Generating an RDF dataset from Twitter data: A Study Using Machine Learning

Thesis submitted in accordance with the requirements of
the University of Liverpool for the degree of Doctor in Philosophy by

Saad Alajlan

June 2021

Dedication

To my father, Abdullah Ajlan Alajlan

Thought you never got to see this, you are in every page

Acknowledgements

The three years of my PhD journey, along with the various feelings and emotions that came along with it, are about to end. As I look back on this time, especially the quarantine period which caused me much stress and depression, I recall the many people who have guided, helped, and supported me. I am grateful to all of them.

First, I would like to express my gratitude and appreciation to my supervisor, Professor Frans Coenen. Without his support, guidance, and advice, I would not have achieved this PhD. He has selflessly offered me all his scientific knowledge and guided me since the first day of my journey. I am honoured and proud to have worked under his supervision. I would also like to thank Dr. Angrosh Mandya and my advisors, Dr. Valentina Tamma and Dr. Xiaowei Huang, for their feedback and suggestions.

I want to thank all my colleagues and friends for their support and advice over the years, not only those in the data mining and machine learning group and Room 211, but also all my friends around the world, especially Dr. Abdulaziz Almanea and those with whom I spent much of the lockdown via Zoom.

Special thanks to Sadeem for her advice, help, and support.

My deep gratitude goes to my mother, Munirah. I am thankful for every opportunity and conversation. Most of all, I appreciate all her love. I would like to acknowledge and thank my older brother, Ajlan, who became a father figure to me when my father passed away when I was six years old. I also thank my other siblings, Abdulaziz, Mohammed, Norah, and Sultan, for their spiritual support and for always standing by me. Of course, I cannot forget to say thank you to my little niece, Alhanouf.

I would like to extend my appreciation to my sponsors, Imam Muhammad ibn Saud Islamic University and the Saudi Cultural Bureau in London, for their support as I completed my postgraduate studies.

Finally, I am grateful to those who helped me finish my thesis, especially since it was written during the COVID-19 pandemic, which is undeniably a tough time for everyone. My appreciation goes to King Salman bin Abdulaziz Al-Saud and His Royal Highness Prince Mohammed bin Salman bin Abdulaziz for their quick action in providing all the necessary help, both medical and financial, to all students studying abroad. I am deeply grateful for their efforts in exhausting the necessary resources to evacuate all students during the pandemic.

Abstract

This thesis presents work focused on finding the most effective and efficient mechanism(s) for generating an RDF dataset from Twitter data. The motivation is the desire to extract knowledge from social media data. Knowledge, that can only be extracted if structure, in the form of RDF, is imposed on the data. The main research question that the thesis seeks to address is, ***"What is the most effective and efficient mechanism whereby an RDF database associated with a particular domain of discourse, described in the form of Twitter free text, can be generated?"***. Four different frameworks for RDF dataset generation from Twitter data are proposed: (i) The Stanford CoreNLP RDF dataset framework, (ii) The GATE RDF dataset framework, (iii) The regular expression RDF dataset framework and (iv) The Shortest Path Dependency Parsing and Word Mover's Distance (SPDP-WMD) framework. Additional contributions include, (i) a regular expression pattern syntax, (ii) a regular expression parser, (iii) a fully labelled motor vehicle pollution evaluation data set and (iv) a partially labelled diabetes evaluation data set. The first RDF dataset generation frameworks was a benchmark, the second an alternative to the benchmark. Both featured a requirement for substantial amounts of training data and, in the case of the second framework, entity lists. The third and fourth frameworks were introduced to address the limitations of the first two. A feature of all the frameworks was that they all utilised Named Entity Recognition (NER) and Relation Extraction (RE) models of some kind. All the frameworks also utilised existing lexical databases such as WordNet, and/or existing schema such as those available from Schema.org, for building the hierarchical structures of classes and relations for the RDFS to enrich the RDF dataset. Apache Jena was used to generate both RDF and RDFS files. The frameworks were evaluated using datasets drawn from two domains of discourse: motor vehicle pollution and diabetes. The motor vehicle pollution data set was used to evaluate all the frameworks; then, the most effective framework was evaluated using the larger diabetes data set. The NER and RE models were evaluated using k-fold cross validation where applicable. The RDFS were populated and then further evaluated by using a set of examples for querying, using the SPARQL query language, so as to extract knowledge. In the case of the diabetes dataset, the populated RDFS was also evaluated by clinicians.

Contents

Illustrations

List of Figures

List of Tables

Notations

The following notations and abbreviations are found throughout this thesis:

ANNIE	A Nearly-New Information Extraction system
BiLSTM	Bidirectional Long Short-Term Memory
BIRCH	Balanced Iterative Reducing and Clustering using Hierarchies
BNF	Backus–Naur Form
CAS	Common Analysis System
CBOW	Continuous Bag of Words Model
CCG	Combinatory Categorical Grammar Parser
CFG	Context Free Grammar Parser
CQs	Competency Questions
CRF	Conditional Random Fields
DIPRE	Dual Iterative Pattern Relation Expansion
EMD	Entity Mention Detection
GATE	General Architecture for Text Engineering
HMM	Hidden Markov Models
JAPE	Java Annotation Patterns Engine
KCV	k-fold Cross-Validation
LHS	Left-Hand-Side
LLDA	Labelled Latent Dirichlet Allocation
LSI	Latent Semantic Indexing
NER	Named entity recognition
NLP	Natural Language Processing
NLTK	Natural Language Tool Kit
NNC	Nearest Neighbour Clustering
OWL	Web Ontology language
POS	Part Of Speech
RDFS	Resource Description Framework Schema

RE	Relation Extraction
RHS	Right-Hand-Sid
RMD	Relation Mention Detection
SEUD	Stanford Enhanced Universal Dependency Parsing
SPDP	The Shortest Path Dependency Parsing
SPDP-WMD	The Shortest Path Dependency Parsing and Word Mover's Distance
	RDF dataset From Twitter Data Framework
Stand. Dev.	Standard Deviation
SUD	Stanford Universal Dependency Parsing
SVM	Support Vector Machines
T-CLASS	Segmenting Named Entities
T-SEG	Segmenting Named Entities
TCV	Ten-fold Cross-Validation
TF-IDF	Term Frequency-Inverse Document Frequency
UIMA	IBM's Unstructured Information Management Architecture
WMD	Word Mover's Distance

Chapter 1

Introduction

1.1 Overview

This thesis presents an investigation of a sequence of frameworks whereby Resource Description Framework (RDF) datasets can be generated from social media data. The idea is to impose structure on such data so that the data can be queried and *actionable* knowledge extracted. Actionable knowledge in this context is defined as valuable and meaningful information that leads to improved decision-making [11, 85, 113, 130]. The motivation, as will be expanded upon later in this introductory chapter, is the wealth of information available within social media data; provided it can be extracted in a structured manner. The focus for the work is Twitter data; so as to limit the scope of the investigation. The work presented in this thesis can thus be more accurately described using the phrase "generation of RDF datasets from Twitter data"; although the proposed frameworks, with some adjustment, could be easily be applied to other forms of free text social media.

In this thesis, the method adopted to identify any Twitter domain of discourse was using Hashtags (#). Hashtags are used by Twitter users to link related tweets and can therefore be seen as being indicative of a given topic [27, 44]. Once the tweets regarding a specific domain of discourse had been collected, using an appropriate hashtag, the data was cleaned so that data mining and Natural Language Processing (NLP) tools and techniques could be applied [60]. The cleaning involved the removal of hashtags, URLs and stop words [21, 60, 62], which have a negative impact on machine learning and the application of NLP as noted in [21].

The remainder of this chapter is organised as follows. In Section 1.2 the motivation for the work contained in this thesis is presented. Section 1.3 presents the research question and subsidiary questions that this thesis seeks to address. Section 1.4 then provides some detail concerning the research methodology that was adopted so as to provide answers to the subsidiary questions, and consequently the overriding research question, presented in the previous section. Sections 1.5 and 1.6 itemise the main contributions arising out of the work described in this thesis and a number of resulting publications.

The structure of the remainder of the thesis is described in Section 1.7. The chapter is concluded with a brief summary in Section 1.8.

1.2 Motivation

Internet usage is increasing rapidly year-on-year, this has been widely noted and observed [37, 110, 120]. This increase has been paralleled with a corresponding increase in social media usage. One has only to look at the statistics on social media usage [63, 108, 134] to get an appreciation of this year-on-year increase. Only 7% of the population of the USA used Social media in 2005, however, this rose to 65% in 2015 [108]. Increased usage of social media has also been influenced by events. The global COVID-19 pandemic is as example. At the height of the pandemic usage of messaging platforms such as Facebook Messenger, WhatsApp, and Twitter increased by approximately 50% [6]. Twitter daily users reached 166 million in the first quarter of 2020, up 24 million over the previous year [134].

Consequently there is an ever-increasing wealth of information contained in social media data. Information that can be turned into actionable knowledge if it can be accessed (queried) appropriately. Actionable knowledge is desirable with respect to a number of sectors such as health, advertisement and marketing, and government. There are a number of reported examples where social media data has been turned into actionable knowledge. These include: election outcome prediction [98]; real-time earthquake event detection [121]; and prediction of health/disease outbreaks such as influenza [7] and COVID-19 [140]. Another example can be obtained from the observation that governments can learn from social media data whether people are content with their government's approach to mitigating health risks pertaining to the COVID-19 pandemic. The desire to extract actionable knowledge from social media is thus the primary motivation for the work presented in this thesis.

The challenge of extracting knowledge from social media data is that it is unstructured, unlike tabular data it is typically in free text format. To solve this problem, we would like to impose structure onto this data so that it can be queried and information extracted. One possible method whereby structure can be imposed on data is by constructing an RDF dataset for the data in question. In the context of extracting knowledge from social media it cannot, at least at time of writing, be anticipated that there is a ready-to-hand RDF dataset available for every Twitter domain of discourse.

Generating an RDF dataset is a non-trivial task; it is both time-consuming and resource intensive. After determining the domain and scope of the RDF dataset the process commences with the identification of the entities of interest using some form of Named Entity Recognition (NER), and then the relations that may exist between the identified entities using some kind of Relation Extraction (RE) mechanism, after which an RDF dataset can be generated using RDF. This is then the focus of this thesis, the automatic generation of RDF datasets from social media data so that structure can

be imposed on this data and consequently the data can be queried so as to extract actionable knowledge.

1.3 Research Questions

From the foregoing, the research question that this thesis seeks to addresses is as follows:

What is the most effective and efficient mechanism whereby an RDF dataset associated with a particular domain of discourse, described in the form of Twitter free text, can be generated?.

To provide an answer to the above research question, five subsidiary research questions were considered:

1. Given that the central building blocks of RDF dataset are entities and relations, can NER and RE be applied using the standard supervised learning tools and techniques available for natural language processing?
2. Given the overhead associated with supervised learning, is there an alternative semi-supervised approach to RE that can be adopted that does not adversely affect the quality of any generated RDF dataset?
3. Following on from the previous two subsidiary questions, is it possible to devise an unsupervised approach to RE that can be adopted, that does not adversely affect the quality of any generated RDF dataset?
4. How do we know when a generated RDF dataset is correct?
5. How can we best cause the generated RDF dataset to be integrable with another dataset that commits the same upper level ontology?

In the context of the above subsidiary questions it should be noted that it was assumed that some kind of supervised, or at least semi-supervised, NER would always be required.

1.4 Research Methodology

This section describe the research methodology that was adopted to provide answers to the above subsidiary questions and consequently the main research question. A four phases programme of work was adopted: (i) supervised, (ii) semi-supervised, (iii) unsupervised and (iv) comparative evaluation. The terms supervised, semi-supervised and un-supervised used here refer to the nature of the machine learning used for RE.

The start point for the work, Phase one, was to consider supervised learning for NER and RE in the context of existing NLP toolkits, namely the Stanford CoreNLP toolkit and the GATE (General Architecture for Text Engineering) toolkit. And to

incorporate these into RDF dataset generation from twitter data frameworks; the first two frameworks presented in this thesis. The operation of both technologies, Stanford and GATE, could then be compared in the context of the RDF dataset generated. The aim here was to provide an answer to subsidiary question one above.

Following on from Phase one, Phase two was directed at the adoption of semi-supervised learning for the RE, the second subsidiary research question to be addressed by this thesis. There are range of semi-supervised learning techniques that could have been adopted. All involve a seed set which is then "grown", in an automated manner, to produce more comprehensive training data. The question was how the seed data can best be grown? The idea was to consider the usage of regular expressions, which in turn would necessitate a bespoke syntax and a parser, and build this into a third framework whose operation could be compared to the frameworks built earlier.

Phase three was directed at an investigation of unsupervised techniques for RE, the third subsidiary question to be addressed by this thesis. In some contexts unsupervised learning refers to clustering techniques of various kinds. In the context of this thesis the term is used to describe mechanisms whereby appropriate training data can be generated in an automated manner. How this was to be done was the subject of the research. As a result of this research, the idea was built into a fourth framework for generating an RDF dataset from Twitter data.

Evaluation of the various NER and RE model generation mechanisms was conducted using established machine learning approaches such k-fold Cross-Validation (kCV). The metrics used were precision, recall and the F1-score measure. Because there were no NER and RE Twitter benchmark datasets available, a bespoke evaluation dataset was created as part of the research,

In each of the above frameworks, a set of entity-relation-entity triples was generated using the developed NER and RE models, which were then used to generate an RDF dataset. There are a number of mechanisms whereby such triples could be converted into RDF, with respect to the work presented in the thesis the Apache Jena tool was used. To allow the generated RDF dataset to have integrability, for example, to allow them to be integrated into another dataset that commits the same upper level ontology, the idea was to map the entities to their classes and then arrange the classes and relations identified into a hierarchy by adding additional supper-classes and super-properties (relation) to the RDF schema (RDFS). In the early stages of the programme of work, the idea was to simply use WordNet for this purpose.

The final phase, Phase 4, was to conduct a comparative evaluation of the four frameworks. In the context of the various NER and RE models their operation could be compared using results obtained earlier. The best performing framework could then be identified. The idea was to then use a larger Twitter dataset to conduct a more comprehensive evaluation with respect to this best performing framework. One of the challenges of the work presented in this thesis was that there were no "gold standard" RDF Twitter data resources available to provide a benchmark. The question was thus

how best to evaluate a given RDF dataset that has been generated in an automated manner; the fourth subsidiary question that this thesis set out to answer. The intention was to consider a number of alternatives, including querying the RDF and eliciting the opinion of domain experts. In Phase 4 consideration was also given to how best to create the hierarchies designed to enhance the integration of the generated RDF. Earlier in the programme of work WordNet was used. During Phase 4 consideration was also given to alternatives. In particular to the usage of the hierarchies available at Schema.org. The aim here was to provide an answer to the final subsidiary research question, subsidiary question five, that this thesis sought to address.

1.5 Contributions

This thesis makes a number of contributions, four RDF Dataset From Twitter Data Frameworks, and a collection of tools and resources developed to support the operation of the individual frameworks and to evaluate the frameworks. Starting with the four frameworks these can be summarised as follows:

1. **The Stanford CoreNLP RDF Dataset From Twitter Data Framework**. A framework that uses the NLP and machine learning (supervised learning) tools and techniques available within the Stanford NLP toolkit to generate NER and RE models, which were then used to generate the desired RDF.

2. **The GATE RDF Dataset From Twitter Data Framework**. A framework similar to the Stanford CoreNLP framework, but instead using the tools and techniques available within GATE.

3. **The Regular Expression RDF Dataset From Twitter Data Framework**. A framework that addressed the limitations of the large manually labelled datasets required by the Stanford and GATE frameworks for RE, by instead using a smaller seed dataset and the idea of regular expressions.

4. **The shortest path dependency parsing and Word Mover's Distance RDF Dataset From Twitter Data Framework**. A fully automated framework, addressing the disadvantages of requiring large manually labelled or seed datasets for RE, as associated with the previous frameworks.

The additional contributions made by the work described in this thesis, additional to the above four frameworks, are with respect to the tools and resources developed to support the operation of the individual frameworks and the evaluation of the frameworks, as follows:

5. **Regular Expression Syntax**. A bespoke syntax for defining regular expressions that can be used for RE.

6. **Regular expression parser**. A bespoke regular expression parser to extract regular expressions, defined using the above syntax, from free text.

7. **Motor Vehicle Pollution Dataset**: A hand labelled collection of Tweets, 300 records, from the motor vehicle pollution domain of discourse, for evaluating all four frameworks

8. **Diabetes Dataset**: A hand labelled collection of Tweets, comprised of 350 records, from the diabetes domain of discourse, for evaluating the NER model used in the final framework.

1.6 Published Work

The main contributions of the thesis, listed in the foregoing section, have been published in two peer-reviewed papers, a conference publication and a book chapter, as follows:

- Alajlan, S., Coenen, F. Konev, B. and Mandya, A. (2019). *Ontology Learning From Twitter Data.* Proc. 11th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2019), Volume 2: Knowledge Engineering and Ontology Development (KEOD), pp94-103.

- Alajlan, S., Coenen, F. and Mandya, A (2020). *From Semi-Automated to Automated Methods of Ontology Learning from Twitter Data.* To appear in the Springer Communications in Computer and Information Science series.

Both publications resulted from work described in Chapters 3, 4, 5 and 6.

1.7 Thesis outline

The remainder of this thesis is divided into seven chapters which are organised as follows:

Chapter 2: Background and Related Work: This chapter presents background information related to data modelling and RDF generation, especially the related work with respect to named entity recognition (NER) and relation extraction (RE). The chapter commences with an overview of data modelling for RDF generation. An overview of ontology languages is included in this part of the chapter; one of which is RDFS. Next, a review of NER approaches that have previously been applied to Twitter data is presented, followed by a review of relevant RE approaches.

Chapter 3: The Stanford CoreNLP RDF Dataset From Twitter Data Framework: In this chapter the first of the four RDF dataset from Twitter data frameworks proposed in this thesis is presented, the Stanford CoreNLP RDF dataset from Twitter data framework. This framework incorporates a supervised method for extracting entities and relations using the Stanford CoreNLP Toolkit and storing them as triples

(subject, predicate, object) to generate an RDF dataset. The RDFS was generated using Apache Jena, and enriched by mapping the entities and relationships to an upper lexicon/schema such as WordNet. The chapter is concluded with an evaluation of the proposed framework.

Chapter 4: The GATE RDF Dataset From Twitter Data Framework: The second RDF dataset from Twitter data framework proposed in this thesis is presented in this chapter. Descriptions are provided of the GATE gazetteer for NER, along with the GATE RE tool. For the framework Apache Jena was used to generate an RDF dataset . And WordNet were again used to enriched the generated RDF and produce the final RDFS. The evaluation of the framework is also presented.

Chapter 5: The Regular Expression RDF Dataset From Twitter Data Framework: The work presented in this chapter attempts to address the limitations encountered with respect to the previous two frameworks; the need for manually labelled training data. In this framework, a semi-supervised approach is used. A small number of tweets, manually labelled, was used as a seed set to generate a NER model and regular expression patterns. These patterns were then used to capture the relations that exist between entities that are defined using the NER model. For the proposed framework Apache Jena was used to generating the RDF which was enriched using the same processes to that used with respect to the first two frameworks. The evaluation of the framework is presented at the end of the Chapter.

Chapter 6: The Shortest Path Dependency Parsing and Word Mover's Distance RDF Dataset From Twitter Data Framework: This chapter presents the final framework considered in this thesis. The framework incorporates a method that automates the preparation of the training data required for RE model extraction. A NER model, which required a training data to generate, was employed to identify entities. Then a RE model was constructed using Shortest Path Dependency Parsing (SPDP) and Word Mover's Distance (WMD). SPDP was used to extract relations from tweets. WMD was used to obtain generic relation labels to describe each relation that links pairs of entity classes. Once the RE model was ready, it was used to extract relations between entities. The final RDF dataset was generated using Apache Jena. The enrichment of the RDF dataset was considered using two alternatives, WordNet and Schema.org to produce an RDFS. The evaluation of the framework is presented withe respect to two evaluation datasets.

