are represented as vectors using embeddings to feed forward a neural network that learns the right transitions of the parser. As shown in Figure 13 we present the dependency between the words for the previous example sentence.

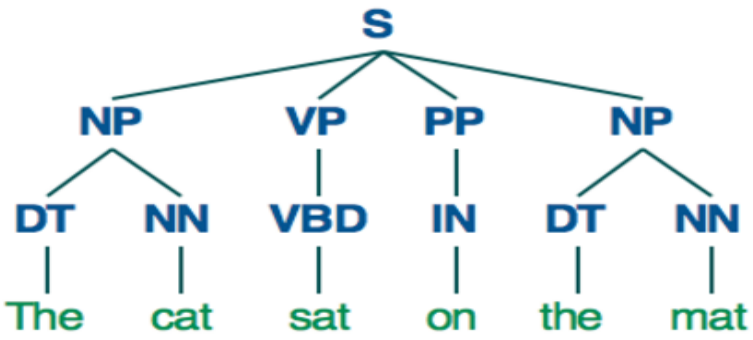


Figure 12: An example output of the syntactic parser.

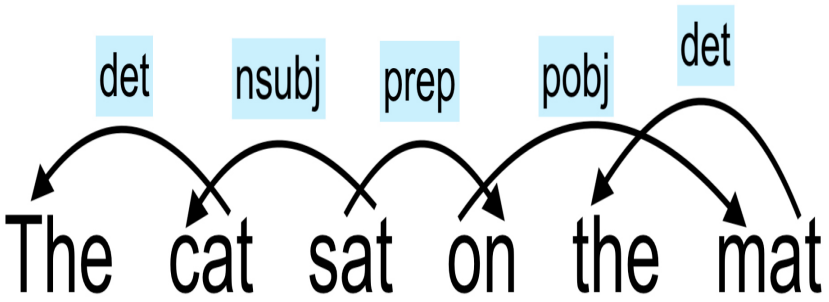


Figure 13: An example output of the dependency parser.

Currently, there are two types of parsing based on the requirements of each NLP task, 1) *Shallow parsing*, used in NLP applications when there is no need to parse all the syntactic information in a given piece of text. It is a non-recursive parsing and it does not look at the internal structures of the parse tree. An example is applying regular expressions on POS tags, and it can be used in different applications such as information extraction and text mining. 2) *Deep parsing*, is used when there is a need to build a more complex NLP application such as a dialogue system, where the parser needs to present the full syntactic structure of each sentence. The parsing algorithms are divided into two approaches according to the methodology of using the grammar rules as follows:

The top-down approach, begins with the highest level of the parse tree that contain the start symbol then moves down the parse tree to the words using the rewriting rules of a formal grammar. The approach operates in exponential time and sometimes need a special form of grammars known as LL1 grammar [PF11].

The bottom-up approach, begins with trees that connect with words, then moves up to the larger trees using the Shift-and-Reduce operations. An example of this approach is the CYK parser [Kas66] where the grammars should be represented in Chomsky Normal Form (CNF), and the LR parser [Knu65].

NLTK has different types of parsers available, that can be used based on the required NLP application. These parsers include 1) *a recursive descent parser*, uses the top-down approach to read ahead one character from the input stream and tries to match it with the implemented grammar, 2) *the shift-reduce parser*, which uses the bottom-up approach to iterate over the sentence and replacing its phrases with the matched grammar until it is reduced, 3) *the regex parser*, uses the output of a POS-tagger to iterate over a set of pre-defined regular expressions that represent the grammar, 4) *the dependency parser*, which is wrapper for the Stanford dependency parser [KM03].

2.3.1.7 *Chunking*

Chunking is another NLP process which aims to identify segments of text that are usually syntactically correlated such as Noun Phrases (NP) and Verb Phrases (VP), and label them as a multi-token sequence. A chunker can replace the parser when there is a need for a lightweight and shallow analysis over the text. Chunkers vary from one use-case to another, based on the definition of the relevant chunks of text. So, for example we can see from Figure 14 the Noun Phrase can be defined in the chunker as a consecutive sequence string that combines between one or more determiner(s), adjective(s), and noun(s). However, in another application, it can be defined to include an extra Prepositional Phrase or Relative Clause. On the other hand, Verb Phrase chunkers also differ from an application to another. So, for instance the chunker may recognize modal verbs, infinitive verbs and negative verbs based on the need. There are learning-based model chunkers [RM99], that can be used with sophisticated text in specific domains. These chunkers are more reliable if they are trained

on each specific domain text as they will get better results than rule-based chunkers. For instance, in the sentence *I bought the baby food* a rule based chunker will not be able to distinguish whether the phrase *baby food* is a single NP representing a compound noun or *the baby* and *food* are two independent NPs.

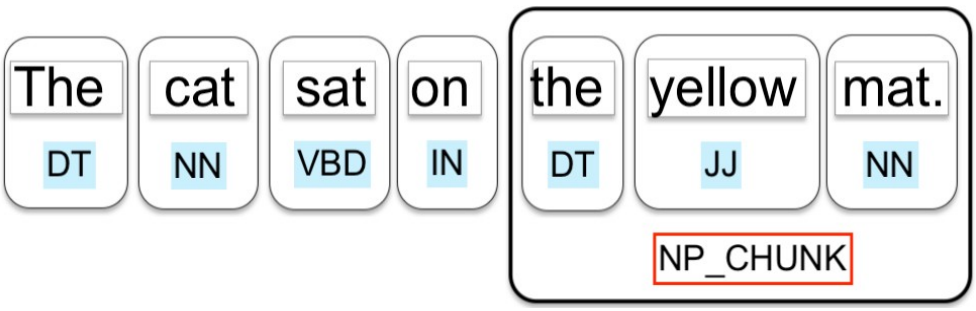


Figure 14: An example output of a Noun Phrase chunker.

NLTK does not have a readily implemented chunker; however, the toolkit allows each user to implement a chunker based on the application requirements. The chunkers in NLTK can be built using rules or machine learning models, with the aid of the NLP components from the toolkit such as POS taggers.

2.3.2 Knowledge Extraction Tasks

Unlike the low-level NLP tasks presented in the linguistic processing part, where the main focus was the syntactic analysis of a textual content, knowledge extraction tasks are more high-level as they focus on deriving semantics from text, that can benefit different applications such as ontology development. Although, ontologies are very critical for various Semantic Web applications, the manual creation of ontologies has a lot of challenges such as labor and cost. So, developing new and existing ontologies would not be possible without automating the ontology development process, taking into consideration the availability of quality data. In this section, we will discuss the most fundamental NLP tasks for automatically developing ontologies.

2.3.2.1 *Co-reference Resolution*

The Co-reference resolution task is considered with connecting the different mentions of the same entity within the text, or when it is referred to some entities in the text in different ways [MBA16]. This can occur due to the use

of pronouns in the corpus. For instance, in the two sentences *Hemoglobin is an iron rich protein. It carries oxygen from the lungs to the rest of the body*, the pronoun *it* refers to the noun *Hemoglobin*. The coreference is a common research problem in the NLP field, and it could be solved using different methods based on the complexity of the corpus. The complexity depends on the ways the same entity is mentioned, for example, it can be easy to recognize and connect *United Kingdom* with *UK* using a system that consists of gazetteer and hand-coded rules. The first method is the rule based approach taking into account the semantic information of each entity. The rule will filter the entities and determine the entities that have the highest probability to be coreferent. Finding the semantics of each entity can be done manually or using other resources such as Wordnet[19]. There are a range of co-reference tools that are based on rules for example ANNIE's orthomatcher with 95% accuracy on unseen text [BDM+02], Stanford Coref tool [RLR+10], and SANAPHOR [PTL+15] which improves the results by mapping the output from Stanford Coref to DBpedia ontology [BLK+09], in order to disambiguate mentions linked with different entities. Another approach for coreference resolution is measuring the distance between the two entities, and if the distance between them is less than a predefined cluster radius, then the two entities belong to the same cluster. Furthermore, decision trees are used to solve this problem as well. The decision tree checks if the two entities are coreferent or not, based on some predefined attributes such as gender, number, and semantic class.

2.3.2.2 *Named Entity Recognition and Classification (NERC)*

Named Entity Recognition (NER), for extracting named entities (NEs), such as persons, locations, or temporal expressions is a critical task in the information extraction process. The process can be divided into the recognition of entities, and mapping them to their relevant classes (Classification). Ambiguity can affect the accuracy of both the recognition task and the classification task, and hence the whole NERC process. For example, a challenge for the classification of entities is to select the right category for *Apple iphone* as it can be categorized as a *device* or a *company name*. Another example to show a challenge for the recognition of entities is to consider the word *May*, where it can be recognized either as a verb or a proper noun referring to the name of a month.

19 https://wordnet.princeton.edu/

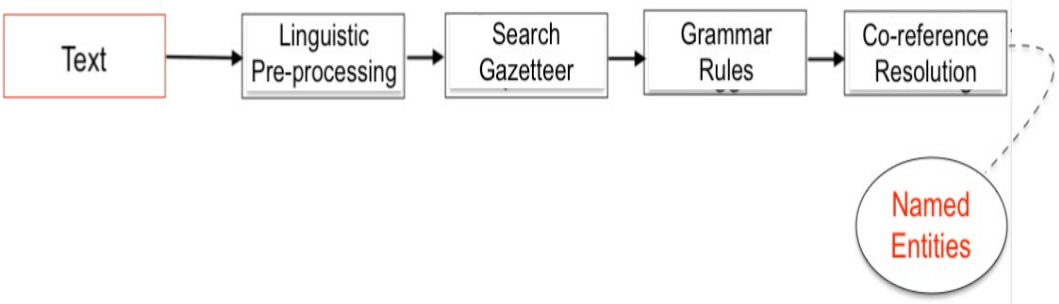


Figure 15: An example of a typical NER processing pipeline.

The NERC task can be applied using three different methods that can be divided as follows:

- *Rule-based approaches*, were widely used, as NERC modules that rely on gazetteers with regular expressions only, can not generate good results due to the ambiguity of the natural languages. For example, the word *apple* can refer to a fruit or an organization name. So as shown in Figure 15, a reliable NERC pipeline is usually divided into two parts, the first part is responsible for the linguistic pre-processing of the text by applying tokenization, segmentation and POS tagging modules. And, the second part is responsible for extracting entities using gazetteers to find common names of locations and people and finite state parsing grammars, followed by a co-reference resolution module to disambiguate these entities.

- *Machine learning (supervised learning) approaches*, use a learning process to train a model on annotated data samples to automatically adjust the weights of features involved in determining the NE types. A variety of supervised models can be used in the NERC process such as the Conditional Random Fields (CRFs) [ML03][FGM05], which uses a sequence labeling model to label tokens in a specific window, or StanfordNER[20], which is one of the most popular NERC frameworks that is based on CRFs. Other supervised models include Support Vector Machines (SVMs) [IK02][MMP03] and neural networks [LBS+16][SG15]. A Neural Network (NN) incorporates advantages over the traditional supervised approaches, and they become more resilient in complex domains. The newer NN deep learning approaches are able learn the latent features in domain corpus quite efficiently, without the need for explicit feature engineering. But to leverage this capability they require

20 https://nlp.stanford.edu/software/CRF-NER.shtml

vast amounts of labeled training data, which is often unavailable or labor intensive to develop.

- *Hybrid approaches*, that incorporates between the two previous approaches [CHR06][SB05].

2.3.2.3 *Entity Linking and Semantic Annotation*

Semantic annotation is the process of associating semantic data to content, and it is needed for indexing and semantic search. The semantic annotation process can be manual, semi-automatic and automatic [Dav13]. Part of the semantic annotation process is entity linking or entity disambiguation [RMD13], which is the next module in the NLP pipeline that usually comes after NERC. It is about annotating ambiguous entities in a document with a link to a unique entity in the ontology or Linked Open Data (LOD) resources such as DBpedia[21] [MJG+11][HYB+11], Freebase [ZSL+12], YAGO [SWL+12]. Typically, the Entity Linking task is challenging since all the LOD resources include several mentions of the same entity in the knowledge base. So, there is a candidate selection module to determine all the possible entries for a given entity in the text, then a reference disambiguation module, which uses a probabilistic model to select the target entity with the right URI, based on some contextual data and ontology information.

2.3.2.4 *Term Extraction*

In order to develop a new domain ontology, we need to recognize and extract the domain relevant entities and terms. Unlike, the Named Entity Recognition (NER) task, which focus on identifying generic entities across all domains such as persons and locations. The definition of terms can vary from one application to another and it is usually formed from noun phrases. However, in some applications noun phrases may vary, so chunkers may extract noun phrases that include prepositional phrases, while it is not extracted in other contexts. In Figure 16, we show the output of a term extractor used to process a sentence from the bio-medical domain. The task is to identify the terms that belong to the *protein* category and those that belong to the *DNA* category.

There are different approaches for term extraction, however, they almost follow the same pipeline except for the way they rank the candidate terms

21 https://wiki.dbpedia.org/

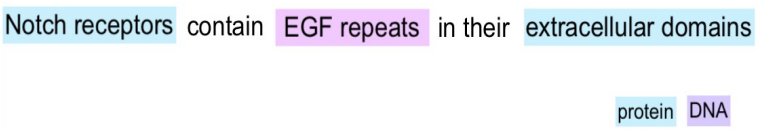


Figure 16: An example output of the term extraction task.

in each corpus. The initial module in the pipeline applies linguistic preprocessing tasks (tokenization, sentence splitting, POS-tagging, and NP chunking) to recognize and filter the candidate terms, and then various grammar rules are applied to restrict chunks such as noun phrases and stopwords. The last module applied is used to rank each term in the corpus and it can use two approaches as follows:

Distributional knowledge approaches [FA99], which typically relies on frequency-based measures to estimate how important a certain term is to a document in a corpus. One of the most commonly used measures is called term frequency-inverse document frequency - Tf/idf [Cho10], which is usually used in information retrieval and text mining. The Tf/idf weight is a statistical measure used to evaluate how important a word in a document is relative to collection of documents. This measure is estimated by counting the number of occurrences of a term in a document (term frequency), while weighting the same term relative to its rarity in the entire document collection. This is effective method in information retrieval and term extraction for filtering out stop words such as ”is”, ”of”, and ”that” which are highly frequent but of little importance.

Contextual knowledge approaches, uses contextual knowledge to produce weights to help with the term ranking. These weights can be generated from different kinds of knowledge by using one technique or more. The weights can be generated based on various techniques such as in the TRUCKS approach [MA00] where the weights are based on frequency of occurrence of the terms with other terms in its context, or the barrier word approach [Bou92] where weights are estimated from the syntactic knowledge based on boundary words before and after a candidate term. Finally, a weight will be assigned to each candidate term and a ranked list is generated. The evaluation of these techniques is subjective depending on the nature of the task. So, a good approach is to test and evaluate different techniques.

2.3.2.5 *Relation Extraction (RE)*

The relation extraction part comes after having the relevant terms and entities extracted from the text, to understand how these entities are related. The main aim of the relation extractor is to extract binary relations between entities. In Figure 17, we can see an example output of the relation extraction task. There are different approaches for relation extraction, some of them are developed to focus on relations between lexical items, and others focus on relations between concepts.

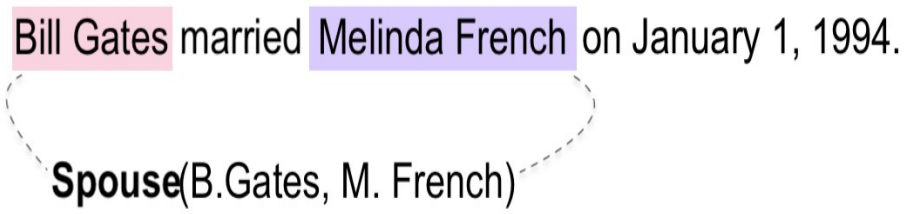
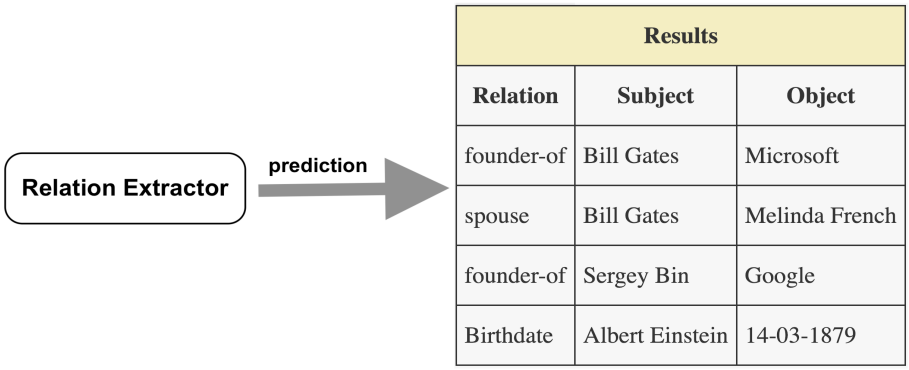


Figure 17: An example output of the relation extraction task.

As shown in Figure 18, the RE task can be considered as a knowledge base population task, where there are a set of relation types provided in a relation schema, that needs to be filled during the task. A typical pipeline of the RE module consists of three main tasks, where the first two are included in the NER task : 1) identifying the boundaries of the arguments, 2) identifying the arguments types, and 3) recognizing the relation types.



Results		
Relation	**Subject**	**Object**
founder-of	Bill Gates	Microsoft
spouse	Bill Gates	Melinda French
founder-of	Sergey Bin	Google
Birthdate	Albert Einstein	14-03-1879

Figure 18: An example output of the relation extraction task.

Generally, there are a wide diversity in the relation extraction techniques, and the most popular approaches could be divided as follows:

- **Rule-based approaches**, where rules can be developed for the RE task based on encoding the domain knowledge [Sod97] [SDN+07] [RRK+08]. This can be achieved by using pre-compiled gazetteers,

together with regular expressions to develop the grammar, or using cascaded phases of regular expressions over annotations with attribute value features pairs [CLR13].

- **Bootstrapping approaches**, are classified as semi-supervised techniques such as the KnowItAll [ECD+04] relation extractor, that has four main components: an extractor, a search engine, assessor, and a bootstrapping module. The extractor works by applying lexico-syntactic extraction rules known as Hearst patterns [Hea92], then applying relation extraction rules representing the relations such as *worksFor(employee, CompanyName)*, then apply the assessor to validate them. The extracted seed of relations are then used to train a probabilistic model and a classifier to select the best patterns, which further can be used to find a new set of instances.

- **Supervised approaches** [AVM15][MBS+09], follow the typical RE pipeline in the pre-processing, then use an annotated dataset labeled with positive and negative examples to extract features (e.g. n-gram of words, POS tags sequence, lemmas) and use them to train a model such as SVM, or recently relation embeddings (deep neural networks) [WBY+13]. These trained models could be used later to predict relations on unseen data. An example of a supervised RE is the *Stanford relation extractor* [22], trained to predict relations such as *Live_In, Located_In, OrgBased_In, Work_For*.

- **Unsupervised approaches**, also called Open Information Extraction (Open IE) systems. These systems work on unlabeled data of any domain, to extract a group of relations from text using clustering techniques. *TextRunner* [YCB+07] is the first released OpenIE system. It utilizes a POS tagger and NP chunker to label the text input with POS tags and NP chunks, then heuristically uses hand-written rules to label a subset of the corpus as positive or negative samples. This labeled data is used later to train a supervised CRF model, in order to extract the relations from unseen data based on the POS tags.

  ReVerb [FSE11], was the successor of TextRunner which tended to suffer from two main drawbacks. The first problem, is that some relations did not comprise significant (incoherent) interpretation, such as the sentence *the book includes good information and omits reliable*

22 https://nlp.stanford.edu/software/relationExtractor.html

resources, results in the incoherent relation *includes omits*. The second problem is uninformative extractions, such as the sentence *Michael made a transaction on the account*, where the output from Textrunner will be *Michael, made, a transaction* instead of *Michael, made a transaction on, the account*. Hence, Reverb offers enhancements to prevent these drawbacks by adding more syntactic and lexical constraints, that are applied before extracting relation phrases, such as checking that a relation phrase has a verb, or a verb followed by a preposition or noun. These constraints are implemented using regular expressions to guarantee the extraction of meaningful and coherent relations.

2.3.3 Challenges

2.3.3.1 *Ambiguity*

Humans can interpret ambiguity in the language correctly by relying on world knowledge. On the other hand, machines can not use world knowledge, so ambiguity in NLP means that text can be interpreted syntactically or semantically by computers in many ways. Syntactic ambiguity means that one sentence might have many parse trees, due to different reasons such as conjunctions, noun group structures, or phrase attachment (i.e. adjective/preposition phrase). An example of ambiguity in prepositional phrase attachment is the sentence *Michael ate a salad with tomato*, where *with tomato* can be attached either to the noun *salad* or to the verb *ate*. Semantic ambiguity means that a sentence can be interpreted in many ways. An example is the sentence *John drove his car, and so did Sam*, which could be interpreted either as *Sam drove John's car* or *Sam drove his own car*. Although these ambiguities are resolved in NLP with a certain accuracy, using either statistical models or rule based techniques, they still represent a big challenge as they can not be resolved completely in all contexts and domains with high accuracy.

2.3.3.2 *Performance*

Generally, ambiguity is not the only challenge in NLP. The performance of linguistic processing approaches is related to their accuracy and usability in terms of their precision and recall scores for different applications. For instance, the performance of the NER task depends on different factors such as, the domain of the text that the NER was trained on [ADB17], and the

types of entities included in the NER Gazetteer. So for example, performing an NER task on a social media corpora is more challenging to perform than on a newswire corpora [MBA16]. In addition, the availability of high quality annotated data is another reason that can affect the performance of the NER task [ADB17]. On the other side, ontology development is a big challenge for the development of Semantic Web applications as they are affected by domain and text genres changes. So, most of the NLP tasks are highly accurate only when they are focused on particular domains and applications. As shown in Figure 19, these challenges lead to adjust a trade-off between generality vs. specificity of domain, complexity of the task, and performance level. For instance, the performance is higher for bag of words and entities extraction, however it starts to degrade for more complex tasks with higher ambiguity such as relations and events extraction. Furthermore, for these complex tasks the more domain specific the higher will be the performance of the NLP modules.

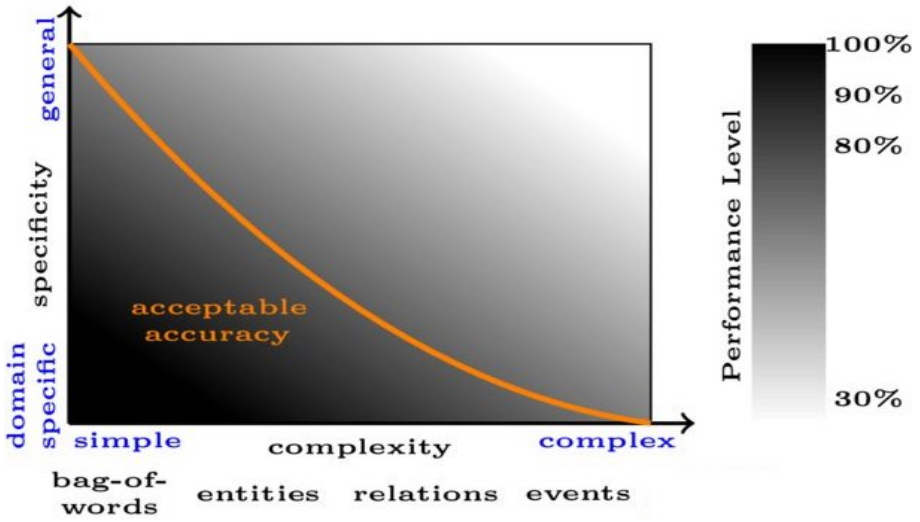


Figure 19: Performance trade-off between generality vs. complexity of some NLP tasks [Cuno5].

2.4 KNOWLEDGE EXTRACTION TOOLS

2.4.1 TEXT2ONTO

Text2Onto [CV05], is an NLP framework developed to learn ontologies from textual resources. Its architecture has various modules for term extraction,

relation extraction and taxonomy construction. In order to extract synonyms, the architecture uses basic linguistic processing such as tokenization and lemmatization, to extract patterns that are used to train a machine learning model. It is considered to be the successor of TextToOnto [MS04], which lacks the flexibility of combining different algorithms and does not facilitate any interactions with the user. These shortcomings are approached by introducing three new features in Text2Onto: 1) the independence of the knowledge representation language after introducing the Probabilistic Ontology Model (POM), which stores the learned knowledge in the form of primitives, 2) providing a user interactive environment with sophisticated visualizations, which allowed to calculate confidence for learned objects, 3) added functionalities to automatically allow updates in the learned knowledge with respect to changes in the corpus without processing from scratch, which helps the users follow the ontology development. The system gives the user the flexibility to apply the desired algorithms and models in the GATE framework.

2.4.2 SPRAT

SPRAT (Semantic Pattern Recognition and Annotation Tool) [MFP09] is another ontology generation and population system, that enables the user to create a new ontology or adjust an already existing ontology. The system is developed for knowledge base construction by linking text to ontologies based on lexico-syntactic patterns. Its architecture combines different NLP modules ranging from named entity recognition, ontology-based information extraction and relation extraction. However, it differs from Text2Onto as it does not rely on statistical clustering for relation extraction, and it uses more lexico-syntactic patterns for extracting entities and relations from text. The system accuracy can be improved by refining the patterns and linking it with another NLP resources, such as WordNet conceptual-semantic categories and lexical relations.

2.4.3 FRED

FRED [DGP+13] is an online web-service that differs from previous tools by providing the user with a ready to use ontologies and linked data, instead of focusing on extracting relevant ontology key terms. The approach uses

Discourse Representation Theory (DRT) [Kam81], linguistic frame semantics [NGP11], and Ontology Design Patterns (ODP) [GP09], to convert text into linked-data that can be used to populate ontologies. The tool relies on a deep parser called Boxer [Bos08], that uses event semantics to generate formal semantic representations (DRS constructs) from natural language sentences. The semantic data from Verbnet or Framenet frames are combined together with ODP best-practice [MSB+14], to develop a collection of ad-hoc transformation rules, that can map the DRS into OWL/RDF triples.

2.4.4 LODIFIER

Lodifier [APR12] is based on NLP techniques such as Named Entity Recognition (NER), Word Sense Disambiguation (WSD) and deep semantic analysis to translate an open domain unstructured text into Linked data on the cloud. The system works by employing an NER system called Wikifier [MW08] to extract entities, and map them into DBPedia URIs. Then, a statistical parser called C&C [CCB07] is used to extract the relations between these entities. After that, the deep semantic parser Boxer [CCB07] is once again leveraged to generate Discourse Representation Structures [KR13]. Finally, the DRS output is transformed into triples to create the RDF graph. Although, the system provides more benefits of additional information over shallow approaches, the architecture does not recover the errors, as it treats the modules independently.

2.5 CNLS BASED APPLICATIONS

Although the previous tools are helpful in the ontology learning and population process, they are only assisting ontology engineers and domain expert users to model OWL ontologies. The tools are based on Ontology Design Patterns (ODPs), which are a group of lexico syntactic patterns which map a natural language input into ODPs. On the other side, non-expert users have barriers using these tools and frameworks, because ontology languages are not readable and understandable. An alternative bottom-up approach is to rely on Controlled Natural Languages (CNLs), which can assist non-expert users by modelling a text input into a readable ontology syntax using syntactic rules through a Natural Language Interface. Examples of these

CNLs include but not limited to Attempto Controlled English (ACE) [KF07], Sydney OWL Syntax [CSM+07], and Rabbit [DHK+07]. We will extensively discuss all the details about CNLs in the next chapter 3. In Section 2.5.1 and Section 2.5.2, we will present some literature approaches that used CNLs for ontology modelling and knowledge extraction purposes.

2.5.1 KANT Project

In [MN95] the authors developed a controlled language rewriting system called KANT. The system uses a strictly defined controlled language with a formally specified syntax to guarantee automatic machine translation with high quality. The KANT controlled language does not limit the size of the vocabulary, and only rules out complex lexical and grammatical constructions. The implementation of the system combines a) constraints on the lexicon to reduce lexical ambiguity and complexity, b) constraints on the grammar to reduce parsing complexity, c) using standardized General Markup Language (SGML), to support the definition of domain terminology and phrasal constructions. The KANT rewriting engine utilizes a prescriptive rewriting approach, where any sentence that is not predefined in the controlled language grammar will not be parsed by the grammar checker, and must be rewritten. The author can resolve the parsing failures by rewriting the sentence and make sure it passes the grammar checker. The system has been deployed by Caterpillar Inc.[23] to guarantee high quality authoring and translation of their complex technical documents.

2.5.2 ITA Project

The authors of [MBP+12] and [XPK+12] apply a CNL-based approach for extracting entities and events in a military domain dataset [GHR11]. Their aim is to build a system that supports knowledge sharing and decision making across different groups from different nations without conversion of the information to a technical format (formal notation). The dataset is centered around entities such as people, locations and events mentioned in a set of reports and messages of real military scenarios. The approach uses Controlled English (CE) [Sow04] as the target representation language to express the extracted information, as well as for expressing queries and

23 https://www.caterpillar.com/

inferences on the extracted information, and for configuring the extraction process as such. The system has a pre-processing module to convert the messages into the form natural language sentences. The Stanford parser [24] is applied on each sentence to produce a syntactic parse tree. The parse tree is then converted into a CE representation. After that, the entity extractor uses the CE grammar rules from the CE store to extract attributes and relations, and to generate CE sentences describing the situation.

2.6 EVALUATION METRICS

Many natural language analysis tasks can be treated as binary classification tasks, whereby a classifier labels samples in a binary decision task as either a positive or a negative sample. The result of this classification process can be represented in a matrix called confusion matrix [DG06]. As shown in Table 1, this matrix includes four categories as follows: 1) True Positives (TP) for the positive correctly classified samples, 2) False Positives (FP) for the negative incorrectly classified samples, 3) True Negatives (TN) for the negative correctly classified samples, 4) False Negatives (FN) for the positive incorrectly classified samples.

For example, let us assume we would like to extract the confusion matrix elements after applying the entity recognition task on the sentence *Satya Nadella is the CEO of Microsoft Corp*. The right output should be the two entities *Satya Nadella* and *Microsoft Corp.*. Supposing the actual extraction has three entities as follows *Satya*, *CEO*, and *Microsoft Corp*. In this case True Positives = 1 (*Microsoft Corp.*, the only exact match), False Positives =2 (*CEO*, and *Satya*, which is not an exact match), False Negatives = 1 (*Satya Nadella*).

Table 1: The Confusion Matrix.

	actual positive	actual negative
Predicted positive	*TP*	*FP*
Predicted negative	*FN*	*TN*

Accuracy is an evaluation metric that represents the ratio between the number of correct predictions and the total number of input samples. However, in order to empirically validate a new algorithm, the authors in [PFK+98] show that it is insufficient to present the results based on ac-

24 http://nlp.stanford.edu/software/lex-parser.shtml

curacy presented in Equation 1. This is because the accuracy metric does not reflect the real results in certain cases, such as when the ratio between the number of samples belonging to each class is big, and when thresholds are used to optimize performance. Hence, it is misleading to present results based on the accuracy estimation only.

$$Accuracy = \frac{TP + TN}{TP + FN + TN + FP} \tag{1}$$

So, they presented another validation criteria called the Receiver Operator Characteristic (ROC) curves that can be used instead of accuracy to clearly evaluate the performance of an algorithm. The ROC curve shows the trade-off between sensitivity and specificity (any increase in sensitivity will be accompanied by a decrease in specificity). As shown in Figure 20, an ROC curve is a two dimensional graph between:

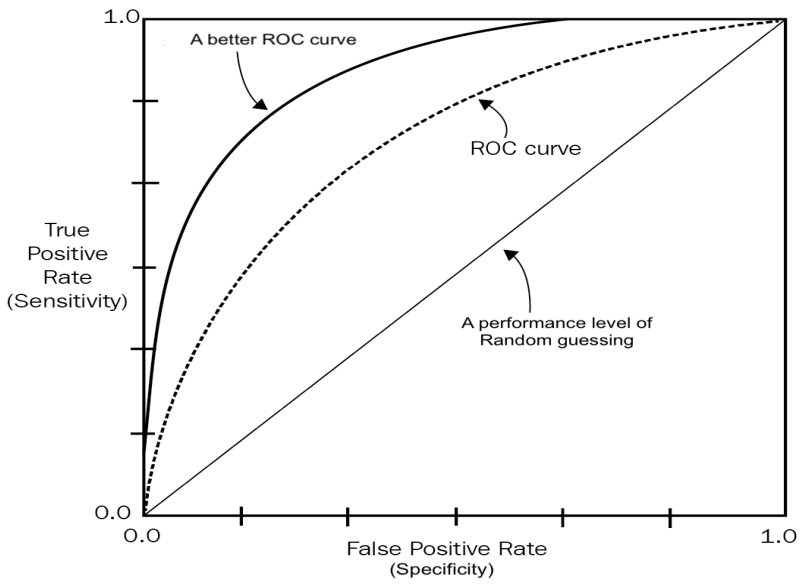


Figure 20: The ROC curve.

1) Specificity (also called false positive rate defined in equation 2), which is the proportion of actual negatives that are correctly identified, plotted on the X-axis,

$$Specificity = \frac{TN}{TN + FP} \tag{2}$$

and 2) Sensitivity (also called true positive rate or the recall defined in equation 3), which is the proportion of actual positives that are correctly identified, plotted on the Y-axis,

$$Recall(sensitivity) = \frac{TP}{TP + FN} \tag{3}$$

Later, it was discovered that ROC curves presented drawbacks especially when the skew size of the class distribution is large, the authors in [DH04] presented cost curves as a solution to this drawback. The authors of [MMS99] recommended Precision-Recall (PR) curves as an alternative to ROC curves. This is because when there is a data imbalance ROC curves will be always the same regardless of any changes in the baseline probability, however the PR curves are more useful for problems where the positive class is more dominating than the negative class.

2.6.1 Precision and Recall

The confusion matrix can be used to define the evaluation metrics known as precision in equation 4 (also known as positive prediction value) that can be seen as measure of quality, and recall in equation 3, that can be seen as a measure of quantity. The precision measures how many samples classified as positive are correct. While recall measures how many samples that should have been retrieved were actually selected.

$$Precision = \frac{TP}{TP + FP} \tag{4}$$

In the PR space, the precision will be plotted on the x-axis and recall on the y-axis. As shown in equation 5, in order to have a single measurement of the system, we can combine both metrics to generate the F1 Score (or f-measure). The F1 score can be seen as the average/mean of the two metrics when they are close.

$$F1 = 2 \times \frac{precision \times recall}{precision + recall} \tag{5}$$

2.7 OTHER RELATED WORK

Although all the technologies based on ontologies and Linked data were introduced as the core of the Semantic Web standardization[25], they did not lead to the original vision of the Semantic Web. This is because lack of trust in using Linked data, lack of public data, and annotations are driven for specific applications. Hence, recently the focus has been diverted into less standardization and more into developing mature technologies [Mik17]. These technologies include open source triple stores and graph databases (e.g. Neo4j[26], Virtuoso[27], and AllegroGraph[28]), and graph extensions to databases (e.g. Oracle RDF Semantic graph[29], and Microsoft Azure Cosmos [30]). Moreover, new research fields have risen such as Semantic search [FHS+16][LUM06], Knowledge graphs, which in return led to improvements in other domains such as text retrieval [DDA14].

On the other side, the rule-based systems might be outdated in the research and academic domains due to the rise of machine learning and deep learning techniques. However, recent research [CLR13] argues that this is not the case in the commercial world and they totally justified with large companies such as IBM, SAP, and Microsoft, which tend to rely on rule-based systems more than machine learning systems. Furthermore, while machine learning approaches are more adaptable and their usage reduces manual effort, they have other drawbacks that might be disregarded in the NLP community. These drawbacks include the need for labeled data, retraining, complexity, and expertise to use or maintain. Hence, arguably the academic NLP community needs to review its perception of the rule based paradigm as been in fact a widely used and non-obsolete technology, which could bridge the gap between the rule-based systems used in industry, and the machine learning approaches that are more used in the academic domain. An example of this might include an initiative to standardize rule languages and data models for different NLP tasks, such as the early attempt of developing Common Pattern Specification Language (CPSL) [AO98], an instance of which was realised as the JAPE language in the GATE NLP system [CMB11]. Another benefit of a standardised rule language would be

25 https://www.w3.org/2013/data/
26 https://neo4j.com/
27 https://virtuoso.openlinksw.com/
28 https://franz.com/agraph/allegrograph/
29 https://www.oracle.com/technetwork/database-options/spatialandgraph
30 https://docs.microsoft.com/en-us/azure/cosmos-db/graph-introduction

its application in investigating the output of machine learning models or defining learning problems for rule induction similar to Rule-based machine learning (RBML) [BGM+11][WI95].

2.8 CHAPTER SUMMARY

In this chapter we reviewed the fundamental technologies underlying this thesis. Section 2.2, presented a brief overview of the Semantic Web, and Semantic Web related technologies known as Ontologies and Linked Data. We have discussed briefly, Resource Description Framework (RDF), Resource Description Framework Schema (RDFS), and Web Ontology Language (OWL). In section 2.3, we provided an overview of the NLP technologies and techniques with respect to their usage in building Semantic Web applications. These technologies include tasks for Linguistic Processing, and NLP Tasks for Knowledge Extraction and Ontology Development. Also, we discussed the related challenges for these tasks in the form of ambiguity and performance.