Chapter 7: Comparative Evaluation: In this chapter, a comparative evaluation of NER and RE models generated and used with respect to the frameworks described in the forgoing four chapters is presented. In this chapter, a query process was used to query the populated RDFS generated using the four proposed frameworks and in one

case human experts.

Chapter 8: Conclusion: In this chapter, the thesis is concluded by summarising the key findings in the context of the research question and the relevant subsidiary questions. Discussions on the potential for future work are also given in this chapter.

1.8 Summary

This introductory chapter has presented an overview of the work presented in this thesis together with the motivation for the work described, the overriding research question and subsidiary research questions that the thesis seeks to provide answers to, and the adopted research methodology whereby answers to the overriding and subsidiary research questions could be arrived at. The chapter also included an itemisation of the main research contributions of the work described and an itemisation of the publications arising out of the work. The chapter was concluded with an overview of the structure of the remainder of the thesis. In the next chapter a literature review is presented to provide the reader with an understanding of the previous effort underpinning the work presented in this thesis.

Chapter 2

Background and Related Work

2.1 Introduction

Resource Description Framework (RDF) dataset generation from structured data is (relatively) straightforward because of the tabular or schema format typically followed in the organisation of such data. RDF dataset generation from unstructured data, such as free text, the focus for the work presented in this thesis, is very much less straightforward.

Before discussing the background to the work presented in this thesis it is useful to provide a number of definitions as follows:

Term : A sequence of characters, delimitated by white space or punctuation, found in free text; typically a word but not necessarily so.

Entity : A term, or a sequence of terms, that describe some "thing". Typically, entities are noun phrases in free text, often proper nouns such as Paris or France.

Concept : A class describing a set of entities, for example the class city or country. A specific entity is therefore an instance of a class. In the context of Resource Description Framework Schema (RDFS) classes are arranged in class hierarchies.

Relationship A term, or a sequence of terms, describing the relationship between two concepts (and by extension pairs of entities). Typically, relationships are verb phrases in free text. For example, "capital of" or "is a kind of".

property : A relation in the context of either the Web Ontology Language (OWL) or RDFS. A property linking two classes (concepts) and/or a class with an instance of a class, a literal of some kind such as a proper noun or a numeric string.

Regardless of whether the intention is to generate a RDF dataset from free or structured text, the initial objective is to identify and capture a set of semantic triples that feature in the text that are then used to define the desired RDF dataset. The triples are of the form $\langle subject, predicate, object \rangle$. Thus, we might have a triple. $\langle Paris, capital\ of, France \rangle$. To identify semantic triples it is necessary to first identify the entities that feature in

the text and then the relationships that link those entities. The first is referred to as Named Entity Recognition (NER) and the second as Relation Extraction (RE). There has been a substantial amount of research directed at both NER and RE in the context of a wide range of application domains.

From the foregoing the work presented in this thesis can therefore be said to encompass three specific areas of research: (i) data modelling in terms of RDF, (ii) NER and (iii) RE. This chapter explores the background and related work associated with these research areas in the context of the proposed RDF dataset generation frameworks presented later in the thesis in Chapters 3, 4, 5 and 6.

This chapter is organised as follows. Ontology languages, namely OWL and RDFS, are first considered in Section 2.2. The purpose of this section is to justify the use of RDFS throughout the remainder of the work described in this thesis to enrich the RDF datasets. Data modelling using RDF from free text is then considered in the following section, Section 2.3. This section first provides a categorisation of Data modelling using RDF and then goes on to consider some specific examples. From the above, NER and RE are key elements of RDF dataset generation. Relevant previous work on NER and RE are therefore presented in Sections 2.4 and 2.5 respectively. Section 2.4 gives an overview of NER and a categorisation of the most frequently used approaches found in the literature: (i) rule-based, (ii) supervised, (iii) semi-supervised, (iv) distantly supervised and (v) unsupervised [131]. Section 2.5 then gives a review of the state of the art of RE; and, as in the case of the discussion concerning NER, provides a categorisation of the most frequently used approaches found in the literature. The categorisation is broadly the same as that for the NER approaches considered earlier: (i) rule-based, (ii) supervisor, (iii) semi-supervised, (iv) distantly supervised and (v) unsupervised. The chapter is concluded with a short summary in Section 2.6.

2.2 Ontology Languages

There are two popular ontology representation languages which can be used to model the schema of a RDF dataset: (i) the Web Ontology language (OWL) [90] and (ii) the Resource Description Framework Schema (RDFS) [87]. Both are supported by the W3C consortium. Both provide vocabularies whereby the concepts and relationships used in a domain of discourse can be specified, and grammars for using the vocabularies [83]. The vocabularies are used to express the concepts used in a domain of discourse and the relationships between these concepts. Both use a triple-based format, $\langle subject, predicate, object \rangle$. The distinction between RDFS and OWL is that OWL vocabularies are more expressive and hence support decidable reasoning about the data that has been captured [5]. This means that OWL is also more expressive. For example the OWL vocabularies include an "equivalent class" property ($owl : equivalentClass$), not available in RDFS, which is used to link classes that have exactly the same set of

individuals (members) [90]. Also, OWL vocabularies included "disjoint" and "restriction" properties ($owl : disjointWith$ and $owl : Restriction$). The first allows two or more classes that do not share any instances (they are disjoint); the second to place restrictions or properties such as each person has only two parents or a university course is taught by only one lecturer [5]. OWL also imposes constraints on the use of the vocabulary, which RDFS does not do. For example, in RDFS something can be both a class and an instance, while this is typically not allowed in OWL.

With respect to the proposed frameworks presented in this thesis, RDFS was used to model the schema of RDF datasets. The reasons for this were as follows:

1. RDFS is less complicated than OWL (because the associated vocabulary and grammar is less expressive).

2. RDFS is adequate for achieving one of the objectives of the work described in this thesis, to enrich the generated RDF, which will allow the generated RDF dataset to integrated with other datasets that have the same upper level ontology/schema.

RDFS provides a data modelling vocabulary for the Resource Description Framework (RDF) data [24]. RDFS provides a specific RDF vocabulary that can be used to define classes, properties, and domains and ranges for properties [72]. By defining concepts (entities) as classes, a system (model) can be prescribed that defines how resources can be conceptualised. It describes the sub-classes and sub-properties that exist in a given domain, thus creating a hierarchical "is-a" relationship between classes and properties [87].

An example RFDS file is given in Figure 2.1 [23]. The example defines a Class "Person" using the "rdfs:Class" vocabulary. The class "Person" has three properties (relations): (i) "maritalStatus", (ii) *ssn* (Social Security Number) and (iii) "age". Each is defined using the "rdf:Property" vocabulary. The first block of RDF states that the RDF is written in English. The following block defines the class "Person" and states that this is a subclass of the class "Animal" by using the "rdfs:subClassOf" vocabulary. In the following three blocks the three properties of the class "Person" are defined. The properties "age" and "ssn" take "integer" values defined using the "rdf:range" vocabulary. Finally, the "Person" class is connected with the class "MaritalStatus" by the "maritalStatus" property, "Person" is considered as a domain and the "MaritalStatus" class is consider as a range. The vocabulary "rdfs:domain" is utilised to state that every resource with a specific property is an instance of a given class. The vocabulary "rdfs:range" is utilised to state that property values are instances of a given class.

2.3 Data Modelling Using RDF

This section provides an overview of the nature of RDF datasets and the existing work on RDF dataset generation from free text. The discussion is focused on free text, as opposed to Twitter data, because, to the best knowledge of the author, there has

```
<rdf:RDF
  xmlns:rdf="http://www.w3.org/TR/WD-rdf-syntax#"
  xmlns:rdfs="http://www.w3.org/TR/WD-rdf-schema#">

<rdfs:Class rdf:ID="Person">
 <rdfs:comment>The class of people.  correspond to a single person.
</rdfs:comment>
 <rdfs:subClassOf rdf:resource="http://www.classtypes.org/useful_classes#Animal"/>
</rdfs:Class>

<rdf:Property ID="maritalStatus">
 <rdfs:range rdf:resource="#MaritalStatus"/>
 <rdfs:domain rdf:resource="#Person"/>
</rdf:Property>

<rdf:Property ID="ssn">
 <rdfs:comment>Social Security Number</rdfs:comment>
 <rdfs:range rdf:resource="http://www.datatypes.org/useful_types#Integer"/>
 <rdfs:domain rdf:resource="#Person"/>
</rdf:Property>

<rdf:Property ID="age">
 <rdfs:range rdf:resource="http://www.datatypes.org/useful_types#Integer"/>
 <rdfs:domain rdf:resource="#Person"/>
</rdf:Property>

<rdfs:Class rdf:ID="MaritalStatus"/>

</rdf:RDF>
```

FIGURE 2.1: Example of an RDFS File [23]

been limited reported work specifically directed at RDF generation from Twitter Data. However. Twitter data is clearly a form of free text with some notable characteristics of its own, namely that the texts are: (i) very short, (ii) feature many abbreviations and colloquialisms and (ii) frequently incorporate poor grammar and spelling. This section is divided into two sub-sections. The first sub-section provides an overview of data modelling using RDF. The second sub-section considers the categorisation of existing RDF dataset generation approaches, which can be used to compartmentalise the individual approaches found in the literature. The categorisation is founded on the nature of the tools and techniques used with respect to individual systems, which in practice comes down to the tools and techniques used for the required NER and RE. The subsection also includes specific examples, drawn from the literature, of proposed systems for RDF dataset generation from free text.

2.3.1 Data Modelling on RDF Overview

RDF has a simple data model that is straightforward to process and manipulate by applications [56]. The data model is independent of any specific serialisation syntax such as XML. RDF is a format for defining named links between resources in the form of triples ($\langle subject, predicate, object \rangle$), often known as statements, which express a relation (predicate) between resources (subject and object). Three types of RDF data that occur in the triples (subjects, predicates, objects) can be as follow:

- IRI (International Resource Identifier): Used to identify a resource. It is an identifier for a resource without implying a location or a way to retrieve the resource. IRIs are a generalisation of URIs (Uniform Resource Identifiers), which allow non-ASCII characters to be utilised. IRI can be used with respect to either a subject, predicate or object.
- Literal: Used for values that are not IRI, such as strings, numbers and dates. An object in an RDF triple could be an IRI, a literal or a blank node.
- Blank node: Used describe a node in a RDF graph not described using an IRI or a literal. A black node can only be a subject or an object. Also referred to as an "anonymous resource" or a "bnode".

A simple example of a RDF triple, $\langle course, isTaught\_by, professor) \rangle$ is as follow:

```
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:ex="http://example.com/">
<rdf:Description rdf:about="http://example.com/course">
  <ex:isTaughtBy rdf:resource="http://example.com/professor"/>
</rdf:Description>
```

The two "xmlns" tags specify that the resources will be using tags from both the "RDF" and "ex" Namespaces. The "rdf:Description" vocabulary begins the definition of the resource being described [101]. The "rdf:about" refers to the resource itself (a course in this case), while "ex:" gives the isTaughtBy property of the description using the "ex" Namespace. Using RDF, the predicate (also called a property) is a relation between subject and object. The Resource, in fourth line of the example, refers to the object (the resource "professor" in this case defined as an IRI). The resource is a "thing" of interest. Resources might be authors, tennis players, places, people and so on [5].

2.3.2 A Categorisation of Data Modelling of Free Text Using RDF

From the literature numerous approaches have been utilised over the past ten years for RDF dataset generation from free text. The tools and techniques of information retrieval, machine learning, data mining, Natural Language Processing (NLP) and knowledge representation and reasoning have all been applied, in various forms. To discuss these different approaches, it is useful to provide a categorisation of these approaches. This

section presents such a categorisation in terms of two headings as shown in Figure 2.2: (i) linguistic approaches, and (ii) hybrid approaches. Using this categorisation the work presented in this thesis can be categorised as a hybrid approach. Further details regarding this categorisation is presented below. In each case some examples are presented to support the discussion, many of these examples are revisited in the following sub-section.

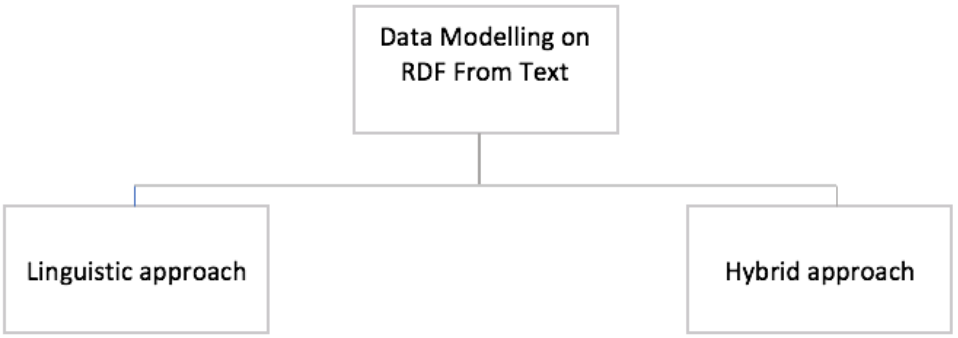

FIGURE 2.2: Taxonomy of approaches to data modelling using RDF

A summarising table is given in Table 2.1 of the examples of previous work on the data modelling of free text using RDF considered in this section; further detail is given in the following sub-sections.

Linguistic approaches

In the linguistic approach to RDF dataset generation, as the name implies, linguistic techniques are used for NER and RE. A range of NLP-based techniques have been applied. The NLP techniques used include Part Of Speech (POS) tagging, sentence parsing, syntactic and structure pattern analysis and dependency parsing. These techniques are directed at examining natural language text at various language levels, typically either the syntactic or the semantic level. The idea is to discover entities and relations using the linguistic indicators present in the text, and then to use these to generate a RDF dataset. The frequently quoted disadvantage of the Linguistic approach is *data sparseness* [2], whereby the given data corpus is not sufficient to model all the nuances required to accurately model the content. However, some examples where these methods have been used in the context of RDF dataset generation can be found in [13, 48, 55].

In [13], Batouche et al. propose a RDF-based method for querying the content of a text. The text is first mapped to RDF triples using existing NLP tools; the generated RDF could then be queried using SPARQL queries. The approach was based on applying a set of rules to extract triples from text and map them to RDF. First, the terms were extracted using a list of terms in a file that was defined by a domain expert. Then,

TABLE 2.1: Summarisation of example approaches to data modelling of free text using RDF considered in Section 2.3.1.

Name	Category	Advantages	Disadvantages
Batouche et al. [13]	Linguistic approach	Using dependency parser to label sentences	Generating a set of rules manually to extract triples from sentences.
Angelin and Vijaya [55]	Linguistic approach	Using dependency parser to label sentences	Using nsubj and nsubj to extract triples that lead to avoiding other important triples in the same sentence
Exner and Nugues [48]	Linguistic approach	Extracting triples automatically.	Using a small number of sentences for evaluation.
text2RDF [30]	Hybrid approach	Extracting triples automatically.	NER and RE are supervised learning that means a training set is needed.
CyberTwitter [95]	Hybrid approach	Using SVCE to identify entities related to the cybersecurity domain	15 alerts only evaluated, raising concerns about the validity of the evaluation
Dimitar et al. [45]	Hybrid approach	Using FEL and SentiStrength to identify terms and sentimentally analyse tweets	RDFS is needed to generate RDF dataset

Stanford dependency parsing was used to parse the text. For example, Figure 2.3 shows the parse of a sentence. It can be noted that each word is labelled with a relation name. A set of rules were generated manually to extract triples such as "SBJ PREP-WITH", SBJ is a nominal subject when PREP-WITH is prepositional object introduced by the preposition "with". Preposition (e.g., with) are "folded into" the arc label rather than being labeled as nodes like the rest of the input sentence is. According to the Figure 2.3, the triple is $\langle pipe, identify-with, caution-label \rangle$. The Stanford dependency parser was used with respect to the fourth proposed framework for generating RDF from Twitter data presented in this thesis described in this thesis (Chapter 6). However, the proposed framework avoid using the set of rules that used in Batouche et al. to extract the triples from text.

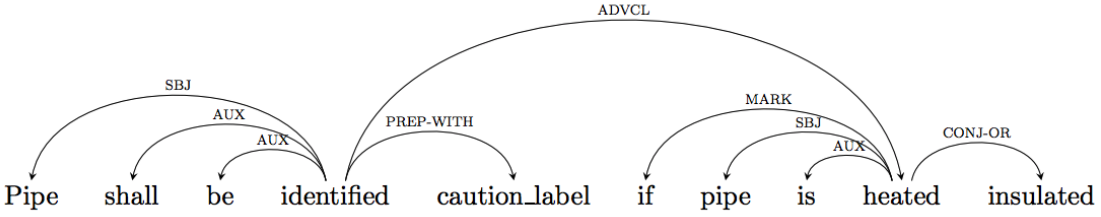

FIGURE 2.3: Dependency Parsing Example

In [55], Angelin and Vijaya present a method to extract triples from text and represent them as an RDF dataset. The input documents were analysed and the sentences were extracted one by one. The dependency parsing of each sentence was again conducted using Stanford dependency parsing so as to extract the desired triples ($\langle subject, predicate, object \rangle$). Two dependencies were considered:

- nsub (nominal subject): A noun phrase used in the system to extract verbs and the subject of each verb. For example, "suspected militants in army fatigues opened fire". The nsubj in this case is "opened, militants".
- dobj (dependent object): The direct object of a Verb Phrase (VP); a noun phrase which is the object of the verb. Used to extract the verb and the object of the verb. Again using the example "suspected militants in army fatigues opened fire", the dobj is "opened, fire".

According to the example above the triple of the sentence "suspected militants in army fatigues opened fire" is $\langle militants, opened, fire \rangle$, which is the represented in RDF format. A disadvantage of the system proposed by Angelin and Vijaya is that using only nsubj and dobj will lead to other important entities and relations being ignored. Regarding the evaluation dataset presented in Chapter 3, the Motor Vehicle Pollution dataset, using only the nsubj and dobj dependencies will result in the extraction of only the entities "Norway" and "petrol" and the relation between them; the entities "Norway" and "2025" and the relation between them will be ignored.

As noted above dependency parsing was used with respect to the fourth framework presented in this thesis (Chapter 6) to prepare a training dataset for the purpose of generating a RE model. The disadvantage of the approach proposed by Angelin and Vijaya , where by not all triples are extracted, was avoided in the case of this fourth framework by using NER coupled Shortest Path Dependency Parsing (SPDP).

In [48], Exner and Nugues proposed a system to automatically extract RDF triples from unstructured data, using DBpedia as the knowledge base. The system operated using a semantic parser [53] and a co-reference solver [76], and used a base-mapping ontology system that used DBpedia to infer relationships between entities. In the reported evaluation, the system was used to process $114,895$ randomly selected articles comprised of a total of $2,156,574$ sentences. The research focused on three entity classes: Persons, Places and Organisations. For the purposes of evaluating the system, 200 randomly selected sentences were analysed manually and an F1 score of 66.3% reported with respect to the mapped triples, showing only limited effectiveness. With specific reference to the evaluation reported in [48] the small number of sentences used in the evaluation (200), compared to the original number of sentences that were processed ($2,156,574$ sentences), raises questions as to the validity of the evaluation.