In Section 2.4, we reviewed the key recent work with respect to knowledge extraction tools for the Semantic Web, where we presented the common used frameworks such as Text2Onto, SPRAT, FRED and Lodifier. Then, we discussed in Section 2.5 some CNLs based applications, as an alternative solution to the previous approaches for tackling the challenges between these technologies and non-expert users. Finally, in Section 2.6, we presented some evaluation metrics used to evaluate the performance of the previous approaches.

3 ANALYSIS OF HUMAN & MACHINE ORIENTED CONTROLLED NATURAL LANGUAGES

The goal of this Chapter is to discuss the state-of-the-art CNLs and their respective tools. Section 3.2, presents a background about history of the different types of CNLs, Section 3.3, provides a comparative analysis of CNLs for the Semantic Web within the context of ontology authoring listed in chronological order of publication. In Section 3.4, tools that use CNLs to perform ontology engineering and querying tasks in the Semantic Web domain are presented. Section 3.5, provides a detailed comparison and analysis of user evaluations for the CNLs. Section 3.6, shows the comparative analysis of the CNL based tools, Section 3.7 presents various human-oriented CNLs from the literature, and finally Section 3.9 offers analytic conclusions with respect to current and future trends with respect to CNLs.
References: Parts of this chapter are based on our publications [SD14] and [SD17].

3.1 INTRODUCTION

Controlled Natural Languages (CNLs) for knowledge creation and management offer an attractive alternative for non-expert users wishing to develop small to medium sized ontologies. Controlled Natural Languages are defined as "subsets of natural language whose grammars and dictionaries have been restricted in order to reduce or eliminate both ambiguity and complexity"[ST04].

In this chapter we will present a comparative review on different areas that will guide us as well as other users to select the appropriate CNL or tool

based on the requirements of the targeted use-case. The analysis process will be based on the following criteria:

- Studying and comparing different CNLs and tools in terms of their parsing mechanism, the goal of developing each CNL and its supported domains, what are the most important features that characterize each CNL and what differentiates it from others, and presenting the advantages and limitations of each CNL.
- A comparison of user evaluations performed on different CNLs in terms of metrics such as the evaluation goal, the number of participants, the evaluation results, whether the CNL is compared to baseline CNLs/tools or whether a CNL is lacking a user evaluations.

3.2 HISTORICAL BACKGROUND OF CNLS

The main goal of CNL was found during the 1930s, to make the English language usable by many societies and persons worldwide [Sch10]. Later, large technology companies such as IBM, and Xerox gave particular interest to CNLs especially in the area of technical documentation and their translations [Sch10][AS92][OBr03]. Typically, Controlled Natural Languages (CNLs) are divided into two broad categories [Huj98], and in this chapter we will be reviewing both types of CNLs as follows:(1) *human-oriented CNLs*, that simplify and improve readability for non-native speakers and second language learners within complex documents (e.g. technical documentation) and facilitate human-human communication for particular objectives, and (2) *machine-oriented CNLs* that was originally developed for improving the translation of technical documents [NM00], and to support knowledge processing and representation especially in the Semantic Web [SKC+08], where CNLs are used to develop user friendly means for ontology authoring, so that end-users can represent formal data within the context of the semantic web, without the need of formal training. Since, the knowledge representation language of the Semantic Web is the Web Ontology Language (OWL) which is based on fragments of the first order logic [WMS04]. Users with no formal background will find it difficult to interact with the ontology engineering process without a special training. This formal barrier may restrict the adoption if not ultimately result in rejection of the semantic technologies [Sma08]. As humans are already familiar with natural languages,

it will be required to develop natural language interfaces to facilitate the interaction of the end users with the semantic web resources while minimizing the need for training. However, it will be very difficult to interpret all the natural language expressions due to the ambiguity of the natural language, so using a controlled or restricted variants of natural language rather than the full or unrestricted version is a possible solution.For instance, the mathematical symbols used in logic to represent ontologies is very difficult to be understood by non-logicians, so another alternative syntax called Manchester OWL Syntax (MOS) [HDG+06] was proposed to replace the formal logic symbols with keywords such as 'some', 'only' and 'not. The syntax is currently being used in tools such as **Protégé-OWL** [KFN+04].

3.3 SEMANTIC WEB RELATED CNLS

PENG [Sch02] is a CNL designed to allow writing unambiguous and precise specifications using a restricted grammar and a domain specific lexicon, which consists of predefined function words, illegal words, and user-defined content words. The content words can be added or modified using the integrated lexical editor. PENG text is easy to be translated into first order logic (FOL) using discourse representation structures (DRS). Also, it uses a sophisticated look-ahead editor to facilitate writing by non linguists without the need to know the grammar rules of the CNL explicitly. This can be achieved by showing what kind of syntactic structures can be used after each word entered by the user. The restricted grammar defines the structure of joining PENG sentences into complex sentences using coordinators and subordinators. The structure is restricted using determiner, pronominal modifier, nominal head, post nominal modifier, negation, verbal head, compliment, adjunct, phrase-level coordination, phrasal level subordination, sentence level coordination, sentence level subordination and constructors. Furthermore, for making it easy to read by non-specialists, PENG avoids ambiguity by applying a set of interpretation principles. PENG ASP [GS17] is an authoring tool for writing PENG sentences with the help of a web-based predictive text editor, that guides the author during the writing process by showing various options for rational sentence completions.

ClearTalk (CT) [Sku03] is a knowledge formulation language for the semantic web that offers a flexible degree of formality with an adequate

expressiveness [Kuh14]. Documents could be automatically translated from CT to knowledge representation structure. It can be used by an English speaker who does not understand formalisms with almost no training. The grammar includes 100 rules discussed in [Sku03] that need to be be learnt before start writing. CT has syntactic restrictions where for instance, basic sentences have the general form of subject, predicate, complement and modifier phrases. The authors implemented a search engine using dtSearch[1] as a basic engine to find things a normal search engine cannot find, for instance, show all occurrences of a certain term and window size that are a) ordered in a useful way, b) summarized by showing levels only, and c)seeking certain words, while integrating WordNet (the engine can require a second word to be present). CT knowledge base consists of 3 modules as follows; linearly organized documents; the lexicon containing all terms and their lexical information; the topic pages. All are indexed by the search engine that allows to search facts.

PENG-D [ST04] is an extension of PENG with all its components plus the support for ontology construction as a start for language layering purposes. Since the meta-modelling architecture of the semantic web is not standardized, it was difficult to layer ontology and rule languages on top of RDFS as some elements in the model will have dual roles in the RDF specification. On the other hand, OWL was not the ideal solution, because of the semantic layering incompatibility of the standardized Description Logic (DL) that is based on first order model theory, with the semantics of RDFS based on non-standard model theory. The interoperability between OWL Lite and Horn Logic is used to create a paradigm called Description Logic Programs (DLP) [GHV+03]. PENG-D has expressivity and formal properties equivalent to DLP, that provides a pathway to layer more expressive constructions on top of it. The authors claim that PENG-D was easy to write with the help of the look-ahead text editor, easier to read than RDF based notations, and easier to be translated into corresponding machine processable format.

Sydney OWL Syntax (SOS) [CSM+07] came after Manchester OWL Syntax [HDG+06] and PENG to overcome their limitations. While Manchester OWL syntax have been well received by non-logicians and is the syntax for Protégé-OWL, it was limited by less focus expressions for property and individual. Also, PENG grammar did not support bidirectionality from PENG sentences to FOL and vice versa. The scope of SOS is to be compati-

1 http://www.dtsearch.com/

ble with OWL 1.1 for expressing ontologies, and to form translations, and to cover anything that can be expressed using OWL 1.1. Furthermore, to provide a two translation between SOS and OWL 1.1 syntax without any information loss. However, the bidirectionality involves using a context sensitive grammar and generating the output during the parsing process. The authors considered assisting the users writing ontologies using SOS by adding an interactive functionality of the lookahead editor. The general issue for designing a CNL is that the authors of the language have to decide how the language will support naturalness, complex constructs, language support for user defined terms and definitions. The authors of SOS choose to be more closer to OWL 1.1 syntax while trying to observe a trade-off to make the expressions as natural as possible. However, this results in SOS having more statements.

Attempto Controlled English[2] (ACE) [FS96] is a well known CNL involving translation into First Order Logic (FOL). The language can be considered a restricted version of standard English developed for the purposes of knowledge representation by interpreting into various formal languages that considered to be machine readable formats based on first-order logic. The language was introduced in 1996 by Fuchs and Schwitter [FS96] followed by many research publications [Kuh10], that either develop and expand the language such as developing the ACE Web Ontology Language known as ACE OWL, a sublanguage of ACE, is a means of writing formal, simultaneously human-and-machine-readable summaries of scientific papers [KF06a] [Kuh06], or implement applications that are based on ACE such as being the interface language for a first-order reasoner [FS03], serving as a query language for the Semantic Web [BKF+04], utilized by [FS07] to build an application for the partial annotation of Webpages, and its usage in the biomedical domain to generate summaries [KRF+06]. The ACE CNL was developed to be domain independent, and hence the ACE grammar is considered to be one of the most expressive CNLs grammar as it is based on a broad grammar coverage handling different grammar complexity levels such as relative clauses, negation, conditional sentences [3]. It is considered one of the most mature CNLs in the literature, for the following reasons: 1) it is a widely adopted CNL with an expressive grammar, 2) the language can be automatically mapped into different formal languages such as Discourse Representation Structures (variant of first order logic)[FKK08a] and subse-

2 http://www.ifi.unizh.ch/atempto/

3 http://attempto.ifi.uzh.ch/site/docs/ace/6.5/ace_constructionrules.html

quently a subset of ACE can be converted to the Web Ontology Language (OWL). 3) it also provides access to different tools and resources to use the language such as the ACE parsing engine - APE[4], and the first-order reasoner RACE [Fuc16], which can check the consistency of ACE axioms.

RABBIT Controlled English [HJD08] is another CNL that is developed by the United Kingdom national mapping agency, to extend the Controlled Language for Ontology Editing named CLOnE [FTB+07]. The main goal is to facilitate the knowledge capture and representation for different domain experts from different backgrounds. The language is more expressive than CLONE as it comes with extended grammar that aids users in ontology authoring tasks. The language is similar to CLONE as it is developed using the GATE framework [CMB+02]. The Rabbit language support three types of sentences known as declarations, axioms, and import statements. Furthermore, it has a feature for supporting the ability to reference different classes in a in more than one ontology [HJD08]. The detailed description of the language expressiveness represented by syntactic patterns and their respective ontology mappings is presented in [DDH+08].

OWL Simplified English is a finite state language for ontology editing [Pow12]. The argument for the finite state approach is that the majority of the OWL expressions created by ontology developers were invariably right branching and hence could be recognised by a finite state grammar. Based on previous studies of ontology corpora, the authors show how the individuals, classes and properties tend to have distinct Part Of Speech (POS) tags. Individuals or instances tend to be either proper nouns, common nouns or numbers, while classes are composed mostly of common nouns, adjectives and proper nouns. Finally, properties tend to open with a verb or auxiliary verb in the present tense. In paper [Pow12], the authors describe a finite state network that is capable of interpreting the CNL sentences in the grammar with minimal knowledge of content words. OWL Simplified English permits the acceptance of some technical phrases that violate normal English. The language can capture ontology operations such as simple negation, cardinality, object intersection but aims to reduce or eliminate structural ambiguity.

Semantic Query and Update High-Level Language (SQUALL) [Fer14] is a CNL for semantic querying and update of RDF graphs on top of SPARQL 1.1. The authors claim that SQUALL is easier to learn, and to formulate complex queries and updates than other CNLs. This is because

4 https://github.com/Attempto/APE

SQUALL combine expressiveness close to SPARQL 1.1, a natural syntax and semantics based on Montague grammars [Mon70] which is a context free generative grammar based on formal logic and calculus. The semantics of SQUALL are translated from this logical intermediate language into SPARQL. The lexical conventions of SQUALL at the lexical level does not differentiate between singular and plural, and between nouns and verbs. However, it does differentiate between them at the syntactic level as it uses URIs for non grammatical words. SQUALL has some ambiguity that the system can resolve using some rules related to priorities of, algebraic operators and smaller syntagms, over, sentence modifiers, and larger syntagms, respectively. Also, ambiguity of two constructs of the same syntagm is resolved by choosing the shorter construct.

AIDA [KBN+13] (Atomic, Independent, Declarative, Absolute) is a proposed approach that can be considered as a CNL for extending the nanopublications concept, to facilitate keeping track of latest research results in modern science using informal representations. AIDA means that natural sentences written in English has to follow a scheme where sentences have to be Atomic, Independent, Declarative and Absolute. This approach introduces a prototype of a nanopublication portal called nanobrowser, based on Apache wicket and the Virtuoso triple store. Nanobrowser looks like a semantic Wiki, where a particular scientific statement is presented with opinions from researchers, about related sentences with its meta-nanopublication, and this shows that AIDA links and relate nanopublications with each other. However, the problem of expressing a sentence in more than one way needs to be taken into account. To solve this, the authors proposed a mixture of clustering and crowdsourcing, so that nanopublications users can identify sentences that have similar meanings. In order to describe scientific results, AIDA assumes that sentences have their own independent existence. Therefore, each AIDA sentence get its own URI to make it first-class citizen in the RDF world. Furthermore, each AIDA sentence should be extractable from its URI without the need to consult any resources, and vice-versa. The authors in [Kuh18] proposed an approach to use AIDA CNL in the domain of scientific publishing to organize scientific claims, where researches can publish their results using AIDA sentences to build a network of scientific claims linked to publications.

As shown in Table 2, we summarize and compare all the previously listed CNL from various aspects. These aspects are intended to help a user

to select the appropriate CNL for a specific task and are summarized as follows:

- *Input and Output*, this aspect is for guiding the user to select the appropriate CNL based on the required output needed for the application whether it is FOL [Smu12], SPARQL [PS+06], OWL [CP15] or RDF [McB04].
- *Parsing*, describes the parsing mechanism used as it is an important piece of information for integrating the CNL with the required application, hence listing the used grammar, programming language, and the mapping techniques.
- *Aim*, indicates the reason behind developing each CNL in terms of the specific use-case and domain.
- *Advantages*, describes the most important feature that can be found in each CNL that distinguish it from other CNLs such as predictive editing, expressivity level and coverage of SPARQL constructs.
- *Limitations*, represents the drawbacks as mentioned in the literature after evaluating each CNL.

Table 2: Summary Table comparing all the Semantic Web Related CNLs listed in chronological order of publication

CNL	Input	Output	Parsing	Aim	Advantages	Limitations
PENG	PENG Texts	FOL	incremental bottom-up chart parser discovers the word form that is currently under investigation and feeds this grammatical information to the look-ahead editor.	write precise and unambiguous specifications for knowledge representation	predictive editing feature	1.Grammar is informed by FOL rather than DL considerations. 2.the grammar is not bidirectional.
CT	CT natural language	formal logic notation and into other natural languages	using syntactic and semantic restrictions	knowledge representation and extraction	the author can choose to leave or remove ambiguity, depending on the need	heavily restricted on both the syntactic and the semantic levels
PENG-D	PENG Texts	RDF and OWL	look-ahead text editor, a controlled language processor, an ontology component and an inference engine.	same as PENG plus the support for ontology construction	1.it has formal properties that are equivalent to DLP. 2.enable ontological definitions to be combined with rules. 3.predictive editing feature	the grammar is not bidirectional.

SOS	Sydney OWL Syntax	OWL 1.1	uses a context-sensitive grammar for bidirectionality which can store the required elements and employ an axiom schema instantiated during parsing then perform one-to-one mapping between controlled natural language and OWL syntax	1.create and edit OWL ontologies 2.provide English translations of OWL ontologies	1.predictive editing feature. 2.bidirectional mapping to OWL 1.1. 3.Coverage of the entire OWL language. 4.easy to implement in tools.	1.only one SOS form for each OWL form. 2.emphasis on variables makes it less readable than pure natural language.
ACE	ACE Texts	FOL	Attempto Parsing Engine (APE) consists of a definite clause grammar written in Prolog	knowledge representation language to create and edit OWL ontologies	syntactically and semantically expressive CNL	1.does not focus on bidirectionality. 2.requires definition of a lexicon.
Rabbit	Rabbit Sentences	OWL	Rabbit parser implemented in java consists of a pipeline of linguistic processing resources	create and edit OWL ontologies	it represents the whole ontology	assumes that the user will have some basic knowledge of the constructs and axioms used in ontology development

OWL Simplified English	predefined function words and verbs listed by the user	OWL ontologies	finite-state transducer	OWL ontology editing	1.disallow structurally ambiguous sentences. 2.fast and reliable implementation of an editing tool using finite state grammar.	very restricted coverage of OWL axioms
SQUALL	SQUALL Texts	SPARQL	translating SQUALL text into an intermediate logical language in the form of Montague grammar and then to SPARQL	query and update RDF graphs	strong adequacy with RDF, and covers all constructs of SPARQL	SQUALL sentences may look unnatural because URIs are invariant with respect to number or person
AIDA	AIDA Sentences	RDF	each AIDA sentence get its own URI in the RDF world. URI encoding, uses the relations to link AIDA sentences and relate them to other entities	allow informal representations of scientific assertions for semantic publishing of nanopublications.	sharing and interlinking of scientific findings, while being more flexible and applicable to a much wider range of scientific results	1.no concrete proposal yet to make the scientific results Atomic 2.URIs encoding is against existing recommendations of keeping URLs opaque.

3.4 TOOLS BASED ON SEMANTIC WEB RELATED CNLS

This section will present tools like editors, wikis and frameworks that use CNLs to perform specific tasks. In 3.4.1, we will discuss the ontology engineering tools presented in the literature for the aim of ontology creation and authoring, 3.4.2, provides the ontology querying tools, and 3.4.3, shows the rest of the tools presented in the literature which use CNLs for different purposes. The end of the section includes Table 3 as a summary table comparing all the tools.

3.4.1 Ontology Engineering Tools

What You See is What You Meant (WYSIWYM) is a popular framework that uses a natural language interface to support users in the knowledge authoring process, by providing a natural language editing feedback using various menu options. This knowledge can be used in generating ontology based CNLs in the Semantic Web context [PSE98].

Guided Input Natural Language Ontology Editor (GINO) is another controlled Natural Language Interface (NLI) that supports ontology editing using user directed feedback. The interface guides the user during the authoring process by suggesting error fixes and alternative sentences that are acceptable by the GINO parser [BK06]. Since the main goal of GINO is to support ontology authoring in the Semantic Web domain, the interface supports the translation of successfully parsed sentences either into triples or SPARQL queries that can be used later by the Jena Semantic Web framework. GINO is based on predefined CNL grammar rules, as well as some rules that can be generated from the ontology.

Round Trip Ontology Authoring (ROA) builds on and extends the existing advantages of the CLOnE software to create and populate an ontology with the addition of a text generation component to form ROA environment. The aim of the text generator is to reproduce the CNL from an ontology, edit the text as required, then parse it back into the ontology until the user gets the desired results. Thus, NLG acts as a feedback to guide the user and reduces the need to learn the Controlled Language by following examples, style guides or CLOnE syntactic rules [DIF+08]. The ROA pipeline consists of GATE NLP modules to annotate the input document, followed by Keyphrase gazetteer and to two JAPE transducers

to identify quoted and unquoted chunks. Then, a Controlled Language for Information Extraction (CLIE) component is connected to the existing ontology to interpret the input sentences. Finally, the ontology is connected with the text generator component to act as an ontology verbaliser to present the ontology in textual form as an ambiguous subset of English [DIF+08].

Rabbit to OWL Ontology authoring (ROO) [DDH+08] is an open source editing tool based on the Rabbit language, that is developed as a plugin in the Protégé framework. The motivation behind developing ROO is to support domain experts in the ontology authoring process, as current CNL based interfaces lack the support of ontology construction. So the authors aimed to fill this gap between domain experts who are engaged in the engineering and conceptualisation of the ontology from the beginning, and the ontology engineer who focus on the logical level of the ontology. A new intelligent model was integrated to ROO to understand the user actions and give feedback accordingly. The model was introduced in [DTD+12] to resolve the modelling errors, by providing a framework for semantic feedback when adding a new fact to an existing ontology. The new framework extends the syntactic analysis performed by Rabbit through categorizing the new ontological facts into four categories concerning inconsistency and novelty of facts. This feedback approach was observed to be repetitive, confusing and sometimes redundant [DDC13]. As a result, a new framework with dialogue interfaces was introduced in [DDC13] as an extension to Rabbit. It provides more appropriate feedback according to different situations by keeping track of the ontology history. In addition, the inputs of the domain experts are analyzed and an intention is assigned to each input.

ACEView is an editor implemented as a plugin for the Protégé editor[5] mapping from ACE to OWL/SWRL and from OWL to ACE. The editor adds more ACE based interfaces to Protégé to engineer an OWL ontology [Kalo8]. Furthermore, it assists the user by providing a querying interface to retrieve latest facts from the knowledge base using the ACE CNL. ACE text is automatically parsed and converted into OWL/SWRL. ACEView comprises vocabulary and wordform view, asserted knowledge and entailed knowledge views. The most beneficial features are, ensuring that consistent naming conventions are used by placing restrictions on OWL entity names, restricting the complexity of the OWL class expressions, and verbalizing complex constructions into simpler sytnax [Kalo8].

5 http://Protege.stanford.edu/

AceWiki [Kuho8] is a monolingual CNL based semantic wiki that takes advantage of ACE for its syntactically user friendly formal language, and of OWL frameworks for applying classification and querying. The AceWiki content is based on ACE predictive editor notation grammar called codeco [Kuh13a]. The main benefit of codeco is that it can translate all AceWiki content to OWL.

AceWiki-GF [KK13] is a multilingual extension of the previously mentioned AceWiki, where users can get all the benefits of AceWiki plus multilinguality after using the Grammatical Framework (GF), discussed in the next section. The implementation was done by modifying the original AceWiki to include GF multilingual Ace grammar, GF parser, GF source editor, and GF abstract tree set. This study included an evaluation about the accuracy of translation in AceWiki-GF. The evaluation showed that the translation accuracy was acceptable, although some errors due to different reasons in terms of Resource Grammar Library (RGL), where incorrect use of RGL by mixing regular and irregular paradigms, using unnatural phrases to native speakers, and negative determiners.

3.4.2 Ontology Querying Tools

Pseudo Natural Language (PNL) [Maro4] is the first query logical system to provide natural, easy and friendly way for people to use the semantic web. The paper introduces the Metalog project[6], to fill the axis of the people to the semantic web. Metalog uses Pseudo Natural Language (PNL) interface that is similar to natural language and extends RDF with the Metalog Model Level (MML). PNL is an unambiguous language with the principle ''one language, one query", designed to be easy to read for users and easy to write for developers, by sacrificing the total freedom of the natural language with the restrictions to the language. Metalog has a smart querying ability that accepts informal queries and normalize it to be mapped into an assertion.

Guided Input Natural language Search ENGine (GINSENG) [BKK05] provides a quasi-natural language guided query interface to the semantic web. Thus, it came out to reduce the gap between real world users and the logic-based semantic web. Ginseng allows users to query any semantic web knowledge base, using a guided input NL vocabulary loaded ontologies that grow with every additional added ontology, but without using any

6 http://www.w3.org/RDF/Metalog/. W3C, 1998-2004

predefined vocabulary. Despite this can limit the users possibilities, it ensures that every query will have a correctly matched result. Ginseng guides the user with a set of possible queries while avoiding grammatical errors, by presenting to the user a choice pop up box that includes suggestions on how to correctly complete the current sentence, and hence the possible choices get reduced as the user continues typing. Ginseng translates queries into a RDF Data Query Language (RDQL) and displays the result. The architecture of ginseng consists of three modules as follows; multilevel grammar; an incremental parser; an ontology access layer through Jena. The multilevel grammar is a domain that contains about 120 independent rules, constructed manually, and divided into two types of rules 1) static grammar rules, to provide the basic structure of sentences and questions, 2) The dynamic grammar rules from the loaded OWL ontologies, created for each class, instance, object, and data-type properties. Furthermore, Ginseng provides ontology annotation option with Ginseng tags. The incremental parser uses the grammar to specify the complete set of parsed sentences without incorrect entries, and to generate the resulting query by creating a complete parse tree.

OWLPath [VGC+11] is an ontology-guided input natural language query editor that combines the advantages of both the natural language interfaces NLIs and CNLs, to reduce the gap between users and the semantic web. It guides the user on how to complete a query using the question and the domain ontologies. The question ontology represents the grammar and the sentence structure, while the domain ontology represents the concepts and relationships in the domain. The main components of OWLPath system are as follows; Ajax interface that loads the domain related set of ontologies and let the users build the query; the suggester generates a list of terms shown in a pop-up list for the user to choose from; the Grammar checker determine only the correct grammatical entries combinations; the SPARQL manager translates the query into SPARQL statements and parse it to the knowledge base through the ontology manager; and finally the results are shown to the user.

3.4.3 Other Tools

Grammatical Framework (GF) is an open source implementation framework that supports the development of multilingual CNLs [AR09] and

[Ran04]. The main goal of the framework is knowledge based Machine Translation (MT) to assist porting any CNL for multilingualism, such as ACE that was reverse engineered for GF [AR09] in five languages. GF is based on a semantic model known as the *abstract syntax*, and a syntactic model known as the *concrete syntax*. Multilinguality can be achieved by mapping the object based on the common abstract syntax of the text, into its respective concrete syntax using the wide coverage grammars of over 15 natural languages. The GF community is active in adding more language resources to the framework, and building useful applications such as patent translation [EES+11], multilingual wikis especially in the culture heritage domain [AR09][Dan08][DDE+12], tax fraud detection framework for supporting fraud experts [CCP16], and interesting projects such as the European project, MOLTO (Multilingual On-Line Translation)[7].

PathOnt [KHL+05] is a PATHological ONTology based application. It uses a controlled ontology from the terminology resources in GALEN [RZ95] for the gross description medical ontology system. The need for this application is to solve the communication problem between pathologists and other professionals who misinterpret the meaning of the gross description. The system consists of three components; the PathOnt semantic to specify the required ontology for the gross description; PathOnt object for visualizing the macroscopic findings stored in the RDF file; PathOnt syntax that generates an XML form for the input update.

Atomate It [VMK+10] is a web based reactive personal information assistance engine, that allows end-users to use data feeds to drive reactive automation. For instance, Atomate can integrate the information out of the RSS/Atom feeds from social networks into RDF model to derive useful behaviors, and thus important reminders can be created, taking into account the rules specified by the user. The CNL interface design is based on GINO and Ginseng interfaces [VMK+10]. The rules in Atomate consists of antecedent to represent the execution conditions, and consequent to specify the actions to be taken. Atomate's data flows from the lost of the data sources provided by the user in the RDF model to the Atomate's feeder. The feeder creates a new entity for each new data source or updates the existing ones. Then, based on these updates the rule chainer retrieves all the rule entities from the world model, and fire all rules whose triggered antecedents depend on the changed entities.

7 http://www.molto-project.eu/

Table 3: Summary Table comparing all Tools based on Semantic Web Related CNLs listed in chronological order of publication

Name	Input	Output	Parsing	Aim	Advantage	Limitations
Ontology Engineering Tools						
WYSIWYM	Natural Language	OWL data	WYSIWYM engine in Prolog, and user interface in JAVA. Also, there is a NL generator to provide feedback for user interactions	defining and editing knowledge bases	allows direct control over semantic features through a NL feedback	the usability of the tool varies according to the structure of the ontology
Gino	NL expressions using a grammar derived by the loaded ontolgies	RDF triples or OWL axioms	parse rules based on CNL grammar from 120 static rules that could be extended dynamically based on the ontology grammar. Then, the sentence is passed to the JENA engine for execution.	1.domain independent ontology editing interface in quasi NL. 2.extension of Ginseng to support procedural statements and introduce new entities	1.guided input natural language editor. 2.easy adaptation to new ontologies using dynamic grammar generation	exploiting NL expression is always limited by the vocabulary from the loaded ontologies.
ROA	CLOne Input	modified or edited ontology	GATE NLP tools plus the gazetteer and 2 JAPE transducers	introducing NLG into CLOnE to facilitate Round-Trip Ontology Authoring.	the text generator reduces the learning curve for users	works for less sophisticated knowledge engineering tasks

ROO	Rabbit Sentences	OWL	Rabbit parser plus OWL conversion using OWL API	1.embed into Protégé as a plug-in. 2.ontology authoring.	show feedback and provide a list of Rabbit templates to help the user	there are common error patterns when using a CNL to define ontological constructs
ACEView	ACE Sentences	OWL/ SWRL	plug-in for Protégé that relies on the OWL API	translating to and from OWL/SWRL	1.naming conventions and consistency. 2.structural complexity of class expressions. 3.verbalizing complex constructions into simpler syntax.	1.no predictive editor is integrated. 2.weak semantic feedback to the user.
ACEWiki	ACE Texts	OWL	bidirectional mapping between ACE and OWL using APE parser	1. Ace Editor for wikis 2.collaborative ontology development using NL	1.easier to use and more expressive than other semantic wikis. 2.predictive editor.	1.still a prototype and not a real-world application. 2.supports a single grammar and allows users to control only the set of (monolingual) content words.

ACEWiki-GF	ACE Texts	OWL	-same as ACEWiki with replacing the ACE parser and English lexical editor, with a GF parser and source editor. -it uses bidirectional mapping between ACE and OWL using APE parser	1.Ace Editor for wikis. 2.multilingual collaborative ontology development using NL.	same advantages as ACEWiki plus supporting multilingualism	still a prototype and not real-world application
Ontology Querying Tools						
PNL	PNL sentences	NL queries based on assertions	builds upon RDF and first-order logic, and uses Prolog to calculate inferences	querying and reasoning on the Web	1.unambiguous and has well-defined semantics. 2.no need to learn a separate, query language as it is the same as its regular expression language	1.unnatural capitalization mitigates the naturalness of the language. 2.complex rules have to be applied in order to resolve ambiguous syntax trees.

Ginseng	NL queries using a grammar derived by the loaded ontolgies	SPARQL	a grammar compiler that generates a set of dynamic rules referred by the loaded ontolgies, and a set of CNL static rules to provide the structure of a basic query	NL query interface to access OWL ontologies	1.allows ontology annotation to create alternatives to the ontology identifiers. 2.works on domain independent ontologies. 3.guided input interface for queries.	writing queries is constrained by the vocabulary of the loaded ontologies.
OWLPath	OWLPath's Guided English queries	SPARQL	-parsing includes Ajax interface, Suggester, Grammar checker, SPARQL Generator and Ontology manager. -the Question ontology represents the grammar & the Domain ontology represents the structure of concepts and relationships	allows non-expert users to easily create SPARQL queries that could be issued over most existing ontology storage systems	1.Domain-independent system that can be applied to any ontology. 2. it has a sophisticated grammar	1.the ontologies used for test purposes are relatively small and are processed in-memory. 2.the question ontologies only include the most common constructors to allow formulating typical queries in the experimental scenarios.
Other Tools						

GF	Grammar Rules	CNL	bidirectional mapping between the concrete language strings and their corresponding abstract trees	1.framework for CNL editing & implementation. 2.Implementing multilingual CNLs	1.support multilingualism using grammar libraries. 2.compatible with many programming languages	The language coverage and the reasoning capabilities are limited to what the grammarian has encoded in the grammar
PathONT	PathOnt CNL	RDF	-developed using the Java-2 platform. -based on the Galen top level ontology. -statements are mapped to RDF triples	1.formalization of the gross description. 2.communication among clinicians and technicians involved in pathology examinations.	1.facilitates the interactions between end-users of medical information and between different applications. 2.multilingual as it supports both English and Korean languages.	covers only simple existential statements
Atomate	Atomate Language	RDF	written entirely in Java script and consists of world model, rule chainer, and triggered rules	personal information assistance engine to define automatic tasks and reminders, while taking context and current activity into account	1.integration of heterogeneous data from RSS feeds. 2.easy to extend its capabilities, add new data sources, and new types of information.	1.lacks a sophisticated DL inference engine. 2.there is no capacities to learn, search, or act on the user's behalf beyond the rules set up by the user.

3.5 EVALUATION OF THE CNLS

According to [Kuh13b], one of the main methods used to evaluate CNLs is **paraphrase-based**, where a group of users provide their feedback and observations about the usability of a CNL. The feedback can be provided via a questionnaire or conversations. With respect to related work, in the next sections we will review existing CNL research, but in the context of user evaluation. The end of the section includes a summary table comparing all the evaluations for these CNLs. Some CNLs do not not have any evaluations mentioned in the literature, and hence they are not listed below.

The authors of **ClearTalk** conducted an evaluation to check if CT helps students to learn [Sku03]. The experiment was informal, where 80 students were divided into two groups to answer questions about applets within 45 minutes, with both groups have text books, and one group only has access to the KB. The results show that the group with access to the KB got 50% higher mark, which proves the main function of the KB that makes students find facts faster. However, some of the difference might be due to better understanding of the subjects.

Recently, an evaluation for **ACE** was presented in [Kuh13b], where it describes an evaluation framework for CNLs based on Ontographs. Ontographs are a graphical notation to enable tool independent and reliable evaluation of the human understanding of a given knowledge representation language. They serve as a common basis for testing and comparing the understandability of two different formal languages and facilitate the design of tool-independent and reliable experiments. An experiment was performed by 64 participants to compare the syntax of ACE versus a simplified version of **Manchester OWL Syntax**, to test which syntax is better in terms of, understandability, learning time, and users acceptance. The results showed that users were able to do better classification using ACE with approximately 5% more accuracy than Manchester OWL, and 4.7 minutes less time for learning. Also, in terms of understandability ACE got 0.67 higher score than Manchester OWL [Kuh13b].

In [EHD09], the authors undertake an evaluation to assess whether domain experts without any ontology authoring development can author and understand declaration and axiom sentences in **Rabbit**. The experiment included 21 participants from the ordnance survey domain and a Rabbit language expert. The participants were given a text that describes a fictional world and were asked to make knowledge statements, then they are anal-

ysed for correctness by independent experts and compared to equivalent statements created by the Rabbit expert. Interestingly, on average 51% of the generated Rabbit sentences contained at least one error. Furthermore, the most common error was the omission of the quantifier at the beginning of "every" sentence.

For the evaluation of **AIDA**, the authors did two evaluations related to the initial stage of the approach [KBN+13]. The first evaluation was done to measure the difficulty of creating nanopublications for scientific results. The experiment involved 16 participants with background in biology and medicine who never knew about AIDA. A random sample was taken from Pubmed abstracts [KBN+13] that have a conclusion section. The evaluation was through an online questionnaire consisting of 3 parts; the first part explains AIDA concept; the second part showed five short texts to be written in one to three AIDA sentences each; and the last part asked about the difficulty to understand AIDA concept and to do rewriting tasks. The results showed that an average sentence required 90 seconds to be created including the time to learn about AIDA concept. Out of 163 sentences created by the user, 70% were perfectly complied with the AIDA restrictions. All participants mentioned that understanding AIDA concept was easy but not very easy, and the rewriting task was of medium difficulty. The second test was to evaluate the quality of automatically extracting AIDA nanopublications from text resources and relate them to each other. The authors used GeneRif[8]dataset, which contains sentences about gene and protein functions. Results showed that 71% of the resulting AIDA sentences were fully complied with AIDA restrictions.

As shown in Table 4, we summarize all the evaluations performed on the previously discussed controlled languages that are presented in the literature. The table describes the performed experiment and the goal of performing it in terms of the metrics it wanted to measure, the number of participants contributed to the experiment, the final results, and finally whether the CNL is compared with other CNLs in the literature or not.

8 http://www.ncbi.nlm.nih.gov/gene/about-generif

Table 4: Summary Table comparing all the evaluations for Semantic Web Related CNLs in chronological order of publication.

Name	Comparison	Evaluation Goal	Experiment	Participants	Results
CT	No	if CT helps students to learn	Informal Test let students answer questions related to IT field	80 students	The group with access to the KB find facts faster and got 50% higher mark. Other than that no other statistical results.
ACE	simplified version of Manchester OWL syntax (MLL)	test which is better in terms of, understandability and learning time using ontographs	users are given instructions about using ontographs then asked to answer a questionnaire	64 students with no background in Logic or CS	-Learning Time: ACE got 4.7 minutes less. - understandability: ACE got 0.67 higher score.
Rabbit	No	test the understandability of users with no background of ontology authoring	convert a text about fictional world topic into rabbit sentences, then the output is compared with the statements created by the Rabbit expert	21 users from Ordnance survey domain and a Rabbit language expert	on average 51% of the generated rabbit sentences contains at least 1 error
AIDA	No	Exp1: difficulty of creating nanopublications using AIDA	online questionnaire that explains how to rewrite sentences using AIDA and asks for the difficulty	16 participants with no knowledge about AIDA	-on average a sentence is created in 90 sec. - understandability: Easy but not very easy. - Rewriting: medium difficulty.