Hybrid approaches

The second category of data modelling using RDF considered in this section is the hybrid approaches. This encompasses combinations of linguistic techniques with a statistical and machine learning techniques [36]. The statistical and machine learning category of approaches to generate an RDF data set from free text also encompass a range of techniques, from information retrieval to data mining. In [137] it was observed that "The lack of consideration for the underlying semantics and relations between the components of a text makes statistics-based techniques more prevalent". The intuition underpinning the combination of techniques is that a combination of techniques will produce a better performance with respect to NER and RE. In this thesis, all the approaches considered are hybrid approaches. This category of approach was adopted because of the generally held view that there is no single approach that, when used in isolation, can produce a high quality of data model. This was argued in [8, 125] where a hybrid approach, combining linguistic, statistical and machine learning techniques, was found to produce better data models than individual approaches. Hybrid approaches also tend to address the data sparseness problem.

Many of the existing studies on data modelling using RDF have adopted a hybrid approach such as [30, 45, 95] In [30] an approach, called text2RDF, was presented to extract triples from text and convert them to RDF. The framework began with a pre-processing phase; splitting the text into "tokens", finding the sentence and paragraph breaks, and then apply shallow parsing to annotate each word with Part Of Speech (POS) tags. The next phase was extracting information from text using two steps (i) Named Entity Recognition (NER) and (ii) Relation Extraction (RE). Both the NER and RE models used in the framework were generated using supervised learning. The final phase was to convert triples extracted, using the NER and RE models, into RDF triples. For the evaluation of the framework, the NER focused on 11 classes; F1-score of 76.57% was obtained.The RE model focused on 5 relations; an F1-score was 75.68%. A similar three-phase process was adopted with respect to the first two RDF from Twitter data frameworks considered in this thesis (described in Chapters 3 and 4).

In [95], Sudip et al. present a hybrid approach to generating an RDF dataset from tweets, called CyberTwitter. CyberTwitter automatically extracted information from tweets regarding cybersecurity and converted them in to an RDF dataset to alerts users. Tweets were collected using the Twitter API and then cleaned. Security Vulnerability Concept Extraction (SVCE) [74] was then used to extract terms that related to security vulnerabilities. SVCE consist of a NER model that has been trained using text from security blogs, such as the Common Vulnerabilities and Exposures (CVE) official security bulletins from Microsoft and Adobe. When the terms were extracted, the terms were mapped to Ontologies and Knowledge Bases (the Unified Cybersecurity Ontology, and the DBpedia and YAGO knowledge bases). For example the term "Adobe Flash" mapped to DBpedia as "dbr:Adobe Flash". A set of properties were defined to be included in the RDF dataset such as "hasVulnerability", a property of the terms that

belong to class "Intelligence and Vulnerability". To evaluate the quality of the approach, a small user study was conducted where five assessors asked to judge the usefulness of generated alerts by choosing three options: "useful", "maybe", "useless". The set of tweets that caused the generated alert were given to the assessors. Out of 15 alerts generated, 13 were categorised as "useful" and the remaining two were categorised as "maybe". The limitation of the evaluation, it can be argued, is that only 15 alerts were evaluated; thus raising concerns about the validity of the evaluation.

In [45], Dimitar et al. present an alternative framework for modelling Twitter free text data using RDF. The focus was a Twitter dataset related to the COVID-19 pandemic. Spam tweets were first removed using a Multinomial Naive Bayes (MNB) classifier trained on the HSpam dataset, which achieves 94% precision on spam labels [123]. Yahoo's Fast Entity Linker (FEL) tool [19] was used to link terms. FEL was designed to link entities within texts. One the linking was complete, the tweets were analysed sentimentally using SentiStrength, a tool for the sentiment analysis of social web data. SentiStrength assigns a positive and a negative label to a short text. Entities and sentiment annotations with the metadata that extracted from the tweets, such as tweet id, post date, username (user who posted the tweet), favourite and retweet count, hashtags (words starting with) and the URLs included in the tweets were converted to an RDF format using RDFS [50]. The reported evaluation of the FEL tool and SentiStrength gave F1-scores of 54% and 52% respectively. It can be argued that this is below an acceptable level; further discussion on what an acceptable F1 score is given later in this thesis. However, the main limitation of the approach presented by Dimitar et al. is that an existing RDFS is needed to generate an RDF dataset; with respect to the scope of the work presented in this thesis it can not be assumed that such an RDFS is available in all cases. However, the frameworks presented in this thesis avoid this concern by generating the RDF dataset from the tweets directly.

2.4 Named Entity Recognition

In the introduction to this chapter it was noted that the identification of triples is the core requirement of any form of RDF dataset. The triples are of the form $\langle subject, predicate, object \rangle$. To identify the subject and object it is necessary to first identify entities, hence Named Entity Recognition (NER). NER is extensively used with respect to a range of NLP applications such as: information extraction, question answering, co-reference resolution and topic modelling [141]. NER can be defined as the task of identifying the names of entity classes such as locations, persons, organizations, and times and dates [141]. As already noted, NER plays a significant role with respect to automated RDF dataset generation as considered in this thesis.

This section presents a review of a range of NER approaches. The various approaches are categorised (see Figure 2.4) as follows: (i) rule-based (ii) supervised (iii) semi-supervised (iv) distantly supervised and (v) unsupervised. A similar categorisation

was used with respect to the RE approaches discussed in the following section. The supervised approach is the approach that was adopted with respect to the work presented in this thesis. Each of the five NER approach categories is discussed in further detail in the following five sub-sections. This section is concluded with Sub-section 2.4.6 where NER in the context of Twitter data, the application focus for the work presented in this thesis, is discussed.

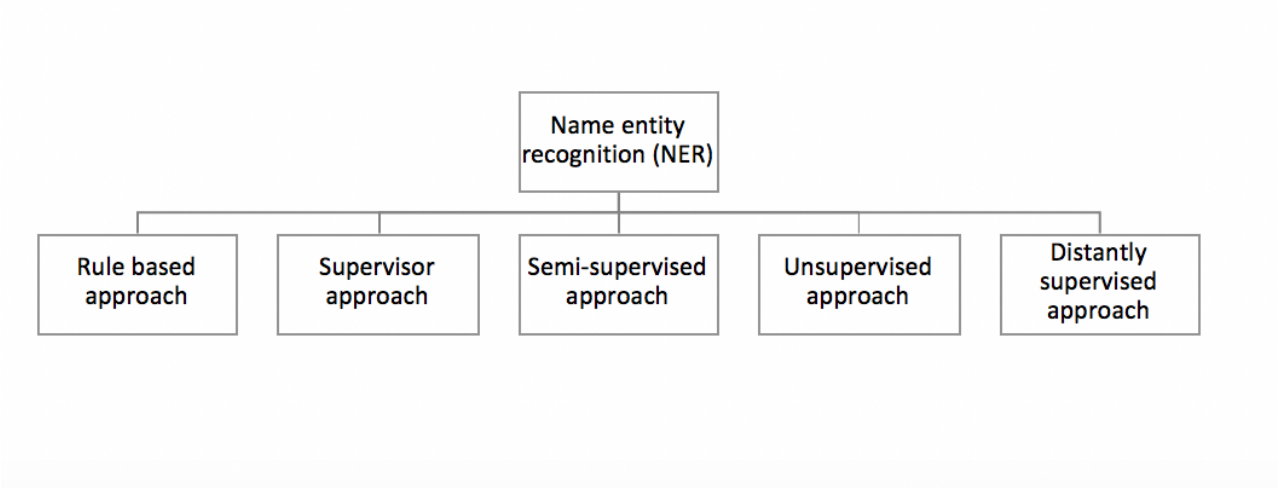

FIGURE 2.4: Categorisation of NER approaches

2.4.1 NER Rule-Based Approach

The Focus of the rule-based approach to NER is the identification of entity names using lexical rules expressing language features. Three variations can be identified: (i) rule only approaches, (ii) combinations of rules and dictionaries such as Gazetteers, and (iii) combinations of rules and feature selection algorithms [131]. Examples of the NER rule-based approaches can be found in [40, 109]. In [145] two examples of rules are given:

Rule 1: PERSON TRIGGER+ $\backslash bUppercaseWord\backslash bUppercaseWord$

Rule 2: PERSON TRIGGER+$\backslash b(\backslash w+)\backslash bKNOWNNAME$

The first rule, Rule 1, will identify any combination of at least two upper case words as the name of a person if they are located next to trigger words. Trigger words are words typically surrounding proper names such as Dr. and Mr., or verbs like "said". In [145] the trigger words were generated manually. For example, given a text containing "Mr. Saad Alajlan" where "Mr." is a "PERSON TRIGGER", and "Saad" and "Alajlan" belong to "UppercaseWord", the rule will extract the name "Saad Alajlan". Note that "$\backslash b$" indicates a word boundary, such as white space or punctuation.

The second rule, Rule 2, will identify an unknown name that exists between the known name and the trigger word. For example, again using the phrase "Mr. Saad

Alajlan", but in this case assuming "Saad" is an unknown name, "Alajlan" is a known name and "Mr" is a "PERSON TRIGGER", the rule will extract "Saad". Note that "$\backslash b$" denotes a word boundary, and that "$\backslash w$" denotes an unknown word.

The advantage of the rule-based category of NER approaches is the simplicity of generating individual rules, which means that individual rules tend to be of high quality. However, generating a comprehensive set of rules to extract all desired entities is a time-consuming task. It is also difficult to measure the comprehensiveness of any generated set of rules. These disadvantages suggested that the rule-based category of NER approach was unsuited to the RDF dataset generation from Twitter data application domain considered in this thesis.

2.4.2 NER Supervised Approach

Supervised approaches use supervised machine learning methods for the NER [68]. Supervised learning is a method that required labelling data as a training set. The work presented in Chapters 3,5 and 6 utilises this technique with respect to the proposed Stanford NER models. Stanford NER used Conditional Random Fields (CRF), which will be defined in Chapter3 in detail. The well-established disadvantage of supervised learning is the need for human resources to prepare a training/test set. Examples of supervised learning techniques that have been used for NER include: (i) Hidden Markov Models (HMM) [16], (ii) Support Vector Machines (SVM) [86], (iii) Decision Trees [124], (iv) Conditional Random Fields (CRF) [88, 126], and (v) Deep Learning [34]. These are the typical techniques that are used in a system for annotation text **??**. Because of the significance of the supervised category of NER approaches with respect to this thesis, further discussion of this approach is left to later in the thesis (Chapters 3, 5 and 6). Example of NER supervised approach from Twitter data will be presented in sub-section 2.4.6.

2.4.3 NER Semi-Supervised Approach

Semi-supervised learning involves the use of a small training set, smaller than in the case of supervised learning, which is then augmented using unlabelled data so that a sufficiently large labelled training set is generated [80]. The idea is to mitigate against the resource overhead associated with supervised learning. One popular semi-supervised technique is "bootstrapping" [99]. An example where bootstrapping has been used for NER can be found in [26]. The benefit of this method is the reduction of the manual effort of labelling data. However, the disadvantage of semi-supervised learning is that labelling errors will be exasperated because of the small seed datasets used. Although Semi-supervised learning can be viewed as an extension of supervised learning it was not adopted with respect to the work presented in this thesis because of the foregoing disadvantage and because of the intuition that (fully) supervised learning would generate better results.

2.4.4 NER Distantly Supervised Approach

Distantly supervised learning involves the use of existing training/test data used with respect to some appliaction, which is adopted for use with respect to a related application, thus avoiding the overhead associated with the supervised and semi-supervised approaches [143]. Knowledge Bases that are readily available online are a good source of training/test data for distantly supervised NER. The advantage of this method is eliminating the need for the human annotation of training data. Examples where the distantly supervised approach has been used for NER can be found in [114], where Wikipedia was used; and [143], where an e-commerce dataset was used. The main disadvantage of the approach is the technology and resource required to adapt existing training/test data to the current application, especially if the two domains are only loosely related. This disadvantage renders it inappropriate for the RDF from Twitter data application considered in this thesis, as any proposed approach needs to be generic (suited to any Twitter domain of discourse). As such, there is no guarantee that appropriate existing training/test data will be available in all cases to support the idea of distantsupervised learning.

2.4.5 NER Unsupervised Approach

The unsupervised method does not require any labelled data for training. The most common genre of unsupervised learning is data clustering [58]. In the context of NER this essentially involves gathering all the different named entities with a similar context in individual clusters. Example of NER using the unsupervised learning approach can be found in [82].A fully unsupervised NER model which only needs to pre-trained word embeddings. This example use k-means clustering method and Euclidean distances measuring. The advantage of the unsupervised approach is that training data is not required. However, the disadvantage is that unsupervised learning tends to deliver suboptimal results with respect to MER model generation. This was evidenced in [47] where Elsner et al. presented an unsupervised generative model that used named entity coreference information [31] in a syntactic context; an F1-score of 48.5% was reported. The system originally described in [47], has been improved upon with the addition of hand-generated rules designed to extract entities based on punctuation, modifiers and pronouns. After improvement, the system achieved an F1-score of 86%. However, the hand-generation of rules is clearly undesirable. In addition, in the context of Twitter data, the use of punctuation is either limited, not used correctly or not used at all.

2.4.6 Previous Work on Named Entity Recognition for Twitter Data

A lot of work has been done regarding NER, using the different categories of approach listed above, in the context of unstructured data such as web page text and social media text (including Twitter data). This sub-section reviews some of the relevant previous work directed at NER with respect to Twitter data, the focus for the work presented

in this thesis. Namely the work of Ritter et al. [116], Limsopatham and Collier [81], and Tan et al. [142]. These examples were chosen because of their prominence in the literature and to illustrate the range of approaches with respect to the foregoing categorisation. A summarisation of the selected example approaches considered in the remainder of this sub-section is given in Table 2.2.

TABLE 2.2: Summarisation of example NER for Twitter data approaches presented in Sub-section 2.4.6.

Name	Category	Advantages	Disadvantages
Ritter et al. [116]	Distantly supervised approach	Building T-class automatically using Labelled-LDA and Freebase.	The topics that have good coverage in Freebase were chosen for evaluation. The topics that do not have good coverage in Freebase would not produce an equivalent performance
Limsopatham and Collier [81]	Supervised approach	Learning orthographic features automatically.	Using a small size of the training data. Deep learning usually needs the data to be substantial in order to perform better than other techniques.
Tan et al. [142]	Rule based approach	Creating rules to extract entities automatically.	Transforming abbreviations words manually. A gap between the F-scores for the formal and informal datasets.

In [116], Ritter et al. presented a distantly supervised approach to identify entities types within Twitter data [112]. A two-step process was presented: (i) Segmenting Named Entities (T-SEG) and (ii) Classifying Named Entities (T-CLASS). For the evaluation presented in [116] the T-SEG system was trained using 2400 tweets labelled with orthographic, contextual and dictionary features. T-CLASS was a learning system that used distantly supervision learning founded on Labelled Latent Dirichlet Allocation (LLDA) topic modelling and the Freebase knowledge base as the source of the distant supervision. The Performance of both T-SEG and T-CLASS was measured using F1-score, an overall F1-score of 51% was obtained. For the evaluation the test data contained another $2,400$ annotated tweets with 10 types of entities popular in Twitter and Freebase. The entity types were: (i) Person, (ii) Geo-Location, (iii) Company, (iv) Product, (v) Facility, (vi) TV-show, (vii) Movie, (viii) Sports-team, (ix) Band and (x) others. The main disadvantage of the approach was that the T-SEG training was computationally

expensive, It should also be noted that for the reported evaluation the authors chose topics that had good coverage in Freebase; it is suggested by the author of this thesis that topics that do not have good coverage in Freebase would not produce an equivalent performance.

In [81], Limsopatham and Collier presented a supervised approach, specifically a deep learning approach, for named entity recognition within Twitter data. The proposed method utilised an orthographic sentence generator, word representations and a Bidirectional Long Short-Term Memory (BiLSTM) neural network. Orthographic sentence generation entails creating an orthographic sentence that includes an orthographic pattern of words in each input sentence. For example, from the sentence "14th MENA FOREX EXPO announced!!", the orthographic sentence generator would generate the following *nncc CCCC CCCCC CCCC ccccccccpp* (where n indicates a numeric digit, c a lower case letter, C an upper case letter and p a punctuation symbol). The orthographic sentence then allowed the BiLSTM to leverage orthographic features automatically when performing NER. The training dataset that was used to evaluate the system contained $2,349$ tweets focusing on 10 types of targeted entities. These targeted entities included "company" with 207 entities, "facility" with 209 entities, "geo-location" with 325 entities, "movie" with 80 entities, "music-artist" with 116 entities, "person" with 664 entities, "product" with 177 entities, "sport team" with 74 entities, "TV show" with 65 entities and others with 545 entities. The Overall F-score was 52.4%. A criticism of the evaluation presented in [81] was the small size of the training data. Data used for training deep learning neural networks usually needs to be substantial [46]. There were only $2,349$ tweets in the training data, focused on 10 entity types, an average of some 246 tweets for each entity type.

In more recent work (2019), Tan et al. in [142] proposed a method for extracting road traffic conditions from Twitter data using NER techniques. In the proposed method, the authors used a rule-based NER approach to extract entities belonging to locations and traffic states. The process commenced with the manual replacement of abbreviations in the tweets to full words; for example, the character "-" to "to" and the word "Acc" to "Accident". Next the tweets were tokenised and tagged using POS tagging. After analysing more than 2000 tweets, rules were handcrafted founded on five different types of word: (i) trigger words, (ii) counter trigger words, (iii) boundary words, (iv) blocking words and (v) extensions words (see Table 2.3). As noted earlier, trigger words are words that trigger a rule. Counter trigger words are words that will cause pausing of the triggering of rules when these words are found around the trigger word. Blocking words are words that are found between trigger words and a desired entity. Extension words are considered to be part of a location entity that are often present after the location entity but are not nouns. The evaluation datasets used in [142] comprised $65,000$ unfiltered formal tweets collected from official sources, such as radio station accounts, government accounts and news accounts; and $80,000$ informal tweets by normal everyday users. For evaluation purposes, 300 formal and 150 informal

English language tweets were selected and labelled. The distinctions between formal and informal tweets were grammar mistakes, acronyms and abbreviations. This distinction was an artefact of the nature of Twitter, which brings together people with different educational backgrounds that leads to the grammatical vagaries frequently found in informal Tweets. The recorded F1-scorers (formal and informal) for the location entity were 88.9% and 81.3% respectively. The corresponding F1-scores for state information were 51.4% and 38.4% respectively. The disadvantages of the system were the associated computational complexity and the effort required to transform manually abbreviations in the tweets to their corresponding complete words. Moreover, a significant gap was observed between the F1-scores for the formal and informal datasets, hence calling into question the effectiveness of the technique.

TABLE 2.3: Word types and their description used in the rule-based approach of Tan et al. [142]

Type of Words	Description	Sample Words
Trigger Words	Words that triggers rules	to, from, near, along, towards
Counter Trigger Words	Words that nullify the triggering of rules	due, available
Boundary Words	Words that marks the ending of an entity	to, till, in
Blocking Words	Extensions or descriptions of words that usually occur between trigger words and the desired entity	the, almost
Extension Words	Words that further describe the "location" entities	roundabout, highway, exit

2.5 Relation Extraction

The previous section reviewed existing work on NER. Once the entity classes that feature in a domain of discourse (represented by a free text corpus) have been identified the next stage is to identify the relations (properties) that link these entity classes. Relation Extraction (RE) is the task of identifying and storing the semantic relationships that exist between entities in text [67]. For example, the text, "Liverpool is located in the UK", has the relation "located-in". Most RE systems focus on extracting binary relations [10] such as *located-in(Liverpool, Uk)* and *founder-of(Bill gates, Microsoft)*. As noted earlier, existing RE approaches can be categorised under five headings (as shown

in Figure 2.5), the same headings used to categorise the NER approaches described above: (i) rule-based, (ii) supervised, (iii) semi-supervised, (iv) distantly supervised and (v) unsupervised. Each of these five categories is considered in further detail in the following five sub-sections. In each case the discussion includes reference to examples. The examples considered are summarised in Table 2.4. The examples were selected because they were representative of the most popular methods of extracting relations from text. Surprisingly, the thesis author has found only a limit number of examples in the literature where RE has been applied to Twitter data, such as [3, 71] where a combined rule-based and supervised method was considered..

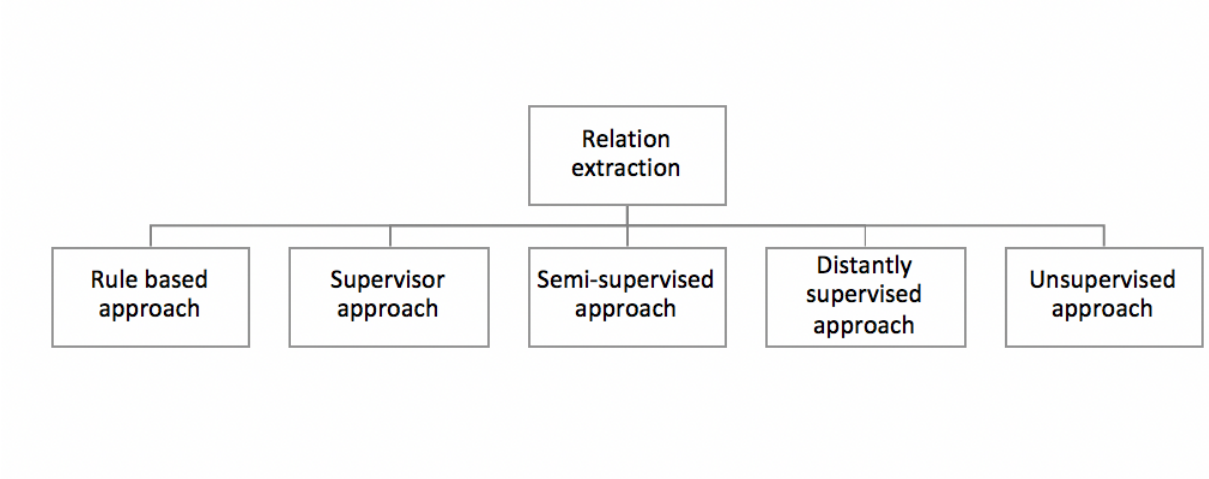

FIGURE 2.5: Categorisation of RE approaches

2.5.1 RE Rule-Based Approach

Rule-based approaches to RE, as the name suggests, typically employ hand-crafted rules to extract relations. For example, in [59] Hearst presented a sequence of rule-based patterns to extract relations from text. Two examples of such patterns are as follows:

Pattern: such NP as {NP ,}* {(or | and)} NP
Sentence: ... works by such authors as Herrick, Goldsmith, and Shakespeare.
relations:
is (“author”, “Herrick”)
is (“author”, “Goldsmith”)
is (“author”, “Shakespeare”)

Pattern: NP {,} especially {NP ,}* {or | and} NP
Sentence: ... most: European countries, especially France, England, and Spain.
Relations:
is (“France”, “European country”)
is (“England”, “European country”)

Table 2.4: Summarisation of example RE approaches presented in Section 2.5.

Name	Category	Advantages	Disadvantages
Wang et at. [136]	Supervised approach.	Extract relation automatically.	Preparing the training set manually.
Chunxiao et al. [35]	Supervised approach.	Retraining relation extraction system easily.	Preparing the training set manually.
Carlson et at. [32]	Semi-supervised approach.	Preparing training set with small amount of labelled.	Appearing errors when the initial classification is used.
Riedel et at. [115]	Distantly supervised approach.	Using knowledge base for extraction.	Lacking an appropriate knowledge source. 30% of entities mentioned on Twitter do not appear in Freebase because they are either too new or are abbreviated or misspelt [116].
Katsios et al. [71]	Rule-Based approach.	Extracting relation automatically.	Extract only the syntax relations.
Anggareska and Purwarianti [3]	Supervised approach.	Extract relation automatically.	Preparing the training set manually.
Li et al. [77]	Supervised approach.	Extract relation automatically.	Deep learning usually needs the data to be substantial in order to perform better than other techniques
Zhang et al. [148]	Supervised approach.	Achieved higher performance than the standard RNN or CNN models.	Deep learning usually needs the data to be substantial in order to perform better than other techniques

is (“Spain”, “European country”)

Alternatively, regular expression patterns may be used to generate rules for relation extraction from text, as suggested in [128]. One of the RDF dataset from Twitter data frameworks presented in this thesis uses the idea of regular expressions to extract relations from tweets (as discussed in Chapter 5). The benefit of the rule-based approach is its simplicity. The challenge is the generation of a set of rules that is sufficiently comprehensive to identify the complete set of relations within a given domain of discourse.

One of the frameworks presented in this thesis, reported on in Chapter 6, is considered to be a rule-based approach, because it is founded on the idea of using the shortest path in a dependency graph, generated using a dependency parsing method, to identify

relations between entities. The assumption was that the shortest path between the two entities, as described by the dependency graph, would hold the information needed to capture the relation between two entities in a sentence. Figure 2.6 present examples of a dependency graphs. One example system where dependency parsing has been used to extract relations from free text can be found in [28] where it was applied to the ACE (Automated Content Extraction) newspaper corpus. The corpus consists of 422 documents, with an additional 97 documents for testing purpose. The authors in [28] utilised the Combinatory Categorical Grammar (CCG) and the Context Free Grammar (CFG) parsers to obtain dependencies; although other dependency parsers could equally well have been used.

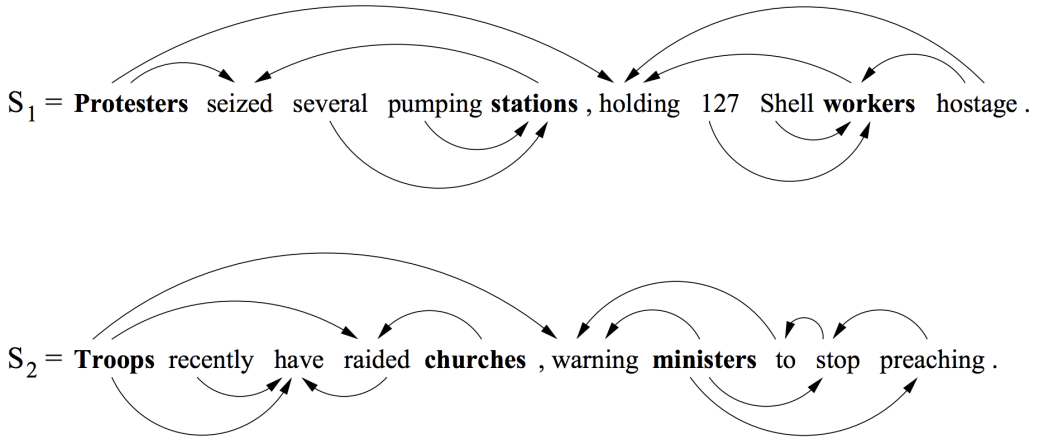

FIGURE 2.6: Sentences as dependency graphs [28]

Another example of a RE rule-based system was presented in [71], where Katsios et al. presented an approach that incorporated both NER and RE. Stanford NER was used to identify the entities in tweets. The targeted entities were Location, Person, Organization, Money, Percent, Date, Time. For the required RE the ClausIE tool was adopted [43], a RE tool founded on Stanford dependency parsing. The next step consisted of selecting the entities and relations that would appear in the final result. First, identifying the most frequent entities in the dataset, and then select the most frequent relations in which these entities occur. The evaluation of the approach was conducted manually. The average precision of the approach was 88%. This main disadvantage of the approach was that it relied on the default Stanford NER, which is not suitable for all domains of discourse, including the motor vehicle pollution and diabetes domains considered in this thesis for evaluation purposes. Moreover, the approach of Katsios et al. resulted in the identification of a lot duplicates relations, because the use of dependency parsing led to the extraction of sets of relations that had the same meaning but a different "shape". This problem can be avoided by grouping all relations that have same semantic meaning, using some form of data clustering technique, as adopted with respect to the fourth framework presented in this thesis (see Chapter 6).

2.5.2 RE Supervised Approach

Supervised learning, as noted above, requires labelled data. Pairs of entities in a given text are typically labelled using predefined relations. RE is then considered as a multi-class classification problem [105]. The principal disadvantage of the supervised approach, as also noted previously, is the need for pre-labelled training/test data. Supervised learning approaches used for RE can be: (i) features-based, (ii) kernel-based or (iii) deep learning-based.

An example of feature-based approach to RE can be found in [3, 35]. Feature based approaches are focused on using either semantic or syntactic features to discover relations. An example of a semantic feature is country names, which may then be used to distinguish between (say) the *citizen of* and *residence of* relation classes, according to whether the country name appears as the first or the second entity in a given sentence [105]. Examples of syntactic features are entities, POS tags, word sequence between entities, the number of words between entities and the path between entities in a given parse [78]. Each can be used to support RE. The main disadvantage of the feature-based approaches is that of identifying the most appropriate features to obtain the best result. Feature based approaches for RE were used with respect to the RDF dataset from Twitter data frameworks presented in Chapters 3, 5, 6 of this thesis. The problem with feature-based approaches to RE is that the features extraction is a very complicated process. It leads to very high dimensional vectors that can lead to significant computational overheads. In [35], Chunxiao et al. introduce a feature-based supervised information extraction system, founded on the Stanford CoreNLP tool. The domain considered in was the USA National Football League (NFL) Scoring corpus. The corpus contained 110 articles relating to the NFL. It was found that the proposed method was easy to retrain with any topic and hence was considered appropriate for use in the context of Twitter domains of discourse. As a result, this method was incorporated into three of the RDF dataset from Twitter data frameworks proposed in this thesis (see Chapters 3, 5 and 6). In [3], Anggareska and Purwarianti presented a system to obtain public complaint information from tweets. Two mechanism were adopted in this system, supervised NER and RE. Seven classes were used to label the NER training set: Object, Location, Time, Conditions, Cause, Suggestion and Link. Seven relation classes were used to label the RE training set: Location, Near, Direction, Section, Condition, Cause and Suggestion. 290 complaint tweets were used as the evaluation dataset. The Ten-Cross Fold Validation (TCV) strategy was used to evaluate the NER and RE models. The average F1-scores of the NER and RE models were 85.6% and 77.2%, respectively. However, the biggest challenge of the Chunxiao et al. and Anggareska and Purwarianti systems was still the preparation of the training data; a problem that is addressed in this thesis by the proposed mechanisms presented in Chapters 5 and 6.

Kernel-based methods have been adopted to address the disadvantage of feature based supervised learning for RE. A number of Kernel function have been proposed to identify similarities between two given relation classes. Three types of kernel function

that are commonly used for RE are: (i) Sequence-based Kernels, (ii) Tree-based Kernels and (iii) Composite Kernels. Essentially, kernel based methods operate through the implicit calculation of vector dot products in very high dimensionality spaces [97]. Some examples where kernel functions have been used for RE can be found in [28, 147]. The SVM model was used for RE with respect to the framework presented in Chapter 4 of this thesis. The biggest limitation of kernel functions, coupled with SVMs, is choosing the most appropriate kernel given a particular application domain [29]. Example of SVM can be founded in [136] where the General Architecture for Text Engineering (GATE) toolkit was used to extract relations from text. This toolkit was also used with respect to one of the proposed frameworks presented later in this thesis. In [136], Wang et al. proposed a model to perform multi-class relation classification using SVM binary classifiers (the one-against-one method). The GATE tool was used for tokenization, sentence splitting, POS tagging, and noun and verb phrase chunking. They also used WordNet to provide word sense disambiguation. The system was able to extract automatically relations from text. The disadvantage, however, in common with all supervised learning methods, was the requirement for training data. The relevance of this example with respect to this thesis is that GATE was used with respect to the second proposed framework (see Chapter 4).

Deep learning is the most recent of the three supervised learning approaches commonly used for RE from text. Deep learning is usually founded on neural network methods where a function that measures the error (or distance) between the actual output and the desired output is used to train the network. Internal parameters (weights) are adjusted until the error is minimised [75]. Of the three approaches (feature-based, kernel-based and deep learning-based) deep learning-based has been shown to be the most effective, provided that sufficient training data is available. One example where deep learning has been used for RE can be found in [77, 148] where Long Shirt Term Memory (LSTM), Convolutional Neural Network (CNNs) and Recurrent Neural Network (RNN) were considered. In [77], Li et al. present a relation extraction method using the Long Shirt Term Memory (LSTM) algorithm to extract relations based on shortest path dependency parsing using the syntactic parse tree of a sentence. This method achieved an F1-score of 84.3%. In [148], Zhang et al. present a hybrid relation extraction model by using the CNN and RNN algorithms to extract biomedical relations based on sentence sequences and shortest path dependency parsing generated from the dependency graph. A fully connected layer was used to combine the output of the CNNs and RNNs. Finally, a Softmax function was applied to extract the relations. The result produced suggested that the hybrid model achieved higher performance than the standard RNN or CNN models. The principal disadvantage of deep learning, as already noted, is the need for large quantities of training data; training data that needs to be hand-crafted. For the purpose of the RDF dataset from Twitter data application considered in this thesis deep learning was therefore not an option because of the lack of sufficient quantities of training data.

2.5.3 RE Semi-Supervised Approach

As noted above with respect to the discussion on NER, semi-supervised learning begins with a small dataset, smaller than in the case of supervised learning, to learn how to extract relations, and then iteratively uses the extracted relations for training [127]. The well documented disadvantage of supervised learning is the need for labelled training data. Hence, the motivations for semi-supervised learning are (i) to minimize the time and effort taken to produce labelled data, and (ii) to take advantage of the unlabelled data that is commonly available and can be used without much effort [105]. As noted earlier in Sub-section 2.4.3, the most popular semi-supervised approach is bootstrapping. Dual Iterative Pattern Relation Expansion (DIPRE) [25] is one example of a bootstrapping method. Some examples where semi-supervised learning has been used for RE can be found in [1, 12]. The disadvantage of semi supervised learning is that it only partially solves the disadvantage of supervised learning, often at the expense of reduced effectiveness.

An example of the use of the semi-supervised approach for extracting relations from text can be found in [32] where an iterative training method, directed at web page free text, was presented that involved "self-supervision". The process presented began with a small amount of labelled (seed) training data to train an initial classifier, which was then used to label additional training data in an iterative manner. The advantage of this method was the small amount of labelled training data that was required. The disadvantage of the method was that labelling errors within the initial classifier would be propagated when the classifier was used to label additional training data. This means any error within the initial classifier will be compounded every time the classifier used to label additional training data. Hence, this approach to RE was deemed unsuitable with respect to the Twitter domain of discourse considered in this thesis.

2.5.4 RE Distantly Supervised Approach

As noted in the foregoing section, distant supervision was originally intended to provide a solution to the limitations of the semi-supervised and unsupervised approaches. Distant supervision operates by using a knowledge base, such as DBPedia [9], as a source of the training data. It assumes that if two entities in a knowledge base have a relationship, then all sentences mentioning these two entities would express that relationship in some way [146]. Some examples where the distantly supervised approach has been used for RE can be found in [94], where freebase [20] was used as the knowledge base, and in [102] where Yago [69] was used as the knowledge base. With respect to the work presented in this thesis it was assumed that for many Twitter domains of interest no suitable knowledge base would be available to support distant supervision (although this might be the case with respect to a small number of domains of discourse). According to the work presented in [116], 30% of entities mentioned in Twitter do not appear in Freebase because they are either too new or are abbreviated or misspelt. A further problem with the distantly supervised approach for RE is that a sentence that mentions two entities

may not express the relationship that connects them in the knowledge base [66]. A distantly supervised approach was therefore considered unsuitable for RE in the context of the generation of RDF datasets from twitter data.

The distantly supervised approach was adopted in many research. In [115], an interesting distantly supervised approach to RE was presented that avoids the use of bespoke labelled training data by using Freebase as an external knowledge base. Stanford NER was used to identify entities that belong to person, organization and location classes. Then, these entities associate with Freebase entities by simply using a string match between entities that defined using Stanford NER and the names of entities in Freebase. For each sentence includes the two pairs that identify using NER tagging and express the relation that linked the two pairs in the freebase, a set of features were extracted such as POS, syntactic features (i.e. features obtained from the dependency parsing tree of a sentence) and so on. Then, all the information, the knowledge base information, the sentences and the features, are used to build the RE classifier. Although the work presented in [115] provided a good example of a distance learning-based approach to RE, in the context of the work on RDF dataset from Twitter data presented in this thesis, it was considered that distance learning-based approaches were unsuitable because of the likely unavailability of an appropriate knowledge source.

2.5.5 RE Unsupervised Approach

The unsupervised approach detects relations using either: (i) some form of clustering model, such as K-means [139] or Hierarchical agglomerative clustering [33]; or (ii) some kind of generative models such as Latent Dirichlet Allocation(LDA) [64]. Generative models, either explicitly or implicitly, model the distribution within a given dataset, which can then be used to label previously unseen data [18]. Generative modelling is therefore an unsupervised learning approach that entails automatically discovering or learning patterns in input data so that the model can be used to generate new examples or group similar parts of the data together that could have been drawn from the original dataset. The alternative to generative modelling is discriminative modelling whereby a model is constructed that can "discriminate" between data points, examples of discriminative modelling include supervised learning, for example the logistic regression and SVM modelling as adopted with respect to the frameworks presented in Chapters 3 and 4. Discriminative models attempt to draw a line in the data space that can be used for classification purposes, whereas generative models attempt to represent how data is distributed over a data space.

The benefit of unsupervised approach is that no training data is required. A disadvantage is that unsupervised learning tends to produce sub-optimal results, which are difficult to interpret [127]. However, some examples where the unsupervised approach has been used for RE can be found in [144] and [138]. With respect to the output produced by these two examples, in [49] it was noted that the output is not always coherent and tends to include irrelevant extractions. In other words the identified relations

are frequently not meaningful and critical relations may be omitted. It was considered that these disadvantages. associated with the unsupervised approach to RE. rendered it unsuitable for the generation of RDF datasets from Twitter data as considered in this thesis.

2.6 Summary

This literature review chapter has presented a review of the background and previous work related to the research presented in this thesis. The review was divided into the three research strands that underpin the thesis: (i) data modelling using RDF, (ii) Named Entity Recognition (NER) and (iii) Relation Extraction (RE). NER and RE play a fundamental role in RDF dataset generation from twitter data. The merits of ontology representation languages to enrich a RDF dataset using the RDFS and OWL were first discussed. Data modelling using RDF was discussed in the context of two categories of approach: (i) linguistic and (ii) hybrid. Potential approaches to both NER and RE were discussed in the context five categories: (i) rule-based, (ii) supervised learning, (iii) semi-supervised learning, (iv) distant supervision and (v) unsupervised learning. In the next chapter the first RDF dataset from Twitter data framework proposed in this thesis is presented, the Stanford CoreNLP Framework.

Chapter 3

The Stanford CoreNLP RDF Dataset From Twitter Data Framework

3.1 Introduction

This chapter presents the first of the four RDF dataset from Twitter data frameworks presented in this thesis; the Stanford CoreNLP framework. This framework was an initial proof of concept framework which was subsequently used as a benchmark with which to compare other frameworks and as a "building block" for the frameworks presented in the later chapters of this thesis. The work presented in the chapter is directed at the following subsidiary research question, Subsidiary Question One from Chapter 1:

1. *Given that the central building blocks of RDF dataset are entities and relations can NER and RE be applied using the standard supervised learning tools and techniques available for natural language processing?*

The fundamental idea was to investigate existing tools and techniques to support Named Entity Recognition (NER) and Relation Extraction (RE) within the context of generating RDF datasets from Twitter data. As noted in Chapter 2, there are a range of such tools available but the most popular, by a significant margin, are those provided within the Stanford CoreNLP toolkit [84]. Stanford Named Entity Recognition and Stanford Relation Extraction are the main tools used with respect to the RDF dataset generation framework considered in this chapter, hence the framework is referred to as the Stanford CoreNLP RDF dataset from Twitter data framework.