3.6 EVALUATION OF CNLS BASED TOOLS

According to [Kuh13b], **task-based experiments** are used to conduct evaluations of tools that use CNLs to perform different tasks. Users are provided with instructions sheet to read and understand, then they are asked to perform some tests using the tool. The statistical data from the tests are observed and recorded to check the tool effectiveness. With respect to related work, in the next sections we will review user evaluation of tools. The end of the section includes Table 5 as a summary table comparing all the evaluations for these tools. Some tools do not not have any evaluations mentioned in the literature, and hence they are not listed below.

3.6.1 Evaluation of Ontology Engineering Tools

An evaluation of **WYSIWYM** was carried out two times. The first evaluation was presented in the CLEF[9] project developed for the medical domain [HSP07]. The experiment was conducted by 15 participants mainly medics and bio-informaticians to test usability, understandability and the difficulty of using the tool. Participants were given a short demonstration on how to construct a simple query using the interface, and then asked to create a set of 4 SQL queries for a database in the medical domain. The sets were given to each participant in a different order to ensure that tasks complexity does not affect the process. The results showed that from the second task onwards all participants achieved 100% success in composing all the queries in a mean completion time 3 to 9 minutes per query, and became faster with each task, especially after the first task. For testing the understandability, the participants were given a paper-based questionnaire of complex queries and asked to select the correct meaning for each query from a list of 3 options. The results showed that on average the participants choose the correct interpretation 84% of the time.

The second evaluation was conducted in [HME08] by 16 researchers and PhD students from the social sciences domain. Users were shown a six minute background video for the main functionalities of the WYSIWYM interface. Descriptions of four resources as paragraphs of English were provided to the users. The goal was to reproduce the descriptions using the WYSIWYM tool. Each subject also received the descriptions in varied

9 http://www.clinical-escience.org/, Retrieved 2008-05-22

order. The descriptions were further divided into eight to ten sub tasks. The successful completion of certain sub-tasks was dependent on the preceding sub-task. Task completion times, number of operations, as well as errors including "avoidable" errors (which imply the result of an error introduced from a previous sub-task), were measured. The results were encouraging, where users mean completion times decreased significantly. Hence, users gained speed over time. The results of the subjective feedback on the tool indicated that the tool perceived positively, where the mean scores on a 1 to 5 (very useful & very difficult) scale was 3.94 for usefulness, and 2.69 for the difficulty level. However, the results in [HSP07] was more positive than [HME08], since the participants of the CLEF project were mainly medics who understand their domain very well. On the other hand, the social science domain tends to be more varied with many different theories and approaches. Consequently, the underlying domain ontology can have a large significant impact on usability. More importantly, users from the social sciences field reported that they were overwhelmed by the large number of options available i.e. thirty properties per one object [HME08].

ROA evaluation is conducted against Protégé [DIF+08], where 20 users were recruited from both research and industrial background, but with no background in either GATE and Protégé tools. The participants were provided by Protégé manual, text generator examples, and two task lists. They were divided into two groups, each group was asked to work on each task list, using either ROA or Protégé, opposite to the other group. Finally, they were asked to complete both a SUS [Bro+96] questionnaire and a comparative questionnaire for each tool. The results showed that the mean SUS score for ROA is 74%, and 41% for Protégé.

An evaluation study of **ROO** was conducted against **ACEView** in [DDH+08] to compare both tools in terms of usability, usefulness and the quality of resultant ontologies. The study involved 16 students from the domains of geography and environmental studies. Student were asked to create ontologies based on hydrology and environmental models, respectively. Both ontology creation tasks were designed to resemble real tasks performed by domain experts at the Ordnance Survey. Ontologies for both domains were produced by the Ordnance Survey's OS MasterMap®10 [10]. The usability results showed that messages in ROO were more helpful, the tool was less

10 http://www.ordnancesurvey.co.uk/osmastermap/, a nationally contiguous vector map containing more than 450 million individual features down to street, address and individual building level, spatial data to approximately 10cm accuracy

complex than ACEView, and users would be more willing to use ROO again. In terms of usefulness, the mean score for ROO users was 5 which is higher than ACEView users who scored 0.38, as the understanding of ontology modelling improves significantly more when using ROO than when using ACEView. The quality of the resultant ontologies, showed that ontologies built with ROO have better readability than those built with ACEView, as ROO encourages users to add annotations for concepts and relationships. Another study presented as an extension of ROO in [DTD+12] showed that 91% of the feedback messages were helpful to the users, and 78% were informative. However, feedback caused confusion and overwhelming for 10% of the cases.

ACEWiki was evaluated to test whether people with no background about ontologies and logic will be able to learn and deal with ACEWiki, without the help of an expert and without spending long time [Kuho8]. The experiment was conducted online, where 20 participants mostly students and graduates with no background about semantic web or logic, were provided by instructions sheet and then asked to add whatever knowledge they like to ACEWiki, following certain restrictions. The participants created 186 sentences, 148 of them were correct, and the other 38 were not. The participants spent on average 11 minutes for creating the first correct sentence, and 8.2 minutes overall for each correct sentence. The feedback for the difficulty level of using ACEWiki was mostly of medium difficulty, and 25% of the users found it difficult.

ACEWiki-GF evaluation [CFK+13] to determine, how much using ACEWiki-GF is effective and efficient to help two users of different languages understand each other. The experiment is to let each user write an article in his native language, and in the post editing stage users read the automatically translated articles written by other users and evaluate, whether the sentences are true or false in their language. The evaluation took place online through the ACEWiki-GF online tool, where 30 participants were asked to create a new wiki page and write true and false statements. Then after finishing, the users were asked to fill a questionnaire about their feedback about the system. The 30 participants created in total 316 sentences on average of 37 minutes. One hundred and seventy one of the sentences were measured as true, and 145 as false. The results show that the translation error rate for ACEWIki-GF is less than 5%. The feedback from the users for the difficulty level of using ACEWiki-GF in general was 2.93 on a 0 (very

difficult) to 4 (very easy) scale. The result was 2.77 for the difficulty level of the sentence editor.

3.6.2 Evaluation of Ontology Querying Tools

Ginseng was evaluated against SQL using SUS evaluation in terms of usability, speed, precision/recall and the ability to parse a large number of real world queries [BKK05]. The evaluation was held by 20 students from the CS department with knowledge of SQL queries. In the experiment, half of the users have to query into Ginseng, and the other half were provided with SQL interface. The results showed that, in terms of speed, Ginseng was faster than SQL with average difference of 1 minute. Also, Ginseng was rated to be better integrated and easier to learn. In terms of parsing power, one knowledge base from geographical Mooney [TM01] was used with 880 queries. Ginseng could execute 40% of the queries out of the box. In addition, the queries that could be parsed resulted in precision of 92.8% and a recall of 98.4%. However, the usability evaluation was limited by a specific subject (CS students), and it was not performed across huge datasets. The authors intend to improve these limitations by extending Ginseng's property tags generation which can be automated with WordNet and machine learning techniques. In contrast to PENG, where knowledge has to be entered into the system using a complete NL processing engine, Ginseng query existing semantically annotated content using a simple querying grammar, where it can be dynamically extended by any OWL ontology structure.

OWLPath evaluation was conducted to test the performance analysis and the user experience [VGC+11]. The performance analysis was in terms of time elapsed between selecting the next entry in the query, and showing the next choice in the pop-up list, taking into account the time for SPARQL statement generation. All the tests were performed on the local machine to avoid internet latency, taking the average time over 10 runs, the results showed that the elapsed time did not change for larger number of words as the number of relations decrease as well. In addition, the time for generating the SPARQL statements is short, since the RDF triples of the words are generated when the user enters each word. On the other hand, regarding the user experience evaluation, precision and recall were not relevant for evaluation, since the OWLPath system is very accurate, as the

resulting queries forced by the pop-up list through the ontologies were always valid. However, the authors designed an experiment to test the advantages of building queries using OWLPath. The experiment involved 4 PhD students with strong background in ontologies and SPARQL to create 10 queries related to a tourism-based ontology. The results showed that, it takes less time to generate a query using OWLPath interface than to do it manually.

3.6.3 Evaluation of the Other Tools

The authors of **Atomate** conducted two evaluations to test whether the users will be able to understand and create rules, and to check whether the users will be interested to use the system in the present or the future [VMK+10]. The first study was a design review with 15 UI researchers to get early feedback before the rule creation process. The second study, involved 3 colleagues from their lab who were asked to create 9 rules ranged from simple to complex, after watching an explanation video. For the design review study, the authors got further feedback for making the rule creation process more clear. For the rule creation study, 33 participants did the study. Twenty six of them completed all the rules and the survey. Fourteen of the participants had programming experience. The first 6 rules were correct over 75% of the time, while the rest of the rules were more problematic. For the complexity of creating the rules, 65% of the users found it easy, while the rest 35% found it difficult. Regarding the usefulness of the system, on a scale of 1 to 7 (7 is very useful), the mean response of the participants was 5.5.

Table 5: Summary Table comparing all the evaluations for the tools based on Semantic Web Related CNLs in chronological order of publication.

Name	Comparison	Evaluation Goal	Experiment	Participants	Results
Ontology Engineering Tools					
WYSIWYM	No	usefulness and difficulty of using the tool	reproduce description of English paragraphs in WYSIWYM	16 researchers and PhD students from social science domain	on 1 to 5 scale (5 is very useful & very difficult). -usefulness=3.94. -difficulty=2.69
ROA	Protégé	usability for ontology editing tasks	ask users to work on task lists using Protégé and ROA	20 participants from research and industry with no background about Protégé	-mean SUS score of usability: ROA = 74% Protégé = 41%
ROO	ACEView	usability and usefulness of the tool	create ontologies based on hydrology & environmental models using ROO and ACEView	16 participants from geography and environment studies	-usability: users are more willing to use ROO than ACEView. -usefulness on 1 to 7 scale (7 is strongly agree): ROO = 5 - ACEView = 0.38
ACEWiki	No	difficulty of using the tool for-non technical users	add knowledge to ACEWiki given some instructions	20 participants mostly 7 students and graduates with no background about logic and semantic web	medium difficulty= 75% of the users. difficult= 25% of the users.

ACEwiki-GF	No	difficulty and translation accuracy of ACEWiki-GF	each user evaluate the output of the tool in his native language through an online questionnaire	30 participants each one is fluent in one of the languages used in the evaluation	-translation error rate was less than 5%. -difficulty on a 0 to 4 scale (4 is very easy)= 2.93
Ontology Querying Tools					
Ginseng	SQL	usability, speed and precision/recall of the paresd queries	write geographical queries using Ginseng and SQL interface	20 students from CS department with knowledge about SQL queries	-speed: on average Ginseng was 1 minute faster. -Usability: Ginseng is easier to learn. -precision=92.8% -recall=98.4%
OWLPath	No	Time to generate a query	create queries related to tourism based ontology	4 Phd students with background in ontologies and SPARQL	it takes less time to generate a query using OWLPath than to do it manually
Other Tools					
Atomate	No	understandability and difficulty usefulness	feedback about the design and creating rules using the tool	15 UI researchers for design review feedback and 33 participants for the rule creation process	-easy to use: 65% of the users -difficult: 35% of the users. -usefulness on 1 to 7 scale (7 is very useful)=5.5

3.7 HUMAN-ORIENTED CNLS

Human-Oriented CNLs have been around for many years. Their development was motivated for the purposes of language learning and unambiguous communication between humans in a domain specific context. Basic English [Ogd30] can be considered the first controlled variant of the English language, developed to improve readability and facilitate communication between non-native speakers and second language learners in different domains. It utilizes a restricted version of the English grammar and lexicon (850 words). Although, it was criticized by linguists and researchers [Fle44][Hin88], for being based on a faulty analysis, and for lacking the scientific approach and empirical evidence to simplify the language. Currently, it has a wide recognition especially from the Basic-English Institute[11], which contributes to its development and usage in different applications. For example, it influenced the Wikipedia community to reproduce the content in Wikipedia using Basic English and call it Simple Wikipedia[12]. The articles are re-authored using style guides[13] to reduce complexity and ambiguity of the language, especially for non-native speakers and juniors. Some of the written style guides recommended for authors in Simple Wikipedia include: *use active voices, avoid compound sentences (e.g conjunctions), avoid idioms (multi-words), keep sentences short and informative*. Another example is the new language of the *Thing Explainer* that uses a restricted language of around one thousand words of the English language to explain complicated concepts such as nuclear reactors, and jet engines [Kuh16]. Furthermore, with the advance in communication it came the need to find better ways to facilitate human-human communication, especially between humans who speak different languages and from different backgrounds. Hence, researchers such as [SSH+95][Che96][Eck98] started to analyse whether these type of languages have an added value, such that they can obtain an empirical proof to utilize them. A comparative analysis between various human-oriented CNLs has been performed by the authors in [JCC16], where they evaluate aspects such as performance, response time and accuracy on various comprehension tasks. The study measures the effect of using these languages that are based on Basic English, versus their natural language

11 http://www.basic-english.org
12 http://simple.wikipedia.org
13 https://simple.wikipedia.org/wiki/Wikipedia:How_to_write_Simple_English_pages

counterpart on non-native speakers. Finally, we have to mention that these human-oriented CNLs have found favour in the production of technical documentation for unambiguous human to human communication. For example, these CNLs facilitated the communication in the aerospace domain between non-native speaker pilot speakers, where they both can understand the technical terms. Furthermore, it was used for machine translation within industry and across various sectors such as aerospace, automotive, and information technology. In the next sections we will review some of the most popular human-oriented CNLs developed for different use cases by big corporates.

Aerospace and Defense Simplified Technical English (ASD-STE[14]), also known as Simplified English and previously as AECMA, developed to improve the readability and comprehensibility in Aircraft maintenance technical documents, within the aerospace industry and across various manufacturers. It is based on a version of the English language that has been restricted lexically, syntactically and semantically. In order to make the language precise, these restrictions are defined using 60 rules [ASD07] to set constrains for using fixed words from a predefined aerospace dictionary, grammar rules, and unambiguous terms. Other human-oriented CNLs in the aviation domain include **Boeing Technical English** [WHH98], that was developed to extend AECMA and to improve the communication between people for air traffic control in a broader range.

COGRAM [AS92], is another human-oriented CNL developed by the telecommunication company Alcatel [15] to fill the gap of the incomplete CNLs such as AECMA Simplified English that was available at this time. It has a lexicon that consists of technical terms and common words, and around 150 rules to restrict the authoring process. The rules were considered as writing style-guides to recommend the usage of certain sentence structures through lexical, syntactic and stylistic restrictions.

ALCOGRAM [AS92] is the successor of COGRAM, that was developed in Alcatel as well to provide more support for the technical writers to author documentation. It is different from COGRAM as it is based on an algorithm that assures certain criteria is met before accepting the authored content. The benchmark of this algorithm is represented in checking the used terms, the syntax complexity, the lexical terms from the gender-specific language, and micro control for the first words in each sentence.

14 http://www.asd-ste100.org/
15 https://www.alcatelmobile.com/

KANT Controlled English (KCE) [MN95], is a CNL developed with focus on machine translation of technical manuals and documentation. The language is used to build a machine translation system called KANT that has restrictions on lexical level, grammar level, and semantics level. Moreover, the system supports resolving ambiguities by enabling the attachment of markup tags to ambiguous phrases, which increases the language precision. The KANT system was acquired later by Caterpillar and integrated to produce the Caterpillar Technical English.

Caterpillar Technical English (CTE) [KAM+98], is a CNL that was developed by Caterpillar[16] to reduce the ambiguity and complexity in its technical documents, which in turn will improve their translation to different languages when used by non-native speakers. The language was supported by an authoring tool to enforce restrictions derived by a lexicon, and some grammar rules, and a language checker to guide the users by providing an interactive disambiguation environment.

IBM EasyEnglish [Ber97], that was developed mainly for improving machine translation using a grammar checker and some lexicon restrictions to reduce ambiguities. The system can detect these ambiguities and suggest other alternatives that can resolve them without enforcing the user to apply these changes. Although, it gives the flexibility to the user for applying the suggested changes, this acted as a drawback since it reduced the precision and simplicity of the language.

Controlled Language for Crisis Management (CLCM) [Tem11], is another human-oriented CNL that facilitates the authoring and comprehension of crisis handling instructions. It consists of simplification rules that define some restrictions on the structure and format of the text. In addition to, lexical constraints to prevent writing technical terms, and grammar constraints such as using direct voices instead of passive voices for writing.

Special English[17], that was developed by Voice of America to be used to broadcast their daily news on the radio and television. It can be considered as a simplified English that is based on Basic English, where they have a restricted lexicon of 1,500 words that is updated regularly. The language recommend the usage of short sentences without enforcing other constrains either on the grammar or the semantics.

16 https://www.caterpillar.com/
17 https://www.voaspecialenglish.com

Controlled Automotive Service Language (CASL) [MG96] is a CNL developed by General Motors[18], with the aim to improve readability and translation of automotive service technical documentations. The language is restricted on the lexical level, and restricted using 62 rules that define some sentence structures and formats that need to be followed. In addition to, the language provides a language checker tool called CASLChecker to guide the authors for the writing process.

Sun Proof [WS02], was a CNL developed by Sun, Inc with the aim to write technical manuals that are easier to understand especially for documents that needs translation. It is restricted using a set of style-guides on the grammar level and lexical level. Also, there are some guidelines to limit the length of the sentence to 25 words, and some semantic restrictions for words that might have more than one meaning.

3.8 OTHER RELATED WORK

3.8.1 PENS classification

In [Kuh14] the study divided the CNLs into different categories based on the application environment and the domain where each language can be used such as improving human-human communication, automatic translation, and providing natural representation for formal notations. Moreover, it presented a scheme called PENS to classify and describe different CNLs based on four dimensions that represent the fundamental language properties. The study used five classes numbered on 1 to 5 scale to define and restrict the top and the bottom of each dimension represented between English on one end and propositional logic on the other end. The dimensions are divided as follows:

- *Precision*, indicates the ability to capture the meaning of text in a certain language without taking context into account. Hence, languages can be classified into five classes: *Imprecise languages*, *Less imprecise languages*, *Reliably interpretable languages*, *Deterministically interpretable languages*, and *Languages with fixed semantics*. These classes are based on how close is each class to formal logic languages (maximum precision), and to natural languages (minimum precision).

18 https://www.gm.com/

- *Expressiveness*, measures the ability of a certain language to describe various propositions. A language is considered to be more expressive if it includes as much of the expressiveness features such as universal quantification and negation.
- *Naturalness*, identifies the readability and understandability of a certain language as features to measure whether it is close to a natural language or not.
- *Simplicity*, measure the complexity level in terms of the amount of effort required to implement both the syntax and semantics for a certain language in a mathematical model. The PENS scheme identified that the number of pages required to describe the language as a good metric to define the simplicity of the language.

3.8.2 O'Brien Common CNLs properties

An analysis between various CNLs has been done in a study by O'Brien [OBro3], where some of the human-oriented CNLs are included such as AECMA Simplified English (SE), Alcatel's COGRAM, IBM's Easy English, Sun Microsystem's Controlled English, and Avaya's Controlled English, and machine-oriented CNLs such as ACE. As shown from Table 6, the study showed that many CNLs tend to share common properties. This thesis is based on this analysis, and tries to test it on the abstracts of the Simple Wikipedia articles by comparing the properties of the text with the common CNLs properties identified in O'Brien's study.

Table 6: The most common properties between various CNLs.

Metric	**Common CNL Properties**
Length of sentences	$\leq$ 20 tokens
Active voices	recommended
Lexicon	Use approved words from the Dictionary (controlled lexicon)
Idioms	Do not make noun clusters of more than three nouns

3.9 CHAPTER SUMMARY

The main aim of this chapter is to analyse all the work in the literature for the CNLs and their respective tools. Then, present their limitations, and what are the requirements, questions, and recommendations for next steps to researchers in the domain. In Section 3.2, we presented the history of CNLs and mentioned the different categories of CNLs developed for knowledge representation purposes and for human-human communication. Section 3.3, analyzed all the Semantic Web related CNLs by comparing them in terms of the required input and output formats, parsing approach, the goal behind its development, as well as the pros and cons of each CNL. In Section 3.4, we presented all the developed tools that are based on the previously discussed CNLs, divided into three categories known as ontology engineering tools, ontology querying tools, and other tools. These tools are compared in terms of the required input and output formats of each tool, the parsing approach, the goal behind their development, the advantages over other tools, and their limitations. Section 3.5, presented the evaluations performed of the previously mentioned CNLs as well as a comparative analysis between them, and Section 3.6, provided the evaluations of the tools based on these CNLs with a comparative analysis between these tools. These evaluations were compared in terms of the main development goal, the use case in which the researchers evaluated each one, the evaluation methodology, the number of participants in each evaluation and the output results. In Section 3.7, we reviewed the most popular human-oriented CNLs in the literature. Finally, from our comparative analysis between different CNLs we have concluded the following:

- The evaluations conducted for the CNLs still need more work and further analysis. There is a need for the CNL community to agree on a concrete methodology for the evaluation of CNLs. It was clear that the authors of each CNL or each CNL tool developed a different (some times even ad-hoc) methodology for their respective evaluation according to the available resources. This makes difficult to compare two or more CNLs with each other at a later stage.

- Researchers should make efforts to ensure the optimum number or at least the minimum number of subjects/users is met for both the task-based and paraphrase based evaluations. The optimum number of users for both evaluations is an open question for research. However,

we refer to the research output from [Nie06], where it was found that 5 users is a sufficient number to find most of usability problems, while for Quantitative (aiming at statistics) evaluation, 20 users is a reasonable confidence interval.

- Many evaluations do not include any statistical evidences, that make it difficult to compare with another CNL of interest. With respect to the paraphrased based approaches Ontographs presented in [Kuh13b], may have potential towards a clear process for comparing CNLs. However, the literature did not show much evaluations for different languages other than English.

- Although there has been task based evaluations for some CNLs. We found a need for a comparative study between all the previously evaluated CNLs, and CNLs that has not been evaluated still. Moreover, there was a need from the CNL community for a comparative study across all tools based on CNLs. Hence, we compared all CNLs and tools and presented them. The aim of this study is to help researchers identify the appropriate CNL for a specific task.

- The comparative study should help users to select a specific CNL or tool based on the used NLP approach and either it is shallow or deep. For example, we would not select CLOnE and RABBIT [HJD08] as they are based on shallow NLP approaches, which would not fit our need for a bidirectional, and lexicalised CNL that has deeper linguistic analysis, such as Grammatical Framework (GF) and Attempto Controlled English (ACE). Another requirement might be the amount of support the CNL is providing towards knowledge creation and the target knowledge representation.

- As per our study, we were able to shortlist ACE and GF for our approach. However, after testing a use-case using GF [SGE+15; SGD+16] based on using ACE as an embedded controlled language [Ran14], we found that although GF is based on a functional language, and can support multiple languages, our intuition was that the framework is still not ready to be used for knowledge extraction purposes, and currently the main focus is on supporting multilingual development of CNLs. Furthermore, we decided to go with a platform independent approach, where we can develop rules that can be generalised to other CNLs.

- Although O'Brien study [OBr03] showed that most of CNLs share common properties, there is a need for a computational linguistic analysis and an empirical corpus based evidence to backup this study. The new analysis should exploit the amount of overlap between the properties of human-oriented CNLs and the properties of machine-oriented CNLs, and investigate if they all share O'Brien properties and how far they are from unrestricted text as identified in research questions **RQ1** and **RQ2**.

- There is a lot of content created using human-oriented CNLs especially for generating various technical documentation in different industry sectors such as aerospace, and information technology. All this content have yet to be exploited for knowledge capture. For example, Simple Wikipedia has been explored in the literature for different purposes such as text summarization, text simplification and triples extraction. However, it has not yet exploited by CNL techniques for knowledge capture as a resource of human-oriented CNL that could be rewritten into a machine-oriented CNL, to act as a valuable knowledge base **(RQ3)**. Furthermore, investigating the effect of rewriting with respect to semantic loss **(RQ4)**.

  It is clear that the above gaps link to our research questions identified above, so we move now to tackle them in Chapter 4 and Chapter 5, respectively.

4 A GOLD-STANDARD DATASET FOR HUMAN TO MACHINE ORIENTED CNLS

4.1 INTRODUCTION

In the previous chapter, we discussed the different types of CNLs in details, and we presented some of the most popular machine-oriented CNLs as well as human-oriented CNLs. Furthermore, we presented the current gaps and research questions in the CNLs community. These research questions focus on measuring the overlap between the properties of both types of CNLs (human and machine oriented respectively), presented in Chapter 1 as follows:

- **RQ1:** What is the overlap[1] between the properties of Simple Wikipedia (human-oriented CNL) and the CNLs properties[2]?
- **RQ2:** What is the overlap between the properties of unrestricted text (e.g. Wikipedia) and the common CNLs properties?

As mentioned in Chapter 1 we refer to the CNLs properties as the shared common rules between the different CNLs defined by O'Brien [OBro3] in her study, which are categorized based on their functionality into three categories: *Lexical*, *Syntactic* and *Textual*.

The study concluded that several key CNLs have shared both shared common rules as well as unique rules. Our motivation for choosing the properties that forms the basis of our analysis in the rest of the chapter is based on both the shared and common rules in the CNLs. As shown in Table 7, we summarized these rules and excluded the rules that can not be measured for example it is difficult to have a metric for a rule mentioning that *Relative pronouns such as who, which or that should not be omitted*.

1 The frequency of existence of two or more similar properties.
2 The common properties described earlier that can be found in most of the CNLs, defined by O'Brien in a research study [OBro3].

Table 7: Measurable and unmeasurable CNL rules from O'Brien's CNL analysis that are used to derive the CNL properties.

Measurable CNL rules	**unmeasurable CNL rules**
Keep procedural sentences as short as possible (20 words maximum).	Relative pronouns such as who, which or that should not be omitted.
When appropriate, use an article (the, a, an) or a demonstrative adjective (this, these) before a noun.	Make your instructions as specific as possible
Avoid using gerunds.	
Do not make noun clusters of more than three nouns.	
Use the active voice.	

The comparative analysis between the different CNLs guided us to select the Simple Wikipedia corpus as it is an open access resource representing a well known human-oriented CNL, and the widely adopted ACE CNL to be the machine-oriented CNL as it is domain independent, provides open access to rich set of tools, and can be mapped into OWL triples. Another gap is the need for a gold-standard dataset mapping between both types of CNLs. This dataset can be used by the community to train machine learning models for automatic rewriting, and hence build interesting CNL based applications such as knowledge based machine translation, ontology development, logic programming, language learning, etc.

This Chapter focuses on our experiments with respect to the computational linguistic analysis, as well as the investigation of the feasibility of rewriting Simple Wikipedia abstracts into ACE CNL [FKKo8a]. Since no ground truth exists, it has been necessary to engineer one for our experiments. We present a linguistic resource[3], that is both human-readable and semantically machine interpretable. This resource is a snapshot taken from the abstracts of the Simple English Wikipedia dump[4]. The selected abstracts are rewritten into a machine-oriented CNL, by applying some rules on the syntactic structures of the sentences to be accepted and parsed by the CNL semantic parser. To our knowledge this resource is the first human to machine CNL aligned dataset. The chapter is structured as follows, Section 4.2 introduces a preliminary case study and a manual analysis for a sample extracted from a human-oriented CNL corpus, Section 4.3 presents the process for the collection, processing, and analysis of the Simple Wikipedia abstracts corpus. In Section 4.4, we discuss the output results from the

3 https://drive.google.com/open?id=1eBUXZ8tESIML3jEptqG4aci4-_BmODkP
4 https://dumps.wikimedia.org/simplewiki/

analysis and the validation of the generated resource, Section 4.5 describes some related work, and finally, Section 4.6 offers a conclusion.
References: Parts of this chapter are based on our publications [SDZ18] and [SZD17].

4.2 PRELIMINARY CASE-STUDY

4.2.1 The Medline Summaries of Diseases Website

As a preliminary study we choose Medlineplus as a data source[5], which includes over 1000 health topic pages and is not available as a dump for download, so we randomly extracted **one summary** for a preliminary manual analysis. This resource contains summaries of common diseases. Each summary includes several sentences about the disease explanation, symptoms, causes, diagnosis, and treatment. Each summary ranges from 10 to 20 sentences. We selected this resource specifically, as it fulfills the properties of a simplified language corpus. The Medlineplus summaries are written using style guides[6] to make the text easy to understand by any person regardless of age, background and reading level. Since the medical concepts and language are usually complex and difficult to read, the aim of the style guides is to help the authors write an easy to read health materials. The style guides involve organizing the writing to keep it within the range of 7^{th} or 8^{th} grade reading level. This could be achieved by finding alternatives for complex words or abbreviations, avoiding abstract syntax language, limit sentences length to be between 10 and 15 words, and using the active voices instead of the passive ones.

4.2.2 Preliminary Manual Analysis

The approach is divided into two phases, the pre-processing phase for preparing the sentences, constructing the ACE and domain lexicons, then generating parse trees using a general parser e.g. Stanford parser[7]. The second phase is the post-processing, responsible for rewriting the simplified

5 https://www.nlm.nih.gov/medlineplus/healthtopics.html
6 https://www.nlm.nih.gov/medlineplus/etr.html
7 http://nlp.stanford.edu:8080/parser/index.jsp

sentences into ACE sentences, and this will involve inserting, replacing, or changing the order of some chunks in the sentence.

In this section, we will go through our analysis of a sample disease from the corpus. We selected a random Medline summary Anemia disease. In the pre-processing phase, we segmented the text to obtain separate sentences. The whole text was normalized to a standard format, for example we removed the apostrophes, bullet points, hyphens and etc. This will generate a corpus of separate sentences that is ready for further processing. Then, we applied a Part Of Speech (POS) tagger on the generated corpus, to identify the structure of each sentence. Since, the ACE grammar can not process a constituent of two or more consecutive nouns, we extracted these constituents to chunk them into one complex noun phrase and process it as a multi-word unit. These units will be added to the domain and ACE lexicons. The next step, is to look for the pronouns in the text and apply a coreference resolution clustering approach (e.g. the pronoun ”it” refers to the noun ”Hemoglobin”) to match different names describing the same entity. After applying the normalization and the coreference resolution tasks, each sentence will be parsed using a natural language parser (e.g. Stanford parser) generating the syntax trees. These trees will be needed for further analysis in the post-processing phase. The Anemia sample corpus includes 10 sentences, each sentence will have more than one ACE alternative in the rewriting process.

Table 8: A table showing a parse tree of a Medline Simplified English sentence and its reconstructed ACE parse tree after applying rewriting rules.

Parse tree of the SE sentence	**Parse tree of the rewritten sentence in ACE CNL**
(ROOT (S (NP (PRP Your)(NN body)) (VP (VBZ needs)(NP (NN iron)) (S (VP (TO to)(VP (VB make)(NP (NN hemoglobin)))))))	(ROOT (S (NP (DT the) (NN body)) (VP (VBZ needs) (NP (NP (DT the)(NN iron)) (PP (IN for)(NP (NP (DT the) (NN making)) (PP (IN of)(NP (DT the)(NN hemoglobin))))))))

An example of the parse tree generated from the Stanford parser for a sample Medline Simplified English sentence *your body needs iron to make Hemoglobin*, and the parse tree of its ACE CNL alternative *the body needs the iron for the making of the Hemoglobin* after applying the rewriting ACE rules

are shown in Table 8. The rules applied here are as follows: 1) remove all the possessive adjectives, 2) each noun should be preceded by an article or a quantifier 3) ACE grammar does not support *to+verb* constituent, so it has to be replaced by another alternative.

Table 9: A Table showing the transformation from the Medline Simplified English grammar fragment into ACE grammar.

Constituent	Grammar	ACE grammar	Constituent in ACE
`your body`	`PRP+NN`	`DET+NN`	`the body`
`iron`	`NN`	`DET+NN`	`the iron`
`to make`	`TO+VP`	`PREP+DET+NN+PREP`	`for the making of`
`hemoglobin`	`NN`	`DET+NN`	`the hemoglobin`

In Table 9 we show in the first column, some nominated constituents of the Medline Simplified English sentence that should be potentially rewritten into ACE. In the second column, the grammar of these constituents is represented in the form of their POS tags form. Using the rewriting rules, the grammar of these constituents should be mapped into their ACE grammar alternatives presented in the third column. In the last column we present the expected ACE constituents that should be generated by the system. Finally, as shown in Figure 21 the new reconstructed ACE sentence will be parsed with the ACE engine to generate the DRS.

PARAPHRASE

There is a body X1.
The body X1 needs an iron for a making of some hemoglobin.

DRS

```
[A,B,C,D,E]
object(E,body,countable,na,eq,1)-1/2
object(D,iron,countable,na,eq,1)-1/5
relation(B,of,C)-1/9
object(B,making,countable,na,eq,1)-1/8
predicate(A,need,E,D)-1/3
modifier_pp(A,for,B)-1/6
object(C,hemoglobin,mass,na,na,na)-1/11
```

Figure 21: The DRS representation from the APE engine.

As shown from Table 9, the rewriting of a Medline Simplified English sentence into ACE CNL may require some modifications in the syntactic structure of the sentence. However, these modifications can change the semantics. So, we extracted all the patterns from the sample corpus that need to be changed, in order for the sentence to be ACE. All the transformed SE sentences from the sample corpus and their ACE alternatives are shown in Table 10.

From Table 10 we can conclude the following grammar rules:

1. Possessive adjectives "you" can not be translated into ACE, so we replace it with a noun "person".
2. Possessive adjectives "your" can not be translated into ACE, so we replace it with an article.
3. Each noun has to be preceded by an article or a quantifier.
4. Each if statement should be transformed into if-then statement.
5. Successive nouns has to be combined into one noun.
6. The constituent "to+verb" is not supported in ACE grammar, so it has to be replaced by "for+the+verb+of".
7. Punctuation: use of comma, colon,semicolon, quotation marks, and parentheses as inter- and intra-sentential punctuation should be replaced with the right terms.
8. Not all modalities are supported such as *will*, so it has to be replaced by an alternative.

In Table 11 we present the grammar rules that are required to transform a Simplified English sentence from the Medline corpus into ACE CNL. Although the Medline corpus of diseases can be considered a Simplified English corpus, as it is based on a similar style-guides of the Basic English, the corpus is domain specific. Hence, the extracted grammar rules can not be generalized to different wider domains. So, in Section 4.4 we decided to perform the rest of the analysis on a corpus that is easier to get access to more data, domain independent, general purpose, includes all the stylistic properties of other sort of human-oriented CNLs such as Simple Wikipedia. We believe all the extracted rules can be generalized to more domains if they have the same linguistic properties, that will be discussed in details in the next section.

Table 10: A Table showing the sample corpus, the selected ACE alternatives and their DRS paraphrases.

Sentence	Medline SE sentence	ACE alternative	DRS paraphrase
1	If you have anemia your blood does not carry enough oxygen to the rest of your body.	if (a person) (has) Anemia (then) (the blood) does not carry (the enough oxygen) to the rest of (the body).	If a person has Anemia then there is a blood X1 and it is false that the blood X1 carries some enough oxygen to a rest of a body.
2	The most common cause of anemia is not having enough iron.	The most common cause of Anemia is (no) enough iron	There is a most common cause X1 of Anemia.If there is an enough iron X2 then it is false that the most common cause X1 is the enough iron X2.
3	Your body needs iron to make hemoglobin.	(the body) needs (the iron) (for the making of) (the hemoglobin).	There is a body X1. The body X1 needs an iron for a making of some hemoglobin.
4	Hemoglobin is an iron rich protein that gives the red color to blood.	Hemoglobin is an (n:iron-rich-protein) that gives the red color to (the blood).	There is an n:iron-rich-protein X1. Hemoglobin is the n:iron-rich-protein X1. The n:iron-rich-protein X1 gives a red color to a blood.
5	Hemoglobin carries oxygen from the lungs to the rest of the body.	Hemoglobin carries (the oxygen) from the lungs to the rest of the body.	Hemoglobin carries some oxygen from at least 2 lungs to a rest of a body.
6	Anemia has three main causes blood loss and lack of red blood cell production and high rates of red blood cell destruction.	Anemia has three main causes (that are) (the n:blood-loss) (and) (the lack of the red n:blood-cell-production) and (the high rates) of (the red n:blood-cell-destruction).	There are 3 main causes X1.Anemia has the main causes X1.The main causes X1 are a lack of at least 2 high rates of a red n:blood-cell-destruction and a red n:blood-cell-production and a n:blood-loss.

7	Anemia can make you feel tired, cold, dizzy, and irritable.	Anemia can make (a person) (a tired person) and (a cold person) and (a dizzy person) and (an irritable person).	There is some anemia X1. It is possible that the anemia X1 makes a person a tired person and a cold person and a dizzy person and an irritable person.
8	You may be short of breath or have a headache.	(the person) may (have) (a short breath) and (a headache).	There is a person X1. It is possible that the person X1 has a short breath and a headache.
9	Your doctor will diagnose anemia with a physical exam and blood tests.	(the doctor) (can diagnose) Anemia with a physical exam and (some n:blood-tests).	There is a doctor X1. It is possible that the doctor X1 diagnoses some anemia with a physical exam and at least 2 n:blood-tests.
10	Treatment depends on the kind of anemia you have.	Treatment depends on the (type) of Anemia.	There is a treatment X1. The treatment X1 depends on a type of some anemia.

Table 11: The grammar from each sentence, the ACE grammar conversion and the reference rule applied from the list

Sentence	Converted constituents	Converted Grammar	Grammar Rules
1	you → a person have anemia → has the anemia your blood → the blood the enough oxygen → enough-oxygen your body → the body	PRP → DT+NN. VBP+NN → VBZ+NN PRP+NN → DT+NN JJ+NN → DT+JJ+NN PRP+NN → DT+NN	1,2,3,4,5
2	not having → no	RB+VBG → DT	3
3	your body → the body iron → the iron to make → for the making of hemoglobin → the hemoglobin	PRP+NN → DET+NN NN → DET+NN TO+VP → PREP+DET+NN+PREP NN → DET+NN	2,3,6
4	iron rich protein → iron-rich-protein blood → the blood	NN+JJ+NN → NNS NN → DT+NN	3,5
5	hemoglobin → the hemoglobin oxygen → the oxygen	NN → DT+NN NN → DT+NN	3
6	blood loss → the n:blood-loss lack → the lack blood cell production/destruction → n:blood-cell-production/destruction high rates → the high rates	NN+NN → DT+NNS NN → DT+NN. NN+NN+NN → DT+NNS JJ+NN → DT+JJ+NN	3,5,7

7	you → a person feel tired → a tired person cold → a cold person dizzy → a dizzy person irritable → an irritable person	PRP → DT+NN VB+JJ → DT+JJ+NN	1,3,7 X+feel+Y→ if+NP+VP+X+ then+NP+VP+Y
8	you → the person may be → may have short of breath → a short breath headache → a headache	PRP → DT+NN MD+VB → MD+VB JJ+IN+NN → DT+JJ+NN NN → DT+NN	1,3 or → and
9	your doctor → the doctor blood tests → some blood-tests	PRP+NN → DT+NN NN → DT+NNS	2,3,5 will → can
10	-	-	kind → type

4.3 SIMPLE WIKIPEDIA CORPUS ANALYSIS

In order to test our intuition with respect to which type of text overlap more with CNLs properties and to what extent, we decided to analyze the presence of the common CNLs properties identified in [OBro3], in the unrestricted text (Wikipedia) and in the simplified text (Simple Wikipedia). In regard to the feasibility of rewriting Simplified Wikipedia sentences to a CNL, our first assumption is that the simplified text should be logically less ambiguous and less complex than standard Wikipedia unstructured text. Consequently, its linguistic properties will overlap significantly more with CNLs than unstructured text.

4.3.1 Corpus Collection and Pre-processing

All experiments are performed on Simple Wikipedia abstracts and their corresponding Wikipedia abstracts after collecting the dumps of all the abstracts in both Wikipedias. Table 12, describes the style guides from Simple Wikipedia on how to write simplified text versus several common stylistic properties observed across several CNLs (both human and machine oriented) [OBro3]. From the Table we can see that both texts significantly overlap in most of the properties.

Table 12: Comparison showing the overlap between CNL rules and Simplified Text rules

Metric	Common CNL Rules	Simplified Text Rules
Length of sentences	$\leq$ 20 tokens	Keep sentences short and informative
Active voices	recommended	recommended
Lexicon	Use approved words from the Dictionary (controlled lexicon)	Basic English Word-list
Idioms	Do not make noun clusters of more than three nouns	Avoid idioms (multi-words)

We collected the XML formatted dumps containing every Simple Wikipedia abstract and its corresponding Wikipedia abstract. Then, we converted the XML format into JSON[8] format. After that we cleaned the text using regular expressions which includes removing special characters, remove incomplete

8 Java Script Object Notation

or blank sentences and abstracts, text between brackets, etc. In Table 13, we show all the steps performed for the pre-processing of the dumps.

Table 13: Corpus collection and Pre-processing steps

Metric	**Simple Wikipedia**	**Wikipedia**
Corpus	Abstracts of all articles (extracted from the dump)	Abstracts of parallel Simple Wikipedia articles
Format	XML dump converted to JSON format	
Pre-processing	Text cleansing using regular expressions (e.g. remove special characters, very short sentences, blank abstracts, text between brackets..etc)	
	Split each abstract into sentences (i.e sentence segmentation)	
	Split each sentence into tokens (i.e tokenization)	
	Run Part of Speech tagging (POS) using NLTK tagger using Penn Treebank Tag Set [MMS93] over each sentence and and create a list of POS tags for each sentence in the corpus (POS structures list).	

Table 14: Comparison between Simple Wikipedia & parallel Wikipedia abstracts

Metric	**Simple Wikipedia**	**Parallel Wikipedia**
No. of abstracts before cleaning	74,067	N/A
No. of abstracts after cleaning	48,880	27,539
Total No. of sentences in the corpus	87,088	39,252
Total No. of tokens in the corpus	968,231	586,732

In Table 14, we show a comparison between the two dumps before and after cleaning. The total number of abstracts before cleaning was around 74k in the Simple Wikipedia dump. Since we had not yet cleaned the dump, we did not count the parallel Wikipedia abstracts. After cleaning the data, we extracted around 48.8k abstracts from Simple Wikipedia and in parallel we found around 27.5k abstracts in Wikipedia as some of the abstracts are found blank, incomplete or missing. The total number of sentences extracted from the cleaned Simple Wikipedia was around 87k, including 968.2k tokens, with most of the abstracts including two sentences. In parallel,

the total number of sentences extracted from the cleaned Wikipedia dump was around 39.2k, including around 586.7k tokens.

4.3.2 Analyzing Common CNLs Properties

In order to measure the common CNLs properties, we analyzed the measurable properties presented in Table 7 that should be present in the Simplified English text. As shown in Table 15, the first metric is the length of sentences (number of tokens/sentence). In Simple Wikipedia we found that more than 90% of the sentences does not exceed the 20 tokens. On the other side, sentences from the parallel Wikipedia abstracts are usually exceeding this limit. Although the guidelines for writing Simple Wikipedia abstracts recommended the authors to avoid using passive voices, we found that 34% of the sentences did not follow this rule. On the other hand, 51% of the sentences in Wikipedia articles, are written using the passive voice. Gerunds are the words that are formed with verbs but act as nouns e.g go swimming, were found to be 6% in Simple Wikipedia sentences and 21% in Wikipedia sentences. CNLs usually use determiners before nouns, so our test found that the tags which preceded nouns in the Simple Wikipedia sentences are ranked as follows: 1) Determiners, 2) Noun Phrases, 3) Prepositions, 4) Adjectives, but in the Wikipedia sentences the list was different as follows 1) Nouns, 2) Noun Phrases, 3) Prepositions, 4) Determiners. Moreover, noun clusters are found in 4% of the Simple Wikipedia sentences and 8% in the Wikipedia sentences. Hence, based on the observations above, we can confirm our hypothesis that the linguistic properties of Simplified Wikipedia text overlap more with CNLs than unstructured text.

Table 15: Results of analysing the CNL properties across Simple Wikipedia and Wikipedia sentences

Metric	**Simple Wikipedia**	**Parallel Wikipedia**
Maximum Tokens/sentence $\leq$20	Yes	No
Passive voices	34%	51%
Gerunds	6%	21%
Articles preceding nouns	`DT,NNP,IN,JJ`	`NN,NP,IN,DT`
Noun clusters	4%	8%

Based on the results above, we conducted a deeper analysis of the POS tags structures of the Simple Wikipedia sentences which overlapped

Table 16: Results of extracting the Simple Wikipedia abstracts that follow the CNL rules.

Metric	Result	Percentage from the total corpus
No. of SE abstracts that are fully overlapping with the common CNL properties.	20,647	42.2%
Total No. of Sentences	36,560	42%
Total No. of Tokens	383,555	39.6%

completely with the CNL rules. This meant extracting all abstracts which follow the common CNLs rules identified by O'Brien [OBr03] in Table 12 from the original dataset dump, excluding the remainder. As shown in Table 16, the total number of abstracts from the Simple Wikipedia dump that follow the CNL rules are found to be around 20.6k. These abstracts include around 36.5k sentences, with 383.5k tokens.

4.4 CORPUS ANALYSIS RESULTS

Since, Simplified English rules and style guides request authors to use preferred sentence forms[9], such as `Subject - Verb - DirectObect`, and `Subject - Verb - IndirectObject`, our second hypothesis (H2) is that *the shared common rules between the different CNLs in O'Brien's analysis can be used to extract corpus based syntactic patterns in the sentence structures through their POS tags structures to rewrite a human-oriented CNL sentence into a machine-oriented CNL sentence, such that the newly rewritten machine-oriented CNL sentence preserves the semantics of the original source human-oriented CNL sentence.* So, we created a dictionary to cluster similar POS tags structures into individual groups. The main aim of creating this dictionary, is to discover to which extent the authors of the Simple Wikipedia abstracts followed the recommended writing guidelines. We grouped all similar POS tags structures together, in order to distinguish between the percentage of unique and repeated POS tags structures per group in the whole corpus. This would thus help approximate the number of rewrite rules needed to map Simplified Wikipedia sentences into ACE CNL. We found a total number around 22k **unique** POS tags structures. Then, we estimated the number of

9 https://simple.wikipedia.org/wiki/Wikipedia:How_to_write_Simple_English_pages

sentences that belong to each POS tags structure. As shown from Table 17, we present five cases from the dictionary. For example, the analysis shows that around 7.8k sentences belong to the same group (**group 1**) containing 102 different POS tag structures, and around 12.6k sentences belong to another group (**group 5**) containing 629 different POS tags structures. This analysis indicates that the corpus contains a significant number of sentences which may only belong to a small group of POS tags structures. Hence by grouping them, we require less linguistic patterns for developing rewriting rules. Although Table 17 shows that there are some repeated POS tags structures in the corpus, the ratio between the number of sentences and POS tags structures means that in order to rewrite 12,630 sentences into ACE CNL we need to implement rules that can cover **group 5** which represents the 629 different POS tags structures within a given cluster, which is a lot of rules that will lead to rewriting only 34.5% of the corpus.

Table 17: Grouping sentences that belong to the top five repeated POS tags structure groups in ascending order.

POS tags structure group	Total No. of POS tags Structures per group	Percentage from the total No. of POS tags Structures	Total No. of sentences	Percentage from the total No. of sentences
1	102	0.4%	7,809	21.3%
2	115	0.5%	8,080	22.1%
3	182	0.8%	9,203	25.1%
4	289	1.3%	10,460	28.6%
5	629	2.8%	12,630	34.5%

In Table 18, we show our analysis after extracting the top five most repeatedly matching individual POS tags structures in the corpus. We experimented by rewriting sentences that belong to POS tags structure number 1, which includes the largest number (634) of sentences. We found that the noun clusters is a common pattern in these sentences. So, to rewrite these sentences into ACE, we need to chunk these nouns. The POS tags structure number 4, could be rewritten using a noun phrase chunker in the beginning of the sentence to chunk the names as one noun, and another noun phrase chunker in the end of the sentence to chunk the combination consisting of a noun followed by an adjective into one chunk. Although it can be shown from Table 18 that POS tags structures numbers 2, 3 and 5 are basic and did not require a rule for rewriting them to be accepted by the ACE parser or converted into DRS, the percentage of these is still very low.

Table 18: A Table showing the Simple Wikipedia sentences that belong to the top 5 dominating POS tags structures and their subsequent translation into ACE CNL.

No.	Individual dominating POS tags structures	Count	Example sentences	SE sentences translation in ACE CNL
1	NNP VBZ DT NN IN NNP IN DT NNP NNPS	634	- Macon is a city of Illinois in the United States. - Agency is a city of Iowa in the United States.	- Macon is a city of Illinois in the n:United-States. - Agency is a city of Iowa in the n:United-States.
2	NNP VBZ DT NN IN NNP	295	- Aalborg is a city in Denmark. - Helene is a moon of Saturn.	- Aalborg is a city in Denmark. - Helene is a moon of Saturn.
3	NNP VBZ DT NN	199	- Thirteen is a number. - Waitby has a castle.	- Thirteen is a number. - Waitby has a castle.
4	NNP NNP NNP VBD DT JJ NN	196	- Guy Henry Ourisson was a French chemist. - Edna May Oliver was an American actress.	- Guy-Henry-Ourisson is a n:French chemist. - Edna-May-Oliver is an n:American-actress.
5	NNP VBZ DT JJ NN	149	- Adelite is a rare mineral. - Zugzwang is a chess term.	- Adelite is a rare mineral. - Zugzwang is a chess term.

4.4.1 Resource Annotation and Validation

So, in order to overcome this problem we found from our analysis that a common pattern in Table 18, most of POS tags structures contain noun clusters and/or adjective noun clusters - noun phrase chunks effectively. So, we applied a chunking rule on all the sentences in the corpus and assigned them a new POS tag `NP_CHUNK`. A set of new POS tags structures (new groups) is created and the results from this analysis is summarized in Table 19. In the Table, we notice that around 19,089 sentences are captured together under one group that includes 300 different POS tags structures, out of total 13,867 different POS tags structures. This means that if we can write linguistic pattern rules that can capture these 300 different POS tags structures, we would be able to map around 19.1k (52.2%) Simple English sentences into ACE CNL format, and consequently into the DRS formal representation which could than be exported to OWL.

Table 19: Grouping sentences that belong to the same POS tags structure group after chunking.

No of new POS tags structures	Percentage from the total No. of POS tags Structures	Total No. of sentences captured	Percentage from the total No. of sentences [10]
300	4.5%	19,089	52.2%
605	9%	21,203	58%

We can see from Table 20 some of the results after extracting the total of around 19,089 sentences that belong to the 300 different POS tags structures from the corpus, then applying the rewriting rules. The Table shows the top five most dominating individual POS tags structures after the chunking took place. The highest occurring new POS tags structure represented 2664 sentences, where the rewriting rule chunked the nouns at the end of the sentence. The **second** most dominating structure represented 1399 sentences, where the rewriting rule chunked the nouns at the beginning and of the sentence. Structure number 3 represented 1088 sentences, and the rewriting rule chunked the nouns at the end of the sentence. Structure number 4, represented 926 sentences and the rewriting rule chunked a noun or group of nouns, followed by POS tag IN, followed by a noun/group of nouns. The last POS tags structure represented 885 sentences, where the past verb of the sentence is rewritten into the present tense, and the nouns at the beginning and end of the sentence are chunked.

Table 20: A Table showing the sentences that belong to the top 5 dominating individual POS tags structures (after chunking) rewritten into ACE CNL.

No.	Individual dominating POS tags structures	SE sentences Count	SE sentences examples	Rewritten sentences in ACE CNL
1	NP_CHUNK VBZ DT NP_CHUNK IN NP_CHUNK	2664	- Alkmene is a person in Greek mythology. - Anyang is a city in South Korea.	- Alkmene is a person in the n:Greek-mythology. - Anyang is a city in the n:South-Korea.
2	NP_CHUNK VBZ DT NP_CHUNK	1399	- Alicia Bridges is an American singer. - Blood transfusion is a medical term.	- Alicia-Bridges is an n:American-singer. - Blood-transfusion is a n:medical-term.
3	NP_CHUNK VBZ DT NP_CHUNK IN NP_CHUNK IN DT NP_CHUNK	1088	- Gilbert is a city of Iowa in the United States. - Iphiclides is a genus of Butterflies in the family Papilionidae.	- Gilbert is a city of Iowa in the n:United-States. - Iphiclides is a genus of Butterflies in the n:family-Papilionidae.
4	NP_CHUNK VBZ DT NP_CHUNK IN DT NP_CHUNK IN NP_CHUNK	926	- Anhui is a province in the east of China. - Agriculture is an important part of the economy of Azerbaijan.	- Anhui is a province in the n:east-of-China. - Agriculture is an important part of the n:economy-of-Azerbaijan.

5	NP_CHUNK VBD DT NP_CHUNK	885	- Abel Ricardo Laudonio was an Argentine boxer. - Braniff International Airways was an American airline.	- Abel-Ricardo-Laudonio is an n:Argentine-boxer. - Braniff-International-Airways is an n:American-airline.

4.5 RELATED WORK

The work from [XCN15] analysed the text in Simple Wikipedia and Wikipedia linguistic features to produce a new resource corpus that can help sentence simplification research. It provides a new simplification dataset that is an improved version over Simple Wikipedia. Their analysis over the text in both Simple Wikipedia and Wikipedia was done for the purpose of showing the amount of simplification between two parallel sentences from both Wikipedias, and introduce a new comparative approach to simplification corpus analysis. The main difference between our work is that we analyse both Wikipedias for the purpose of rewriting simplified English into a formal knowledge using CNLs as they tend to be less ambiguous.

In the context of human-oriented CNL for the medical domain, the authors in [EY09] present an approach to convert a biomedical query written in a CNL named BIOQUERYCNL, into a program in a knowledge representation and reasoning paradigm called Answer Set Programming (ASP) [Bar03] via Discourse Representation Structures (DRS) [KR13]. The advantage of this is to automate reasoning about biomedical ontologies using the ASP solvers. Our work is different, as we focus on the knowledge creation and representation and we do not include the querying part. Other approaches studied transforming the Natural Language (NL) into ASP include [BDT07], where the authors present a research to transform simple sentences into ASP using Lambda calculus. As well as, the work presented in [Son08], where the authors use the Boxer framework [CCB07] to perform semantic analysis over the output of the C&C parser, and then the reasoning is done using ASP. Although Boxer/C&C tools can parse and create a semantic representation in DRS for a natural language, the parser will not perform well in the any domain since it was trained on newspaper text corpus [Bos08].

In [BBE+12] the author, presents a semantically annotated corpus developed using a bootstrapping approach using NLP techniques for parsing and semantic interpretation, together with an interface for collaborative annotation of experts and crowd-sourcing community. Although they generated a semantic resource with deep semantics, the resource is small as it includes less than 5k sentences which is much smaller than ours.

The other resources of ProPBank [PGK05] and Framenet [BFL98] are semantically annotated corpora. However, they do not combine the various layers of linguistic annotation into one common (Semantic Web, ontology

aware) formalism [BBE+12], which is needed for our knowledge capture task.

The majority for knowledge extraction and machine reading is focusing around Ontology Learning and Population [BCM05]. Approaches usually involve machine learning techniques which require large corpora involving the usual challenges in supervised learning such as the costly and manual annotation of training data.

In [WL14] the authors present a text rewriting approach to increase the amount of labeled data available for model training. They analysed Simple Wikipedia and Wikipedia parallel corpora to automatically extract rewrite rules, then generate multiple versions of sentences annotated with gold standard labels for the purposes of semantic role labeling. Our work is similar to them in that we also use rewrite rules to generate a gold standard resource. However, the main difference is that we use rewrite rules to generate CNL sentences from Simple Wikipedia sentences, while their task is focused on semantic role labeling.

Our approach is different from the previous existing approaches in terms of being completely based on rules, and does not rely on machine learning methods, since machine-oriented CNLs by design do **not** circumvent linguistic disambiguation and can be deterministically parsed into knowledge structures.

4.6 CHAPTER SUMMARY

The work in this chapter helped us to answer both research questions named RQ1 and RQ2, as well as validate our hypothesis H1 stating that *Lexical, syntactic and textual properties derived from the shared common rules between the different CNLs in O'Brien analysis have higher statistical presence in Simplified text than unstructured text due to the recommended style guides imposed on authors to simplify the text.* From the analysis presented in Table 15, we have showed that the hypothesis is right as SE text overlap more with common CNL properties than unrestricted text, in all the measurable properties metrics. Hence, we can conclude that rewriting Simple Wikipedia abstracts to a machine-oriented CNL is an achievable task. In addition, we were able to produce and present a gold-standard dataset that is both machine readable

and human understandable. The resource is available for download[11], and includes the 17,749 validated sentences from the original Simple Wikipedia abstracts and their parallel sentences rewritten in ACE. This should be a very valuable resource for the following reasons:

- The first gold-standard dataset mapping between human-oriented CNL sentences and its aligned machine-oriented CNL sentences.
- This dataset can be used for exploitation in different applications related to interesting CNL based applications such as knowledge based machine translation, ontology development, logic programming, and text simplification.

This chapter is summarized as follows, Section 4.1, introduced and motivated the problem. In Section 4.2, we presented our initial manual analysis and a case-study for a sample text from the Medline summaries of diseases corpus that can be considered as human-oriented CNL text. This analysis helped us to identify a rewriting approach from a human-oriented CNL into a machine-oriented CNL, using rules based on the POS tags structures. In Section 4.3, we show the different steps required for the comparative and statistical analysis of the common CNLs properties between the Simple Wikipedia abstracts and the Wikipedia abstracts. These steps involved data collection, then applying corpus pre-processing using different NLP techniques. Section 4.4 shows the results of the corpus analysis after the annotation and validation steps, and the initial evaluation of the resource based on the system coverage and the semantic loss. Finally, in Section 4.5, we presented some work from the literature that analysed and compared the text in Simple Wikipedia versus the text in Wikipedia for various purposes, and other literature work who presented textual resources such as semantically annotated corpora. In Chapter 5, we will present the detailed architecture of the rule based rewriting system and discuss the implementation of each rewriting rule. In addition to, a more detailed evaluation for the coverage and the semantic loss results of the approach when applied on a test set.

11 https://drive.google.com/open?id=1eBUXZ8tESIML3jEptqG4aci4-_BmODkP

5 FROM SIMPLIFIED TEXT TO CONTROLLED NATURAL LANGUAGE FOR KNOWLEDGE EXTRACTION

5.1 INTRODUCTION

In the previous chapter, we presented a linguistic resource mapping between a statistically representative corpus of human-oriented CNL represented in Simple Wikipedia sentences, and the ACE machine-oriented CNL. Such a dataset can be exploited for developing a variety of CNL based applications for knowledge creation. Such easily deterministically parsed language content reduce or eliminates the need for these statistical and machine learning models, taken into consideration the inherent less ambiguous nature of human/machine orientated CNLs. Furthermore, we presented corpus statistics concerning the the resource and compared between the Simple Wikipedia and the Wikipedia abstracts to measure the overlap of the common CNLs properties in both corpora.

In this chapter we will discuss in more details the architecture and the methodology to build this resource and the feasibility of rewriting Simplified English into a ACE CNL [FKK08a]. The SE selected abstracts are rewritten into ACE, by applying rules on the syntactic structures of the SE sentences such that they are accepted and parsed by ACE parser[1]. We conduct a series of experiments to test the following research questions:

- **RQ3:** What is the feasibility[2] of rewriting human-oriented CNL (Simplified text) into machine processable text (e.g. ACE CNL)?
- **RQ4:** What is the effect of rewriting with respect to semantic loss?

1 https://github.com/Attempto/APE

2 Refers to the coverage of the rewriting system in terms of the percentage of sentences that could be successfully rewritten.

The rest of this chapter is structured as follows: Section 5.2 describes the system architecture, the main approach and the developed rules, Section 5.3 presents the knowledge extraction process in the form of DRS and OWL triples. In Section 5.4, we discuss the results and the evaluation, and finally, Section 5.5 offers a conclusion

References: Parts of this chapter are based on the publications [SDZ18] and [SZD18].

5.2 THE REWRITING APPROACH AND RULES

As discussed in Chapter 4, our approach is mainly based on rules that are developed after extracting patterns from the POS tags structures of each sentence. As shown in Table 21, the system includes 10 rules that are developed using different NLP techniques. The rules can be applied to extract DRS structures and OWL triples from text, however we separated the two processes in the next experiments to measure the coverage of the approach for the purposes of extracting i) DRS statements, and ii) OWL triples, respectively.

Table 21: A table showing the rules used to rewrite the simplified text sentences into ACE CNL.

Rule	Description
1	Chunk nouns and/or adjectives
2	Insert determiners before noun/adj chunks
3	Coreference resolution on personal/possessive pronouns.
4	Rewrite past verbs into present
5	Modify upper/lower case of noun and/or noun/adj chunks
6	Chunk nouns separated by POS tag IN e.g. republic-of-India.
7	Convert unparsed numbers into words e.g. 3rd
8	Resolve Pre-determiners in beginning of sentences e.g. all
9	Resolve/Replace particles e.g. since, for, about..etc
10	Resolve/Replace adverbs e.g. occasionally

5.2.1 System Architecture

The system represented in the architecture shown in Figure 22 is implemented and applied on a sampled dataset of 19,089 sentences that was

Table 22: Description of the different modules in the architecture.

Phase	Modules	Description
Pre-processing	Segmentation	Divide the abstract into separate sentences based on fullstops, which marks the end of a sentence.
	Normalization	Sentences are normalized to a standard format, for example we removed the apostrophes, bullet points, hyphens and etc.
	POS tagging	Apply the NLTK Part of Speech (PoS) tagger on the sampled dataset, to identify the POS tag structure of each sentence.
	Chunking	Since, the ACE grammar can not process a constituent of two or more consecutive nouns/adjectives, we extracted these constituents to chunk them as one chunk and process it as a multi-word unit. This will generate a new set of POS tags structures that will represent more SE sentences.
	Grouping	Similar POS tags structures are regrouped into the new generated POS tags structures, and stored into POS tags structures list. This list will be needed for further analysis in the post-processing phase.
Post-processing	Rewriting rules	Rewrite the SE sentences into ACE sentences, and this will involve inserting, replacing, or changing the order of some predicates in the sentence.
	ACE validation	Validate that the generated ACE sentences could be parsed with APE engine. If the ACE sentence is not acceptable by APE, iterate over the next ACE rewriting rule.
	DRS extraction	Map the rewritten ACE sentence into DRS formalism.
	OWL triples extraction	If possible, extract an OWL triple from the rewritten ACE sentence, and store it in a triple store.

extracted from the main dataset. As mentioned in Chapter 4, this sampled dataset includes only the sentences that follow the CNL rules, and the sentences are captured together under one group that includes 300 different POS tags structures. As shown in Table 22, the architecture is divided into two phases namely the *pre-processing phase*, for preparing the sentences, and generating POS tags structures list, and the *post-processing phase*, responsible for rewriting of the simplified sentences into ACE sentences. The whole process will involve a percentage of information loss which will be evaluated based on the percentage of successfully rewritten sentences. Figure 22 shows that in the post-processing phase each sentence will be in the form of a syntax tree. At this stage the ACE rules are applied on each tree to reconstruct it into an ACE tree.

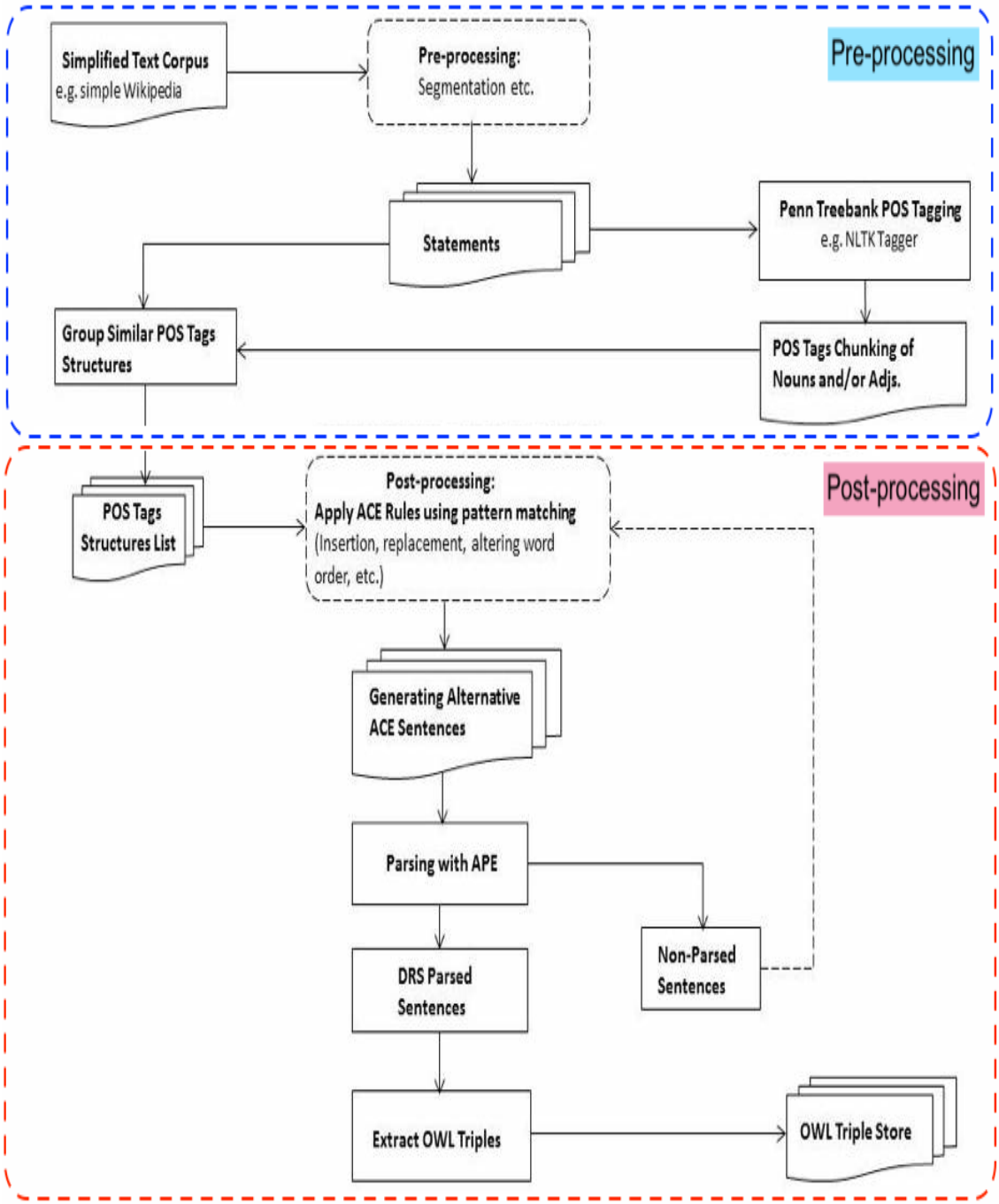


Figure 22: System architecture for rewriting simplified text into ACE CNL to extract OWL Triples.

For instance, in ACE each noun should be preceded by an article or a quantifier (e.g. the, each, every, all, etc) which means that we have to insert an article or quantifier before each noun. In the following sentence *your body needs iron to make hemoglobin*, there are three nouns *body, iron, hemoglobin*. The noun *body* is preceded by a possessive *your* that needs to be replaced by either an article or a quantifier in order to follow ACE grammar rules. Also, both nouns *iron* and *hemoglobin* need to be preceded by either an article or a quantifier. In addition to, the constituent *to make* is not acceptable in ACE grammar. Thus, one alternative would be to replace the constituent *to make* with *for the making of*. Finally, the whole sentence will require 2 replacements and 2 insertions. A possible reconstructed ACE sentence that is acceptable by the ACE grammar and processable by the APE engine could be *the body needs the iron to make the hemoglobin*.

5.2.2 The Rewriting Rules

The process involved a pre-processing module to chunk noun clusters in order to generate new POS tags structures, that represent a larger grouped population of the SE sentences. The algorithm of the this process is explained in Algorithm 1, and the used regular expressions in Table 23. The input is all the Simple Wikipedia sentences that are overlapping with CNL properties, and the output is 19,089 sentences that are mapped to the top 300 most frequent POS tags structures. After applying the pre-processing step we parsed all the sentences using the APE parser to extract the sentences that are **already accepted** without the need of rewriting, as they seem to be basic sentences. The output is 1,536 sentences rewritten into ACE CNL and converted into DRS, and 554 sentences converted into OWL triples. This is because some sentences could be accepted for parsing as DRS but not to OWL, as the DRS parser is more relaxed and the OWL parser is restricted to triples only. Hence, even if there is an accepted OWL triple that can be extracted from the parsed ACE CNL sentence, the OWL parser does not accept it if there are extra tokens that need to be removed first. For example, if we consider the ACE sentence *"Presov is a city in the n:eastern-Slovakia"*, which is accepted by the DRS parser, but refused by the OWL parser. Therefore, in order to be accepted we need to eliminate from the sentence the last 3 tokens *"in the n:eastern-Slovakia"*, and process only the part of the sentence

"Presov is a city", which will be accepted as an OWL triple by the parser.

Algorithm 1 Pre-processing

INPUT: Simple Wikipedia sentences overlapping with CNL properties
OUTPUT: Sentences matching the top 300 repeated POS tags structures.

for all sentences in the dataset **do**
  Remove Punctuation
  Tokenize sentence into tokens T_i using NLTK tagger
  Generate tuples U_i of POS tags P_i for each T_i in the sentence
  Define a regex X_i for compound Nouns/Adjectives e.g. multi-word expressions
  Use RegexpParser to parse U_i and generate a tree E_i
  for all U_i in tree E_i **do**
    if Sequence of tuples (subtrees) equals X_i **then**
      Extract the matched subtrees MS_i and store them in a sorted list.
      Replace P_i of each MS_i by a new POS tag *COMP* in the POS tags serial PS_i of the sentence.
      Append sentence and its new PS_i into 2 lists L_s and L_P respectively.
    else
      Append sentence and its old PS_i into L_s and L_P respectively .
    end if
  end for
  for all L_s and L_P **do**
    if PS_i in L_P not in Dictionary D_i keys **then**
      Store the new PS_i as key in D_i and its parallel sentence from L_s as value in D_i.
    else
      Append sentence to sentences list as value to the matched D_i key
    end if
  end for
  Sort the D_i descending by longest list in values.
  Extract the top 300 repeated PS_i from D_i.
  Parse all the extracted sentences using the ACE validation engine APE.
end for

For the rest of the rules and as explained in Figure 22, the input sentences for each rule will be the output result of application of each rule, that were not accepted by the APE parser.

Table 23: Description of the rules used in the post-processing stage in Figure 22 and their respective expressions.

Rule	Description of transformation	NLTK Regular Expression	Output Result
Pre-processing	The regex should chunk Nouns/Adjectives e.g. multi-word expressions. Then extract the matched patterns (subtrees), and store them in a sorted list, where each subtree content is replaced by the new *NP_CHUNK* POS tag. Store the new POS tags sequences in a dictionary with their matched sentences. Finally, extract the top 300 repeated POS tags sequences from the dictionary and validate them using the ACE validation engine APE.	$NP\_CHUNK = \{< NN.* > + < JJ >? < NN.* > +\| < JJ >? < NN.* > +\| < NN.* > + < JJ >?\}$	A.1.1
1	Chunk all matched patterns of multi-word expressions by replacing spaces between tokens with *hyphens*. Then add a prefix *n:* if the chunk is not in the beginning of the sentence, finally replace all the patterns with the new chunk.	$pos\_tag\_pattern = \{< NN.* > +< CD\|WDT\|CC\| IN\|DT\|JJ.*\|RBS\|FW > * < NN.* > +\}$	A.1.2
2	use a regex to extract compound Nouns/Adjectives that are not preceded with determiners, then extract the matched subtrees. Add a determiner before the chunk if the chunk is not in the beginning of the sentence.	$pos\_tag\_pattern = \{< DT >? < NN.* > \| < DT >? < JJ.* >\} \} < DT >< NN.* > \| < DT >< JJ.* > \{$	A.1.3
3	Match a personal/possessive pronoun pronoun in the sentence with the referenced Noun/Adjective in the abstract, otherwise apply the Stanford Co-reference resolution module to replace the corresponding coreference.	$pos\_tag\_pattern = \{< PRP\|PRP\$ >\}$ $replace\_corref(pos\_tag\_pattern, reference\_token)$	A.1.4

4	if the sentence has past tense verb (VBD), apply **pattern.en**[3] library to get the present conjugated forms. Then, use a regex pattern to replace the past tense verb with its present conjugated form.	$pos\_tag\_pattern = \{< VBD > \mid < VBN >\}$ $new\_tense = conjugate(past\_verb, tense = PRESENT)$	A.1.5
5	Use a regex to search for Nouns/Adjectives not preceded by determiners then replace the first letter in each pattern to its upper case form.	$pos\_tag\_pattern = \{< DT >< NN.* > \mid < DT >< JJ.* >\}$ $\}< DT >\{$ `// Chinking individual determiners`	A.1.6
6	Use a regex to extract POS sequence patterns tagged as *Nouns +TO + Nouns* e.g. republic-of-India. Then extract the matched subtrees and replace spaces between tokens with *hyphens* to generate a new chunk that replaces the old sequence of tokens.	$pos\_tag\_pattern = \{< NN.* \mid JJ.* \mid CD > + < IN > < NN.* \mid JJ.* \mid CD > +\}$	A.1.7
7	Use a regex to extract cardinal patterns tagged *CD* and are digits. Then use the *num2words*[4]) library to rewrite digits into words. Replace the spaces between words in the string with *hyphens* to generate a new chunk. Create a dictionary where the keys are digits and values are generated chunk, then use this dictionary to replace the digits with the new chunk in the sentence.	$dict[CD] = num2words(CD, to = words)$	A.1.8
8	Apply a regex to extract a pattern of a determiner preceded with pre-determiners e.g. *all the* in the POS tag sequence, and apply another regex to remove the determiner.	$pos\_tag\_pattern = re.sub(\{< PDT >< DT >\}, \{< DT >, text\}$	A.1.9

3 https://www.clips.uantwerpen.be/pages/pattern
4 https://anaconda.org/auto/num2words

9	Implement a dictionary that contains particle strings and their alternative replacement e.g. *Since* replaced by *from*, then search in the dictionary for the replacement of the particle. Finally, use a regex to replace it from the dictionary with its alternative particle	$pos\_tag\_pattern = re.sub(dict[RP1], [RP2], text)$	A.1.10
10	Use a regex to search for adverbs followed/preceded by nouns/adjectives, generate subtrees of tuples of all the matched patterns in the sentence, then chunk each subtree by replacing spaces between tokens with *hyphens*. If the pattern is not detected in the beginning of the sentence, denote a prefix *n:* at the beginning of the chunk, then replace the pattern with the new chunk in the sentence.	$pos\_tag\_pattern = \{<NN.*|JJ.*>+<RB|RBR|RBS><NN.*|JJ.*>+\}$	A.1.11

Rule 1, is developed to chunk nouns and/or adjectives clusters. The main algorithm is described in Algorithm 2 and Table 23, where the input is a total of 17,553 sentences for DRS and 18,535 sentences for translation into OWL that were not parsed from the pre-processing step. Since, the number of OWL triples extraction is less than that of DRS, we will try to process sentences by removing extra tokens in the sentence that are not part of the triple, for the sake of extracting more OWL triples in a later stage.