With respect to the work presented in the foregoing chapter, Chapter 2, the Stanford CoreNLP framework can best be described as hybrid approach because it incorporates both elements from linguistics, namely Natural Language Processing (NLP), and elements from machine learning. NLP is used to pre-process tweets. Machine learning is

then used to extract entities and relations from the pre-processed tweets. The nature of the machine learning used, for NER and RE, was supervised learning, which means that the system needs examples (training data) to conduct the required learning.

The rest of this chapter is structured as follows. Section 3.2 provided an overview of the Stanford Core NLP toolKit. Section 3.3 provides an overview of the proposed Stanford CoreNLP RDF dataset from Twitter data. Section 3.3 presents further detail concerning the various sub-processes that make-up the parent process. Section 3.4 discusses the evaluation process applied to the Stanford CoreNLP RDF dataset from Twitter data framework. The chapter is completed with a summary of the proposed Stanford CoreNLP framework in Section 3.5.

3.2 Stanford CoreNLP

This section provides an overview of the Stanford CoreNLP toolkit so as to facilitate the reader's understanding with respect to the remaining sections in this chapter. Stanford CoreNLP is an annotation toolkit that supports many NLP from tokenisation through to co-reference resolution. The toolkit is implemented using the Java programming language. Although there are many useful toolkits for the analysis of natural language, Stanford CoreNLP is one of the most utilised [84]. The most significant tools provided within the Stanford CoreNLP toolkit include: (i) a tokeniser for turning text into tokens, (ii) an XML processing tool, CleanXML, to remove XML tags from text, (iii) a Part Of Speech (POS) tagger, (iv) a lemmatisation tool, (v) a NER tool, (vii) a parser, (viii) a sentiment analyser and (ix) a RE tool. With respect to the proposed framework the NER and RE tools were used.

To further justify the use of Stanford CoreNLP with respect to the first framework presented in this thesis the key benefits of Stanford CoreNLP can be summarised as follows:

1. **Effectiveness and Efficiency**: The Stanford CoreNLP toolkit features stable and high-quality linguistic analysis components [84]. For example, in [100], a comparative study of two popular Natural Language Processing tools, Stanford CoreNLP and Apache OpenNLP, a machine learning toolkit for NLP [96], was presented. The comparison was in terms of time taken and accuracy to annotate the text. Stanford CoreNLP demonstrated a better performance than Apache OpenNLP. Stanford CoreNLP as found to be more accurate and more efficient than Apache OpenNLP.

2. **General Applicability**: The Stanford toolkit can be applied at both the large document level and the single sentence level.

3. **Features High Compatibility**: The toolkit has interfaces with respect to many programming languages, including Python, Ruby, Perl, Scala, Clojure, Javascript (node.js), and .NET languages (C and F).

4. **Ease of Use**: The toolkit is relatively straight forward to use, unlike comparable toolkits such as IBM's Unstructured Information Management Architecture (UIMA) [52] and the General Architecture for Text Engineering (GATE) [38]. UIMA is a Java software development kit design to support the implementation of applications directed at the analysis of free text. GATE was developed at the University of Sheffield and is discussed in further detail in the following chapter.

In 2014, Stanford starting support different spoken languages such as Arabic, Chinese, French and German, in addition to English. However, some annotations can only deal with specific languages. For example, NER can only be used with English and Chinese. Having provided an overview of the Stanford CoreNLP toolkit the following sections presents the proposed Stanford CoreNLP framework.

3.3 RDF Dataset Framework Using Stanford CoreNLP

This section provides an overview of the proposed Stanford CoreNLP RDF dataset from Twitter data framework. A schematic of the proposed framework is given in Figure 3.1. From the figure it can be seen that the workflow (pipeline) supported by the framework starts with a collection of tweets describing a domain of discourse. There are then four process that are sequentially applied to the tweet collection: (i) Tweet cleaning, (ii) NER (iii) RE, (iv) RDF dataset generation NER and RE are bundled together under the heading of Knowledge Extraction. NER and RE are applied so as to extract entities and relations from tweets in the form of a set of triples, $\langle subject, predicate, object \rangle$. After that, the RDF dataset is generated using this set of triples. Two further steps were included in the framework, not shown in the figure, to enrich the RDF dataset: (i) Class mapping and augmentation and (ii) RDFS generation. The RDF dataset is enriched by mapping the entities and relationships to an upper lexicon/schema. The identified terms (entities) are mapped to their classes. For example, the entities UK, USA, China and France would be mapped to the class "Location".The classes and relations are then used for RDFS generation [23]. The reason for selecting RDFS was discussed in Section 2.2 of Chapter 2. Each of these processing steps is discussed in further detail in the following five sub-sections.

3.3.1 Tweet Cleaning

As noted above, the input to the proposed Stanford CoreNLP framework is a collection of tweets. Many methods are available to collect tweets pertaining to a specific domain, such as the Twitter API or crawling the Twitter www site using some bespoke software. The limitation of the Twitter API is that it only provides access to tweets from the previous seven days. To solve this problem the "GetOldTweets" program was used [61]. GetOldTweets is a Python library, available from GitHub [1] that provides additional

[1] https://github.com/Jefferson-Henrique/GetOldTweets-python

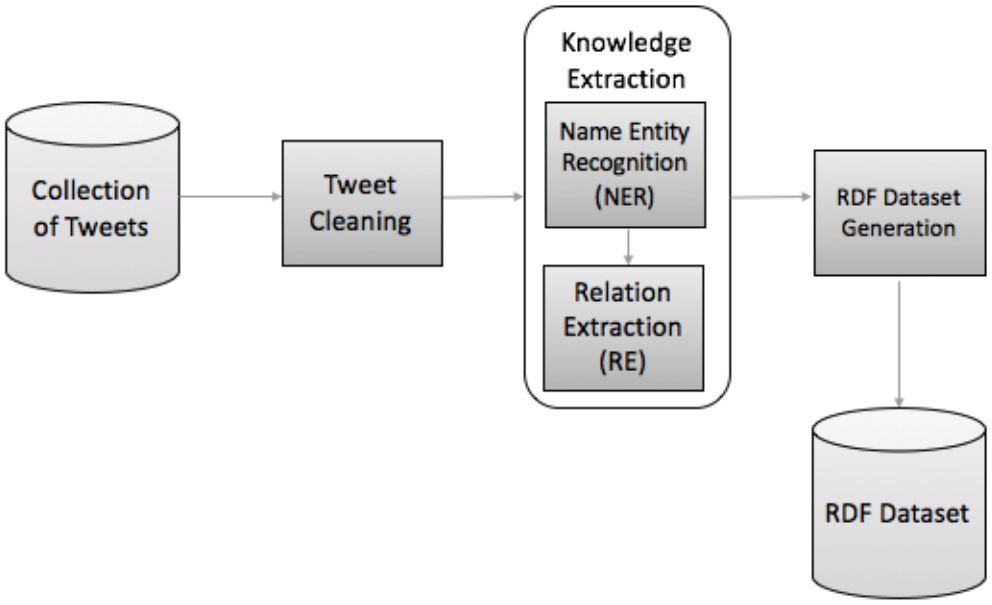

FIGURE 3.1: Schematic of the Stanford CoreNLP RDF Dataset from Twitter Data Framework

functionality than the Twitter API itself. GetOldTweets, as the name suggests, can be used to get tweets from within a period greater than the last seven days with respect to a given topic.

Once a set of tweets, covering a domain of discourse of interest, had been obtained the first process within the proposed Stanford CoreNLP framework workflow (Figure 3.1) was applied, the tweet cleaning process. During this process the collected tweets were transformed ready for the application of the following process. URLs that exist in the tweets were first removed, because they do not provide any useful information. Next, user-names were deleted because information concerning the individual tweet writers was not relevant for RDF generation. Finally, hash symbols and punctuation marks were removed. The data cleaning was conducted to ensure effective operation of the NLP tools to be applied in the following stages.

3.3.2 Knowledge Extraction

The second stage of the framework, shown in Figure 3.1, is knowledge extraction. This stage contains two sub-processes NER and RE. Each is discussed in further detail below.

Named Entity Recognition (NER)

The second process in the workflow presented in Figure 3.1 is NER. As noted earlier, NER is the process of identifying named entities and categorising these entities using suitable class labels [70]. The primary purpose of the NER, with respect to the proposed framework, was to identify entities within tweets to be included in the RDF dataset and

then define the classes that should be included in the final RDFS. The process is best described by considering the example given in Figure 3.2, which shows the sentence "Liverpool city is a metropolitan borough in Merseyside, England.". This sentence has some words in it labelled using NER; the entity Liverpool is labelled with the class "City", the entities Merseyside with the class "Location" and England with the class "Country". For the proposed framework the Stanford NER tool [54] was used to extract entities from tweets and associate them with classes.

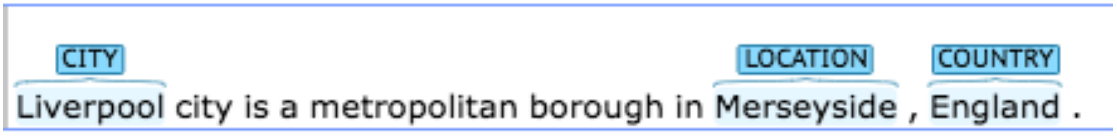

FIGURE 3.2: Example application of NER

The Stanford NER tool uses a Conditional Random Field (CRF) model to classify entities according to a set of predefined classes. CRFs are a type of sequence labelling models [54]. The Stanford NER tool labels words in a given text according to a number of features: (i) the nature of the current (targeted) word, and the nature of the previous word and follow on word (if they exist); (ii) the current word character n-gram; (iii) the POS tag for the current word and the POS tags for previous and follow on words (if they exist); (iv) the current word shape (for example weather it is capitalised or not), (v) any surrounding orthographic information and (vi) the presence or otherwise of a word on the left and to the right of a specific window size [54]. Window size is the size of the context window and determines the number of words before and after a given word.

The Stanford NER tool has three default models, trained using a supervised learning mechanism, that identify "standard" sets of classes, as follows:

Model 1 : Identifies there classes: (i) Location (ii) Person and (iii) Organisation; and was trained on the CoNLL 2003 [2], MUC 6 [3], MUC 7 [4] and ACE 2002 [5] training datasets.

Model 2 : Identifies four classes: (i) Location, (ii) Person, (iii) Organisation and (v) Misc; and was trained on the CoNLL 2003 training dataset.

Model 3 : Identifies seven classes: (i) Location, (ii) Person, (iii) Organisation, (iv) Money, (v) Percent, (vi) Date and (vii) Time; and was trained on the MUC 6 and MUC 7 datasets.

These NER models have all been successfully applied in identifying entities belonging to these standard classes. However, it cannot be expected that every Twitter domain

[2] https://www.clips.uantwerpen.be/conll2003/ner/
[3] https://www-nlpir.nist.gov/related_projects/muc/
[4] https://www-nlpir.nist.gov/related_projects/muc/
[5] https://www.nist.gov/speech/tests/ace

of interest will be subject to these standard classes. A bespoke Stanford NER model is therefore required that considers the entities within a given domain of discourse. For the evaluation presented later in this Chapter, Section 3.4, a "motor vehicle pollution" domain of discourse was considered. Stanford NER provides the ability to train a model so that the entities of interest can be captured; entities that belong to specific user-defined classes. For example, in the case of the motor vehicle pollution domain of discourse, it might be desirable for "petrol" and "diesel" entities to belonging to the class "Fuel_V". Some further explanation regarding the class "Fuel_V" is appropriate here. The class "Fuel_V" stands for Fuel Vehicle (for example, petrol, diesel and so on) and is used as an example throughout this thesis especially in the context of the car pollution domain Twitter dataset used as one of the evaluation datasets (see section 3.4).

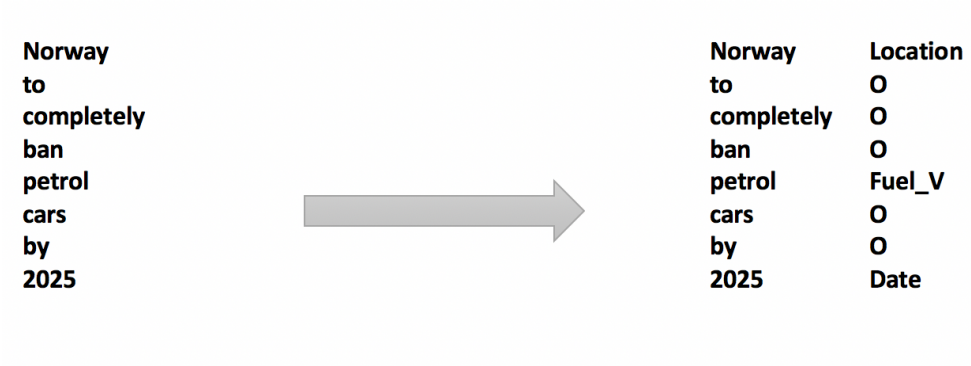

FIGURE 3.3: Example of a Stanford NER training data record

To generate a Stanford NER model an appropriately constructed training set is required; a training set where the entities of interest have been annotated with class labels. This needs to be presented in a prescribed format. Figure 3.3 presents an example training record. In the figure each row represents a word. On the left is an example tweet, "Norway to completely ban petrol cars by 2025". On the right the same tweet is given, but transposed into the syntactic format required by the NER tool. In the example the training record is intended for a Stanford NER model that can identify entities that belong to the classes: Location, Date and Fuel_V. To generate a training record of the syntactic form given in Figure 3.3, from a given tweet, a two-step process was followed:

- The tweet was first cleaned and then tokenised.
- Words that belonged to one of the classes of interest were then annotated with the appropriate class label; the annotation "O" was allocated otherwise.

The mechanism for selecting class labels is a matter for the end user and can be expected to be application dependent. However, for the purpose of generating RDFS from Twitter data, it is suggested here that this can best be done with reference to a lexical database such as WordNet [93] or an existing schema such as Schema.org

[104]. The reason being that this will readily support the construction of a hierarchical structure with respect to the generated RDFS that will include parent classes that are not specifically included in the input Twitter data. In the example NER training record given in Figure 3.3, the label "Location" indicates the class Location. The label "Fuel_V" indicates the class Fuel Vehicle and the label "Date" indicated the class Date. All tweets in the constructed training set would need to be of a similar format. The training set could then be used to generate a Stanford NER model.

For the proposed framework using Stanford CoreNLP the generated bespoke Stanford NER Model was used twice:

1. To support RE.
2. To map entities of interest to classes.

RE is discussed in the following subsection.

Relation Extraction

The third process in the workflow, presented in Figure 3.1, for the proposed Stanford CoreNLP RDF dataset from Twitter data Framework is RE. This process comprises the application of the Stanford RE tool [35] to extract the relations that exist between pairs of entities in the given collection of tweets; entities identified by the NER model described in the foregoing process. From the literature, the Stanford RE tool has been successfully utilised to extract relations from a range of different domains. For example, in [35] the Stanford RE tool was used to extract relations with respect to an American football domain.

The Stanford RE tool comprises a number of components. With respect to the RE for the proposed framework, the Entity Mention Detection (EMD) and the Relation Mention Detection (RMD) components were used. Both components are required to train the RE model. EMD is used to define entities in a sentence. For EMD the Stanford NER model described earlier in the Named Entity Recognition (NER) section was used. RMD was then used to define relations between entity mentions in the same sentence. The default Stanford RE is designed to detect four types of relation: (i) $Live\ In$ (ii) $Located\ In$ (iii) $OrgBased\ In$ (v) $Work\ For$. However, as in the case of the Stanford NER tool, the Stanford RE tool has the ability to be retrained to fit a particular domain of discourse. This retraining requires a bespoke training set (unless, of course, the Twitter domain of discourse of interest happens to fit with one of the defaults). The training set needs to be represented in a prescribed syntactic format (see for example [119] or [118]).

Figure 3.4 presents an example of a Stanford RE training record using the same tweet used for the NER training record example given in Figure 3.3: "Norway to completely ban petrol cars by 2025". The training record is expressed in the form of a table comprised of nine columns and a number of rows. The first eight rows in the example represent the tokens in the given Tweet. The columns represent various parameters

associated with the tweet. Column two gives the class for the entities included in the given tweet; "Location" and "Other" in the example. The "Other" class indicates that the associated entity belongs to a class that is relevant to the domain of discourse, but not the"Location" class. Column three gives the sequential number of each token in the tweet starting with the number 0. Column five gives the Part Of Speech (PoS) tag. Column six gives the content of the tweet. The other columns are labelled using the "O" character, because these columns are not required for the desired RE task; however, these columns are important for syntactic reasons. The adopted format is recommended by [35] where it was adapted from [119] and [118]. The last two rows, after an empty row, express the relations included in the training record. This is arranged in three columns. The first and second columns give the identifiers for the relevant entities, and the third the relation connecting them. Tweets can express one or more relations. In the example two relations are included, linking three entities: (i) the relation "ban" between entity 0 ("Norway") and entity 4 ("petrol") and (ii) the relation "ban Fuel_V Date" between entity 0 ("Norway") and entity 7 ("2025").

O	Location	0	O	NNP	Norway	O	O	O
O	O	1	O	TO	to	O	O	O
O	O	2	O	RB	completely	O	O	O
O	O	3	O	VB	ban	O	O	O
O	Other	4	O	NN	petrol	O	O	O
O	O	5	O	NNS	cars	O	O	O
O	O	6	O	IN	by	O	O	O
O	Other	7	O	CD	2025	O	O	O
0	4	ban						
0	7	ban_Fuel_V_Date						

FIGURE 3.4: Example of a Stanford RE training data record

The RE training set is then used to train a bespoke RE model. To gain confidence in the model its performance can be validated using a test set.

The output from the knowledge extraction is a set of triples of the form $\langle subject, predicate, object \rangle$, subject is the first entity, predicate is the relation between the two entities and the object is the second entity. Figure 3.5 presents the output that would be generated given the example tweet given in Figure 3.4. The way the Stanford RE model works means that additional relationships that are not relevant to the relations identified in the training set will also be identified. For the proposed framework, the output set of relation triples were filtered so that only those relations pertinent to the desired RDF dataset were retained. In other words, only the relations identified in the training set will be used to generate the RDF dataset.

3.3.3 RDF Dataset Generation

Many tools are recommended by W3C to generate an RDF dataset. Of these, at time of writing, Apache Jena was one of the more popular tools used to generate RDF datasets. Apache Jena is a free, open source, Java framework for developing Semantic Web and Linked Data applications [65]. Apache Jena also includes an API for

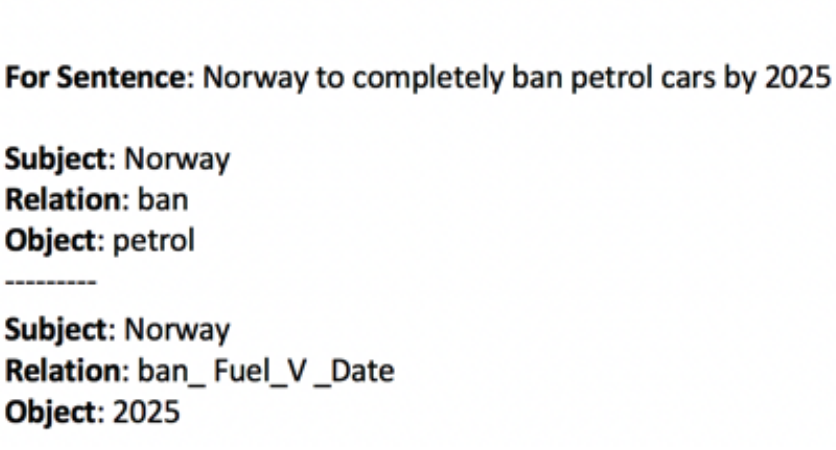

FIGURE 3.5: Example output using a Stanford RE model trained with respect to a car pollution domain of discourse

collecting data from and writing to RDF files. Apache Jena was applied to the triples ($\langle subject, predicate, object \rangle$), produced using the processes described above, to produce the desired RDF dataset.