Algorithm 2 Chunking (Rule1)

for all Sentences **do**
  Remove punctuation, split sentence into tokens T_i (tokenization), Generate tuples U_i of POS tags P_i (POS tagging)
  Define a regex X_i for compound Nouns/Adjectives e.g. multi-word expressions
  Use RegexpParser to parse U_i and generate a tree E_i
  for all U_i in tree E_i **do**
    if Sequence of tuples (subtrees) equals X_i **then**
      Extract the matched subtrees MS_i and sort them by number of tokens in a list L_i.
    end if
  end for
  for all MS_i in list L_i **do**
    Chunk MS_i by replacing spaces between tokens with *hyphens*.
    if MS_i not in beginning of sentence **then**
      Denote a prefix *n:* at the beginning of the chunk.
    end if
    Replace MS_i with the new chunk from the list in the main sentence.
  end for
end for

Algorithm 3 Inserting Determiners (Rule2)

for all Sentences **do**
  Remove punctuation, split sentence into tokens T_i (tokenization), Generate tuples U_i of POS tags P_i (POS tagging)
  Define a regex X_i for compound Nouns/Adjectives not preceded with determiners
  Use RegexpParser to parse U_i and generate a tree E_i
  for all U_i in tree E_i **do**
    if Sequence of tuples (subtrees) equals X_i **then**
      Extract the matched subtrees MS_i and sort them by the number of tokens in a list L_i.
    end if
  end for
  for all MS_i in list L_i **do**
    if MS_i not in beginning of sentence **then**
      Add determiner before the chunk.
    end if
    Replace MS_i with the new chunk from the list in the main sentence.
  end for
end for

Rule 2, is developed to check the existence of determiners before noun/adjective clusters and add a determiner in case it is not present. The main algorithm is described in Algorithm 3 and Table 23, where the input is a total of 12,936 sentences for DRS and 16,609 sentences for translation into OWL, that was not accepted by the APE parser after applying rule 1.

Algorithm 4 Coreference Resolution (Rule3)

for all Sentences **do**
  Remove punctuation, split sentence into tokens T_i (tokenization), Generate tuples U_i of POS tags P_i (POS tagging)
  if P_i is a **Personal pronoun** or a **Possessive pronoun then**
    Extract U_i and append to a list.
    Match each sentence with its first sentence in the abstract from the Simple Wikipedia corpus.
  end if
  if the first token(s) in the abstract is a Noun/Adjective N_i **then**
    Extract N_i and append to list.
  else
    Apply Stanford Co-reference resolution module
    Store the sentence in a dictionary with the extracted token(s) of its corresponding coreference token antecedents C_i.
  end if
  Use Regex pattern to replace P_i with C_i in sentence 2.
end for

Rule 3, is developed to resolve Coreference of pronouns between sentences in each abstract. Most of the sentences with pronouns were referring to nouns existing at the beginning of each abstract. So, the algorithm looks for these pronouns and replaces them with the extracted nouns. In addition, we used Stanford Coreference module [LPC+11] to resolve pronouns that did not have nouns in the beginning of their abstracts. The algorithm is described in Algorithm 4 and Table 23, where the input is a total of 9,140 sentences for parsing into DRS and 15,590 sentences for translation into OWL triples.

Algorithm 5 Modifying Verb Tense (Rule4)

for all Sentences **do**
  Remove punctuation, split sentence into tokens T_i (tokenization), Generate tuples U_i of POS tags P_i (POS tagging)
  if P_i is a **past tense verb** (VBD) **then**
    Extract U_i and append verb to a Dictionary D_i.
    Apply **pattern.en** library on all verbs in D_i to get their present conjugated forms.
  end if
  Use Regex pattern to replace the past tense verb in all sentences with its present conjugated form in D_i.
end for

Rule 4, is developed to replace past tense verbs with their present forms using **pattern.en**[5] library. So, we stored all the past tense verbs in the corpus, and their present forms in a dictionary and used a regular expression to replace them in the sentences. The algorithm is described in Algorithm 5 and Table 23, where the input is a total of 7,530 sentences for DRS and 14,867 sentences translation into OWL triples.

Algorithm 6 Modifying Lower/Upper Case (Rule5)

for all Sentences **do**
  Remove punctuation, split sentence into tokens T_i (tokenization), Generate tuples U_i of POS tags P_i (POS tagging)
  Define a regex X_i for Nouns/Adjectives not preceded by determiners
  Use RegexpParser to parse U_i and generate a tree E_i
  for all U_i in tree E_i **do**
    if Sequence of tuples (subtrees) equals X_i **then**
      Extract the matched subtrees MS_i and store them in a sorted list.
      Replace the first letter in each MS_i to its upper case form.
    end if
  end for
end for

Rule 5, is developed to search for nouns that are not preceded by determiners, and replace the first letter from lower case into upper case. The algorithm is described in Algorithm 6 and Table 23, where the input is a total of 5,422 sentences for transformation into DRS and 13,499 sentences translation into OWL triples.

Algorithm 7 Chunking Expressions with **IN** POS Tag (Rule6)

for all Sentences **do**
  Remove punctuation, split sentence into tokens T_i (tokenization), Generate tuples U_i of POS tags P_i (POS tagging)
  Define a regex X_i for patterns tagged *Nouns +TO + Nouns*
  Use RegexpParser to parse U_i and generate a tree E_i
  for all U_i in tree E_i **do**
    if Sequence of tuples (subtrees) equals X_i **then**
      Extract the matched subtrees MS_i and sort them by number of tokens in a list L_i.
    end if
  end for
  for all MS_i in list L_i **do**
    Chunk MS_i by replacing spaces between tokens with *hyphens*.
    if MS_i not in beginning of sentence **then**
      Denote a prefix *n:* at the beginning of the chunk.
    end if
    Replace MS_i with the new chunk from the list in the main sentence.
  end for
end for

5 https://www.clips.uantwerpen.be/pages/pattern

Rule 6, is developed to search for *NN+IN+NN* POS tags clusters, and chunk them as one compound Noun e.g. *Republic-of-India*. The algorithm is described in Algorithm 7 and Table 23, where the input is a total of 3,832 sentences for DRS and 13,059 sentences translation into OWL triples.

Algorithm 8 Rewriting Cardinals/Numbers (Rule7)

for all Sentences **do**
  Remove punctuation, split sentence into tokens T_i (tokenization), Generate tuples U_i of POS tags P_i (POS tagging)
  Define a regex X_i for cardinal patterns tagged *CD* and are digits.
  Use RegexpParser to parse U_i and generate a tree E_i
  for all U_i in tree E_i **do**
    if Sequence of tuples (subtrees) equals X_i **then**
      Extract the matched subtrees MS_i and sort them by number of tokens in a list L_i.
    end if
  end for
  for all MS_i in list L_i **do**
    Use *num2words*[6]) library to rewrite digits into string.
    Chunk MS_i by replacing spaces between tokens with *hyphens*.
    Replace MS_i with the new chunk from the list in the main sentence and store in new list NL_i.
  end for
end for

Rule 7, is developed to search for digit cardinals represented in *CD* POS tags, and rewrite them into a string of one Noun cluster e.g. *3rd* will be rewritten into *third*. The algorithm is described in Algorithm 8 and Table 23, where the input is a total of 3,502 sentences for parsing into DRS and 12,386 sentences translation into OWL triples, which were not parsed after applying rule 6.

Rule 8, is developed to resolve pre-determiners in the beginning of each sentence. We used a regular expression to search and remove determiners that are preceded with pre-determiners e.g. *All the*. The algorithm is described in Algorithm 9 and Table 23, where the input is a total of 2,099 sentences for DRS and 11,616 sentences translation into OWL triples.

Rule 9, is developed to resolve and replace particles that could be accepted and parsed by ACE CNL e.g rewrite *since* to *from*. We used a regular expression to match a predefined list of particles and replace them with another alternative that can be accepted and pared by the ACE parser. The algorithm is described in Algorithm 10 and Table 23, where the input is a total of 1,838 sentences for DRS and 11,420 sentences translation into OWL triples. After applying this rule, the system successfully extracted DRS from 270 sentences and OWL triples from 117 sentences.

Algorithm 9 Resolving Pre-determiners (Rule8)

for all Sentences **do**
  Remove punctuation, split sentence into tokens T_i (tokenization), Generate tuples U_i of POS tags P_i (POS tagging)
  Define a regex X_i for determiner patterns preceded with pre-determiners tagged $\{< PDT >< DT >\}$
  Use RegexpParser to parse U_i and generate a tree E_i
  for all U_i in tree E_i **do**
    if Sequence of tuples (subtrees) equals X_i **then**
      Extract the matched subtrees MS_i and sort them by number of tokens in a list L_i.
    end if
  end for
  for all MS_i in list L_i **do**
    Remove the determiners that are preceded with predeterminers in MS_i.
    Use Regex pattern to replace MS_i with the new one.
  end for
end for

Algorithm 10 Resolving Particles (Rule 9)

for all Sentences **do**
  Remove punctuation, split sentence into tokens T_i (tokenization), Generate tuples U_i of POS tags P_i (POS tagging)
  Initialize a dictionary D_i that contains particle strings and their alternative replacement e.g. *Since* replace by *from*
  if P_i is a **particles**(RP) **then**
    Search in D_i for the replacement of the particle
    Use Regex to replace the particle with its alternative string.
  end if
end for

Algorithm 11 Resolving Adverbs (Rule 10)

for all Sentences **do**
  Remove punctuation, split sentence into tokens T_i (tokenization), Generate tuples U_i of POS tags P_i (POS tagging)
  Define a regex X_i for adverbs followed/preceded by nouns/adjectives
  Use RegexpParser to parse U_i and generate a tree E_i
  for all U_i in tree E_i **do**
    if Sequence of tuples (subtrees) equals X_i **then**
      Extract the matched subtrees MS_i and sort them by number of tokens in a list L_i.
    end if
  end for
  for all MS_i in list L_i **do**
    Chunk MS_i by replacing spaces between tokens with *hyphens*.
    if MS_i not in beginning of sentence **then**
      Denote a prefix *n:* at the beginning of the chunk.
    end if
    Replace MS_i with the new chunk from the list in the main sentence.
  end for
end for

Rule 10, is developed to resolve adverbs in sentences e.g chunk *third largest* as one compound noun to be *n:third-largest*. We used a regular expression to search for adverbs and chunk them. The algorithm is described in Algorithm 11 and Table 23, where the input is a total of 1,568 sentences for DRS and 11,303 sentences for translation into OWL triples. After applying this rule, the system successfully extracted DRS from 228 sentences and OWL triples from 125 sentences.

Table 24: Examples of simplified text sentences after their rewriting into alternative ACE and generating their equivalent DRS and OWL outputs.

Simplified text sentences	**Rewritten ACE sentences**	**Applied rules**
- Parametric statistics is a branch of statistics. - Herbs zoster is a disease in Humans.	- Parametric-statistics is a branch of Statistics. - Herbs-zoster is a disease in Humans.	1
- Lucca is a city in Italian region of Tuscany. - The Manx is a breed of domestic cat.	- Lucca is a city in the n:Italian-region of Tuscany. - The Manx is a breed of the n:domestic-cat.	1,2
- He was an American actor, comedian, and television personality.	- Reynaldo-Rey is an n:American-actor-comedian and a n:television-personality.	1,2,3,4
- The Afrotheria is a group of mammals.	- The Afrotheria is a n:group-of-Mammals.	6
- It is a commune of 264 people.	- Buisson is a n:commune of a n:two-hundred-and-sixty-four people.	1,2,3,7
- Alice in Videoland is an electroclash band from Sweden.	- Alice-in-Videoland is an n:electroclash-band from Sweden.	1,6

In order to show the effect of the previously presented rules on the corpus, we present in Table 24 some examples of Simplified Text sentences before and after the rewriting process took place. In the second column we show the sentences after implementing and applying the architecture in Figure 22 and the rules in Table 21, where the *n:* is a prefix that comes before the chunked nouns, after applying the chunker and the rewriting rules. The

third column, shows the index of the rewriting rules applied on the SE sentences to generate their ACE CNL alternatives. These ACE sentences are accepted and parsed by the ACE parser, where we can successfully extract DRS and OWL outputs.

5.3 KNOWLEDGE EXTRACTION AS DRS AND OWL TRIPLES

In Table 25 we show an example for a Simple Wikipedia sentence extracted from the corpus, and its equivalent ACE sentence after the rewriting process take place. The DRS and OWL triples generated from the equivalent ACE sentence are presented as well.

Table 25: The DRS and OWL outputs from the ACE parser for an example sentence.

Metric	Result
Simplified text	Parametric statistics is a branch of statistics.
Equivalent ACE text	Parametric-statistics is a branch of Statistics.
DRS	[A,B]object(A,branch,countable,na,eq,1)-1/4 relation(A,of,named(Statistics))-1/5 predicate(B,be,named(Parametric-statistics),A)-1/2
OWL XML	<Ontology xml:base="http://www.w3.org/2002/07/owl#" xmlns="http://www.w3.org/2002/07/owl#" ontologyIRI="http://attempto.ifi.uzh.ch/ontologies/owlswrl/test"> <ObjectPropertyAssertion> <ObjectProperty IRI="http://attempto.ifi.uzh.ch/ontologies /owlswrl/ test#branch"/> <NamedIndividual IRI="http://attempto.ifi.uzh.ch/ontologies /owlswrl/test#Statistics"/> <NamedIndividual IRI="http://attempto.ifi.uzh.ch/ontologies /owlswrl /test#Parametric-statistics"/> </ObjectPropertyAssertion></Ontology>

As seen from Table 26, since the extraction of OWL triples has limited coverage on full sentences, we decided to process the sentences that we were not able to extract triples from, by eliminating any non triple tokens that represent extra information in the sentence. This means we reprocess these sentences by keeping only entities that can be accepted by the parser as an OWL triple and removing any extra tokens. This is performed by iterating over all the sentences removing a token each time, and try to extract the triples from the maximum possible tokens in each sentence. For example, if we consider the ACE sentence *"Presov is a city in the n:eastern-Slovakia"*, which is accepted by the DRS parser, but refused by the OWL parser. Therefore,

Table 26: Knowledge extraction DRS/OWL after applying each rule (Development set).

Output	Rule	Parsed	Non-parsed	Total
DRS	Pre-processing	1536	17553	19089
	1	4617	12936	17553
	2	3255	9681	12936
	3	1604	7536	9140
	4	2108	5422	7530
	5	1590	3832	5422
	6	330	3502	3832
	7	1403	2099	3502
	8	261	1838	2099
	9	270	1568	1838
	10	228	1340	1568
OWL	Pre-processing	554	18535	19089
	1	1926	16609	18535
	2	453	16156	16609
	3	715	14875	15590
	4	1368	13499	14867
	5	440	13059	13499
	6	673	12386	13059
	7	770	11616	12386
	8	196	11420	11616
	9	117	11303	11420
	10	125	11178	11303

Table 27: OWL Triples Extraction Results (Development Set).

Iteration	Total	Parsed	Non-parsed
1	11178	9	11169
2	11169	105	11064
3	11064	5666	5398
4	5398	23	5375
5	5375	157	5218
6	5218	2137	3081
7	3081	13	3068
8	3068	19	3049
9	3049	342	2707
10	2707	3	2704
11	2704	55	2649
12	2649	162	2487

in order to be accepted we need to eliminate from the sentence the last 3 tokens *"in the n:eastern-Slovakia"*, and process only the part of the sentence *"Presov is a city"*, which will be accepted as an OWL triple by the parser.

The results of this experiment is shown in Table 27, where we can observe a total of 12 iterations required to extract the maximum amount of OWL triples from the remainder of the sentences. The algorithm stops iterating when there are no more OWL triples to generate. In each iteration we show, the number of sentences presented as input in each iteration and the output represented in the number of parsed and unparsed sentences.

5.4 SYSTEM EVALUATION

The evaluation of the system is performed on three perspectives as follows:

1. The system coverage, which is crucial to evaluate **RQ3**: *What is the feasibility of rewriting human-oriented CNL (Simplified text) into machine processable text (e.g. ACE CNL)?*
2. The semantic loss, to evaluate **RQ4**: *What is the effect of rewriting with respect to semantic loss?*
3. The accuracy of the extracted triples, by evaluating our approach against other state-of-the-art systems in terms of a binary relation classification task.

5.4.1 Rewriting System Coverage

The system coverage is evaluated on both the development set and the test set. Since we developed syntactic rules to cover only the top 300 POS tags structures, the evaluation of the system coverage is to estimate how many sentences the system was able to rewrite out of the whole corpus, and hence the coverage will show the amount of knowledge the system was able to extract from the whole corpus, in the form of DRS and OWL triples. By the end of this evaluation, we need to answer the following questions:

1. How much of the corpus, the system was able to rewrite?
2. How much knowledge the system was able to extract?

Table 28: Development Set Final Results.

Output	Total	Parsed	Non-parsed	Coverage
DRS	19,089	17,749	1340	92.9%
OWL		16,602	2487	87%

In the development set the number of successfully rewritten sentences into DRS is 17,749 sentences, which reports a coverage of 92.9%. The number of extracted OWL triples from the fully parsed sentences is 7,911 triples. In order to extract more OWL triples from the rest of the unparsed sentences, we need to remove any tokens that are not part of the OWL triples entities. As shown in Table 27, the remaining of the 11,169 unparsed sentences, were parsed after eliminating any tokens that are not part of the triple entities and represent extra information in the sentence. They were processed over 12 iterations to extract additional 8,691 triples, so the total is 16,602 OWL triples. The final result can be shown in Table 28, where the total extracted OWL triples from the development set has reported coverage of 87%.

In addition to, the Table shows a portion of sentences that could not be parsed by the system, and this also means a portion of unextracted knowledge. This happened because the developed rules could not rewrite them into an interpretable ACE CNL structure, thus they were refused by the APE parser. The rewrite rules are developed for the most repeated POS structures to rewrite as much sentences as possible, neglecting individual POS structures. All the rewritten sentences are passed to the APE parser and validated for being successfully parsed.

On the other hand, in order to test the coverage of the system on the unseen data, we extracted a test set from the main dataset that matches 20% when mapped with the development set. So, we assumed that the total number of 19,089 sentences represents 80% of the corpus. Hence, to estimate the total number of sentences that represents 20% of the corpus that represents the test set we used equation 6 as follows:

$$Testset = \frac{19089 \times 0.2}{0.8} \approx 4773 \; sentences \tag{6}$$

As shown in Table 29, the number of successfully rewritten sentences into DRS is 3,589 sentences with coverage of 75.1%. Furthermore, the number of extracted OWL triples from these sentences is 3,400 triples which reports a coverage of 71.2%. As shown in Table 30, the total amount of knowledge in the form of DRS extracted after using the rewriting system on both the

Table 29: Test Set Final Results.

Output	Total	Parsed	Non-parsed	Coverage
DRS	4773	3589	1184	75.1%
OWL		3400	1373	71.2%

Table 30: Evaluation of the coverage of the rewriting system (development+test) datasets.

Extracted Knowledge	Total No. of sentences	Parsed	Non-parsed	Coverage
DRS	23,862	21,338	2,524	89.4%
OWL		20,002	3,860	83.8%

development and test datasets is 89.4%. In addition to, the total knowledge in the form of OWL triples is 83.8% of the input SE sentences.

5.4.2 Semantic Loss

The second evaluation was conducted to ensure that the semantics of the Simplified English input sentence has not changed after the rewriting took place. In this evaluation, we need to answer whether each generated ACE CNL sentence preserved the semantics of its source SE sentence after rewriting or not, and if not, what is the loss?. Since, the evaluation of the semantic loss needs a human evaluator to assess the percentage of change in semantics after rewriting, we extracted randomly a sample from the entire corpus of sentences to decrease the amount of sentences to be annotated. While the purpose of random sampling is to guarantee an unbiased extracted sample from the corpus, we need to ensure that the extracted amount of data is a representative sample of the entire corpus as well. For a representative and unbiased sample we followed the statistical estimation recommended by [Smi13], where we need to estimate four main values as follows:

- *Population size*, corresponds to the total number of sentences in the corpus of 23,862 successfully parsed sentences.
- *Margin of error (confidence interval)*, determines how much error we allow to let our sample mean fall, where the study defines 5% to be a usual acceptable number.

- *Confidence level*, corresponds to a constant called the Z-score used to determine the confidence factor for the actual mean to be within the confidence interval. As per the study, the average confidence interval is 95% confident.
- *Standard of deviation*, represents the variance factor and recommended to be 0.5 to ensure a large sample.

So, with 5% confidence interval and 95% confidence level, the representative sample is calculated to be 379 sentences. The semantic loss is evaluated by measuring whether all the contextual information in the SE sentence is covered in the rewritten ACE CNL sentence. The coverage of information is evaluated on a 0 to 5 scale based on the evaluation methodology used in [BJR16]. Where 5 means that the rewritten ACE sentence has the same semantics of the original SE sentence. A score less than 5 means that there is a loss of semantics in one or more parts of the sentence i.e score of 4 means 1 mistake, score of 3 means 2 mistakes, and so on. Any sentence with score less than 5 is annotated as a negative sample. For example, the SE sentence *"Eastwood is a census-designated place in Bossier Parish, Louisiana, United States"* is rewritten into the ACE sentence *"Eastwood is a n:censusdesignated-place in the n:Bossier-Parish-Louisiana-United-States"* , this rewriting has given a score of 3 by deducting 2 points as a result of (1) The missing hyphen due to the incorrect chunking of *n:censusdesignated-place*, (2) The addition of the inappropriate determiner *the* before the chunked noun, is not needed. In Table 31 we summarize the results of the evaluation, in terms of the number of sentences with semantic loss (negative samples), the most common reasons for semantic loss, and precision reported. Semantically correct sentences represent a total of 330 sentences out of 379, where the semantics has not been changed after the rewriting. Semantically incorrect sentences represent the amount of lost information from the original sentence, where 42 sentences are labeled with a score of 4 as they included one rewriting mistake, and 7 sentences are labeled with a score of 3 as they included two rewriting mistakes. Hence, we report a total of 49 semantically incorrect sentences.

5.4.3 Comparison with Reverb

Lastly, we evaluated the quality of the relations (and arguments) in the extracted triples of our rewriting system versus one state-of-the-art binary

Table 31: Semantic Loss Evaluation Results

Semantically correct	Semantically incorrect	Common Reasons	Precision
330	49	-incorrect chunking -wrong determiner -inappropriate verb tense	87%

relation extraction system called Reverb [FSE11] discussed in 2.4. Reverb has a set of predefined syntactic and lexical constraints to reduce erroneous extractions. We performed manual evaluations on both systems using the previous extracted representative sample. We manually evaluated the output of Reverb represented in binary triples relations (arg1; rel; arg2) with entities as arguments after processing the original SE sentences. On the other hand, we manually evaluated the extracted binary triples of the rewritten ACE CNL sentences. The evaluation is done against a baseline dataset[7]. This dataset includes all the 379 sentences in the representative sample dataset, where each sentence is manually annotated by identifying how many binary triple relations in each sentence, and what are the exact entities that are included in each relation.

Table 32: Evaluation of the accuracy (complete and informative) of the extracted triples.

Metric	Precision	Recall	F1
Reverb	0.81	0.96	0.87
CNL Rewriting	0.86	0.94	0.89

We evaluated the extracted triple relations in terms of being informative and complete, on both approaches using a confusion matrix. The evaluation is performed after counting false positives (wrong or partially-right entity) and false negatives (wrong triples). For example, if we consider the sentence *It is a tributary of the river Seine*, where the triple generated from Reverb is *it, be tributary of, the river*. Although we did not penalize Reverb for co-reference resolution, we can see that Reverb drops a part of the *object* entity, which is the name of the river *Seine* in the previous extraction. On the other side, a FN is counted when the approach drops a full triple. As shown in Table 32, we extracted a total of 539 triples from all the 379 sentences, the

7 https://drive.google.com/drive/folders/1EP2UwTEEtJDr_15ntz8IEH8-JkpLtXCf?usp=sharing

extractions in both approaches show a very close recall but our rewriting system is still better in terms of precision and F1.

5.5 CHAPTER SUMMARY

In Section 5.1 we introduced and presented the main aim of this Chapter, in terms of validating hypothesis **H2**:

The shared common rules between the different CNLs in O'Brien's analysis can be used to extract corpus based syntactic patterns in the sentence structures through their POS tags structures to rewrite a human-oriented CNL sentence into a machine-oriented CNL sentence, such that the newly rewritten machine-oriented CNL sentence preserves the semantics of the original source human-oriented CNL sentence.

And to answer research questions **RQ3** and **RQ4**, about the rewriting approach from SE sentences into ACE CNL. Furthermore, we discussed that the approach can be generalized into other corpora that follow the authoring style-guides of the Basic English, as they have the same common CNL properties defined in the literature. Section 5.2, discussed the system architecture and the detailed explanation of the algorithms for the used rules, as well as the process of knowledge extraction. As seen from the output results the rewriting approach of the SE sentences helped us to extract a valuable amount of knowledge of about 89.4% from the input text represented in DRS and OWL. In Section 5.3 we presented the evaluation of the approach on **three** perspectives to measure i) its coverage, ii) semantic loss, and iii) the accuracy of the extracted triple relations, to ensure triples are complete and are not discarding important information. The results of the system coverage was presented for both the development and the testing data sets, to evaluate the amount of the extracted knowledge in the form of DRS statements and OWL triples from the whole corpus. In the evaluation of the semantic loss after rewriting the SE sentences into ACE CNL sentences we verified that our process produces low semantic loss with 87% precision which strengthens the feasibility of the rewriting approach. However we presented the common reasons that caused the generation of incorrect ACE sentences and led to loss in semantics such as inaccurate chunking and selecting improper determiners before nouns. In the last evaluation, we compared the accuracy of our rewriting approach

versus Reverb in terms of the precision and recall of the quality of the extracted triples in the form of binary relations. The quality of the extracted triples is measured based on the entities being complete and informative. Although, Reverb has slightly higher recall, it fully dropped around 15 triple relations, and extracted 95 incomplete triple relations in the 379 sentences. Our approach has higher precision as well as higher F1 as it not only cater for binary relations, but for other types of relations.

6 CONCLUSIONS AND FUTURE WORK

6.1 GENERAL CONCLUSIONS

This thesis began with introducing the latest work in the literature related to ontology authoring and knowledge extraction for the Semantic Web, together with presenting the limitations and the current requirements in terms of a set of challenges in the Semantic Web community. In this thesis we tackle one of these challenges and we make the case for Controlled Natural Languages (CNLs) as an alternative solution for knowledge creation and management, especially for non-expert users and small organizations who would like adopt semantic technologies. We have shown that rewriting a human-oriented CNL such as Simplified English into a machine processable CNL with minimal loss of semantics, will assist in reducing the knowledge acquisition bottleneck.

We presented our rewriting approach from Simplified English [SSH+95] sentences to Attempto Controlled English (ACE) [FKK08a] sentences, together with the system architecture and the implemented algorithms. We have also presented the first dataset that includes natural language sentences and their parallel machine processable sentences in the form of ACE CNL, which could be used by the CNL community to developing future CNL related applications for ontology development and knowledge capture. Finally, we have evaluated our approach from different perspectives, the system coverage, the amount of extracted knowledge, semantic similarity to ensure that semantics has not changed after applying our approach, and lastly the quality of the entities in the extracted OWL triples versus state-of-the-art system for binary relation classification and extraction namely Reverb [FSE11].

This concluding chapter is divided into three sections: (1) we revisit the research contributions and findings of this thesis, (2) Limitations of our

approach and system, (3) Future work with respect to system enhancement and various use-cases.

6.2 RESEARCH CONTRIBUTIONS AND FINDINGS

With respect to the analysis of human-oriented CNLs and machine-oriented CNLs we have find the following:

- In the last few years, the trend in the CNLs community is moving from theoretical outcomes towards useful applications, interfaces and frameworks that have a useful impact on real-life use-cases. These use-cases include using CNL frameworks to model and extend the computational grammar for various natural languages such as Afrikaans [PM18], in order to be used in different CNLs applications , using CNLs for financial services compliance and checking [ACP18], a CNL for question generation and marking of language learning exercises for different languages other than English [GK18], a CNL for hazard analysis and risk assessment [CMW+18], using CNLs to organize scientific claims [Tob18], using CNLs for language learning [LL18], CNLs for object oriented programming that aids in producing computer games [Hsi18], a CNL for tax fraud detection [CCP16], a CNL for legal sources on the Semantic Web [WNL16], developing a CNL approach for Human-Robot interaction in a shopping mall [DGE16].

- *Human-oriented* CNLs, unlike *machine-oriented* CNLs [Huj98] do not attempt to unambiguously map into formal knowledge structures, as the communication goal is: *human to human* and not *human to machine*. Hence, the development and planning of human-oriented CNLs appears often community driven, many of which at first glance, like Simplified Wikipedia, may have been inspired by Basic English [Ogd30]. However, the simplicity is restricted by the amount of effort the authors did to follow the writing style-guides (e.g. recommended sentence structures).

- Given that *human-oriented* CNLs have higher user uptake in industry than *machine-oriented* CNLs (at the cost of introducing some ambiguity to the machine), it is worth investigating the similarities between human-oriented CNLs to machine-oriented CNLs linguistically and

how this can be exploited for formal knowledge capture. Furthermore, our analysis helped us to focus on one of the most important gaps in the CNLs domain, which is the need for a CNL dataset representing a human-oriented CNL (Simplified Wikipedia) aligned with a machine-oriented CNL (ACE). In addition to, identify the appropriate CNL for each task, and hence selecting ACE as the machine-oriented CNL for developing our approach.

With respect to introducing a CNL gold-standard dataset we make the following contributions:

- We have provided corpus statistics and linguistic analysis, which answered our research questions **RQ1** and **RQ2** with respect to the properties of Simplified English (Simplified Wikipedia) text are closer to the properties of CNLs than unrestricted text. Hence, it is possible to rewrite Simplified English to ACE and its subsequent transformation into logical and knowledge representations such as DRS and OWL respectively.
- We have presented a linguistic resource that is both human-readable and semantically machine interpretable. To our knowledge this resource is the first aligned dataset across a human-oriented and machine-oriented CNL. The dataset is well represented in that it takes almost the entire Simplified Wikipedia abstract population post-cleansing.

With respect to providing a CNL rewriting approach for knowledge extraction purposes we make the following contributions:

- We have conducted a series of experiments to answer research question **RQ3**, about using rewriting rules to perform subsequent transformation of the SE sentences into ACE CNL, which led to the extraction of logical and knowledge representations such as DRS and OWL.
- Our approach shows that we can successfully rewrite 89.4% of human Simplified English sentences into ACE CNL with an acceptable semantic loss, which answers research question **RQ4** and strengthens the feasibility of the approach .
- Furthermore, we extracted around 20k OWL triples from the transformation. So our approach benefits more fully from the deterministic parse of the ACE parser.

- As discussed in Chapter 3, since many human-oriented CNLs tend to follow the rules of Basic English, we would argue that based on applying the rewriting rules on a representative sample from the Simple Wikipedia abstracts corpus, these rules could also generalize by induction to vast amounts of other human-oriented CNLs content in industry such as Caterpillar Technical English (CTE) [KAM+98], Special English[1], IBM Controlled English, Ericsson English (EE) [AS92] and others. The authors in [Kuh14] claim that more than 20 languages were inspired by Basic English, which is ripe for knowledge capture.

- Our approach increases precision from 0.86 instead of 0.81 with respect to capturing binary triple relations in comparison to a well known baseline - Reverb [FSE11]. Moreover as shown in Chapter 5, by rewriting SE to ACE we can capture more relations than Reverb, without the need for customization due to the discourse relation structures captured by ACE, and without having a predefined relation schema.

6.3 OPEN QUESTIONS AND LIMITATIONS

- The main aim of developing a few set of rules (ten rules) is that we wanted to capture the most common patterns in the corpus, and as per the CNLs analysis in [OBr03] this could be generalized to other types of simplified English corpora, as their properties are inherited from the human-oriented CNLs properties and are derived by the writing style-guides of Basic English.

  The upside of adding more rules is that we can achieve wider coverage and extra amount of knowledge that could be extracted. However the downside is, increasing the system complexity by adding more POS structures, and hence more rules to capture their patterns. This will require the involvement of a linguist to investigate the patterns in these POS structures and implement more rules to capture them.

- Although our selection of ACE was derived by the comparative analysis we did in Chapter 3, we were also limited by the capabilities of this CNL in terms of grammar, syntax and semantics. For instance, ACE CNL was was often driven by projects related to the semantic web

1 https://www.voaspecialenglish.com

and machine translation, and hence the CNL did not support times, events and states as temporality was not a requirement. So, the CNL only supports the simple present tense, active and passive but not past tenses. This gives an opportunity to find other alternatives that can be used to resolve these challenges. An example of this is to add temporal logic and temporal operators such as "before", "after" and "while" into ACE and the DRS, another example is resolving explicit times and dates in the sentences by either using prepositional phrases such as the sentence *Mark uses a card into the ATM at a time T1.*, or by adding a string for the date such as *John goes to the doctor on March 3, 2014.* that can be rewritten into *John goes to the doctor on "2014-03-03".* which will be accepted by the ACE parser.

Another restriction is that not all ACE constructions map to OWL, for instance the mapping does not support enumerations, and datatype properties, and this takes part in degrading the amount of extracted knowledge when we map from ACE to OWL.

6.4 FUTURE DIRECTIONS

Future work with respect to this thesis will involve working on two directions as follows:

- **Use the gold-standard dataset to train machine learning models:** Since we generated a gold-standard dataset of simplified sentences with their equivalent CNL sentences, and the developed rules can be generalized by induction to other human-oriented CNLs corpora, as they all share the common CNLs properties. We can use the generated dataset and train machine learning models such as neural networks models, to capture the patterns in other human-oriented CNL corpus, and automatically rewrite the sentences using a pre-trained model. This will lead to generating different resources of CNLs which in return could boost the ontology authoring and knowledge extraction processes from these resources.

- **Testing the approach on other CNL platforms:** We initially tested a use-case [SGE+15; SGD+16] based on using Grammatical Framework (GF) [Ran11]. The use-case is based on the concept of Embedded

Controlled Language [Ran14] that allows for extracting formal semantic knowledge from text corpus. It is characterized by a two-level approach for natural language representation. The First level is called the abstract syntax, which accounts for language-independent aspects, and the second level is called the concrete syntax, which accounts for language-specific aspects. The same abstract syntax can be equipped with many concrete syntaxes – reversible mappings from Abstract Syntax Trees (AST) to feature structures and strings – making the grammar multilingual. The translation is achieved by parsing a string in the source language into an AST, followed by linearizing the AST into one or more concrete syntaxes, each for a different target language. Most importantly, GF provides a general-purpose Resource Grammar Library (RGL) [REK09] for currently 30 languages, all implementing the same abstract syntax. The concept and technology of Embedded Controlled Language, allows that a CNL can be specified as an extension to a general-purpose wide-coverage grammar, the host language which is already defined in the GF RGL. The rationale behind this is that the probability that an arbitrary sentence, taken form a text corpus, will be accepted by a strict ACE CNL grammar is close to zero. Instead, the approach aims for wide coverage parsing by default, as the parser will fall back to the robust RGL grammar whenever the CNL grammar fails to accept a phrase in a sentence. This can be achieved by handling the embedded CNL phrases as separate chunks in the abstract syntax tree. The GF parser (i.e., the application grammar) can be adjusted by tweaking probabilities of the CNL rules and by excluding highly ambiguous but less relevant rules from the wide-coverage grammar (w.r.t. the domain of interest), so that ASTs containing CNL chunks appear among the k-best analysis.

The advantage of GF, is that multilinguality is optionally possible, as the resulting ASTs can be linearized into concrete syntax of any of the languages covered by GF RGL. The GF translation facility might allow for multilingual verbalisation of the extracted knowledge base (e.g. via the readily available multilingual ACE/OWL grammar in GF[2]) and/or for cross-lingual IE where the input corpus might be in a language other that English, and it would be still translated

2 https://github.com/Attempto/ACE-in-GF

to ACE during the extraction process. On the other side, the main disadvantage with embedded CNLs approach that it is specific to GF implementation and does not generalize as easy to other toolkits. So, we favored the rewriting approach as the approach is based on rules and regular expressions patterns, which is platform agnostic and can be re-implemented in any NLP toolkit such as JAPE [CMB11], IBM Content Analytics [3], Stanford CoreNLP [4], NLTK [BL04] and GF itself, and across any NLP toolkit with shallow parsing tools.

- **Exploiting other Semantic Web related applications and resources:** This can be possible after augmenting the aligned human to machine oriented CNL content with semantic data such as RDF[5] generated from ACE2OWL[6][KF06b]. In addition to, exploit the generated resource in other fields beyond CNLs for knowledge based population. Potential applications include generating general knowledge for expert and knowledge based systems and ontology aware NLP applications as well as knowledge based MT, automated reasoning, language learning as well as teaching and learning resources for knowledge engineering and logic programming.

3 https://www.ibm.com/products/watson-explorer
4 https://stanfordnlp.github.io/CoreNLP/
5 https://www.w3.org/RDF/
6 https://www.w3.org/2001/sw/wiki/ACE

A KNOWLEDGE EXTRACTION RESULTS AND EVALUATION

A.1 DRS RESULTS OF EACH RULE

Total Count	6671	17553	1536	19089
pos	num_pos_structures	non_parsed	parsed	total
CHUNK VBZ DT CHUNK IN CHUNK	366	2307	357	2664
CHUNK VBZ DT CHUNK	125	1098	301	1399
CHUNK VBZ DT CHUNK IN CHUNK IN DT CHUNK	77	1087	1	1088
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK	224	879	47	926
CHUNK VBD DT CHUNK	82	885	0	885
DT CHUNK VBZ DT CHUNK IN CHUNK	194	430	39	469
CHUNK VBD DT CHUNK CC CHUNK	98	460	0	460
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN DT CHUNK IN CHUNK IN CHUNK	31	50	352	402
CHUNK VBZ DT CHUNK IN DT CHUNK	158	290	28	318
PRP VBZ DT CHUNK IN CHUNK	50	299	0	299
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK IN CHUNK	47	287	2	289
CHUNK VBZ DT CHUNK CC CHUNK	83	261	2	263
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN CHUNK	79	133	104	237
PRP VBZ DT CHUNK IN DT CHUNK	61	220	0	220
CHUNK VBZ DT CHUNK IN CHUNK IN CHUNK	122	191	21	212
DT CHUNK VBZ DT CHUNK	72	169	36	205
PRP VBZ IN DT CHUNK IN DT CHUNK IN DT CHUNK IN CHUNK	12	203	0	203
PRP VBZ IN CHUNK	23	198	0	198
PRP$ CHUNK VBZ CHUNK	17	194	0	194
DT CHUNK VBZ DT CHUNK IN DT CHUNK	104	156	30	186

Figure A.1.1: A snapshot for the results after applying the pre-processing module on the corpus.

Total Count	6553	12936	4617	17553
pos	num_pos_structures	non_parsed	parsed	total
CHUNK VBZ DT CHUNK IN CHUNK	363	1920	387	2307
CHUNK VBZ DT CHUNK	124	12	1086	1098
CHUNK VBZ DT CHUNK IN CHUNK IN DT CHUNK	76	117	970	1087
CHUNK VBD DT CHUNK	82	885	0	885
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK	223	522	357	879
CHUNK VBD DT CHUNK CC CHUNK	98	460	0	460
DT CHUNK VBZ DT CHUNK IN CHUNK	191	353	77	430
PRP VBZ DT CHUNK IN CHUNK	50	299	0	299
CHUNK VBZ DT CHUNK IN DT CHUNK	151	37	253	290
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK IN CHUNK	47	44	243	287
CHUNK VBZ DT CHUNK CC CHUNK	81	256	5	261
PRP VBZ DT CHUNK IN DT CHUNK	61	219	1	220
PRP VBZ IN DT CHUNK IN DT CHUNK IN DT CHUNK IN CHUNK	12	203	0	203
PRP VBZ IN CHUNK	23	198	0	198
PRP$ CHUNK VBZ CHUNK	17	194	0	194
CHUNK VBZ DT CHUNK IN CHUNK IN CHUNK	119	171	20	191
PRP VBZ DT CHUNK	24	184	0	184
PRP VBZ IN DT CHUNK	23	175	0	175
DT CHUNK VBZ DT CHUNK	72	35	134	169
PRP VBZ IN DT CHUNK IN CHUNK	43	163	0	163

Figure A.1.2: A snapshot for the results after applying rule 1 on the corpus.

Total Count	5570	9681	3255	12936
pos	num_pos_structures	non_parsed	parsed	total
CHUNK VBZ DT CHUNK IN CHUNK	304	360	1560	1920
CHUNK VBD DT CHUNK	82	885	0	885
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK	168	148	374	522
CHUNK VBD DT CHUNK CC CHUNK	98	460	0	460
DT CHUNK VBZ DT CHUNK IN CHUNK	159	154	199	353
PRP VBZ DT CHUNK IN CHUNK	50	299	0	299
CHUNK VBZ DT CHUNK CC CHUNK	76	179	77	256
PRP VBZ DT CHUNK IN DT CHUNK	61	219	0	219
PRP VBZ IN DT CHUNK IN DT CHUNK IN DT CHUNK IN CHUNK	12	203	0	203
PRP VBZ IN CHUNK	23	198	0	198
PRP$ CHUNK VBZ CHUNK	17	194	0	194
PRP VBZ DT CHUNK	24	184	0	184
PRP VBZ IN DT CHUNK	23	175	0	175
CHUNK VBZ DT CHUNK IN CHUNK IN CHUNK	106	60	111	171
PRP VBZ IN DT CHUNK IN CHUNK	43	163	0	163
VBD DT IN CHUNK CC IN CHUNK	94	148	0	148
CHUNK VBD DT CHUNK IN CHUNK	3	126	0	126
CHUNK VBZ DT CHUNK IN CHUNK IN DT CHUNK	57	35	82	117
DT CHUNK VBZ CHUNK	61	80	24	104
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN CHUNK	58	28	75	103

Figure A.1.3: A snapshot for the results after applying rule 2 on the corpus.

Total Count	4403	7536	1604	9140
pos	num_pos_structures	non_parsed	parsed	total
CHUNK VBD DT CHUNK	82	885	0	885
CHUNK VBD DT CHUNK CC CHUNK	98	460	0	460
CHUNK VBZ DT CHUNK IN CHUNK	102	360	0	360
PRP VBZ DT CHUNK IN CHUNK	50	84	203	287
PRP VBZ DT CHUNK IN DT CHUNK	61	47	148	195
PRP$ CHUNK VBZ CHUNK	17	41	149	190
PRP VBZ IN CHUNK	23	25	164	189
CHUNK VBZ DT CHUNK CC CHUNK	37	179	0	179
DT CHUNK VBZ DT CHUNK IN CHUNK	53	154	0	154
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK	67	148	0	148
VBD DT IN CHUNK CC IN CHUNK	94	148	0	148
PRP VBZ DT CHUNK	24	12	133	145
PRP VBZ IN DT CHUNK	23	19	125	144
PRP VBZ IN DT CHUNK IN CHUNK	43	32	110	142
CHUNK VBD DT CHUNK IN CHUNK	3	126	0	126
CHUNK VBZ DT CHUNK IN CHUNK IN DT CHUNK IN CHUNK	35	100	0	100
CHUNK VBZ DT CHUNK IN CD CHUNK	50	60	34	94
PRP VBZ IN DT CHUNK IN DT CHUNK	21	86	7	93
CHUNK VBZ DT CD CHUNK	34	91	0	91
PRP VBD DT CHUNK IN CHUNK	52	47	40	87

Figure A.1.4: A snapshot for the results after applying rule 3 on the corpus.

Total Count	3947	5422	2108	7530
pos	num_pos_structures	non_parsed	parsed	total
CHUNK VBD DT CHUNK	82	13	872	885
CHUNK VBD DT CHUNK CC CHUNK	98	385	75	460
CHUNK VBZ DT CHUNK IN CHUNK	102	360	0	360
CHUNK VBZ DT CHUNK CC CHUNK	37	179	0	179
DT CHUNK VBZ DT CHUNK IN CHUNK	53	154	0	154
VBD DT IN CHUNK CC IN CHUNK	94	36	112	148
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK	67	148	0	148
CHUNK VBD DT CHUNK IN CHUNK	3	121	5	126
CHUNK VBZ DT CHUNK IN CHUNK IN DT CHUNK IN CHUNK	35	100	0	100
CHUNK VBZ DT CD CHUNK	34	52	39	91
PRP VBZ IN DT CHUNK IN DT CHUNK	19	86	0	86
PRP VBZ DT CHUNK IN CHUNK	22	84	0	84
DT CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK	36	2	80	82
DT CHUNK VBD CD IN DT CD CHUNK	56	25	50	75
DT CHUNK VBZ CHUNK	39	25	48	73
PRP VBD DT CHUNK IN DT CHUNK	47	65	0	65
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN DT CHUNK	31	25	39	64
CHUNK VBD DT CHUNK IN DT CHUNK	36	52	9	61
PRP VBZ CHUNK CC CHUNK	27	61	0	61
CHUNK VBZ DT CHUNK IN CHUNK IN CHUNK	44	60	0	60

Figure A.1.5: A snapshot for the results after applying rule 4 on the corpus.

Total Count	3251	3832	1590	5422
pos	num_pos_structures	non_parsed	parsed	total
CHUNK VBD DT CHUNK CC CHUNK	71	42	343	385
CHUNK VBZ DT CHUNK IN CHUNK	102	153	207	360
CHUNK VBZ DT CHUNK CC CHUNK	37	16	163	179
DT CHUNK VBZ DT CHUNK IN CHUNK	53	45	109	154
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK	67	113	35	148
CHUNK VBD DT CHUNK IN CHUNK	3	121	0	121
CHUNK VBZ DT CHUNK IN CHUNK IN DT CHUNK IN CHUNK	35	100	0	100
PRP VBZ IN DT CHUNK IN DT CHUNK	19	86	0	86
PRP VBZ DT CHUNK IN CHUNK	22	61	23	84
PRP VBD DT CHUNK IN DT CHUNK	47	63	2	65
PRP VBZ CHUNK CC CHUNK	27	48	13	61
CHUNK VBZ DT CHUNK IN CHUNK IN CHUNK	44	29	31	60
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK	5	55	0	55
CHUNK VBD DT CHUNK IN DT CHUNK	29	11	41	52
CHUNK VBZ DT CD CHUNK	14	45	7	52
CHUNK VBD DT JJ CHUNK	41	19	29	48
PRP VBZ DT CHUNK IN DT CHUNK	22	45	2	47
PRP VBD DT CHUNK IN CHUNK	26	44	3	47
PRP$ CHUNK VBZ CHUNK	8	41	0	41
DT CHUNK VBP DT CHUNK IN CHUNK	21	39	1	40

Figure A.1.6: A snapshot for the results after applying rule 5 on the corpus.

Total Count	2583	3502	330	3832
pos	num_pos_structures	non_parsed	parsed	total
CHUNK VBZ DT CHUNK IN CHUNK	72	91	62	153
CHUNK VBD DT CHUNK IN CHUNK	3	121	0	121
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK	49	59	54	113
CHUNK VBZ DT CHUNK IN CHUNK IN DT CHUNK IN CHUNK	35	100	0	100
PRP VBZ IN DT CHUNK IN DT CHUNK	19	86	0	86
PRP VBD DT CHUNK IN DT CHUNK	45	59	4	63
PRP VBZ DT CHUNK IN CHUNK	19	35	26	61
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK	5	54	1	55
PRP VBZ CHUNK CC CHUNK	18	45	3	48
PRP VBZ DT CHUNK IN DT CHUNK	21	45	0	45
DT CHUNK VBZ DT CHUNK IN CHUNK	20	45	0	45
CHUNK VBZ DT CD CHUNK	12	44	1	45
PRP VBD DT CHUNK IN CHUNK	23	44	0	44
CHUNK VBD DT CHUNK CC CHUNK	18	42	0	42
PRP$ CHUNK VBZ CHUNK	8	41	0	41
DT CHUNK VBP DT CHUNK IN CHUNK	21	39	0	39
PRP VBD DT CHUNK	31	38	0	38
CHUNK VBZ DT CHUNK IN DT CHUNK	37	37	0	37
DT CHUNK VBZ DT JJ CHUNK	19	37	0	37
CHUNK VBZ DT CHUNK WDT VBZ IN CHUNK	4	36	0	36

Figure A.1.7: A snapshot for the results after applying rule 6 on the corpus.

Total Count	2419	2099	1403	3502
pos	num_pos_structures	non_parsed	parsed	total
CHUNK VBD DT CHUNK IN CHUNK	3	121	0	121
CHUNK VBZ DT CHUNK IN CHUNK IN DT CHUNK IN CHUNK	35	0	100	100
CHUNK VBZ DT CHUNK IN CHUNK	57	11	80	91
PRP VBZ IN DT CHUNK IN DT CHUNK	19	42	44	86
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK	39	26	33	59
PRP VBD DT CHUNK IN DT CHUNK	41	8	51	59
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK	4	54	0	54
DT CHUNK VBZ DT CHUNK IN CHUNK	20	9	36	45
PRP VBZ CHUNK CC CHUNK	15	17	28	45
PRP VBZ DT CHUNK IN DT CHUNK	21	24	21	45
PRP VBD DT CHUNK IN CHUNK	23	36	8	44
CHUNK VBZ DT CD CHUNK	11	2	42	44
CHUNK VBD DT CHUNK CC CHUNK	18	6	36	42
PRP$ CHUNK VBZ CHUNK	8	37	4	41
DT CHUNK VBP DT CHUNK IN CHUNK	21	20	19	39
PRP VBD DT CHUNK	31	37	1	38
DT CHUNK VBZ DT JJ CHUNK	19	37	0	37
CHUNK VBZ DT CHUNK IN DT CHUNK	37	14	23	37
CHUNK VBZ DT CHUNK WDT VBZ IN CHUNK	4	36	0	36
PRP VBZ DT CHUNK IN CHUNK	15	9	26	35

Figure A.1.8: A snapshot for the results after applying rule 7 on the corpus.

Total Count	1627	1838	261	2099
pos	num_pos_structures	non_parsed	parsed	total
CHUNK VBD DT CHUNK IN CHUNK	3	121	0	121
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK	4	54	0	54
PRP VBZ IN DT CHUNK IN DT CHUNK	16	42	0	42
DT CHUNK VBZ DT JJ CHUNK	19	9	28	37
PRP$ CHUNK VBZ CHUNK	5	37	0	37
PRP VBD DT CHUNK	30	10	27	37
CHUNK VBZ DT CHUNK WDT VBZ IN CHUNK	4	36	0	36
PRP VBD DT CHUNK IN CHUNK	22	36	0	36
PRP VBZ DT CHUNK IN DT CHUNK IN CHUNK	24	33	0	33
DT CHUNK VBZ DT CHUNK IN CHUNK IN DT CHUNK	5	30	0	30
PRP VBZ DT CHUNK IN CD CHUNK	21	29	0	29
PRP VBZ CHUNK IN DT CHUNK	23	3	25	28
CHUNK VBZ DT CHUNK CC CHUNK IN DT CHUNK IN DT CHUNK IN CHUNK	3	28	0	28
PRP VBZ IN CHUNK CC CHUNK CC VBZ DT CHUNK	9	27	0	27
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK	21	26	0	26
CHUNK VBZ DT JJS CHUNK IN CHUNK	16	25	0	25
PRP VBZ DT CHUNK IN DT CHUNK	12	24	0	24
PRP VBZ RB DT CHUNK	19	8	15	23
PRP VBZ DT CHUNK IN CD	19	23	0	23
PRP VBD CHUNK IN CHUNK	5	21	0	21

Figure A.1.9: A snapshot for the results after applying rule 8 on the corpus.

Total Count	1702	1568	270	1838
pos	num_pos_structures	non_parsed	parsed	total
DT CHUNK VBZ IN DT CHUNK CC VBZ IN DT CHUNK	3	0	127	127
CHUNK VBZ CHUNK IN DT CHUNK	36	61	0	61
CHUNK VBZ CD IN DT CHUNK IN CHUNK	25	59	0	59
CHUNK VBZ IN DT CHUNK	6	16	23	39
CHUNK VBZ DT CHUNK IN CHUNK IN DT CHUNK IN CD	4	0	34	34
CHUNK VBZ CD IN DT CHUNK IN DT CHUNK	29	34	0	34
CHUNK VBZ RB DT CHUNK	14	11	19	30
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK	23	29	0	29
DT CHUNK VBZ CD IN DT CHUNK	4	27	0	27
CHUNK CC CHUNK VBZ IN DT CHUNK IN CHUNK	8	22	0	22
CHUNK VBZ CHUNK IN DT CHUNK IN CHUNK	12	22	0	22
CHUNK VBZ DT CHUNK IN DT CHUNK	20	20	2	22
CHUNK VBZ DT CHUNK IN DT CHUNK CHUNK	13	21	0	21
DT CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK	14	8	13	21
DT CHUNK VBZ DT CHUNK IN DT CHUNK	16	13	3	16
DT CHUNK VBZ IN DT CHUNK	11	2	13	15
CHUNK VBZ DT CHUNK	15	13	1	14
CHUNK VBZ DT CHUNK CC CHUNK IN DT CHUNK IN DT CHUNK IN CHUNK	5	12	0	12
CHUNK VBP DT CHUNK	11	12	0	12
CHUNK IN DT CHUNK VBZ CHUNK	3	12	0	12

Figure A.1.10: A snapshot for the results after applying rule 9 on the corpus.

Total Count	num_pos_structures	non_parsed	parsed	total
pos	1380	1340	228	1568
CHUNK CHUNK VBP	10	67	0	67
CHUNK VBZ CD IN DT CHUNK IN CHUNK	7	59	0	59
CHUNK VBZ CD IN DT CHUNK IN DT CHUNK	8	34	0	34
CHUNK VBZ CHUNK IN DT CHUNK	12	28	33	61
CHUNK CC CHUNK VBZ IN DT CHUNK IN CHUNK	4	22	0	22
CHUNK VBZ DT CHUNK IN DT CHUNK CHUNK	7	21	0	21
CHUNK VBZ DT CHUNK IN DT CHUNK	11	20	0	20
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK	7	18	11	29
CHUNK VBZ IN DT CHUNK	7	14	2	16
CHUNK IN DT CHUNK VBZ CHUNK	3	12	0	12
CHUNK VBP DT CHUNK	8	12	0	12
CHUNK VBZ DT CHUNK CC CHUNK IN DT CHUNK IN DT CHUNK II	4	12	0	12
CHUNK VBZ DT CHUNK	10	12	1	13
CHUNK VBZ IN CHUNK CC CHUNK CC VBZ DT RB CHUNK	4	11	0	11
CHUNK VBZ DT CHUNK IN DT CHUNK CHUNK CHUNK	5	11	0	11
DT CHUNK IN CHUNK VBZ CD IN DT CHUNK IN CHUNK	4	11	0	11
CHUNK VBZ RB DT CHUNK	6	11	0	11
DT CHUNK CHUNK IN DT CHUNK CC VBZ IN DT CHUNK	4	11	0	11
CHUNK VBZ CHUNK IN DT CHUNK IN CHUNK	6	10	12	22
CHUNK VBZ DT RB DT CHUNK	4	10	0	10

Figure A.1.11: A snapshot for the results after applying rule 10 on the corpus.

A.2 OWL TRIPLES RESULTS OF EACH RULE

Total Count	6671	18535	554	19089
pos	num_pos_structures	non_parsed	parsed	total
CHUNK VBZ DT CHUNK IN CHUNK	366	2606	58	2664
CHUNK VBZ DT CHUNK	125	1098	301	1399
CHUNK VBZ DT CHUNK IN CHUNK IN DT CHUNK	77	1088	0	1088
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK	224	921	5	926
CHUNK VBD DT CHUNK	82	885	0	885
DT CHUNK VBZ DT CHUNK IN CHUNK	194	450	19	469
CHUNK VBD DT CHUNK CC CHUNK	98	460	0	460
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN DT CHUNK IN CHUNK IN CHUNK	31	402	0	402
CHUNK VBZ DT CHUNK IN DT CHUNK	158	303	15	318
PRP VBZ DT CHUNK IN CHUNK	50	299	0	299
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK IN CHUNK	47	289	0	289
CHUNK VBZ DT CHUNK CC CHUNK	83	261	2	263
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN CHUNK	79	237	0	237
PRP VBZ DT CHUNK IN DT CHUNK	61	220	0	220
CHUNK VBZ DT CHUNK IN CHUNK IN CHUNK	122	212	0	212
DT CHUNK VBZ DT CHUNK	72	169	36	205
PRP VBZ IN DT CHUNK IN DT CHUNK IN DT CHUNK IN CHUNK	12	203	0	203
PRP VBZ IN CHUNK	23	198	0	198
PRP$ CHUNK VBZ CHUNK	17	194	0	194
DT CHUNK VBZ DT CHUNK IN DT CHUNK	104	177	9	186

Figure A.2.12: A snapshot for the results after applying the pre-processing module on the corpus.

Total Count	6643	16609	1926	18535
pos	num_pos_structures	non_parsed	parsed	total
CHUNK VBZ DT CHUNK IN CHUNK	366	2437	169	2606
CHUNK VBZ DT CHUNK	124	12	1086	1098
CHUNK VBZ DT CHUNK IN CHUNK IN DT CHUNK	77	1088	0	1088
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK	224	876	45	921
CHUNK VBD DT CHUNK	82	885	0	885
CHUNK VBD DT CHUNK CC CHUNK	98	460	0	460
DT CHUNK VBZ DT CHUNK IN CHUNK	193	417	33	450
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN DT CHUNK IN CHUNK IN CHUNK	31	402	0	402
CHUNK VBZ DT CHUNK IN DT CHUNK	155	231	72	303
PRP VBZ DT CHUNK IN CHUNK	50	299	0	299
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK IN CHUNK	47	289	0	289
CHUNK VBZ DT CHUNK CC CHUNK	81	258	3	261
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN CHUNK	79	237	0	237
PRP VBZ DT CHUNK IN DT CHUNK	61	220	0	220
CHUNK VBZ DT CHUNK IN CHUNK IN CHUNK	122	212	0	212
PRP VBZ IN DT CHUNK IN DT CHUNK IN DT CHUNK IN CHUNK	12	203	0	203
PRP VBZ IN CHUNK	23	198	0	198
PRP$ CHUNK VBZ CHUNK	17	194	0	194
PRP VBZ DT CHUNK	24	184	0	184
DT CHUNK VBZ DT CHUNK IN DT CHUNK	104	104	73	177

Figure A.2.13: A snapshot for the results after applying rule 1 on the corpus.

Total Count	6218	16156	453	16609
pos	num_pos_structures	non_parsed	parsed	total
CHUNK VBZ DT CHUNK IN CHUNK	350	2227	210	2437
CHUNK VBZ DT CHUNK IN CHUNK IN DT CHUNK	77	1087	1	1088
CHUNK VBD DT CHUNK	82	885	0	885
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK	211	853	23	876
CHUNK VBD DT CHUNK CC CHUNK	98	460	0	460
DT CHUNK VBZ DT CHUNK IN CHUNK	179	355	62	417
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN DT CHUNK IN CHUNK IN CHUNK	31	402	0	402
PRP VBZ DT CHUNK IN CHUNK	50	299	0	299
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK IN CHUNK	47	289	0	289
CHUNK VBZ DT CHUNK CC CHUNK	78	258	0	258
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN CHUNK	79	237	0	237
CHUNK VBZ DT CHUNK IN DT CHUNK	129	231	0	231
PRP VBZ DT CHUNK IN DT CHUNK	61	220	0	220
CHUNK VBZ DT CHUNK IN CHUNK IN CHUNK	122	208	4	212
PRP VBZ IN DT CHUNK IN DT CHUNK IN DT CHUNK IN CHUNK	12	203	0	203
PRP VBZ IN CHUNK	23	198	0	198
PRP$ CHUNK VBZ CHUNK	17	194	0	194
PRP VBZ DT CHUNK	24	184	0	184
PRP VBZ IN DT CHUNK	23	175	0	175
PRP VBZ IN DT CHUNK IN CHUNK	43	163	0	163

Figure A.2.14: A snapshot for the results after applying rule 2 on the corpus.

Total Count	6019	14875	715	15590
pos	num_pos_structures	non_parsed	parsed	total
CHUNK VBZ DT CHUNK IN CHUNK	314	2227	0	2227
CHUNK VBZ DT CHUNK IN CHUNK IN DT CHUNK	76	1087	0	1087
CHUNK VBD DT CHUNK	82	885	0	885
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK	206	853	0	853
CHUNK VBD DT CHUNK CC CHUNK	98	460	0	460
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN DT CHUNK IN CHUNK IN CHUNK	31	402	0	402
DT CHUNK VBZ DT CHUNK IN CHUNK	144	355	0	355
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK IN CHUNK	47	289	0	289
PRP VBZ DT CHUNK IN CHUNK	50	113	174	287
CHUNK VBZ DT CHUNK CC CHUNK	78	258	0	258
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN CHUNK	79	237	0	237
CHUNK VBZ DT CHUNK IN DT CHUNK	129	231	0	231
CHUNK VBZ DT CHUNK IN CHUNK IN CHUNK	120	208	0	208
PRP VBZ DT CHUNK IN DT CHUNK	61	73	123	196
PRP$ CHUNK VBZ CHUNK	17	41	149	190
PRP VBZ IN CHUNK	23	189	0	189
VBD DT IN CHUNK CC IN CHUNK	94	148	0	148
PRP VBZ DT CHUNK	24	13	132	145
PRP VBZ IN DT CHUNK	23	144	0	144
PRP VBZ IN DT CHUNK IN CHUNK	43	142	0	142

Figure A.2.15: A snapshot for the results after applying rule 3 on the corpus.

Total Count	5900	13499	1368	14867
pos	num_pos_structures	non_parsed	parsed	total
CHUNK VBZ DT CHUNK IN CHUNK	314	2227	0	2227
CHUNK VBZ DT CHUNK IN CHUNK IN DT CHUNK	76	1087	0	1087
CHUNK VBD DT CHUNK	82	13	872	885
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK	206	853	0	853
CHUNK VBD DT CHUNK CC CHUNK	98	460	0	460
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN DT CHUNK IN CHUNK IN	31	402	0	402
DT CHUNK VBZ DT CHUNK IN CHUNK	144	355	0	355
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK IN CHUNK	47	289	0	289
CHUNK VBZ DT CHUNK CC CHUNK	78	258	0	258
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN CHUNK	79	237	0	237
CHUNK VBZ DT CHUNK IN DT CHUNK	129	231	0	231
CHUNK VBZ DT CHUNK IN CHUNK IN CHUNK	120	208	0	208
PRP VBZ IN CHUNK	23	189	0	189
VBD DT IN CHUNK CC IN CHUNK	94	106	42	148
PRP VBZ IN DT CHUNK	23	144	0	144
CHUNK VBD DT CHUNK IN CHUNK	9	141	0	141
PRP VBZ IN DT CHUNK IN CHUNK	43	140	0	140
PRP VBZ DT CHUNK IN CHUNK	33	113	0	113
DT CHUNK VBD IN CHUNK CD CC VBD IN CHUNK CD	12	112	0	112
DT CHUNK VBZ CHUNK	66	67	38	105

Figure A.2.16: A snapshot for the results after applying rule 4 on the corpus.

Total Count	5603	13059	440	13499
pos	num_pos_structures	non_parsed	parsed	total
CHUNK VBZ DT CHUNK IN CHUNK	314	2067	160	2227
CHUNK VBZ DT CHUNK IN CHUNK IN DT CHUNK	76	1087	0	1087
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK	206	842	11	853
CHUNK VBD DT CHUNK CC CHUNK	98	460	0	460
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN DT CHUNK IN CHUNK IN CHUNK	31	402	0	402
DT CHUNK VBZ DT CHUNK IN CHUNK	144	265	90	355
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK IN CHUNK	47	289	0	289
CHUNK VBZ DT CHUNK CC CHUNK	78	258	0	258
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN CHUNK	79	237	0	237
CHUNK VBZ DT CHUNK IN DT CHUNK	129	231	0	231
CHUNK VBZ DT CHUNK IN CHUNK IN CHUNK	120	208	0	208
PRP VBZ IN CHUNK	23	189	0	189
PRP VBZ IN DT CHUNK	23	144	0	144
CHUNK VBD DT CHUNK IN CHUNK	9	141	0	141
PRP VBZ IN DT CHUNK IN CHUNK	43	140	0	140
PRP VBZ DT CHUNK IN CHUNK	33	91	22	113
DT CHUNK VBD IN CHUNK CD CC VBD IN CHUNK CD	12	112	0	112
VBD DT IN CHUNK CC IN CHUNK	74	99	7	106
DT CHUNK VBZ DT CHUNK IN DT CHUNK	67	103	1	104
PRP VBD DT CHUNK IN DT CHUNK	76	104	0	104

Figure A.2.17: A snapshot for the results after applying rule 5 on the corpus.

Total Count	5463	12386	673	13059
pos	num_pos_structures	non_parsed	parsed	total
CHUNK VBZ DT CHUNK IN CHUNK	300	1614	453	2067
CHUNK VBZ DT CHUNK IN CHUNK IN DT CHUNK	76	1085	2	1087
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK	201	837	5	842
CHUNK VBD DT CHUNK CC CHUNK	98	460	0	460
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN DT CHUNK IN CHUNK IN CHUNK	31	402	0	402
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK IN CHUNK	47	288	1	289
DT CHUNK VBZ DT CHUNK IN CHUNK	121	230	35	265
CHUNK VBZ DT CHUNK CC CHUNK	78	258	0	258
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN CHUNK	79	237	0	237
CHUNK VBZ DT CHUNK IN DT CHUNK	129	231	0	231
CHUNK VBZ DT CHUNK IN CHUNK IN CHUNK	120	203	5	208
PRP VBZ IN CHUNK	23	189	0	189
PRP VBZ IN DT CHUNK	23	144	0	144
CHUNK VBD DT CHUNK IN CHUNK	9	141	0	141
PRP VBZ IN DT CHUNK IN CHUNK	43	140	0	140
DT CHUNK VBD IN CHUNK CD CC VBD IN CHUNK CD	12	112	0	112
PRP VBD DT CHUNK IN DT CHUNK	76	104	0	104
DT CHUNK VBZ DT CHUNK IN DT CHUNK	67	103	0	103
CHUNK VBZ DT JJ CHUNK	23	102	0	102
CHUNK VBZ DT CHUNK IN CHUNK IN DT CHUNK IN CHUNK	36	101	0	101

Figure A.2.18: A snapshot for the results after applying rule 6 on the corpus.

Total Count	5367	11616	770	12386
pos	num_pos_structures	non_parsed	parsed	total
CHUNK VBZ DT CHUNK IN CHUNK	279	1586	28	1614
CHUNK VBZ DT CHUNK IN CHUNK IN DT CHUNK	75	1084	1	1085
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK	198	835	2	837
CHUNK VBD DT CHUNK CC CHUNK	98	460	0	460
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN DT CHUNK IN CHUNK IN CHUNK	31	402	0	402
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK IN CHUNK	46	288	0	288
CHUNK VBZ DT CHUNK CC CHUNK	78	258	0	258
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN CHUNK	79	237	0	237
CHUNK VBZ DT CHUNK IN DT CHUNK	129	223	8	231
DT CHUNK VBZ DT CHUNK IN CHUNK	114	185	45	230
CHUNK VBZ DT CHUNK IN CHUNK IN CHUNK	119	202	1	203
PRP VBZ IN CHUNK	23	189	0	189
PRP VBZ IN DT CHUNK	23	144	0	144
CHUNK VBD DT CHUNK IN CHUNK	9	141	0	141
PRP VBZ IN DT CHUNK IN CHUNK	43	140	0	140
DT CHUNK VBD IN CHUNK CD CC VBD IN CHUNK CD	12	112	0	112
PRP VBD DT CHUNK IN DT CHUNK	76	58	46	104
DT CHUNK VBZ DT CHUNK IN DT CHUNK	67	88	15	103
CHUNK VBZ DT JJ CHUNK	23	102	0	102
CHUNK VBZ DT CHUNK IN CHUNK IN DT CHUNK IN CHUNK	36	2	99	101

Figure A.2.19: A snapshot for the results after applying rule 7 on the corpus.

Total Count	5047	11420	196	11616
pos	num_pos_structures	non_parsed	parsed	total
CHUNK VBZ DT CHUNK IN CHUNK	271	1586	0	1586
CHUNK VBZ DT CHUNK IN CHUNK IN DT CHUNK	74	1084	0	1084
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK	196	835	0	835
CHUNK VBD DT CHUNK CC CHUNK	98	460	0	460
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN DT CHUNK IN CHUNK IN CHUI	31	402	0	402
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK IN CHUNK	46	288	0	288
CHUNK VBZ DT CHUNK CC CHUNK	78	258	0	258
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN CHUNK	79	237	0	237
CHUNK VBZ DT CHUNK IN DT CHUNK	124	223	0	223
CHUNK VBZ DT CHUNK IN CHUNK IN CHUNK	118	202	0	202
PRP VBZ IN CHUNK	23	189	0	189
DT CHUNK VBZ DT CHUNK IN CHUNK	109	185	0	185
PRP VBZ IN DT CHUNK	23	144	0	144
CHUNK VBD DT CHUNK IN CHUNK	9	141	0	141
PRP VBZ IN DT CHUNK IN CHUNK	43	140	0	140
DT CHUNK VBD IN CHUNK CD CC VBD IN CHUNK CD	12	112	0	112
CHUNK VBZ DT JJ CHUNK	23	102	0	102
PRP VBZ IN DT CHUNK IN DT CHUNK	21	93	0	93
DT CHUNK VBZ DT CHUNK IN DT CHUNK	57	88	0	88
PRP VBD DT CHUNK IN CHUNK	52	87	0	87

Figure A.2.20: A snapshot for the results after applying rule 8 on the corpus.

Total Count	5551	11303	117	11420
pos	num_pos_structures	non_parsed	parsed	total
CHUNK VBZ DT CHUNK IN DT CHUNK	579	1921	0	1921
CHUNK VBZ DT CHUNK IN CHUNK IN DT CHUNK	83	1078	0	1078
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK	277	634	0	634
CHUNK VBZ DT CHUNK CC CHUNK	135	619	0	619
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK	113	490	0	490
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN DT CHUNK IN CHUNK IN CHUNK	29	394	0	394
CHUNK VBZ IN DT CHUNK	74	277	31	308
DT CHUNK VBZ DT CHUNK IN DT CHUNK	186	300	1	301
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK IN CHUNK	52	298	0	298
CHUNK VBZ DT CHUNK IN CHUNK	119	291	0	291
CHUNK VBZ DT CHUNK CC DT CHUNK	145	236	0	236
CHUNK VBZ IN DT CHUNK IN DT CHUNK	81	150	1	151
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN CHUNK	31	144	0	144
DT CHUNK VBZ IN DT CHUNK CC VBZ IN DT CHUNK	3	133	0	133
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN DT CHUNK	46	129	0	129
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK IN CHUNK IN CHUNK	14	124	0	124
CHUNK CHUNK IN DT CHUNK	45	119	0	119
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN DT CHUNK IN CHUNK	15	105	0	105
DT CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK	83	98	1	99
DT CHUNK VBZ CD IN DT CHUNK	6	90	0	90

Figure A.2.21: A snapshot for the results after applying rule 9 on the corpus.

A.3 OWL TRIPLES RESULTS AFTER ITERATIONS

Total Count	5414	11169	9	11178
pos	num_pos_structures	non_parsed	parsed	total
CHUNK VBZ DT CHUNK IN DT CHUNK	577	1919	0	1919
CHUNK VBZ DT CHUNK IN CHUNK IN DT CHUNK	82	1076	0	1076
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK	277	633	0	633
CHUNK VBZ DT CHUNK CC CHUNK	135	619	0	619
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK	113	489	0	489
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN DT CHUNK IN CHUNK IN CHUNK	29	394	0	394
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK IN CHUNK	52	298	0	298
DT CHUNK VBZ DT CHUNK IN DT CHUNK	181	291	0	291
CHUNK VBZ DT CHUNK IN CHUNK	117	287	2	289
CHUNK VBZ IN DT CHUNK	70	275	0	275
CHUNK VBZ DT CHUNK CC DT CHUNK	145	236	0	236
CHUNK VBZ IN DT CHUNK IN DT CHUNK	80	150	0	150
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN CHUNK	31	144	0	144
DT CHUNK VBZ IN DT CHUNK CC VBZ IN DT CHUNK	3	133	0	133
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN DT CHUNK	46	129	0	129
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK IN CHUNK IN CHUNK	14	124	0	124
CHUNK CHUNK IN DT CHUNK	45	119	0	119
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN DT CHUNK IN CHUNK	15	105	0	105
DT CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK	81	97	0	97
DT CHUNK VBZ CD IN DT CHUNK	6	90	0	90

Figure A.3.22: A snapshot for the results of extracting an OWL triple after eliminating text (iteration 1).