As described in Chapter 2, the vocabulary "rdf:Description" begins the definition of the resource being described. The "rdf:about" refers to the subject while the tag "rdf:resource" refers to the object. "predicate" is the property that link the subject and the object. The triple ($\langle subject, predicate, object \rangle$) are defined using the notation <rdf:Description rdf:about="subject"> for the subject and <predicate rdf:resource= "object" / > for the predicate and the object. For example, we might have a triple, extracted from a tweet, of the form $\langle Norway, ban, petrol \rangle$. This triple will be presented in the RDF dataset as follows:

```
<rdf:Description rdf:about=‘‘http://example.com/Norway’’>
     <NS:ban rdf:resource=‘‘http://example.com/petrol’’/>
</rdf:Description>
```

Where "Norway" is the subject, "ban" is the predicate (property) and "petrol" is the object.

3.3.4 Class Mapping and Augmentation

The next step is the mapping of triples to an upper lexicon/schema to enrich the RDF dataset, which will allow the RDF dataset to be integrated into another dataset that has the same upper level "ontology". It was considered appropriate to map the entities to their class for inclusion in the final RDFS. The entity-class mapping was obtained from the Stanford NER model as trained with respect to a given domain of discourse.

For example, the entity *Norway* is mapped to the class *Location*, the entity *petrol* to the class *Fuel_V* and the entity 2025 to the class *Date*.

It was proposed that a class and property (relation) hierarchy could be constructed by using hypernyms extracted from an existing scheme repository or lexical database, such as Schema.org [57] or WordNet [93]. Schema.org is a collaborative, community-based vocabulary aimed at creating, maintaining and promoting schema for structured online data [104]. More than 10 million websites already use this to markup their web pages and email messages [104]. Schema.org covers topics such as people, places, events, products, offers and health [57]. WordNet is a lexical database of semantic relations between words (nouns, verbs, adjectives and adverbs). The relations include synonyms, hyponyms, and meronyms [93]. WordNet was constructed in 1986 by a research team at the University of Princeton, and continues to grow and develop [51]. Words are grouped into some 117,000 sets called synsets. These are interconnected by bidirectional arcs that express lexical (word-word) and semantic (synset-synset) relations, including hyper/hyponymys (for example tree and oak/oak and tree), meronymys (for example tree and branch), antonymys (for example long and short) and various entailment relations (such as buy and pay, show and see, and untie and tie) [89]. WordNet was mapped to the upper level ontology SUMO [103]. WordNet and Schema.org use a hierarchical concept model in which more specific concepts inherit information from more general concepts.

Therefore, for the proposed Stanford CoreNLP RDF dataset from Twitter data framework, the RDFS was constructed using the obtained properties and classes augmented with the hypernyms of the identified classes (super-class) and the hypernyms of the identified properties (super-properties) obtained from WordNet. Some examples are given in Figure 3.6 for the classes “Location” and “Data”, and the relation “ban”. The hypernyms in each case are expressed as lists of decreasing specificity.

Location
[‘object ’, ‘physicalentity’, ‘entity’]

Date
[‘day ’, ‘timeunit’, ‘measure ’, ‘abstraction’, ‘entity’]

ban
[‘outlaw’, ‘forbid’, ‘command’, ‘order’, ‘request’, ‘ask’, ‘request’, ‘communicate’, ‘convey’, ‘transfer’, ‘move’]

FIGURE 3.6: Example hypernyms for the “Location” and “Data” classes and the relation “ban”.

Returning to the example considered previously, the identified classes and properties would each be associated with their hypernyms. For example, using WordNet, the class “Location” would have a super-class (hypernym) “object” which has a super-class “physicalentity”, which has a super-class “entity”. Similarly, the class “Date” would have the super-class “day”, which has a super-class “timeunit”. Each sequence will

eventually reach the broadest class in WordNet, the class "entity" (the "Thing" class if using Schema.org). The same process can be applied to properties. For example, using WordNet, the property "ban" would have the super-property (hypernym) "outlaw", which has a super-property "forbid", and so on until the sequence reaches the super-property "move". A special case should be mentioned here, that is the case where a class or a property is expressed using more than one word. In this case it will not be possible to find the label in WordNet because WordNet, in most cases, does not include multi-word expressions. With respect to the proposed framework the adopted solution was that, if a class or a property comprised a multi-word expression, such as the property "ban Fuel_V Date", the multi-word expression should be tokenised and the first noun or verb word used as the label for the class or relation under consideration. The intuition here was that the first noun (verb) in a multi-word class (property) expression will usually be the most relevant.

3.3.5 RDF Schema Generation

This section presents the RDFS generation process. As discussed in Chapter 2, there are two popular ontology specification languages: (i) the Web Ontology language (OWL) [90], and (ii) the Resource Description Framework Schema (RDFS) [87]. With respect to the proposed framework RDFS was adopted, for reasons discussed earlier. Apache Jena generates an RDFS model corresponding to the given classes and properties.

Once a set of classes, properties and class/property hypernyms have been obtained the RDFS vocabulary were used to express the class hierarchies and associated properties. With respect to the proposed framework the vocabularies "rdfs:Class", "rdf:Property", "rdfs:domain", "rdfs:range", "rdfs:subPropertyOf" and "rdfs:subClassOf" were used. The "rdfs:Class" and "rdf:Property" constructs were used to define classes and properties. "rdfs: domain" was used to define the domain of a property, the classes (resources) that can be used as the subject of a predicate; whilst the "rdfs:range" was utilised to define the range of a property, the classes (resources) that can be used as the object of a predicate. "rdfs:subClassOf" and "rdfs:subPropertyOf" were used to define sub-class relationships and sub-property relationships, according to identified hyponyms.

3.4 Evaluation

The previous sections presented details of the proposed Stanford CoreNLP RDF dataset from Twitter data framework, the first of the four frameworks considered in this thesis. This section presents the evaluation of this proposed framework. For the purpose of the evaluation a specific Twitter dataset was collected related to the motor vehicle pollution domain of discourse. Further details concerning this datasets is presented next in Subsection 3.4.1. The objective of the evaluation is:

- To determine the effectiveness of the RDF dataset generation process by evaluating the Stanford NER and RE models generation processes.

The evaluation results with respect each of these objective are presented below in Sub-sections 3.4.2 and 3.4.3. In sub-section 3.4.4, example of the generated RDFS was presented.

3.4.1 Datasets

This sub-section presents the motor vehicle pollution Twitter evaluation dataset. The motor vehicle pollution domain was used because it represented a topic that was easy to understand, and therefore any RDF dataset generation method utilising the dataset could easily be manually evaluated. The criteria for collecting tweets to be included in the motor vehicle pollution dataset was that a tweet had to mention issues related to banning fuel vehicles. or encouraging the concept of using environment friendly vehicles. in a particular location using #pollution and #climatechange. A total of 300 motor vehicle pollution tweets were collected and hand labelled.

Analysis of the motor vehicle pollution dataset showed that four entity classes were contained in the tweets: (i) Location, (ii) Fuel_V , (iii) Environment_friendly_V, and (iv) Date; and four relations labels: (i) ban, (ii) use, (iii) ban_Fuel_V_Date, and (iv) use_Environment_friendly_V_Date. Figures 3.7 and 3.8 illustrate the distribution of the instances of the classes and relations across the dataset; 768, 384, 198 and 1162 for the classes; and 241, 125, 166 and 87 for the relations. The Figures show that there were more examples of entities than relations. This was to be expected. An average tweet comprised eight entities and two relations; not all entities were paired.

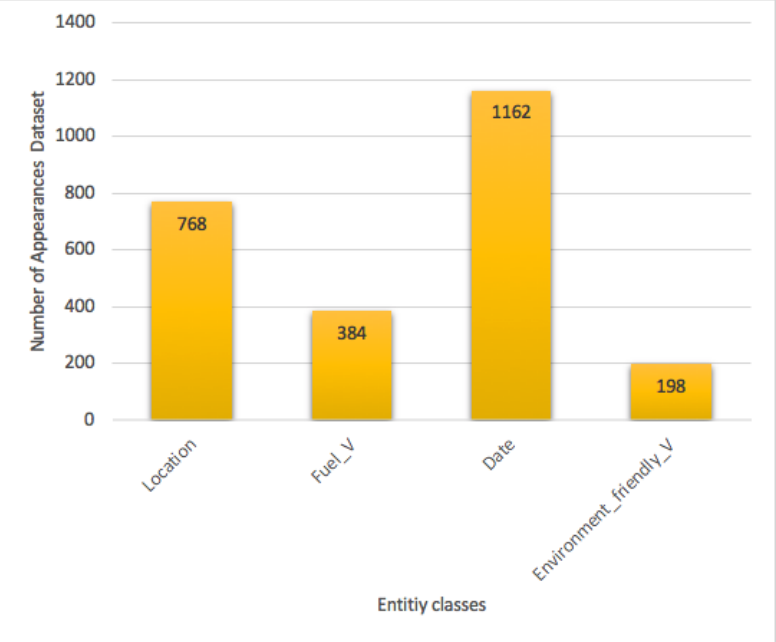

FIGURE 3.7: Distribution of the entities per class across the motor vehicle pollution dataset

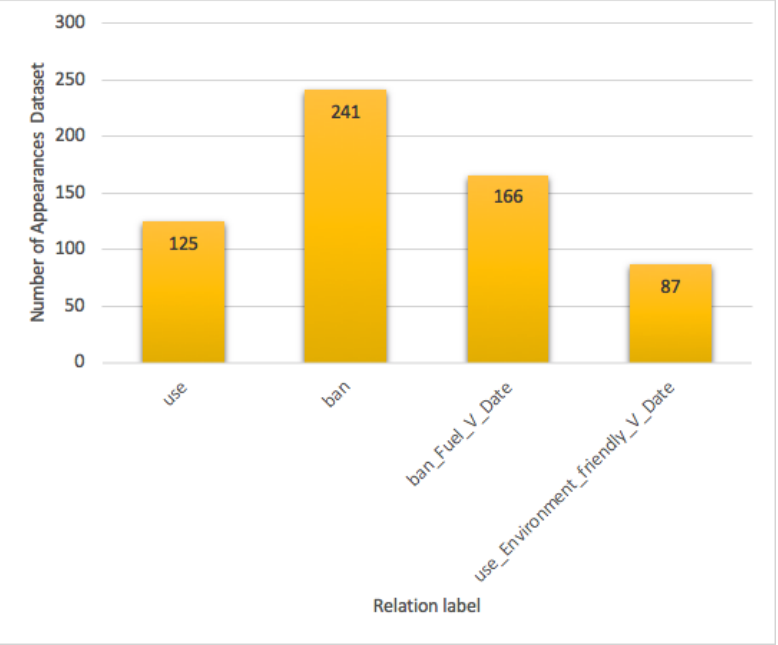

FIGURE 3.8: Distribution of the relations per class across the motor vehicle pollution dataset.

A further 311 unlabelled tweets from the motor vehicle pollution domain were used to populate the generated RDFS and query it using predefined queries as presented in Chapter 7.

3.4.2 Named Entity Recognition (NER) Evaluation

The effectiveness of the Stanford NER model generation process was determined in terms of the accuracy of the model. For this purpose, Ten-fold Cross Validation (TCV) was used with respect to the 300 labelled motor vehicle pollution domain tweets. Using TCV the dataset is split into 10 independent folds. All but one of these fold are utilised to train the model, while the remaining tenth is utilised to test the model [4]. We use Training data to learn a prediction/classification model, and Test data to determine the performance of the model. The two should be disjoint. It is generally acknowledged that the larger the training data set the better the learning process. So we want a data split that maximises the learning but retains a sufficient range of examples for testing. What a sufficient number of examples is depends on the nature of the data, more specifically: the number of classes, number of attributes and the range of values for those attributes. We make the assumption that if we select a range of examples from the data we have, we will have a sufficiently representative test set. Accuracy metrics derived from a small number of test examples are not sufficiently rigorous. Generally speaking, a few hundred test examples is seen as sufficient. However, In many cases, we do not have the luxury of a few hundred test examples so we use cross-validation instead. Ten Cross-Validation (TCV) is the most commonly used in the literature [3, 35, 77, 136, 148], because: (i) ten rounds of training is seen as robust, and (ii) ten simplifies the maths. We used ten

because we believed this maximised the training whilst at the same time provided for robust testing, even though we only have 300 records. There is evidence in the literature where TCV has been used for smaller numbers of Tweets, for example in [3] only 290 Tweets were used. However, where fewer than 300 records were available, for example only 100 as will be presented in Chapter 5, TCV was considered unsuitable because there would only be 10 records in the test set for each run and this was deemed insufficiently robust. Therefore, in the case where only 100 tweets were available Three-fold Cross Validation was used. The metric used was F1-score, calculated as follows:

$$F1 - score = 2 \; . \; \frac{Precision \; . \; Recall}{Precision + Recall}$$

$$Precision = \frac{Number \; of \; correct \; entities}{Number \; of \; entities \; retrieved}$$

$$Recall = \frac{Number \; of \; correct \; entities}{Number \; of \; entities \; that \; exist \; in \; gold \; standard}$$

The F1-score metric is commonly used in the literature for evaluating information extraction models such as NER, and RE models. Some examples where F1-score has been used for NER and RE can be found in [3, 35, 77, 81, 136, 148]. The results are presented in Table 3.1, which shows the F1-score evaluation of the NER model for the motor vehicle pollution domain dataset using TCV. Note that the individual recall and precision values are not included in the table because the Stanford NER tool does not output this information. The average F1-score was 78.0% obtained with a Standard Deviation (SD) of 0.71. The low SD indicated a consistent performance. The F1-scores obtained were deemed to be sufficient to provide a bench-mark with which alternative approaches could be compared..

3.4.3 Relation Extraction Evaluation

The effectiveness of the generated RE model, as in the case of the NER model, was also determined using TCV. The metrics used in this case were Precision, Recall and F1-score calculated as presented in sub-section 3.4.2. The results are presented in Table 3.2. The average Precision, Recall and F1-scores were 77.74%, 82.26% and 79.56% respectively, with associated SD values of 13.5, 9.2 and 10.9. The high SD values indicate a high degree of variability which was not a feature of the NER model. It was found that the reasons for this were: (i) the relatively small number of relations in the dataset compared to the number of entities (619 versus 2512) and (ii) the imbalanced of the different relations

Table 3.1: TCV results for the Stanford NER model evaluation of the Stanford CoreNLP Framework

Fold Number	F1-score
Fold 1	77.3%
Fold 2	77.7%
Fold 3	77.2%
Fold 4	79.3%
Fold 5	77.2%
Fold 6	78.0%
Fold 7	78.0%
Fold 8	78.9%
Fold 9	78.4%
Fold 10	78.3%
Average	78.0%
Standard Deviation (SD)	0.71

in the data. Fold 5 was the worst performing fold. The reason for this was that the precision and recall of the relation "use_Environment_friendly_V_Date" was 5.6% and 22.2% respectively, which had a significant impact on the overall precision, recall and F-score. The reason for this was because the relation "use_Environment_friendly_V_Date" featured 87 times in the dataset, approximately nine times per fold. In comparison the precision and recall scores of the relation "ban" in the same fold, which appeared 241 times in the dataset, were 77.1% and 79.4%, respectively. Overall, the model presented good average scores. Details are included in Appendix A.

Table 3.2: TCV results for the Stanford RE model evaluation of the Stanford CoreNLP Framework

Fold Number	Precision	Recall	F-score
Fold 1	67.6%	88.5%	76.7%
Fold 2	81.8%	88.5%	85.0%
Fold 3	81.6%	75.5%	78.4%
Fold 4	94.2%	98.0%	96.1%
Fold 5	47.2%	69.0%	56.1%
Fold 6	85.2%	74.2%	79.3%
Fold 7	74.3%	76.4%	75.3%
Fold 8	70.2%	75.5%	72.7%
Fold 9	88.6%	87.3%	87.9%
Fold 10	86.7%	89.7%	88.1%
Average	77.74%	82.26%	79.56%
Standard Deviation (SD)	13.5	9.2	10.9

3.4.4 RDF Schema Generation Using Motor Vehicle Pollution Dataset

To get a better understanding of the nature of the RDFS generation process, and the hierarchy construction mechanism, incorporated into the framework, the framework was

used to generate a motor vehicle pollution RDFS dataset.

The RDFS generated using the proposed framework is presented in graph form in Figure 3.9. The RDFS was generated using Apache Jena as detailed in Section 3.3.5. WordNet was used to construct the RDFS hierarchy using class and property hypernyms. From Figure 3.9 it can be seen that the RDFS comprised four classes: (i) Location, (ii) Fuel_V, (iii) Environment_friendly_V, and (v) Date. From the figure it can also be seen that the four classes were related using four properties: (i) ban, (ii) use, (iii) use_Environment_friendly_V_Date, and (iv) ban_Fuel_V_Date. The "Location" class featured relationships with the classes "Fuel_V" and "Environment_friendly_V", namely "ban" and "use", respectively. The "Location" class also featured "ban Fuel_V_Date" and "use Environment_friendly_V_Date" relationships with the "Date" class.

As noted above, WordNet was used to build the hierarchical relations using the hypernyms of the classes and properties as super-class and super-properties (super-properties are not shown in the Figure). From Figure 3.9 the "Location" class had three super-classes: "object", "physicalentity" and "entity". The "Fuel_V" class has four super-classes: "substance", "matter", "physicalentity" and "entity" and so on. The property "ban" had ten super-properties "outlaw", "forbid", "command", "order", "request", "ask", "communication", "convey", "transfer" and "move". The property "use" did not have any hypernyms, and thus no super-properties. For properties described using multi-word phrases, such as "ban_Fuel_V_Date", the first verb within the property phrase was considered, and WordNet searched for hypernyms of this verb. For example, the first verb in the property "ban_Fuel_V_Date" is "ban". The super-properties for the verb "ban" was presented above. To give another example, the first verb in the property "use_Environment_friendly_V_Date" is "use". The property use does not have any super-properties, thus the property "use_Environment_
friendly_V_Date" does not have any super-properties. If a class or property does not exist in WordNet, as opposed to not having any listed hypernyms, the class or property will not have any super-classes (properties).

The generated RDFS was populated and queried using pre-designed queries with known responses. Example of such queries will be presented later in Chapter 7.