Total Count	5409	11064	105	11169
pos	num_pos_structures	non_parsed	parsed	total
CHUNK VBZ DT CHUNK IN DT CHUNK	577	1919	0	1919
CHUNK VBZ DT CHUNK IN CHUNK IN DT CHUNK	82	1076	0	1076
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK	277	633	0	633
CHUNK VBZ DT CHUNK CC CHUNK	135	595	24	619
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK	113	489	0	489
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN DT CHUNK IN CHUNK IN CHUNK	29	394	0	394
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK IN CHUNK	52	298	0	298
DT CHUNK VBZ DT CHUNK IN DT CHUNK	181	291	0	291
CHUNK VBZ DT CHUNK IN CHUNK	115	278	9	287
CHUNK VBZ IN DT CHUNK	70	275	0	275
CHUNK VBZ DT CHUNK CC DT CHUNK	145	236	0	236
CHUNK VBZ IN DT CHUNK IN DT CHUNK	80	150	0	150
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN CHUNK	31	144	0	144
DT CHUNK VBZ IN DT CHUNK CC VBZ IN DT CHUNK	3	133	0	133
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN DT CHUNK	46	129	0	129
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK IN CHUNK IN CHUNK	14	124	0	124
CHUNK CHUNK IN DT CHUNK	45	119	0	119
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN DT CHUNK IN CHUNK	15	105	0	105
DT CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK	81	97	0	97
DT CHUNK VBZ CD IN DT CHUNK	6	90	0	90

Figure A.3.23: A snapshot for the results of extracting an OWL triple after eliminating text (iteration 2).

Total Count	5341	5398	5666	11064
pos	num_pos_structures	non_parsed	parsed	total
CHUNK VBZ DT CHUNK IN DT CHUNK	577	16	1903	1919
CHUNK VBZ DT CHUNK IN CHUNK IN DT CHUNK	82	17	1059	1076
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK	277	560	73	633
CHUNK VBZ DT CHUNK CC CHUNK	129	6	589	595
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK	113	317	172	489
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN DT CHUNK IN CHUNK IN CHU	29	388	6	394
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK IN CHUNK	52	293	5	298
DT CHUNK VBZ DT CHUNK IN DT CHUNK	181	2	289	291
CHUNK VBZ DT CHUNK IN CHUNK	112	13	265	278
CHUNK VBZ IN DT CHUNK	70	275	0	275
CHUNK VBZ DT CHUNK CC DT CHUNK	145	2	234	236
CHUNK VBZ IN DT CHUNK IN DT CHUNK	80	141	9	150
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN CHUNK	31	105	39	144
DT CHUNK VBZ IN DT CHUNK CC VBZ IN DT CHUNK	3	133	0	133
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN DT CHUNK	46	125	4	129
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK IN CHUNK IN CHUNK	14	122	2	124
CHUNK CHUNK IN DT CHUNK	45	119	0	119
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN DT CHUNK IN CHUNK	15	104	1	105
DT CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK	81	75	22	97
DT CHUNK VBZ CD IN DT CHUNK	6	0	90	90

Figure A.3.24: A snapshot for the results of extracting an OWL triple after eliminating text (iteration 3).

Total Count	3261	5375	23	5398
pos	num_pos_structures	non_parsed	parsed	total
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK	226	560	0	560
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN DT CHUNK IN CHUNK IN CHUNK	29	388	0	388
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK	67	317	0	317
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK IN CHUNK	48	293	0	293
CHUNK VBZ IN DT CHUNK	70	275	0	275
CHUNK VBZ IN DT CHUNK IN DT CHUNK	75	141	0	141
DT CHUNK VBZ IN DT CHUNK CC VBZ IN DT CHUNK	3	133	0	133
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN DT CHUNK	43	125	0	125
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK IN CHUNK IN CHUNK	14	122	0	122
CHUNK CHUNK IN DT CHUNK	45	119	0	119
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN CHUNK	26	105	0	105
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN DT CHUNK IN CHUNK	15	104	0	104
CHUNK	67	89	0	89
CHUNK VBZ IN DT CHUNK IN CHUNK	23	83	0	83
CHUNK CHUNK	5	77	0	77
DT CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK	62	75	0	75
CHUNK CHUNK VBP	6	67	0	67
CHUNK CHUNK IN CHUNK	12	67	0	67
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK IN DT CHUNK	52	64	0	64
CHUNK VBZ CD IN DT CHUNK IN CHUNK	24	58	0	58

Figure A.3.25: A snapshot for the results of extracting an OWL triple after eliminating text (iteration 4).

Total Count	3235	5218	157	5375
pos	num_pos_structures	non_parsed	parsed	total
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK	226	555	5	560
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN DT CHUNK IN CHUNK IN CHUNK	29	388	0	388
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK	67	315	2	317
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK IN CHUNK	48	293	0	293
CHUNK VBZ IN DT CHUNK	70	275	0	275
CHUNK VBZ IN DT CHUNK IN DT CHUNK	75	141	0	141
DT CHUNK VBZ IN DT CHUNK CC VBZ IN DT CHUNK	3	133	0	133
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN DT CHUNK	43	125	0	125
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK IN CHUNK IN CHUNK	14	122	0	122
CHUNK CHUNK IN DT CHUNK	45	119	0	119
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN CHUNK	26	105	0	105
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN DT CHUNK IN CHUNK	15	104	0	104
CHUNK	67	89	0	89
CHUNK VBZ IN DT CHUNK IN CHUNK	23	83	0	83
CHUNK CHUNK	5	77	0	77
DT CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK	62	75	0	75
CHUNK CHUNK IN CHUNK	12	67	0	67
CHUNK CHUNK VBP	6	67	0	67
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK IN DT CHUNK	52	64	0	64
CHUNK VBZ CD IN DT CHUNK IN CHUNK	24	58	0	58

Figure A.3.26: A snapshot for the results of extracting an OWL triple after eliminating text (iteration 5).

Total Count	3106	3081	2137	5218
pos	num_pos_structures	non_parsed	parsed	total
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK	222	5	550	555
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN DT CHUNK IN CHUNK IN CHUNK	29	237	151	388
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK	65	12	303	315
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK IN CHUNK	48	35	258	293
CHUNK VBZ IN DT CHUNK	70	275	0	275
CHUNK VBZ IN DT CHUNK IN DT CHUNK	75	141	0	141
DT CHUNK VBZ IN DT CHUNK CC VBZ IN DT CHUNK	3	133	0	133
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN DT CHUNK	43	32	93	125
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK IN CHUNK IN CHUNK	14	38	84	122
CHUNK CHUNK IN DT CHUNK	45	119	0	119
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN CHUNK	26	9	96	105
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN DT CHUNK IN CHUNK	15	7	97	104
CHUNK	67	89	0	89
CHUNK VBZ IN DT CHUNK IN CHUNK	23	83	0	83
CHUNK CHUNK	5	77	0	77
DT CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK	62	2	73	75
CHUNK CHUNK IN CHUNK	12	67	0	67
CHUNK CHUNK VBP	6	67	0	67
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK IN DT CHUNK	52	56	8	64
CHUNK VBZ CD IN DT CHUNK IN CHUNK	24	58	0	58

Figure A.3.27: A snapshot for the results of extracting an OWL triple after eliminating text (iteration 6).

Total Count	2216	3068	13	3081
pos	num_pos_structures	non_parsed	parsed	total
CHUNK VBZ IN DT CHUNK	70	275	0	275
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN DT CHUNK IN CHUNK IN CHUNK	27	237	0	237
CHUNK VBZ IN DT CHUNK IN DT CHUNK	75	141	0	141
DT CHUNK VBZ IN DT CHUNK CC VBZ IN DT CHUNK	3	133	0	133
CHUNK CHUNK IN DT CHUNK	45	119	0	119
CHUNK	67	89	0	89
CHUNK VBZ IN DT CHUNK IN CHUNK	23	83	0	83
CHUNK CHUNK	5	77	0	77
CHUNK CHUNK IN CHUNK	12	67	0	67
CHUNK CHUNK VBP	6	67	0	67
CHUNK VBZ CD IN DT CHUNK IN CHUNK	24	58	0	58
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK IN DT CHUNK	46	56	0	56
DT CHUNK VBZ IN DT CHUNK	31	53	0	53
CHUNK VBZ IN CHUNK	8	46	0	46
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK IN CHUNK IN CHUNK	11	38	0	38
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK IN CHUNK	29	35	0	35
CHUNK VBZ CD IN DT CHUNK IN DT CHUNK	29	34	0	34
CHUNK CHUNK IN CD	5	34	0	34
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN DT CHUNK	14	32	0	32
CHUNK VBZ CHUNK IN DT CHUNK	26	31	0	31

Figure A.3.28: A snapshot for the results of extracting an OWL triple after eliminating text (iteration 7).

Total Count	2201	3049	19	3068
pos	num_pos_structures	non_parsed	parsed	total
CHUNK VBZ IN DT CHUNK	70	275	0	275
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN DT CHUNK IN CHUNK IN CHUNK	27	236	1	237
CHUNK VBZ IN DT CHUNK IN DT CHUNK	75	141	0	141
DT CHUNK VBZ IN DT CHUNK CC VBZ IN DT CHUNK	3	133	0	133
CHUNK CHUNK IN DT CHUNK	45	119	0	119
CHUNK	67	89	0	89
CHUNK VBZ IN DT CHUNK IN CHUNK	23	83	0	83
CHUNK CHUNK	5	77	0	77
CHUNK CHUNK VBP	6	67	0	67
CHUNK CHUNK IN CHUNK	12	67	0	67
CHUNK VBZ CD IN DT CHUNK IN CHUNK	24	58	0	58
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK IN DT CHUNK	46	55	1	56
DT CHUNK VBZ IN DT CHUNK	31	53	0	53
CHUNK VBZ IN CHUNK	8	46	0	46
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK IN CHUNK IN CHUNK	11	38	0	38
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK IN CHUNK	29	35	0	35
CHUNK CHUNK IN CD	5	34	0	34
CHUNK VBZ CD IN DT CHUNK IN DT CHUNK	29	34	0	34
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN DT CHUNK	14	31	1	32
CHUNK VBZ CHUNK IN DT CHUNK	26	31	0	31

Figure A.3.29: A snapshot for the results of extracting an OWL triple after eliminating text (iteration 8).

Total Count	2176	2707	342	3049
pos	num_pos_structures	non_parsed	parsed	total
CHUNK VBZ IN DT CHUNK	70	275	0	275
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN DT CHUNK IN CHUNK IN CHUNK	27	133	103	236
CHUNK VBZ IN DT CHUNK IN DT CHUNK	75	141	0	141
DT CHUNK VBZ IN DT CHUNK CC VBZ IN DT CHUNK	3	133	0	133
CHUNK CHUNK IN DT CHUNK	45	119	0	119
CHUNK	67	89	0	89
CHUNK VBZ IN DT CHUNK IN CHUNK	23	83	0	83
CHUNK CHUNK	5	77	0	77
CHUNK CHUNK IN CHUNK	12	67	0	67
CHUNK CHUNK VBP	6	67	0	67
CHUNK VBZ CD IN DT CHUNK IN CHUNK	24	58	0	58
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK IN DT CHUNK	45	0	55	55
DT CHUNK VBZ IN DT CHUNK	31	53	0	53
CHUNK VBZ IN CHUNK	8	46	0	46
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK IN CHUNK IN CHUNK	11	1	37	38
CHUNK VBZ DT CHUNK IN DT CHUNK IN DT CHUNK IN CHUNK	29	3	32	35
CHUNK CHUNK IN CD	5	34	0	34
CHUNK VBZ CD IN DT CHUNK IN DT CHUNK	29	34	0	34
CHUNK VBZ CHUNK IN DT CHUNK	26	31	0	31
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN DT CHUNK	13	0	31	31

Figure A.3.30: A snapshot for the results of extracting an OWL triple after eliminating text (iteration 9).

Total Count	1965	2704	3	2707
pos	num_pos_structures	non_parsed	parsed	total
CHUNK VBZ IN DT CHUNK	70	275	0	275
CHUNK VBZ IN DT CHUNK IN DT CHUNK	75	141	0	141
DT CHUNK VBZ IN DT CHUNK CC VBZ IN DT CHUNK	3	133	0	133
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN DT CHUNK IN CHUNK IN CHUNK	13	133	0	133
CHUNK CHUNK IN DT CHUNK	45	119	0	119
CHUNK	67	89	0	89
CHUNK VBZ IN DT CHUNK IN CHUNK	23	83	0	83
CHUNK CHUNK	5	77	0	77
CHUNK CHUNK VBP	6	67	0	67
CHUNK CHUNK IN CHUNK	12	67	0	67
CHUNK VBZ CD IN DT CHUNK IN CHUNK	24	58	0	58
DT CHUNK VBZ IN DT CHUNK	31	53	0	53
CHUNK VBZ IN CHUNK	8	46	0	46
CHUNK CHUNK IN CD	5	34	0	34
CHUNK VBZ CD IN DT CHUNK IN DT CHUNK	29	34	0	34
CHUNK VBZ CHUNK IN DT CHUNK	26	31	0	31
CHUNK CHUNK IN DT CHUNK IN CHUNK	16	23	0	23
CHUNK VBP IN DT CHUNK	14	23	0	23
CHUNK CHUNK IN DT CHUNK IN DT CHUNK	22	23	0	23
CHUNK VBZ DT CHUNK CC CHUNK IN DT CHUNK IN DT CHUNK IN CHUNK	6	23	0	23

Figure A.3.31: A snapshot for the results of extracting an OWL triple after eliminating text (iteration 10).

Total Count	1956	2649	55	2704
pos	num_pos_structures	non_parsed	parsed	total
CHUNK VBZ IN DT CHUNK	70	275	0	275
CHUNK VBZ IN DT CHUNK IN DT CHUNK	75	141	0	141
DT CHUNK VBZ IN DT CHUNK CC VBZ IN DT CHUNK	3	133	0	133
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN DT CHUNK IN CHUNK IN CHUNK	13	97	36	133
CHUNK CHUNK IN DT CHUNK	45	119	0	119
CHUNK	67	89	0	89
CHUNK VBZ IN DT CHUNK IN CHUNK	23	83	0	83
CHUNK CHUNK	5	77	0	77
CHUNK CHUNK IN CHUNK	12	67	0	67
CHUNK CHUNK VBP	6	67	0	67
CHUNK VBZ CD IN DT CHUNK IN CHUNK	24	58	0	58
DT CHUNK VBZ IN DT CHUNK	31	53	0	53
CHUNK VBZ IN CHUNK	8	46	0	46
CHUNK VBZ CD IN DT CHUNK IN DT CHUNK	29	34	0	34
CHUNK CHUNK IN CD	5	34	0	34
CHUNK VBZ CHUNK IN DT CHUNK	26	31	0	31
CHUNK VBP IN DT CHUNK	14	23	0	23
CHUNK VBZ DT CHUNK CC CHUNK IN DT CHUNK IN DT CHUNK IN CHUNK	6	9	14	23
CHUNK CHUNK IN DT CHUNK IN CHUNK	16	23	0	23
CHUNK CHUNK IN DT CHUNK IN DT CHUNK	22	23	0	23

Figure A.3.32: A snapshot for the results of extracting an OWL triple after eliminating text (iteration 11).

	1938	2487	162	2649
pos	num_pos_structures	non_parsed	parsed	total
CHUNK VBZ IN DT CHUNK	70	275	0	275
CHUNK VBZ IN DT CHUNK IN DT CHUNK	75	141	0	141
DT CHUNK VBZ IN DT CHUNK CC VBZ IN DT CHUNK	3	133	0	133
CHUNK CHUNK IN DT CHUNK	45	119	0	119
CHUNK VBZ DT CHUNK IN DT CHUNK IN CHUNK IN DT CHUNK IN CHUNK IN CHUNK	12	3	94	97
CHUNK	67	89	0	89
CHUNK VBZ IN DT CHUNK IN CHUNK	23	83	0	83
CHUNK CHUNK	5	77	0	77
CHUNK CHUNK VBP	6	67	0	67
CHUNK CHUNK IN CHUNK	12	67	0	67
CHUNK VBZ CD IN DT CHUNK IN CHUNK	24	58	0	58
DT CHUNK VBZ IN DT CHUNK	31	53	0	53
CHUNK VBZ IN CHUNK	8	46	0	46
CHUNK CHUNK IN CD	5	34	0	34
CHUNK VBZ CD IN DT CHUNK IN DT CHUNK	29	34	0	34
CHUNK VBZ CHUNK IN DT CHUNK	26	31	0	31
CHUNK CHUNK IN DT CHUNK IN CHUNK	16	23	0	23
CHUNK VBP IN DT CHUNK	14	23	0	23
CHUNK CHUNK IN DT CHUNK IN DT CHUNK	22	23	0	23
CHUNK VBZ IN DT CHUNK IN DT CHUNK IN DT CHUNK IN CHUNK	10	22	0	22

Figure A.3.33: A snapshot for the results of extracting an OWL triple after eliminating text (iteration 12).

Total Count	2487	2487
pos	non_parsed	total
CHUNK VBZ IN DT CHUNK	275	275
CHUNK VBZ IN DT CHUNK IN DT CHUNK	141	141
DT CHUNK VBZ IN DT CHUNK CC VBZ IN DT CHUNK	133	133
CHUNK CHUNK IN DT CHUNK	119	119
CHUNK	89	89
CHUNK VBZ IN DT CHUNK IN CHUNK	83	83
CHUNK CHUNK	77	77
CHUNK CHUNK IN CHUNK	67	67
CHUNK CHUNK VBP	67	67
CHUNK VBZ CD IN DT CHUNK IN CHUNK	58	58
DT CHUNK VBZ IN DT CHUNK	53	53
CHUNK VBZ IN CHUNK	46	46
CHUNK CHUNK IN CD	34	34
CHUNK VBZ CD IN DT CHUNK IN DT CHUNK	34	34
CHUNK VBZ CHUNK IN DT CHUNK	31	31
CHUNK CHUNK IN DT CHUNK IN CHUNK	23	23
CHUNK CHUNK IN DT CHUNK IN DT CHUNK	23	23
CHUNK VBP IN DT CHUNK	23	23
CHUNK CC CHUNK VBZ IN DT CHUNK IN CHUNK	22	22
CHUNK VBZ IN DT CHUNK IN DT CHUNK IN DT CHUNK IN CHUNK	22	22

Figure A.3.34: A snapshot for the results of extracting an OWL triple after eliminating text (iteration 13).

A.4 EVALUATION

In this section we present snapshots for the process of the manual evaluation performed to create the gold standard annotated dataset, the evaluation of the semantic loss between the SE sentences and the rewritten ACE sentences, the evaluation of reverb, and the evaluation of the accuracy of the extracted triples from the ACE sentences after rewriting. We guide the reader to this shared drive[1] to view the full results.

1 https://drive.google.com/drive/folders/1m2A2VKKGx2n_ZrEGgPluPNtZXe9rXPiM?usp=sharing

main_sent	ace_sent	Semantic Loss Score	Reason
The Archamoebae are an important group of amoebae.	all n:arch-Amoebae are an n:important-group of the n:Amoebae.	4	over-specified Pre-determiners
Nurse Jackie is an American comedy-drama television series.	Nurse-Jackie is an n:American-comedydrama-television-series.	4	incorrect chunking
They were against the Pittsburgh Pirates.	In-1903-the-Boston-Red-Sox is against the n:Pittsburgh-Pirates.	4	Verb Tense
Michael Valentine Doonican was an Irish singer and television presenter.	Michael-Valentine-Doonican is an n:Irish-singer and the n:Television-Presenter.	4	over-specified DT
Danny DeVito directed the movie.	Danny-DeVito directs the movie.	4	Verb Tense
Olpe is a Kreis in the south-east of North Rhine-Westphalia, Germany.	Olpe is a kreis in the n:southeast of the n:North-RhineWestphalia-Germany.	4	incorrect chunking
Marc Chagall was a Jewish Russian-French artist.	Marc-Chagall is a n:Jewish-RussianFrench-artist.	4	incorrect chunking
Sophie Gengembre Anderson was a French-born British artist.	Sophie-Gengembre-Anderson is a n:Frenchborn-British-artist.	4	incorrect chunking
Liviu Librescu was a Romanian-born Israeli and American scientist and engineer.	Liviu-Librescu is a n:Romanianborn-Israeli and the n:American-scientist and Engineer.	4	incorrect chunking
Joseph Wright , , was an English landscape and portrait painter.	Joseph-Wright is an n:English-landscape and the n:portrait-painter.	4	inappropriate DT
Ronald Carlile Buxton was a British politician and civil engineer.	Ronald-Carlile-Buxton is a n:british-Politician and the n:civil-engineer.	4	inappropriate DT
The 1996 Summer Paralympics took place in Atlanta, Georgia, United States.	the n:nineteen-hundred-and-ninety-six-Summer-Paralympics takes the Place in the n:Atlanta-Georgia-United-States.	3	Verb Tense over-specified DT
Ann Sothern was an American movie, stage and television series actress.	Ann-Sothern is an n:American-movie-stage and the n:television-series-actress.	4	inappropriate DT
His real name was Leonard Alfred Schneider.	Lenny-Bruce-real-name is the n:Leonard-Alfred-Schneider.	3	incorrect chunking over-specified DT
Peter's Friends is a 1992 comedy-drama movie.	Peters-Friends is a n:nineteen-hundred-and-ninety-two-comedydrama-movie.	4	incorrect chunking
Eastwood is a census-designated place in Bossier Parish, Louisiana, United States.	Eastwood is a n:censusdesignated-place in the n:Bossier-Parish-Louisiana-United-States.	3	incorrect chunking over-specified DT"
The Invisible Woman is a 2013 romance-drama movie.	the n:Invisible-Woman is a n:two-thousand-and-thirteen-romancedrama-movie.	4	incorrect chunking
The Chiffons was an all-woman singing group.	The Chiffons is an n:allwoman-singing-group.	4	incorrect chunking
Reynaldo Rey was an American actor, comedian, and television personality.	Reynaldo-Rey is an n:American-actor-comedian and the n:television-personality.	4	over-specified DT
His career began in 1935.	Guido-Gorgatti-career begins in 1935.	4	Verb Tense
The Driver's Champion was Michael Schumacher in a close battle.	the n:Drivers-Champion is the n:Michael-Schumacher in a n:close-battle.	4	over-specified DT
Hainburg is a village in the Offenbach Rural District in the state of Hesse, Germany.	Hainburg is a village in the n:Offenbach-Rural-District in the state of the n:Hesse-Germany.	4	over-specified DT
A contronym is a word with two meanings.	a n:contronym is a n:word with the n:two-meanings.	4	over-specified DT
Molineux Stadium is a football stadium in Wolverhampton, England.	Molineux-Stadium is a n:football-stadium in the n:Wolverhampton-England.	4	over-specified DT

Figure A.4.35: A snapshot showing the semantic loss evaluation results.

	triple rel	Non-triple rel	S1	P1	O1	Non-rel1	S2	P2	O2
The name in Turkic means pure water falls .	1	0	name in turkic	mean	pure water falls				
It is about sports related issues around Atlanta .	1	1	it	be about	sports related issues	Around Atlanta			
He wrote the book Goodfellas .	1	0	he	write	the book Goodfellas				
She wrote a short story A Jury of Her Peers .	1	0	she	write	short story A Jury of Her Peers .				
ReachOut Healthcare America is an American company that manages dental clinics .	2	0	ReachOut Healthcare America	be	an american company		an american company	manages	dental clinics
The name Pirituba means a place with many marsh plants .	1	1	Pirituba	mean	a place	with many marsh plants			
He is a right arm spin bowler .	1	0	he	be	right arm spin bowler				
It has images articles templates downloads and other attractions .	1	0	it	have	images articles templates downloads and other attractions .				
It is about old friends gathering at a New Years party .	1	1	it	be about	old friends gathering	at a New Years party .			
It is about the son of a pearl dealer living in the Lower California peninsula .	2	0	it	be about	the son of a pearl dealer		the son of a pearl dealer	living	Lower California peninsula .

Figure A.4.36: A snapshot showing the evaluation of the gold standard annotated dataset.

				triple rel	Non-triple rel	Full right triple	wrong triple or right but partial entiy	missed full right triple
The name in Turkic means pure water falls .	turkic	mean	pure water	1	0	0	1	0
It is about sports related issues around Atlanta .	it	be about	sports	1	1	0	1	0
He wrote the book Goodfellas .	he	write	the book	1	0	0	1	0
She wrote a short story A Jury of Her Peers .	she	write	a short story	1	0		1	0
ReachOut Healthcare America is an American company that manages dental clinics .	healthcare america	be	an american company stdin	2	0	1	1	0
The name Pirituba means a place with many marsh plants .	pirituba	mean	a place	1	1	1	0	0
He is a right arm spin bowler .	he	be	a right arm stdin	1	0	0	1	0
It has images articles templates downloads and other attractions .	it	have	images articles stdin	1	0	0	1	0
It is about old friends gathering at a New Years party .	it	be about	old friends	1	1	0	1	0
It is about the son of a pearl dealer living in the Lower California peninsula .	it	be about	the son of a pearl dealer	2	0	1	0	1

Figure A.4.37: A snapshot showing the evaluation of reverb on the Simple Wikipedia sentences.

Full right triple	wrong triple or right but partial entity	missed full right triple
TP	FP	FN
428	95	15
PRECISION= 428/(428+95)=0.81 RECALL= 428/(428+15)=0.96		

Figure A.4.38: A snapshot showing the statistical analysis of the evaluation results of reverb on the Simple Wikipedia sentences.

				Full right triple	wrong triple or right but partial entiy	missed full right triple
ACE_sent	drs_sent	triple rel	Non-triple rel	TP	FP	FN
The name in Turkic means the n:pure-water-falls.	<?xml version="1.0" encoding="UTF-8"?> <DRS domain="A B C"> <object ref="C", noun="name", struct="countable", unit="na", numrel="eq" num="1", sentid="1", tokid="2"/> <object ref="B", noun="pure-water-falls", struct="countable", unit="na", numrel="eq" num="1", sentid="1", tokid="9"/> <modifier_pp ref="A", prep="in", obj="named('Turkic')", sentid="1", tokid="3"/> <predicate ref="A" verb="mean" subj="C" obj="B" sentid="1" tokid="5"/> </DRS>	1	0	0	1	0

Figure A.4.39: A snapshot showing the evaluation of an example entry after the automatic rewriting of Simple English to ACE.

Full right triple	wrong triple or right but partial entity	missed full right triple
TP	FP	FN
446	67	25
PRECISION= 446/(446+67)=0.86 RECALL= 446/(446+25)=0.94		

Figure A.4.40: A snapshot showing the statistical estimation of Precision and Recall metrics after the automatic rewriting of Simple English to ACE.

BIBLIOGRAPHY

[ACP18] Shaun Azzopardi, Christian Colombo, and Gordon J. Pace. "A Controlled Natural Language for Financial Services Compliance Checking". In: *Controlled Natural Language - Proceedings of the Sixth International Workshop, CNL 2018, Maynooth, Co. Kildare, Ireland, August 27-28, 2018*. 2018, pp. 11–20.

[ADB17] Isabelle Augenstein, Leon Derczynski, and Kalina Bontcheva. "Generalisation in named entity recognition: A quantitative analysis". In: *Computer Speech & Language* 44 (2017), pp. 61–83.

[AFV05] Grigoris Antoniou, Enrico Franconi, and Frank Van Harmelen. "Introduction to semantic web ontology languages". In: *Reasoning web*. Springer, 2005, pp. 1–21.

[AO98] Douglas E Appelt and Boyan Onyshkevych. "The common pattern specification language". In: *Proceedings of a workshop on held at Baltimore, Maryland: October 13-15, 1998*. Association for Computational Linguistics. 1998, pp. 23–30.

[APR12] Isabelle Augenstein, Sebastian Padó, and Sebastian Rudolph. "Lodifier: Generating linked data from unstructured text". In: *Extended Semantic Web Conference*. Springer. 2012, pp. 210–224.

[AR09] Krasimir Angelov and Aarne Ranta. "Implementing controlled languages in GF". In: *International Workshop on Controlled Natural Language*. Springer. 2009, pp. 82–101.

[AS92] Geert Adriaens and Dirk Schreors. "From COGRAM to ALCOGRAM: Toward a controlled English grammar checker". In: *Proceedings of the 14th conference on Computational linguistics-Volume 2*. Association for Computational Linguistics. 1992, pp. 595–601.

[ASD07] ASD ASD-STE100. "Simplified Technical English". In: *ASD standard* (2007).

[AVM15] Isabelle Augenstein, Andreas Vlachos, and Diana Maynard. "Extracting relations between non-standard entities using distant supervision and imitation learning". In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. 2015, pp. 747–757.

[Bar03] Chitta Baral. *Knowledge representation, reasoning and declarative problem solving*. Cambridge university press, 2003.

[BB] David Beckett and Tim Berners-Lee. *Turtle–Terse RDF Triple Language. W3C Team Submission, January 2008*.

[BBE+12] Valerio Basile, Johan Bos, Kilian Evang, and Noortje Venhuizen. "Developing a large semantically annotated corpus". In: *LREC 2012, Eighth International Conference on Language Resources and Evaluation*. 2012.

[BCM05] Paul Buitelaar, Philipp Cimiano, and Bernardo Magnini. "Ontology learning from text: An overview". In: *Ontology learning from text: Methods, evaluation and applications* 123 (2005), pp. 3–12.

[BCS+07] Michele Banko, Michael J Cafarella, Stephen Soderland, Matthew Broadhead, and Oren Etzioni. "Open information extraction from the web." In: *IJCAI*. Vol. 7. 2007, pp. 2670–2676.

[BDM+02] Kalina Bontcheva, Marin Dimitrov, Diana Maynard, Valentin Tablan, and Hamish Cunningham. "Shallow methods for named entity coreference resolution". In: *Chaınes de références et résolveurs d'anaphores, workshop TALN*. 2002.

[BDT07] Chitta Baral, Juraj Dzifcak, and Luis Tari. "Towards overcoming the knowledge acquisition bottleneck in answer set prolog applications: Embracing natural language inputs". In: *International Conference on Logic Programming*. Springer. 2007, pp. 1–21.

[BEP+08] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. "Freebase: a collaboratively created graph database for structuring human knowledge". In: *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*. AcM. 2008, pp. 1247–1250.

[Ber+98] Tim Berners-Lee et al. *Semantic web road map*. 1998.

[Ber97] Arendse Bernth. "EasyEnglish: a tool for improving document quality". In: *Proceedings of the fifth conference on Applied natural language processing*. Association for Computational Linguistics. 1997, pp. 159–165.

[BFL98] Collin F Baker, Charles J Fillmore, and John B Lowe. "The berkeley framenet project". In: *Proceedings of the 17th international conference on Computational linguistics-Volume 1*. Association for Computational Linguistics. 1998, pp. 86–90.

[BGM+11] George W Bassel, Enrico Glaab, Julietta Marquez, Michael J Holdsworth, and Jaume Bacardit. "Functional network construction in Arabidopsis using rule-based machine learning on large-scale data sets". In: *The Plant Cell* 23.9 (2011), pp. 3101–3116.

[BHL01] Tim Berners-Lee, James Hendler, and Ora Lassila. "The semantic web". In: *Scientific american* 284.5 (2001), pp. 34–43.

[BJR16] Nikita Bhutani, HV Jagadish, and Dragomir Radev. "Nested propositions in open information extraction". In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. 2016, pp. 55–64.

[BK06] Abraham Bernstein and Esther Kaufmann. "GINO–a guided input natural language ontology editor". In: *International Semantic Web Conference*. Springer. 2006, pp. 144–157.

[BKF+04] Abraham Bernstein, Esther Kaufmann, Norbert E Fuchs, and June Von Bonin. "Talking to the semantic web-a controlled english query interface for ontologies". In: *14th Workshop on Information Technology and Systems*. 2004, pp. 212–217.

[BKK05] Abraham Bernstein, Esther Kaufmann, and Christian Kaiser. "Querying the semantic web with ginseng: A guided input natural language search engine". In: *15th Workshop on Information Technologies and Systems, Las Vegas, NV*. Citeseer. 2005, pp. 112–126.

[BL04] Steven Bird and Edward Loper. "NLTK: the natural language toolkit". In: *Proceedings of the ACL 2004 on Interactive poster and demonstration sessions*. Association for Computational Linguistics. 2004, p. 31.

[BLK+09] Christian Bizer, Jens Lehmann, Georgi Kobilarov, Sören Auer, Christian Becker, Richard Cyganiak, and Sebastian Hellmann. "DBpedia-A crystallization point for the Web of Data". In: *Web Semantics: science, services and agents on the world wide web* 7.3 (2009), pp. 154–165.

[BM10] Dan Brickley and Libby Miller. *FOAF vocabulary specification 0.91*. 2010.

[Bos08] Johan Bos. "Wide-coverage semantic analysis with boxer". In: *Proceedings of the 2008 Conference on Semantics in Text Processing*. Association for Computational Linguistics. 2008, pp. 277–286.

[Bou92] Didier Bourigault. "Surface grammatical analysis for the extraction of terminological noun phrases". In: *Proceedings of the 14th conference on Computational linguistics-Volume 3*. Association for Computational Linguistics. 1992, pp. 977–981.

[Bri04] Dan Brickley. "RDF vocabulary description language 1.0: RDF schema". In: *http://www. w3. org/TR/rdf-schema/* (2004).

[Bri92] Eric Brill. "A simple rule-based part of speech tagger". In: *Proceedings of the third conference on Applied natural language processing*. Association for Computational Linguistics. 1992, pp. 152–155.

[Bro+96] John Brooke et al. "SUS-A quick and dirty usability scale". In: *Usability evaluation in industry* 189.194 (1996), pp. 4–7.

[CCB07] James R Curran, Stephen Clark, and Johan Bos. "Linguistically motivated large-scale NLP with C&C and Boxer". In: *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*. Association for Computational Linguistics. 2007, pp. 33–36.

[CCP16] Aaron Calafato, Christian Colombo, and Gordon J Pace. "A controlled natural language for tax fraud detection". In: *International Workshop on Controlled Natural Language*. Springer. 2016, pp. 1–12.

[CFK+13] Laura Canedo, Norbert E Fuchs, Kaarel Kaljurand, Maarit Koponen, Tobias Kuhn, Jussi Rautio, and Victor Ungureanu. "Evaluations of ACE-in-GF and of AceWiki-GF". In: (2013).

[Che96] Steven Gregory Chervak. "The effects of simplified english on performance of a maintenance procedure". PhD thesis. State University of New York at Buffalo, 1996.

[Cho10] Gobinda G Chowdhury. *Introduction to modern information retrieval*. Facet publishing, 2010.

[CHR06] Philipp Cimiano, Matthias Hartung, and Esther Ratsch. "Finding the appropriate generalization level for binary relations extracted from the Genia corpus". In: *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*. 2006.

[CK04] Jeremy J Carroll and Graham Klyne. "Resource Description Framework ({RDF}): Concepts and Abstract Syntax". In: (2004).

[CLR13] Laura Chiticariu, Yunyao Li, and Frederick R Reiss. "Rule-based information extraction is dead! long live rule-based information extraction systems!" In: *Proceedings of the 2013 conference on empirical methods in natural language processing*. 2013, pp. 827–832.

[CM14] Danqi Chen and Christopher Manning. "A fast and accurate dependency parser using neural networks". In: *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 2014, pp. 740–750.

[CMB+02] Hamish Cunningham, Diana Maynard, Kalina Bontcheva, and Valentin Tablan. "GATE: an architecture for development of robust HLT applications". In: *Proceedings of the 40th annual meeting on association for computational linguistics*. Association for Computational Linguistics. 2002, pp. 168–175.

[CMB11] Hamish Cunningham, Diana Maynard, and Kalina Bontcheva. *Text processing with gate*. Gateway Press CA, 2011.

[CMW+18] Paul CHOMICZ, Armin MULLER-LERWE, Götz-Philipp WEGNER, Rainer BUSCH, and Stefan KOWALEWSKI. "Controlled Natural Languages for Hazard Analysis and Risk Assessment". In: (2018).

[CP15] Jorge Cardoso and Alexandre Miguel Pinto. "The Web Ontology Language (OWL) and its Applications". In: *Encyclopedia of Information Science and Technology, Third Edition*. IGI Global, 2015, pp. 7662–7673.

[CRA16] Francesco Corcoglioniti, Marco Rospocher, and Alessio Palmero Aprosio. "A 2-phase frame-based knowledge extraction framework". In: *Proceedings of the 31st Annual ACM Symposium on Applied Computing*. ACM. 2016, pp. 354–361.

[CSM+07] Anne Cregan, Rolf Schwitter, Thomas Meyer, et al. "Sydney OWL Syntax-towards a Controlled Natural Language Syntax for OWL 1.1." In: *OWLED*. Vol. 258. 2007.

[Cun05] Hamish Cunningham. "Information extraction, automatic". In: *Encyclopedia of language and linguistics*, (2005), pp. 665–677.

[CV05] Philipp Cimiano and Johanna Völker. "text2onto". In: *International conference on application of natural language to information systems*. Springer. 2005, pp. 227–238.

[Dan08] Dana Dannells. "Generating tailored texts for museum exhibits". In: *The 2nd Workshop on Language Technology for Cultural Heritage (LaTeCH 2008)*. 2008, pp. 17–20.

[Dav13] Brian Patrick Davis. "On applying controlled natural languages for ontology authoring and semantic annotation". PhD thesis. 2013.

[DDA14] Jeffrey Dalton, Laura Dietz, and James Allan. "Entity query feature expansion using knowledge base links". In: *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*. ACM. 2014, pp. 365–374.

[DDC13] Ronald Denaux, Vania Dimitrova, and Anthony G Cohn. "Interacting with ontologies and linked data through controlled natural languages and dialogues". In: *Do-Form: Enabling Domain Experts to Use Formalised Reasoning-AISB Convention 2013*. Society for the study of artificial intelligence. 2013, pp. 18–20.