3.5 Summary

In this chapter, the first of four proposed RDF dataset rom Twitter data Frameworks, the Stanford CoreNLP framework, has been presented. This framework features a workflow comprised of five processes: (i) Tweet cleaning (ii) NER (iii) RE, (iv) RDF generation (v) Class mapping and augmentation, and (vi) RDFS generation. For the NER and RE, tools within the Stanford CoreNLP Toolkit were used. These operate using a machine learning supervised approach. By default the tools operate using a generic predefined set of classes and relations. It was anticipated that these would be unsuitable for most Twitter domains of discourse so both models required retraining. Once the

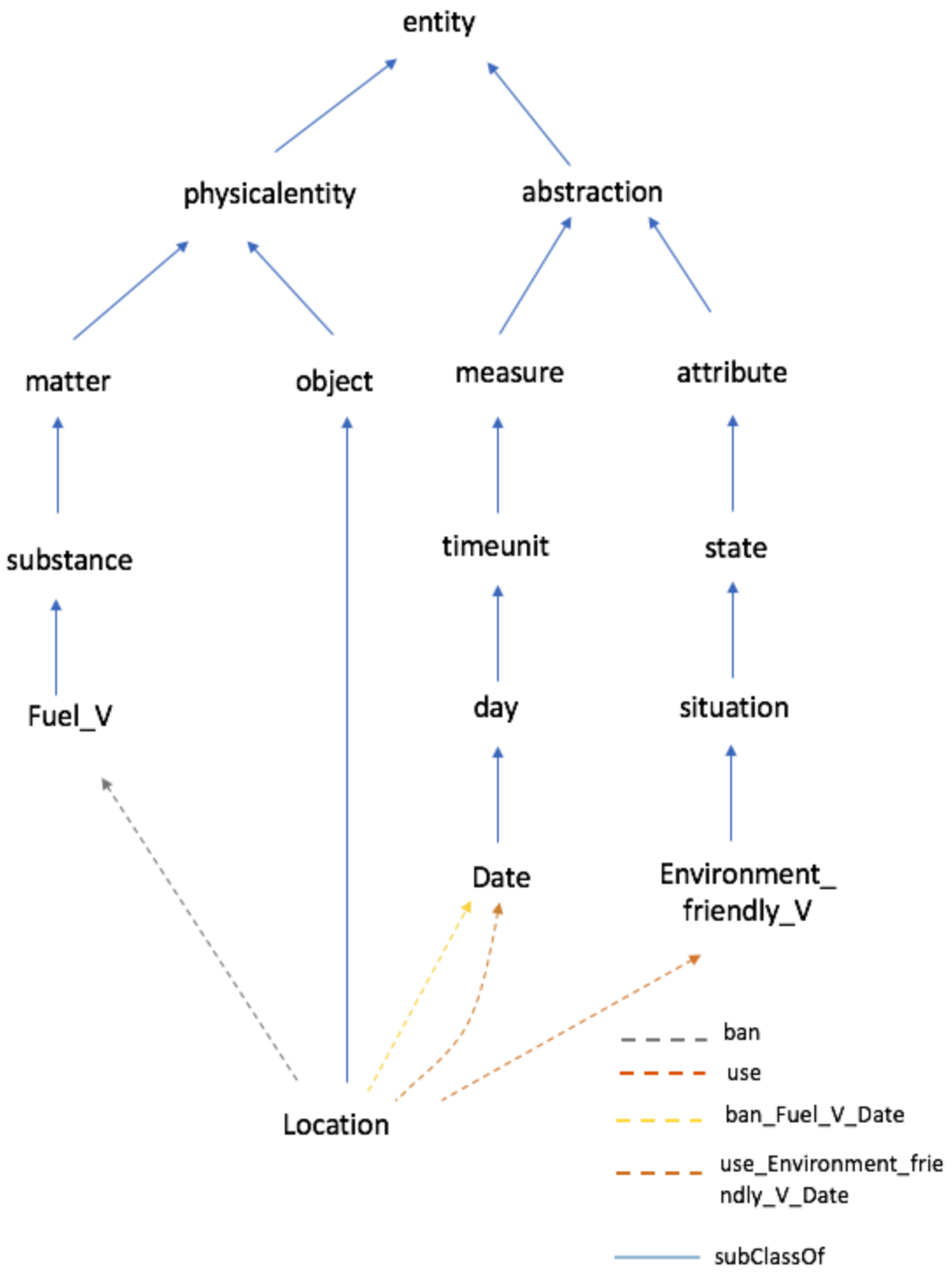

FIGURE 3.9: Node and link diagram illustrating the motor vehicle pollution RDFS, generated using the Stanford framework and extended using WordNet

relevant entities and relations have been extracted and stored as triples, they were used to generate an RDF dataset. To enrich the RDF dataset, the entities were mapped onto classes. Each class and relation (property) was then augmented with a set of hypernyms to form a class and property hierarchy. It was suggested that Wordnet be used for this

purpose; although alternatives, such a schema.org, do exist. The final stage was the RDFS generation stage, it was recommended that this was done using Apache Jena to generate the desired RDFS file. An evaluation of the proposed framework was also presented. It was found that the framework was suitably effective considering that this was a first benchmark framework. The next chapter describes the second proposed RDF dataset from Twitter data framework, the GATE framework.

Chapter 4

The GATE RDF Dataset From Twitter Data Framework

4.1 Introduction

In Chapter 3 a first, benchmark, RDF dataset from Twitter Data framework was presented, the Stanford CoreNLP framework. This was categorised as a hybrid approach because it featured both machine learning and linguistic techniques that were combined to generate a RDF dataset from a given collection of Twitter data. Stanford CoreNLP [84] was selected for this first framework because of its popularity and its many claimed advantageous. However, it was conjectured that the same fundamental ideas as incorporated into the Stanford CoreNLP framework could be used to explore the advantages, with respect to NER and RE, that might be offered using alternative NLP toolkits such as IBM's Unstructured Information Management Architecture (UIMA) [52], the Python NLTK (Natural Language ToolKit) platform [17] or the General Architecture for Text Engineering (GATE) toolkit [38]. The work presented in this chapter sought to explore this idea in further detail by investigating RDF dataset generation from Twitter data with respect to the GATE NLP toolkit. GATE was selected for this purpose because it was another well established toolkit, first proposed in 1995, whereas UIMA and NLTK were more recent. The main two distinctions between the Stanford CoreNLP toolkit and the GATE toolkit is that in the latter NER is achieved using a "gazetteer" and RE using a bespoke GATE model.

This chapter thus presents the GATE RDF dataset from Twitter data framework. At the same time, as in the case of the work presented in the previous chapter, the work presented in the chapter is directed at deriving an answer to Subsidiary Question One from Chapter 1:

1. *Given that the central building blocks of RDF dataset are entities and relations can NER and RE be applied using the standard supervised learning tools and techniques available for natural language processing?*

The rest of this chapter is structured as follows. Section 4.2 provides an introduction to the NLP GATE toolkit. Section 4.3 presents the proposed GATE RDF dataset from Twitter data framework. The section is divided into four sub-sections, each describing one of the core processes that make up the overall proposed framework: (i) Named Entity Recognition (NER), (ii) Relation Extraction (RE), (iii) RDF generation and (iv) RDFS generation and Augmentation. Section 4.4 presents the evaluation of the proposed GATE framework for RDF dataset from Twitter data framework. The chapter is concluded with a summary of the framework in Section 4.5. The comparative evaluation of the framework with the Stanford CoreNLP framework is left to Chapter 7.

4.2 General Architecture for Text Engineering

In this section an overview is provided of the GATE NLP toolkit so as to facilitate the reader understanding with respect to the remaining material in this chapter. GATE is described as an architecture to build and distribute software systems directed at applications designed to process human language. The first version of GATE was released in the middle of the 1990s; thus, it is well established. GATE is essentially a toolkit comprised of a set of tools that can be pipelined in various application dependent ways. The following were of significance with respect to the work presented in this chapter:

1. The gazetteer packaged with the A Nearly-New Information Extraction (ANNIE) system, which was used for NER with respect to the proposed framework.
2. A number of machine learning resources, both batch learning and machine learning resources. Batch learning was used for RE with respect to the proposed framework.
3. Verb and noun group chunkers and the Part of Speech (POS) Tagger.
4. A pattern matcher called Jape (Java Annotation Patterns Engine).

In common parlance the term "gazetteer" refers to a domain specific dictionary or lexicon, for example a list of street names in a city. In the GATE context the term "gazetteer" describes:

1. A set of lists of entity names, referred to as *entity lists* or *entity dictionaries*, one list per class to be considered, and
2. The software system used to find the entity names in a given text corpus and label the identified entities with their associated class names.

ANNIE is an exemplar information extraction system distributed with GATE and founded on the Java Annotation Patterns Engine (JAPE) language and a finite state algorithm [39]. ANNIE includes: (i) a tokeniser, (ii) a gazetteer, (iii) a sentence splitter, (iv) a POS tagger, and (v) a semantic tagger that can be used to annotate free text using

JAPE grammar rules. It is a foundation system that helps to extract entities and relations from text. ANNIE includes components for pre-processing text that was used with respect to the proposed framework to pre-process tweets and identified entities (as described in sub-section 4.3.1), and to pre-process tweets before using machine learning resources for training a RE model (as described in sub-section 4.3.2).

For many applications, including RDF dataset generation from Twitter data, the existing entity lists that are provided with the ANNIE gazetteer are not sufficient. GATE therefore includes a facility to allow users to generate additional entity lists. This facility was utilised with respect to the framework proposed in this chapter.

To run a GATE application, a configuration file is required. A GATE configuration file specifies the parameters, expressed as XML, needed by the application. For example, it might be necessary to specify a learning algorithm, such as Support Vector Machine (SVM), or a validation approach, such as Ten Cross-fold Validation (TCV). Given a specific GATE application the associated configuration file will need to be edited.

More detail concerning how the above elements were incorporated into the proposed framework are presented in the next section.

4.3 The GATE RDF Dataset From Twitter Data Framework

This section provides a detailed description of the proposed GATE RDF dataset from Twitter data framework. A schematic of the proposed framework is given in Figure 4.1. From the figure it can be seen that the workflow starts with a collection of tweets describing a domain of discourse. There are then four processes that are sequentially applied: (i) Tweet cleaning, (ii) NER, (iii) RE and (iv) RDF dataset generation As before, the NER and RE processes are bundled together under the heading of Knowledge Extraction.

The tweet cleaning process was identical to that used for the Stanford CoreNLP framework described in the previous chapter; removal of URLs, user names, hash symbols and punctuation marks. The tweet cleaning process is therefore not discussed further in this chapter. NER is discussed in Sub-section 4.3.1 and RE in Sub-section 4.3.2. The RDF dataset generation process is then considered in Sub-section 4.3.3. The RDFS generation and augmentation process is discussed in Sub-section 4.3.4.

4.3.1 Named Entity Recognition

As in the case of the Stanford CoreNLP framework the start point for the GATE framework is the identification of entities to be included in the triples that will then be used to define an appropriate RDF dataset As already noted, GATE uses a gazetteer that includes *entity lists* (also sometimes referred to as entity dictionaries) to identify entities. A given gazetteer will comprise one or more entity lists (each associated with a class) and the software to utilise such lists [132]. Typical entity lists comprise names

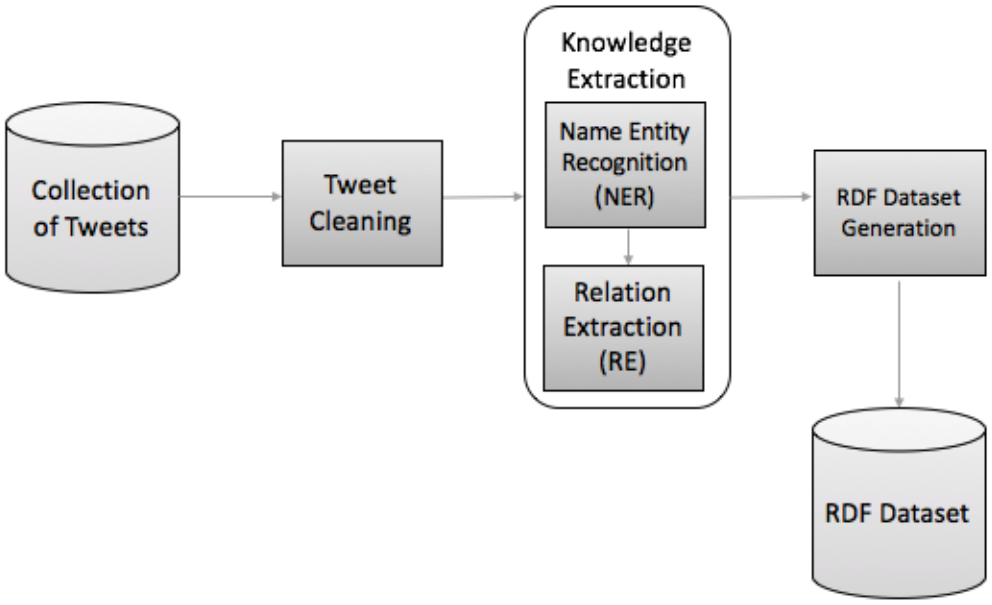

FIGURE 4.1: Schematic of the GATE RDF Dataset from Twitter Data Framework

of people, organizations, days of the week, units of currency and so on. The ANNIE gazetteer comes with a number of default entity lists.

Figure 4.2 presents an example of the ANNIE gazetteer entity list interface. From the figure it can be seen that the interface comprises two panels. The left-hand panel shows the available entity lists, the right hand panel the contents of a selected entity list. In the figure the "country code" list has been selected (highlighted). Each entity list is described in terms of a tuple of the form:

$$\langle listName : majorType : minorType : language : annotationType \rangle$$

where: (i) $listName$ is the entity lists name, (ii) $majorType$ is the primary tag, (iii) $minorType$ is the secondary tag, (iv) $language$ is the language of the entity list (GATE supports a number of commonly spoken language) and (v) $annotationType$ is the annotation tag. The major and minor types facilitate the housekeeping and administration of entity lists (the minor type is optional). The $annotationType$ defines the tag that will be used when the entity list is used to annotate free text (tweets). In Figure 4.2 the left-hand panel only shows the name and major type of each entity list.

As noted above, the ANNIE gazetteer, by default, cannot be expected to identify and label every entity belonging to every class of interest in a given collection of tweets drawn from some random domain of discourse. For the purpose of RDF dataset generation there is therefore a need to be able to add bespoke entity lists in addition to those that come with ANNIE. For example, regarding the pollution domain of discourse used for evaluation purposes with respect to the work presented in this thesis (see Section 4.4) a class "Fuel_V", which included entities such as "petrol" and "diesel", was required.

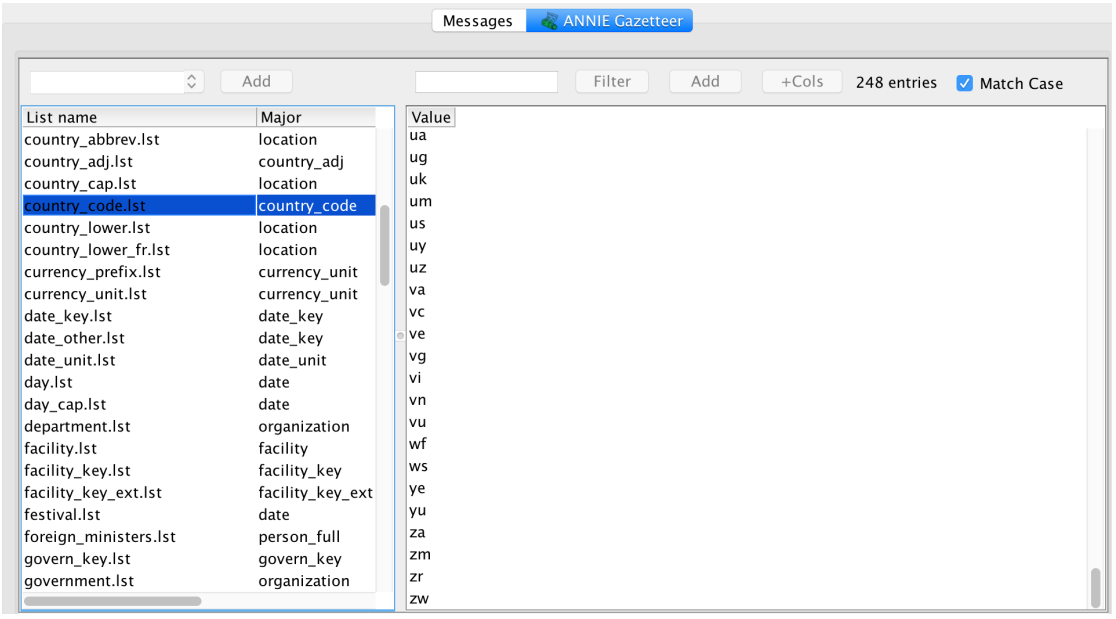

FIGURE 4.2: ANNIE gazetteer entity list interface

GATE provides two mechanisms whereby additional entity lists can be added to the ones already available within a given gazetteer:

1. The hand-crafting of additional entity lists using an external text editor, independent of GATE, and then importing the constructed entity list files into the gazetteer.

2. Construction of additional entity lists using the ANNIE gazetteer entity list interface (Figure 4.2).

The first entails a two-step process:

1. Create a collection of entity list files using a text editor, each listing the entities that belong to the additional classes to be considered, one file per class.

2. Add the entity lists generated in Step 1 to the *definitions file*, a file used by GATE that includes all predefined entity list names plus any user defined entity list names. The definitions file is named `lists.def`. The required syntax for adding a newly created entity list to the *definitions file* is the tuple format described above

For example we might have:

$$\langle day.lst : date : day : en : DAY \rangle$$

where "day.lst" is the list name, "date" is the major type, "day" is the minor type, "en" (English) is the language and "DAY" is the annotation type.

The second mechanism for adding a user defined entity list to the ANNIE gazetteer is by using the ANNIE gazetteer entity list interface shown in Figure 4.2. The Figure

shows two text boxes at the top, which can be used to create a row in a new entity lists or add rows to an existing entity list. This is a very laborious way of creating an entity list, it is therefore much more convenient to use the first mechanism. The second mechanism is better used for adding rows to the existing entity lists or editing entity lists.

One reason for editing entity lists is to change the annotation type. By default the annotation type will be "Lookup". This is used with respect to many of the predefined entity lists that come with GATE. However, for RDFS generation, as presented later in this Chapter, it is best to use a more descriptive annotation name (class name); "Lookup" will not provide any information regarding the entities when an RDFS is generated. To change the annotation type to a more appropriate name that describes the entities in a list, the user can use the ANNIE gazetteer entity list interface. However, it is more efficient to use an appropriately defined set of JAPE grammar rules. JAPE is an annotation patterns engine written in Java [39]. A JAPE grammar consists of what are referred to as "phases". Each phase consists of one or more rules. Jape rules are of the form:

$$\langle CONDITION \rangle \rightarrow \langle ACTION \rangle$$

where the Left-Hand-Side (LHS) expresses some annotation pattern description that must be met for the annotation manipulation described on the Right-Hand-Side (RHS) to be performed.

Some example JAPE code to change an annotation type from "Lookup" to something more informative is given in Listing 4.1. The listing starts, line 1, by declaring the name of the phase, "locationCode" in this case. Next, line 2, the annotations against which the rule will be matched are defined, in this case the annotation "Lookup" is given. Next, line 3, the control method for the rule matching to be employed is expressed. In the example the control method is "appelt", which means that only one rule can be fired for the same region of text (in the example, for simplicity, only one rule is given). Line 5 then gives the name of the rule, "TestLocation". The actual rule is then presented in lines 6 to 10. The LHS (lines 6, 7 and 8) defines a pattern where the "Lookup.majortype" is set to "location" and states that each identified instance that subscribes to this pattern is to be temporarily named using the label "match". Whenever a match is found this triggers the RHS of the rule, which in this case (lines 9 and 10) states that the temporary label "match", from line 7, is to be changed to "Location". For debugging purposes the right-hand-side also gives the name of the rule that was fired.

```
1.      Phase: locationCode
2.      Input: Lookup
3.      Options: control = appelt
4.
5.      Rule: TestLocation
6.      (
```

```
7.        {Lookup.majorType == location}
8.  ):match
9.  -->
10.  :match.Location = {rule=TestLocation}
```

LISTING 4.1: JAPE code example to change the annotation type of an entity list from "Lookup" to "Location"

Given an ANNIE gazetteer that includes all the required entity lists, this can be used to identify entities within a given text, tagging them as dictated by the annotation type attribute value assigned to each entity list. Figure 4.4 presents an annotated tweet as it appears in the ANNIE gazetteer NER interface. As in the case of the ANNIE gazetteer entity list interface, the NER interface comprises two panels. The left-hand panel shows the annotated text of interest. The right-hand panel lists the entity lists to be used plus some alternative annotation options. In the example the left-hand panel features the tweet "Norway to completely ban petrol cars by 2025" (used previously for illustrative purposes). The right-hand panel in the example shows four entity lists (Date, Fuel_V, Location and Lookup) and four alternative annotation options. In the example the entity lists Date and Location are predefined, the entity list Fuel_V is a bespoke entity list, and the Lookup entity list is a default list. In the example the first three options have been selected. The significance of the "Relationinstance" annotation option listed in Figure 4.4 will be clarified in Sub-section 4.3.2. The rest of the annotation options given in Figure 4.4 have the obvious meaning. Inspection of the figure indicates that: the entity "Norway" is labeled as belonging to the class "Location" because "Norway" is an entity in the "Location" entity list, the entity "petrol" is labeled as belonging to the class "Fuel_V" because "petrol" is an entity in the "Fuel_V" entity list, and the entity "2025" is labeled as belonging to the class "Date" because "2025" is an entity in the "Date" entity list.