[DDE+12] Dana Dannells, Mariana Damova, Ramona Enache, and Milen Chechev. "Multilingual online generation from semantic web ontologies". In: *Proceedings of the 21st International Conference on World Wide Web*. ACM. 2012, pp. 239–242.

[DDH+08] Vania Dimitrova, Ronald Denaux, Glen Hart, Catherine Dolbear, Ian Holt, and Anthony G Cohn. "Involving domain experts in authoring OWL ontologies". In: *International Semantic Web Conference*. Springer. 2008, pp. 1–16.

[DG06] Jesse Davis and Mark Goadrich. "The relationship between Precision-Recall and ROC curves". In: *Proceedings of the 23rd international conference on Machine learning*. ACM. 2006, pp. 233–240.

[DGE16] Ezgi Demirel, Kamil Doruk Gur, and Esra Erdem. "Human-robot interaction in a shopping mall: a CNL approach". In: *International Workshop on Controlled Natural Language*. Springer. 2016, pp. 111–122.

[DGP+13] Francesco Draicchio, Aldo Gangemi, Valentina Presutti, and Andrea Giovanni Nuzzolese. "Fred: From natural language text to rdf and owl in one click". In: *Extended Semantic Web Conference*. Springer. 2013, pp. 263–267.

[DH04] Chris Drummond and Robert C Holte. "What ROC Curves Can't Do (and Cost Curves Can)." In: *ROCAI*. Citeseer. 2004, pp. 19–26.

[DHK+07] Cathy Dolbear, Glen Hart, Katalin Kovacs, John Goodwin, and Sheng Zhou. "The Rabbit language: description, syntax and conversion to OWL". In: *Ordnance Survey Research Labs Technical Report* (2007).

[DIF+08] Brian Davis, Ahmad Ali Iqbal, Adam Funk, Valentin Tablan, Kalina Bontcheva, Hamish Cunningham, and Siegfried Handschuh. "Roundtrip ontology authoring". In: *International Semantic Web Conference*. Springer. 2008, pp. 50–65.

[DKP+09] Nilesh Dalvi, Ravi Kumar, Bo Pang, Raghu Ramakrishnan, Andrew Tomkins, Philip Bohannon, Sathiya Keerthi, and Srujana Merugu. "A web of concepts". In: *Proceedings of the twenty-eighth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*. ACM. 2009, pp. 1–12.

[DTD+12] Ronald Denaux, Dhavalkumar Thakker, Vania Dimitrova, and Anthony G Cohn. "Interactive Semantic Feedback for Intuitive Ontology Authoring." In: *FOIS*. 2012, pp. 160–173.

[ECD+04] Oren Etzioni, Michael Cafarella, Doug Downey, Stanley Kok, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S Weld, and Alexander Yates. "Web-scale information extraction in knowitall:(preliminary results)". In: *Proceedings of the 13th*

international conference on World Wide Web. ACM. 2004, pp. 100–110.

[Eck98] Douglas Eckert. "The use of Simplified English to improve task comprehension for non-native English speaking aviation maintenance technician students." In: (1998).

[EES+11] Cristina Espana Bonet, Ramona Enache, Adam Slaski, Aarne Ranta, Lluis Marquez Villodre, and Meritxell Gonzalez Bermudez. "Patent translation within the MOLTO project". In: *4th Workshop on Patent Translation, MT Summit XIII*. 2011, pp. 70–78.

[EHD09] Paula Engelbrecht, Glen Hart, and Catherine Dolbear. "Talking rabbit: a user evaluation of sentence production". In: *International Workshop on Controlled Natural Language*. Springer. 2009, pp. 56–64.

[EY09] Esra Erdem and Reyyan Yeniterzi. "Transforming controlled natural language biomedical queries into answer set programs". In: *Proceedings of the Workshop on Current Trends in Biomedical Natural Language Processing*. Association for Computational Linguistics. 2009, pp. 117–124.

[FA99] Katerina T Frantzi and Sophia Ananiadou. "The C-value/NC-value domain-independent method for multi-word term extraction". In: *Journal of Natural Language Processing* 6.3 (1999), pp. 145–179.

[FBC+10] David Ferrucci, Eric Brown, Jennifer Chu-Carroll, James Fan, David Gondek, Aditya A Kalyanpur, Adam Lally, J William Murdock, Eric Nyberg, John Prager, et al. "Building Watson: An overview of the DeepQA project". In: *AI magazine* 31.3 (2010), pp. 59–79.

[Fer14] Sebastien Ferre. "SQUALL: The expressiveness of SPARQL 1.1 made available as a controlled natural language". In: *Data & Knowledge Engineering* 94 (2014), pp. 163–188.

[FGM05] Jenny Rose Finkel, Trond Grenager, and Christopher Manning. "Incorporating non-local information into information extraction systems by gibbs sampling". In: *Proceedings of the 43rd annual meeting on association for computational linguistics*. Association for Computational Linguistics. 2005, pp. 363–370.

[FHS+16] Zhangjie Fu, Fengxiao Huang, Xingming Sun, Athanasios Vasilakos, and Ching-Nung Yang. "Enabling semantic search based on conceptual graphs over encrypted outsourced data". In: *IEEE Transactions on Services Computing* (2016).

[FKK08a] Norbert E Fuchs, Kaarel Kaljurand, and Tobias Kuhn. "Attempto controlled english for knowledge representation". In: *Reasoning Web*. Springer, 2008, pp. 104–124.

[FKK08b] Norbert E Fuchs, Kaarel Kaljurand, and Tobias Kuhn. *Discourse representation structures for ACE 6.0*. Department of Informatics, 2008.

[Fle44] Rudolf Flesch. "How basic is basic English". In: *Harper's Magazine* 188.1126 (1944), pp. 339–343.

[Fra79] W Nelson Francis. "Brown corpus manual". In: *http://icame. uib. no/brown/bcm. html* (1979).

[FS03] Norbert E Fuchs and Uta Schwertel. "Reasoning in attempto controlled english". In: *International Workshop on Principles and Practice of Semantic Web Reasoning*. Springer. 2003, pp. 174–188.

[FS07] Norbert E Fuchs and Rolf Schwitter. "Web-annotations for humans and machines". In: *European Semantic Web Conference*. Springer. 2007, pp. 458–472.

[FS96] Norbert E Fuchs and Rolf Schwitter. "Attempto controlled english (ace)". In: *arXiv preprint cmp-lg/9603003* (1996).

[FSE11] Anthony Fader, Stephen Soderland, and Oren Etzioni. "Identifying relations for open information extraction". In: *Proceedings of the conference on empirical methods in natural language processing*. Association for Computational Linguistics. 2011, pp. 1535–1545.

[FTB+07] Adam Funk, Valentin Tablan, Kalina Bontcheva, Hamish Cunningham, Brian Davis, and Siegfried Handschuh. "Clone: Controlled language for ontology editing". In: *The Semantic Web*. Springer, 2007, pp. 142–155.

[Fuc16] Norbert E Fuchs. "Reasoning in attempto controlled english: non-monotonicity". In: *International Workshop on Controlled Natural Language*. Springer. 2016, pp. 13–24.

[GFC06] Asunción Gómez-Pérez, Mariano Fernández-López, and Oscar Corcho. *Ontological Engineering: with examples from the areas of Knowledge Management, e-Commerce and the Semantic Web*. Springer Science & Business Media, 2006.

[GHM+08] Bernardo Cuenca Grau, Ian Horrocks, Boris Motik, Bijan Parsia, Peter Patel-Schneider, and Ulrike Sattler. "OWL 2: The next step for OWL". In: *Web Semantics: Science, Services and Agents on the World Wide Web* 6.4 (2008), pp. 309–322.

[GHR11] Jacob L Graham, David L Hall, and Jeffrey Rimland. "A synthetic dataset for evaluating soft and hard fusion algorithms". In: *Defense Transformation and Net-Centric Systems 2011*. Vol. 8062. International Society for Optics and Photonics. 2011, 80620F.

[GHV+03] Benjamin N Grosof, Ian Horrocks, Raphael Volz, and Stefan Decker. "Description logic programs: combining logic programs with description logic". In: *Proceedings of the 12th international conference on World Wide Web*. ACM. 2003, pp. 48–57.

[GK18] Nikhil Gilbert and C Maria Keet. "Automating question generation and marking of language learning exercises for isiZulu". In: *Controlled Natural Language: Proceedings of the Sixth International Workshop, CNL 2018, Maynooth, Co. Kildare, Ireland, August 27-28, 2018*. Vol. 304. IOS Press. 2018, p. 31.

[GP09] Aldo Gangemi and Valentina Presutti. "Ontology design patterns". In: *Handbook on ontologies*. Springer, 2009, pp. 221–243.

[Gru93] Thomas R Gruber. "A translation approach to portable ontology specifications". In: *Knowledge acquisition* 5.2 (1993), pp. 199–220.

[GS17] Stephen C Guy and Rolf Schwitter. "The PENG ASP system: architecture, language and authoring tool". In: *Language Resources and Evaluation* 51.1 (2017), pp. 67–92.

[HDG+06] Matthew Horridge, Nick Drummond, John Goodwin, Alan L Rector, Robert Stevens, and Hai Wang. "The Manchester OWL syntax." In: *OWLed*. Vol. 216. 2006.

[Hea92] Marti A Hearst. "Automatic acquisition of hyponyms from large text corpora". In: *Proceedings of the 14th conference on Computational linguistics-Volume 2*. Association for Computational Linguistics. 1992, pp. 539–545.

[Hin88] Don E Hinson. "Simplified English—Is it really simple?" In: *Proceedings of the 38th International Technical Communication Conference*. 1988.

[HJD08] Glen Hart, Martina Johnson, and Catherine Dolbear. "Rabbit: Developing a control natural language for authoring ontologies". In: *European Semantic Web Conference*. Springer. 2008, pp. 348–360.

[HME08] Feikje Hielkema, Chris Mellish, and Peter Edwards. "Evaluating an ontology-driven wysiwym interface". In: *Proceedings of the Fifth International Natural Language Generation Conference*. Association for Computational Linguistics. 2008, pp. 138–146.

[HS13] S Harris and A Seaborne. *SPARQL 1.1 Overview. W3C Recommendation*. 2013.

[Hsi18] Michael S Hsiao. "Automated program synthesis from object-oriented natural language for computer games". In: *Controlled Natural Language-Proceedings of the Sixth International Workshop, CNL 2018, Maynooth, Co. Kildare, Ireland, August 27-28*. 2018, pp. 71–74.

[HSP07] Catalina Hallett, Donia Scott, and Richard Power. "Composing questions through conceptual authoring". In: *Computational linguistics* 33.1 (2007), pp. 105–133.

[Huj98] W-O Hujisen. "Controlled language: an introduction". In: *Proc. 2nd Int. Workshop on Controlled Language Applications (CLAW), 1998*. 1998, pp. 1–15.

[HYB+11] Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Furstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard Weikum. "Robust disambiguation of named entities in text". In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics. 2011, pp. 782–792.

[IK02] Hideki Isozaki and Hideto Kazawa. "Efficient support vector classifiers for named entity recognition". In: *Proceedings of the 19th international conference on Computational linguistics-Volume 1*. Association for Computational Linguistics. 2002, pp. 1–7.

[JCC16] Nataly Jahchan, Anne Condamines, and Emmanuelle Cannesson. "To What Extent Does Text Simplification Entail a More Optimized Comprehension in Human-Oriented CNLs?" In: *International Workshop on Controlled Natural Language*. Springer. 2016, pp. 69–80.

[KAC+02] Gregory Karvounarakis, Sofia Alexaki, Vassilis Christophides, Dimitris Plexousakis, and Michel Scholl. "RQL: a declarative query language for RDF". In: *Proceedings of the 11th international conference on World Wide Web*. ACM. 2002, pp. 592–603.

[Kal08] Kaarel Kaljurand. "ACE View—an Ontology and Rule Editor based on Attempto Controlled English." In: *OWLED*. 2008.

[KAM+98] Christine Kamprath, Eric Adolphson, Teruko Mitamura, and Eric Nyberg. "Controlled language for multilingual document production: Experience with Caterpillar technical English". In: *Proceedings of the Second International Workshop on Controlled Language Applications*. Vol. 146. 1998.

[Kam81] Hans Kamp. "A theory of truth and semantic representation". In: *Formal semantics-the essential readings* (1981), pp. 189–222.

[Kas66] Tadao Kasami. "An efficient recognition and syntax-analysis algorithm for context-free languages". In: *Coordinated Science Laboratory Report no. R-257* (1966).

[KBM08] Vipul Kashyap, Christoph Bussler, and Matthew Moran. "Ontology Authoring and Management". In: *The Semantic Web: Semantics for Data and Services on the Web* (2008), pp. 137–159.

[KBN+13] Tobias Kuhn, Paolo Emilio Barbano, Mate Levente Nagy, and Michael Krauthammer. "Broadening the scope of nanopublications". In: *Extended Semantic Web Conference*. Springer. 2013, pp. 487–501.

[KC06] Graham Klyne and Jeremy J Carroll. "Resource description framework (RDF): Concepts and abstract syntax". In: (2006).

[KF06a] Kaarel Kaljurand and Norbert E Fuchs. "Bidirectional mapping between OWL DL and attempto controlled english". In: *International Workshop on Principles and Practice of Semantic Web Reasoning*. Springer. 2006, pp. 179–189.

[KF06b] Kaarel Kaljurand and Norbert E Fuchs. "Mapping Attempto Controlled English to OWL DL". In: *3rd European Semantic Web Conference. Demo and Poster Session, Budva, Montenegro*. 2006.

[KF07] Kaarel Kaljurand and Norbert E Fuchs. "Verbalizing OWL in Attempto Controlled English." In: *OWLED*. Vol. 258. 2007.

[KFN+04] Holger Knublauch, Ray W Fergerson, Natalya F Noy, and Mark A Musen. "The Protege OWL plugin: An open development environment for semantic web applications". In: *International Semantic Web Conference*. Springer. 2004, pp. 229–243.

[KHL+05] Hong-Gee Kim, Byung-Hyun Ha, Jae-Il Lee, and Myeng-Ki Kim. "A multi-layered application for the gross description using semantic web technology". In: *International journal of medical informatics* 74.5 (2005), pp. 399–407.

[KK13] Kaarel Kaljurand and Tobias Kuhn. "A multilingual semantic wiki based on Attempto Controlled English and Grammatical Framework". In: *Extended Semantic Web Conference*. Springer. 2013, pp. 427–441.

[KM03] Dan Klein and Christopher D Manning. "Accurate unlexicalized parsing". In: *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*. Association for Computational Linguistics. 2003, pp. 423–430.

[Knu65] Donald E Knuth. "On the translation of languages from left to right". In: *Information and control* 8.6 (1965), pp. 607–639.

[KR13] Hans Kamp and Uwe Reyle. *From discourse to logic: Introduction to modeltheoretic semantics of natural language, formal logic and discourse representation theory*. Vol. 42. Springer Science & Business Media, 2013.

[KRF+06] Tobias Kuhn, Loic Royer, Norbert E Fuchs, and Michael Schroeder. "Improving text mining with controlled natural language: A case study for protein interactions". In: *International Workshop on Data Integration in the Life Sciences*. Springer. 2006, pp. 66–81.

[KS06] Tibor Kiss and Jan Strunk. "Unsupervised multilingual sentence boundary detection". In: *Computational Linguistics* 32.4 (2006), pp. 485–525.

[Kuh06] Tobias Kuhn. "Attempto Controlled English as ontology language". In: *REWERSE Annual Meeting*. 2006.

[Kuh08] Tobias Kuhn. "Acewiki: A natural and expressive semantic wiki". In: *arXiv preprint arXiv:0807.4618* (2008).

[Kuh10] T Kuhn. "Controlled English for Knowledge Representation (Doctoral dissertation, PhD thesis, Faculty of Economics, Business Administration and Information Technology of the University of Zurich)". In: (2010).

[Kuh13a] Tobias Kuhn. "A principled approach to grammars for controlled natural languages and predictive editors". In: *Journal of Logic, Language and Information* 22.1 (2013), pp. 33–70.

[Kuh13b] Tobias Kuhn. "The understandability of OWL statements in controlled English". In: *Semantic Web* 4.1 (2013), pp. 101–115.

[Kuh14] Tobias Kuhn. "A survey and classification of controlled natural languages". In: *Computational Linguistics* 40.1 (2014), pp. 121–170.

[Kuh16] Tobias Kuhn. "The controlled natural language of Randall Munroe's Thing Explainer". In: *International Workshop on Controlled Natural Language*. Springer. 2016, pp. 102–110.

[Kuh18] Tobias Kuhn. "Using the AIDA Language to Formally Organize Scientific Claims". In: *arXiv preprint arXiv:1806.01507* (2018).

[LBS+16] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. "Neural architectures for named entity recognition". In: *arXiv preprint arXiv:1603.01360* (2016).

[Lin03] Dekang Lin. "Dependency-based evaluation of MINIPAR". In: *Treebanks*. Springer, 2003, pp. 317–329.

[LL18] Herbert Lange and Peter Ljunglöf. "Putting control into language learning". In: *SIGCNL 2018* (2018).

[LPC+11] Heeyoung Lee, Yves Peirsman, Angel Chang, Nathanael Chambers, Mihai Surdeanu, and Dan Jurafsky. "Stanford's multi-pass sieve coreference resolution system at the CoNLL-2011 shared task". In: *Proceedings of the fifteenth conference on computational natural language learning: Shared task*. Association for Computational Linguistics. 2011, pp. 28–34.

[LUM06] Yuangui Lei, Victoria Uren, and Enrico Motta. "Semsearch: A search engine for the semantic web". In: *International Conference on Knowledge Engineering and Knowledge Management*. Springer. 2006, pp. 238–245.

[MA00] Diana Maynard and Sophia Ananiadou. "Identifying terms by their family and friends". In: *Proceedings of the 18th conference on Computational linguistics-Volume 1*. Association for Computational Linguistics. 2000, pp. 530–536.

[Mar04] Massimo Marchiori. "Towards a people's web: Metalog". In: *Proceedings of the 2004 IEEE/WIC/ACM International Conference on Web Intelligence*. IEEE Computer Society. 2004, pp. 320–326.

[MBA16] Diana Maynard, Kalina Bontcheva, and Isabelle Augenstein. "Natural language processing for the semantic web". In: *Synthesis Lectures on the Semantic Web: Theory and Technology* 6.2 (2016), pp. 1–194.

[MBP+12] David Mott, Dave Braines, Stephen Poteet, Anne Kao, and Ping Xue. "Controlled Natural Language to facilitate information extraction". In: *Proceedings of the Sixth Annual Conference of the International Technology Alliance, London, UK*. 2012.

[MBS+09] Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. "Distant supervision for relation extraction without labeled data". 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: Volume 2-Volume 2*. Association for Computational Linguistics. 2009, pp. 1003–1011.

[McB04] Brian McBride. "The resource description framework (RDF) and its vocabulary description language RDFS". In: *Handbook on ontologies*. Springer, 2004, pp. 51–65.

[MFP09] Diana Maynard, Adam Funk, and Wim Peters. "SPRAT: a tool for automatic semantic pattern-based ontology population". In: *International conference for digital libraries and the semantic web, Trento, Italy*. 2009.

[MG96] Linda Means and Kurt Godden. "The Controlled Automotive Service Language (CASL) Project". In: *CLAW 96: Proceedings of the First International Workshop on Controlled Language Applications*. 1996, pp. 106–114.

[Mik17] Peter Mika. "What Happened To The Semantic Web?" In: *Proceedings of the 28th ACM Conference on Hypertext and Social Media*. ACM. 2017, pp. 3–3.

[MJG+11] Pablo N Mendes, Max Jakob, Andres Garcia-Silva, and Christian Bizer. "DBpedia spotlight: shedding light on the web of documents". In: *Proceedings of the 7th international conference on semantic systems*. ACM. 2011, pp. 1–8.

[ML03] Andrew McCallum and Wei Li. "Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons". In: *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*. Association for Computational Linguistics. 2003, pp. 188–191.

[MMM+04] Frank Manola, Eric Miller, Brian McBride, et al. "RDF primer". In: *W3C recommendation* 10.1-107 (2004), p. 6.

[MMP03] James Mayfield, Paul McNamee, and Christine Piatko. "Named entity recognition using hundreds of thousands of features". In: *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*. Association for Computational Linguistics. 2003, pp. 184–187.

[MMS93] Mitchell P Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. "Building a large annotated corpus of English: The Penn Treebank". In: *Computational linguistics* 19.2 (1993), pp. 313–330.

[MMS99] Christopher D Manning, Christopher D Manning, and Hinrich Schütze. *Foundations of statistical natural language processing*. MIT press, 1999.

[MN95] Teruko Mitamura and Eric Nyberg. "Controlled English for knowledge-based MT: Experience with the KANT system". In: *Proceedings of TMI-95* 145.146-147 (1995), p. 216.

[Mon70] Richard Montague. "Universal grammar". In: *Theoria* 36.3 (1970), pp. 373–398.

[MS04] Alexander Maedche and Steffen Staab. "Ontology learning". In: *Handbook on ontologies*. Springer, 2004, pp. 173–190.

[MSB+14] Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. "The Stanford CoreNLP natural language processing toolkit". In: *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*. 2014, pp. 55–60.

[MV+04] Deborah L McGuinness, Frank Van Harmelen, et al. "OWL web ontology language overview". In: *W3C recommendation* 10.10 (2004), p. 2004.

[MW08] David Milne and Ian H Witten. "Learning to link with wikipedia". In: *Proceedings of the 17th ACM conference on Information and knowledge management*. ACM. 2008, pp. 509–518.

[NGP11] Andrea Giovanni Nuzzolese, Aldo Gangemi, and Valentina Presutti. "Gathering lexical linked data and knowledge patterns from FrameNet". In: *Proceedings of the sixth international conference on Knowledge capture*. ACM. 2011, pp. 41–48.

[Nie06] Jakob Nielsen. "Quantitative studies: How many users to test". In: *Alertbox, June* 26 (2006), p. 2006.

[NM00] Eric Nyberg and Teruko Mitamura. "The KANTOO machine translation environment". In: *Conference of the Association for Machine Translation in the Americas*. Springer. 2000, pp. 192–195.

[NP10] Roberto Navigli and Simone Paolo Ponzetto. "BabelNet: Building a very large multilingual semantic network". In: *Proceedings of the 48th annual meeting of the association for computational linguistics*. Association for Computational Linguistics. 2010, pp. 216–225.

[OBr03] Sharon O'Brien. "Controlling controlled english. an analysis of several controlled language rule sets". In: *Proceedings of EAMT-CLAW* 3.105-114 (2003), p. 33.

[Ogd30] Charles Kay Ogden. "Basic English: A general introduction with rules and grammar". In: (1930).

[Pau17] Heiko Paulheim. "Knowledge graph refinement: A survey of approaches and evaluation methods". In: *Semantic web* 8.3 (2017), pp. 489–508.

[PDG12] Valentina Presutti, Francesco Draicchio, and Aldo Gangemi. "Knowledge extraction based on discourse representation theory and linguistic frames". In: *International conference on knowledge engineering and knowledge management*. Springer. 2012, pp. 114–129.

[PF11] Terence Parr and Kathleen Fisher. "LL (*): the foundation of the ANTLR parser generator". In: *ACM Sigplan Notices*. Vol. 46. 6. ACM. 2011, pp. 425–436.

[PFK+98] Foster J Provost, Tom Fawcett, Ron Kohavi, et al. "The case against accuracy estimation for comparing induction algorithms." In: *ICML*. Vol. 98. 1998, pp. 445–453.

[PGK05] Martha Palmer, Daniel Gildea, and Paul Kingsbury. "The proposition bank: An annotated corpus of semantic roles". In: *Computational linguistics* 31.1 (2005), pp. 71–106.

[PM18] Laurette PRETORIUS and Laurette MARAIS. "Modelling Negation of the Afrikaans Declarative Sentence in GF". In: *Controlled Natural Language: Proceedings of the Sixth International Workshop, CNL 2018, Maynooth, Co. Kildare, Ireland, August 27-28, 2018*. Vol. 304. IOS Press. 2018, p. 92.

[Poo06] Jonathan Pool. "Can controlled languages scale to the Web?" In: *International Workshop on Controlled Language Applications 5*. 2006.

[Pow12] Richard Power. "OWL Simplified English: a finite-state language for ontology editing". In: *International Workshop on Controlled Natural Language*. Springer. 2012, pp. 44–60.

[PS+06] Eric Prud, Andy Seaborne, et al. "SPARQL query language for RDF". In: (2006).

[PSE98] Richard Power, Donia Scott, and Roger Evans. "What You See Is What You Meant: direct knowledge editing with natural language feedback." In: *ECAI*. Vol. 98. 1998, pp. 677–681.

[PTL+15] Roman Prokofyev, Alberto Tonon, Michael Luggen, Loic Vouilloz, Djellel Eddine Difallah, and Philippe Cudré-Mauroux. "SANAPHOR: Ontology-based coreference resolution". In: *International Semantic Web Conference*. Springer. 2015, pp. 458–473.

[Ran04] Aarne Ranta. "Grammatical framework". In: *Journal of Functional Programming* 14.2 (2004), pp. 145–189.

[Ran11] Aarne Ranta. *Grammatical framework: Programming with multilingual grammars*. Vol. 173. CSLI Publications, Center for the Study of Language and Information, 2011.

[Ran14] Aarne Ranta. "Embedded controlled languages". In: *International Workshop on Controlled Natural Language*. Springer. 2014, pp. 1–7.

[REK09] Aarne Ranta, Ali El Dada, and Janna Khegai. "The GF resource grammar library". In: *Linguistic Issues in Language Technology* 2.2 (2009), pp. 1–63.

[RLR+10] Karthik Raghunathan, Heeyoung Lee, Sudarshan Rangarajan, Nathanael Chambers, Mihai Surdeanu, Dan Jurafsky, and Christopher Manning. "A multi-pass sieve for coreference resolution". In: *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics. 2010, pp. 492–501.

[RM99] Lance A Ramshaw and Mitchell P Marcus. "Text chunking using transformation-based learning". In: *Natural language processing using very large corpora*. Springer, 1999, pp. 157–176.

[RMD13] Delip Rao, Paul McNamee, and Mark Dredze. "Entity linking: Finding extracted entities in a knowledge base". In: *Multisource, multilingual information extraction and summarization*. Springer, 2013, pp. 93–115.

[RP16] Petar Ristoski and Heiko Paulheim. "Semantic Web in data mining and knowledge discovery: A comprehensive survey". In: *Web semantics: science, services and agents on the World Wide Web* 36 (2016), pp. 1–22.

[RRK+08] Frederick Reiss, Sriram Raghavan, Rajasekar Krishnamurthy, Huaiyu Zhu, and Shivakumar Vaithyanathan. "An algebraic approach to rule-based information extraction". In: *Data Engineering, 2008. ICDE 2008. IEEE 24th International Conference on*. IEEE. 2008, pp. 933–942.

[RZ95] AL Rector and P Zanstra. "Terminology Services for Clinical Information Systems". In: *Health in the New Communications Age: Health care telematics for the 21st century* (1995), p. 90.

[SB05] Alexander Schutz and Paul Buitelaar. "Relext: A tool for relation extraction from text in ontology extension". In: *International semantic web conference*. Springer. 2005, pp. 593–606.

[SBF+98] Rudi Studer, V Richard Benjamins, Dieter Fensel, et al. "Knowledge engineering: principles and methods". In: *Data and knowledge engineering* 25.1 (1998), pp. 161–198.

[Sch02] Rolf Schwitter. "English as a formal specification language". In: *Database and Expert Systems Applications, 2002. Proceedings. 13th International Workshop on*. IEEE. 2002, pp. 228–232.

[Sch10] Rolf Schwitter. "Controlled natural languages for knowledge representation". In: *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*. Association for Computational Linguistics. 2010, pp. 1113–1121.

[SD14] Hazem Safwat and Brian Davis. "A brief state of the art of CNLs for ontology authoring". In: *International Workshop on Controlled Natural Language*. Springer. 2014, pp. 190–200.

[SD17] Hazem Safwat and Brian Davis. "CNLs for the semantic web: a state of the art". In: *Language Resources and Evaluation* 51.1 (2017), pp. 191–220.

[SDN+07] Warren Shen, AnHai Doan, Jeffrey F Naughton, and Raghu Ramakrishnan. "Declarative information extraction using datalog with embedded extraction predicates". In: *Proceedings of the 33rd international conference on Very large data bases*. VLDB Endowment. 2007, pp. 1033–1044.

[SDZ18] Hazem Safwat, Brian Davis, and Manel Zarrouk. "Engineering an aligned gold-standard corpus of human to machine oriented Controlled Natural Language". In: *2018 IEEE/WIC/ACM International Conference on Web Intelligence (WI)*. IEEE. 2018.

[Sea04] Andy Seaborne. "Rdql-a query language for rdf". In: *http://www.w3. org/Submission/RDQL/* (2004).

[SG15] Cicero Nogueira dos Santos and Victor Guimaraes. "Boosting named entity recognition with neural character embeddings". In: *arXiv preprint arXiv:1505.05008* (2015).

[SGD+16] Hazem Safwat, Normunds Gruzitis, Brian Davis, and Ramona Enache. "Extracting semantic knowledge from unstructured text using embedded controlled language". In: *Semantic Computing (ICSC), 2016 IEEE Tenth International Conference on*. IEEE. 2016, pp. 87–90.

[SGE+15] Hazem Safwat, Normunds Gruzitis, Ramona Enache, and Brian Davis. "Embedded controlled language to facilitate information extraction from eGov policies". In: *Proceedings of the 17th International Conference on Information Integration and Web-based Applications & Services*. ACM. 2015, p. 71.

[SKC+08] Rolf Schwitter, Kaarel Kaljurand, Anne Cregan, Catherine Dolbear, Glen Hart, et al. "A Comparison of three Controlled Natural Languages for OWL 1.1." In: *OWLED (Spring)*. 2008.

[Sku03] Doug Skuce. *A controlled language for knowledge formulation on the semantic Web*. 2003.

[SKW07] Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. "Yago: a core of semantic knowledge". In: *Proceedings of the 16th international conference on World Wide Web*. ACM. 2007, pp. 697–706.

[Sma08] Paul R Smart. "Controlled natural languages and the semantic web". In: (2008).

[Smi04] Michael K Smith. "Owl web ontology language guide". In: *http://www. w3. org/TR/owl-guide/* (2004).

[Smi13] Scott Smith. "Determining sample size: How to ensure you get the correct sample size". In: *E-Book (c) Qualtrics Online Sample* (2013).

[Smu12] Raymond R Smullyan. *First-order logic*. Vol. 43. Springer Science & Business Media, 2012.

[Sod97] Stephen G Soderland. "Learning text analysis rules for domain-specific natural language processing". PhD thesis. Citeseer, 1997.

[Son08] Tran Cao Son. "Using answer set programming and lambda calculus to characterize natural language sentences with normatives and exceptions". In: (2008).

[Sow04] John Sowa. "Common logic controlled english (2004)". In: *Draft available online at: http://www. jfsowa. com/clce/specs. htm* (2004).

[SSH+95] Serena K Shubert, Jan H Spyridakis, Heather K Holmback, and Mary B Coney. "The comprehensibility of simplified English in procedures". In: *Journal of technical writing and communication* 25.4 (1995), pp. 347–369.

[ST04] Rolf Schwitter and Marc Tilbrook. "Controlled natural language meets the semantic web". In: *Proceedings of the Australasian Language Technology Workshop 2004*. 2004, pp. 55–62.

[Swa04] Aaron Swartz. *application/rdf+ xml Media Type Registration*. Tech. rep. 2004.

[SWL+12] Wei Shen, Jianyong Wang, Ping Luo, and Min Wang. "Linden: linking named entities with knowledge base via semantic knowledge". In: *Proceedings of the 21st international conference on World Wide Web*. ACM. 2012, pp. 449–458.

[SZD17] Hazem Safwat, Manel Zarrouk, and Brian Davis. "From simplified text to knowledge representation using Controlled Natural Language". In: *International Journal of Computational Linguistics and Applications* 2 (2017).

[SZD18] Hazem Safwat, Manel Zarrouk, and Brian Davis. "Rewriting Simplified Text into a Controlled Natural Language". In: *Controlled Natural Language - Proceedings of the Sixth International Workshop, CNL 2018, Maynooth, Co. Kildare, Ireland, August 27-28, 2018*. 2018, pp. 85–91.

[Tem11] Irina Temnikova. "Establishing implementation priorities in aiding writers of controlled crisis management texts". In: *Proceedings of the International Conference Recent Advances in Natural Language Processing 2011*. 2011, pp. 654–659.

[TM01] Lappoon R Tang and Raymond J Mooney. "Using multiple clause constructors in inductive logic programming for semantic parsing". In: *European Conference on Machine Learning*. Springer. 2001, pp. 466–477.

[TMS03] Ann Taylor, Mitchell Marcus, and Beatrice Santorini. "The Penn treebank: an overview". In: *Treebanks*. Springer, 2003, pp. 5–22.

[Tob18] KUHN Tobias. "Using the AIDA Language to Formally Organize Scientific Claims". In: *Controlled Natural Language: Proceedings of the Sixth International Workshop, CNL 2018, Maynooth, Co. Kildare, Ireland, August 27-28, 2018*. Vol. 304. IOS Press. 2018, p. 52.

[VCM08] Kashyap Vipul, Bussler Christoph, and Moran Matthew. "The Semantic Web-Semantics for Data and Services on the Web". In: *Berlin and Heidelberg* (2008).

[VGC+11] Rafael Valencia-Garcia, Francisco Garcia-Sanchez, Dagoberto Castellanos-Nieves, et al. "OWLPath: An OWL ontology-guided query editor". In: *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans* 41.1 (2011), pp. 121–136.

[VMK+10] Max Van Kleek, Brennan Moore, David R Karger, Paul André, et al. "Atomate it! end-user context-sensitive automation using heterogeneous information sources on the web". In: *Proceedings of the 19th international conference on World wide web*. ACM. 2010, pp. 951–960.

[WBY+13] Jason Weston, Antoine Bordes, Oksana Yakhnenko, and Nicolas Usunier. "Connecting language and knowledge bases with embedding models for relation extraction". In: *arXiv preprint arXiv:1307.7973* (2013).

[WH96] Richard H Wojcik and Heather Holmback. "Getting a controlled language off the ground at Boeing". In: *Proceedings of the First International Workshop on Controlled Language Applications*. Vol. 22. 1996, p. 31.

[WHH98] Richard H Wojcik, Heather Holmback, and James Hoard. "Boeing Technical English: An extension of AECMA SE beyond the aircraft maintenance domain". In: *Proceedings: Second International Workshop on Controlled Language Applications (CLAW 98), Pittsburgh, PA*. Vol. 215. 1998.

[WI95] Sholom M Weiss and Nitin Indurkhya. "Rule-based machine learning methods for functional prediction". In: *Journal of Artificial Intelligence Research* 3 (1995), pp. 383–403.

[WL14] Kristian Woodsend and Mirella Lapata. "Text rewriting improves semantic role labeling". In: *Journal of Artificial Intelligence Research* 51 (2014), pp. 133–164.

[WLB12] Wilson Wong, Wei Liu, and Mohammed Bennamoun. "Ontology learning from text: A look back and into the future". In: *ACM Computing Surveys (CSUR)* 44.4 (2012), p. 20.

[WMS04] Chris Welty, Deborah L McGuinness, and Michael K Smith. "Owl web ontology language guide". In: *W3C recommendation, W3C (February 2004) http://www. w3. org/TR/2004/REC-owl-guide-20040210* (2004).

[WNL16] Adam Wyner, Adeline Nazarenko, and François Lévy. "Towards a high-level controlled language for legal sources on the semantic web". In: *International Workshop on Controlled Natural Language*. Springer. 2016, pp. 92–101.

[WS02] Jennifer Wells Akis and William R Sisson. "Improving translatability: A case study at Sun Microsystems". In: *Inc. The Globalization Insider* 4 (2002).

[XCN15] Wei Xu, Chris Callison-Burch, and Courtney Napoles. "Problems in current text simplification research: New data can help". In: *Transactions of the Association of Computational Linguistics* 3.1 (2015), pp. 283–297.

[XPK+12] Ping Xue, Stephen Poteet, Anne Kao, David Mott, Dave Braines, Cheryl Giammanco, and Tien Pham. *Information extraction using controlled english to support knowledge-sharing and decision-making*. Tech. rep. BOEING CO SEATTLE WA RESEARCH and TECHNOLOGY, 2012.

[YCB+07] Alexander Yates, Michael Cafarella, Michele Banko, Oren Etzioni, Matthew Broadhead, and Stephen Soderland. "Textrunner: open information extraction on the web". In: *Proceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*. Association for Computational Linguistics. 2007, pp. 25–26.

[ZSL+12] Zhicheng Zheng, Xiance Si, Fangtao Li, Edward Y Chang, and Xiaoyan Zhu. "Entity disambiguation with freebase". In: *Proceedings of the The 2012 IEEE/WIC/ACM International Joint Conferences on Web Intelligence and Intelligent Agent Technology-Volume 01*. IEEE Computer Society. 2012, pp. 82–89.