When an ANNIE gazetteer is applied to a text example each character is given a sequential start and end character IDs, $\langle c1_s, c1_e \rangle$. Thus, given a tweet T this can be described as a list of characters of the form $[\langle c1_s, c1_e \rangle, \langle c2_s, c2_e \rangle, \ldots]$. A word can thus be described in terms of the start character ID of the first character and the end character ID of the last character $\langle c1_s, cn_e \rangle$. This numbering mechanism is illustrated in Figure 4.3, again using the tweet "Norway to completely ban petrol cars by 2025". Inspection of Figure 4.3 indicates that the word "Norway" is delimited using character IDs $\langle 0, 6 \rangle$, which means that the "N" character is delimited using character IDs $\langle 0, 1 \rangle$, the "o" character is delimited using character IDs $\langle 1, 2 \rangle$ and so on. GATE assigns a unique Entity ID, e_{id}, to each identified entity that in turn references a sequence of characters in T. Therefore, identified entities are stored using the form:

$$\langle e_{id}, e_{annotationType}, i_{start}, i_{end} \rangle$$

Where e_{id} is the unique ID for the entity, $e_{annotationType}$ is the class name for the entity, and i_{start} and i_{end} are the start and end character indexes for the entity.

```
<!-- The document content area with serialized nodes -->

<TextWithNodes><Node id="0" />Norway<Node id="6" /> <Node id="7" />to<Node id="9" />
<Node id="10" />completely<Node id="20" /> <Node id="21" />ban<Node id="24" />
<Node id="25" />petrol<Node id="31" /> <Node id="32" />cars<Node id="36" />
<Node id="37" />by<Node id="39" /> <Node id="40" />2025<Node id="44" />
</TextWithNodes>
```

FIGURE 4.3: Example of the sequential indexing used by GATE to identify characters in free text (tweets)

Using the ANNIE gazetteer, as described above, a given collection of tweets is processed so that the entities of interest are identified. Within the context of the proposed framework, these are stored temporarily (in an XML file format) ready for use within the following process, the RE process. This follow-on process is discussed in the next sub-section.

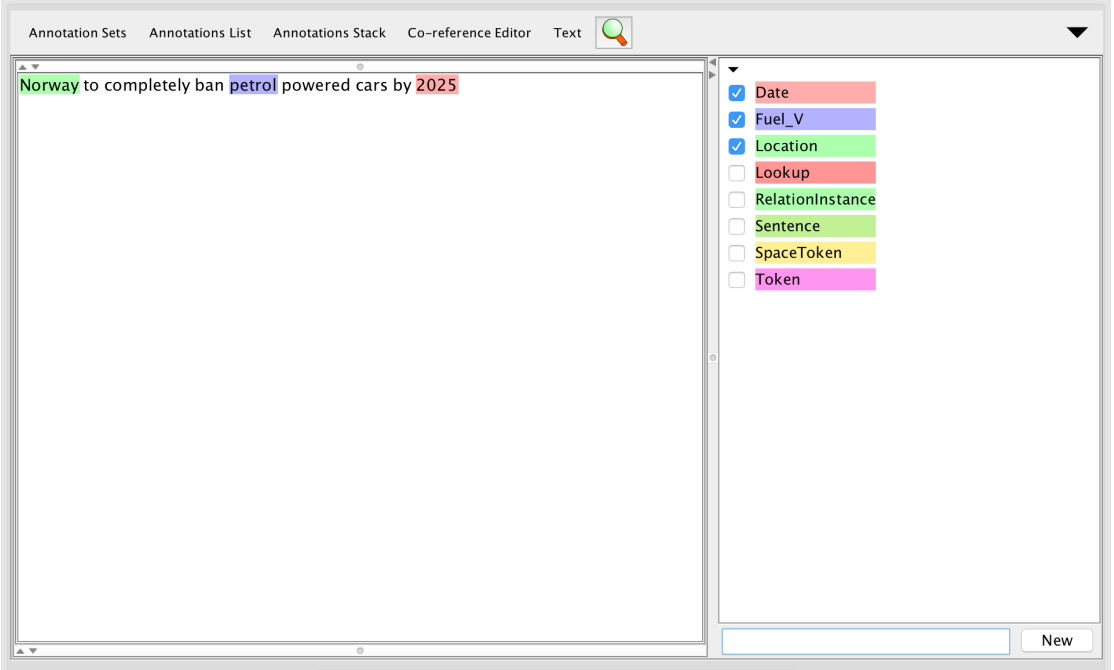

FIGURE 4.4: ANNIE gazetteer NER interface

4.3.2 Relation Extraction

As in the case of the Stanford CoreNLP framework, the third process of the GATE RDF dataset from Twitter data framework, outlined in Figure 4.1, is RE. Again, the RE model is created using a supervised learning process. To do this using GATE, a three steps process is followed: (i) paring of the instances of the classes of interest that were identified using the NER facilitated by the ANNIE gazetteer, (ii) producing a training

set which is then used to train the RE model and (iii) creating an appropriate *RE configuration file*. All three of these steps will be presented in detail below.

Paring classes: The proposed mechanism for the pairing of entities that feature in a tweet is founded on the use of appropriately defined JAPE grammar rules. The JAPE grammar takes, as input, tweets that have been annotated using the ANNIE gazetteer as described in Sub-section 4.3.1. Given a set of annotated tweets $T = \{t_1, t_2, \dots\}$ the JAPE grammar rules are applied to each tweet $t_i \in T$ in turn and zero, one or more entity pairs extracted. Conceptually each identified entity pair, e_1 and e_2, is defined as follows:

$$\langle annotationType, e1_{annotationType}, e1_{id}, e2_{annotationType}, e2_{id}, i_{start}, i_{end} \rangle$$

where: $annotationType$ is the annotation tag, $e1_{annotationType}$ and $e2_{annotationType}$ are the classes name of the entities, $e1_{id}$ and $e2_{id}$ are unique IDs for the entities e_1 and e_2, and i_{start} and i_{end} are the start character index for entity e_1 and the end character index for entity e_2 respectively. Thus, using the example tweet "Norway to completely ban petrol cars by 2025" used earlier, this features instances of the classes "Location", "Fuel_V" and "Date". Hence, the Jape grammar would extract two entity pairs. The first of these pairs would be:

$$\langle RelationInstance, Location, 513, Fuel\_V, 512, 0, 31 \rangle$$

"RelationInstance" is the annotation tag of the class pair, 512 and 513 are the unique identifiers for the instances Norway (belonging to the class "Location") and petrol (belonging to the class "Fuel_V"), 0 is the start character index for the entity Norway (see Figure 4.3) and 31 is the end character index for the entity petrol (see Figure 4.3) with respect to the first entity pair. The second entity pair would be:

$$\langle RelationInstance, Location, 513, Date, 511, 0, 44 \rangle$$

Training set generation: Training set generation involves annotating the input document (Twitter) collection in terms of the entry pairs identified previously in Step 1. For each identified entity pair, the user needs to manually assigning relation labels. An example GATE RE training record is presented in Figure 4.5. As in the case of the previous examples, the example is based on the tweet "Norway to completely ban petrol cars by 2025" given in Figure 4.3. The example training record defines a relation "ban" between a subject (entity) belonging to the class "Location" and an object (entity) belonging to the class "Fuel_V". The record starts, line 1, with an ID number which links it to the relevant tweet (111 is used in the example), a user defined annotation type ("RelationClass" in this case), and the start and end character indexes, identified during entity pairing, for the phrase in question. Next, lines 2 to 5, the class name and value for the relation are defined. The subject entity is the defined on lines 6 to 9, and the object entity on lines 10 to 13. In the example the subject entity belongs to the

class "Location" and the object entity to the class "Fuel_V". The values are 512 and 513; these are the unique entity IDs assigned during the entity pairing (see example above). The same process is repeated with respect to all the documents (tweets) in the input dataset so that a complete training set is generated. This training set is then used to create a GATE RE model to identify relations between entities in previously unseen tweets, but before this can be done the relevant configuration file must be edited.

```
<Annotation Id="1111" Type="RelationClass" StartNode="0" EndNode="31">
<Feature>
  <Name className="java.lang.String">rel-type</Name>
  <Value className="java.lang.String">ban</Value>
</Feature>
<Feature>
  <Name className="java.lang.String">Location</Name>
  <Value className="java.lang.String">513</Value>
</Feature>
<Feature>
 <Name className="java.lang.String">Fuel_V</Name>
  <Value className="java.lang.String">512</Value>
</Feature>
</Annotation>
```

FIGURE 4.5: Example GATE relation extraction training record

Configuration file: As noted above, GATE uses a number of Configuration files which define a range of parameters and features which can be adjusted with respect to a variety of applications. The RE configuration file needs to be edited before RE model generation can be commenced. This is necessary because the RE model generator needs a number of settings to be defined, such as the learning algorithm to be used (a SVM with respect to the evaluation presented later in Section 4.4) and the nature of the entities and relations marked up in the training data.

To give the reader a greater understanding of the nature of GATE RE configuration files, a fragment of an edited configuration file is given in Listing 4.2. The fragment tells the RE model generator that the training data includes an entity called Location (GATE refers to such entities as "arguments") which has a specified form and Part of Speech (GATE uses the term "attribute" for such features of an entity/argument). In more detail, lines 2 to 7 tell the RE model generator to expect an entity (argument) class called "Location" that has semantic type "NOMINAL", meaning it is a string (the alternative is NUMERIC), is indicated by the annotation tag "Location" and has a given id (e_{id}). Lines 8 to 14 tell the GATE RE model generator of the linguistic structure of the entity "Location" referred to above, a structure called "Form", specifying that the entity has "semantic type" normal (is a string), is indicated by a token which is also a string and is at the current position in processing (-1 would indicate before and 1 after). Lines 16 to 22 tell the GATE RE model generator that the entity has a Part of Speech, called "POS", specifying that with semantic type NOMINAL (is a string) and that it

is indicated by a token which is a part of speech tag (GATE uses the term "category" for such tags) and is at the current position in processing. Similar declarations will be made for the other entities, and the relations, in the training data. The rest of the configuration file is presented in AppendixB.

```
1. <FEATURES-ARG1>
2.     <ARG>
3.       <NAME>Location</NAME>
4.       <SEMTYPE>NOMINAL</SEMTYPE>
5.           <TYPE>Location</TYPE>
6.           <FEATURE>id</FEATURE>
7.     </ARG>
8.     <ATTRIBUTE>
9.       <NAME>Form</NAME>
10.      <SEMTYPE>NOMINAL</SEMTYPE>
11.      <TYPE>Token</TYPE>
12.      <FEATURE>string</FEATURE>
13.      <POSITION>0</POSITION>
14.    </ATTRIBUTE>
15.    <ATTRIBUTE>
16.      <NAME>POS</NAME>
17.      <SEMTYPE>NOMINAL</SEMTYPE>
18.          <TYPE>Token</TYPE>
19.          <FEATURE>category</FEATURE>
20.      <POSITION>0</POSITION>
21.    </ATTRIBUTE>
22.</FEATURES-ARG1>
```

Listing 4.2: Example of the Configuration File

The GATE RE model, once trained, can be used to predict the relations between entities from tweets. The results are stored in an XML format as shown in Figure 4.6. The figure shows an example using the tweet "Norway to completely ban petrol cars by 2025" used earlier. The XML records the existence of a relation "ban" between an entity belonging to the class "Location" and an entity belonging to the class "Fuel_V", the entities 513 and 512. Note that the XML includes a probability value for the accuracy of the prediction, 0.92425094 in this case.

4.3.3 RDF Dataset Generation

The final process in Figure 4.1 was RDF dataset generation. This process was the same as that described in Chapters 3 using Apache Jena to generate the RDF dataset. The process is therefore not discussed further here, other than to note that Apache Jena generates an RDF dataset.

```
<AnnotationSet Name="ML">
<Annotation Id="1158" Type="RelationClass" StartNode="0" EndNode="31">
<Feature>
  <Name className="java.lang.String">Location</Name>
  <Value className="java.lang.String">513</Value>
</Feature>
<Feature>
  <Name className="java.lang.String">rel-type</Name>
  <Value className="java.lang.String">ban</Value>
</Feature>
<Feature>
  <Name className="java.lang.String">Fuel_V</Name>
  <Value className="java.lang.String">512</Value>
</Feature>
<Feature>
  <Name className="java.lang.String">prob</Name>
  <Value className="java.lang.Float">0.92415094</Value>
</Feature>
</Annotation>
```

FIGURE 4.6: Example of a GATE XML format used to record RE results

4.3.4 RDF Schema Generation and Augmentation

The RDFS generation process commences after completion of the RDF dataset generation processes described above. This involves the mapping of the triples $\langle subject, predicate, object \rangle$ to an upper lexicon/schema to enrich the RDF dataset. The process adopted for this purpose which is the same process proposed with respect to the Stanford Core NLP framework described in Chapter 3. It was considered appropriate to map the entities to their class for inclusion in the final RDFS. GATE RE provides the classes of the entities as presented in Figure 4.5.

The entity classes and properties were used to generate the desired RDFS. As in the case of the Stanford CoreNLP framework described in Chapter 3, to allow the RDF dataset to be integrated into another dataset that has the same upper level ontology, a hierarchical structure of classes and properties was generated using WordNet (an alternative might have been Schema.org). The hierarchy construction, using WordNet, was conducted in the same manner as described previously; by importing all the hypernyms (superclassses/superproperties) for the identified classes and properties in a text file format (see subsection 3.3.4).

4.4 Evaluation

The previous section presented the proposed GATE RDF dataset from Twitter data framework. In this section, the evaluation of the framework is presented. For the purpose of evaluating the framework, the Motor Vehicle Pollution evaluation data set also used to evaluate the Stanford framework described in Chapter 3 was used. The objectives of the evaluation were:

- To determine the effectiveness of the RDF dataset generation process by evaluating the GATE RE model generation process.

The effectiveness of the NER was not evaluated because the GATE framework used the ANNIE gazetteer. As described above, the ANNIE gazetteer uses user-defined *entity lists* (also sometimes referred to as entity dictionaries) to conduct NER. There is no learning that takes places, the NER is as good as the entity lists supplied by the user. There was therefore nothing to evaluate with respect to NER in the context of the GATE framework.

4.4.1 Relation Extraction Evaluation

This sub-section presents the evaluation of the RE model used with respect to the GATE framework. Ten Cross-Fold Validation was again used, and the Precision, Recall and F1-score metrics, as in the case of the evaluation presented in Chapter 3. Recall that the motor vehicle pollution training set comprised 300 records, thus 30 records per fold. The results are presented in Table 4.1. Inspection of the table indicates high Standard Deviation values. As noted in Chapter 3, it was conjectured that this was because of the imbalanced nature of the training data. The F1-score Standard Deviation was 9.4, the recall was 9.6% and the precision 11.9%. The precision, recall and F1-score values were between 48.6% and 85.4%, 60.5% and 93.7%, and 50.8% and 79.2%, respectively. The average F1-score of the folds was 66.8%, less than that obtained with respect to the Stanford RE model. The averages for the precision and recall were 70.7% and 67.7%, respectively. The worst recorded F1-score was for Fold 5, a value of 50.8%. The reason was that the precision of the relations "use" and "ban_Fuel_V_Date" were 39.5% and 42.1%, which means the RE model predicted correctly only 39.5% of "use" relation cases, and 42.1% of the "ban_Fuel_V_Date" ban relations. The results for all ten folds are presented in Appendix B.

In conclusion, the GATE RE model demonstrated reasonable performance, but not as effective as the Stanford RE model. It seems that POS tagging and the inclusion of words between entities, used with Stanford RE play a significant role with respect to the effectiveness of the RE. ANNIE also uses POS tagging, but a comparative study on the effectiveness of POS tagging presented in [133] demonstrated that Stanford POS tagging was more effective than ANNIE POS tagging. ANNIE does not consider the words between entities.

.

4.4.2 RDF Schema Generation Using Motor Vehicle Pollution Dataset

RDFS generation using the motor vehicle pollution domain following the same processes as described for the Stanford framework as discussed in Chapter 3, Sub-section 3.4.4. As to be expected, using the GATE framework, the same classes and properties were extracted from the Twitter data. Four entity classes: (i) Location, (ii) Fuel_V,

TABLE 4.1: TCV results for the GATE RE model evaluation of the GATE Framework

Fold Number	Precision	Recall	F1-score
Fold 1	73.8%	68.2%	70.6%
Fold 2	85.4%	77.0%	79.2%
Fold 3	69.2%	86.7%	71.1%
Fold 4	50.5%	70.2%	57.6%
Fold 5	48.6%	60.5%	50.8%
Fold 6	75.2%	93.7%	82.1%
Fold 7	53.9%	72.2%	61.4%
Fold 8	69.3%	75.0%	70.0%
Fold 9	72.7%	77.8%	72.0%
Fold 10	70.7%	67.7%	66.8%
Average	67.0%	75.0%	68.2%
Standard Deviation (SD)	11.9	9.6	9.4

(iii) Environment_friendly_V, and (iv) Date; and four properties: (i) ban, (ii) use, (iii) ban_Fuel_V_Date, and (iv) use_Environment_friendly_V_Date. As before, an RDFS file was generated using Apache Jena as explained in Sub-section 4.3.4. And also as before, WordNet was utilised to build the hierarchy structure using class and property hypernyms. The final RDFS was the same as that shown previously in Figure 3.9 in Chapter 3. What can therefore be concluded is that both RDFS files have the same classes and properties. The reason for this similarity is a result of how the two frameworks have been trained, both the NER and RE models were created using the same insights and understandings of the motor vehicle pollution domain

4.5 Summary

This chapter has presented the second proposed framework for RDF dataset from Twitter data, the GATE framework. This framework comprised four stages: (i) Tweet cleaning (ii) NER (iii) RE and (iv) RDF dataset generation. The collected tweets were first cleaned. Then, some pre-processing steps, using the GATE toolkit, were applied to the input data, namely tokenisation and the assigning of a sequential index ID number to each character in each Tweet. The ANNIE gazetteer was then used to identify the entities of interest. The GATE gazetteer comes with a set of predefined entity lists, but these were considered insufficient for most domains of discourse that an end user might wish to consider. Therefore, additional entity lists needed to be created; the mechanism for doing this was fully described. The RE tool that comes with GATE operates using a supervised machine learning approach. A training set therefore needs to be expressed by manually assigning relations. Once a training set has been produced a GATE RE model was generated and subsequently used, within the context of the proposed framework, to extract entities and the relationships between these entities (triples) from collections of tweets to generate a RDF dataset. To enrich the RDF dataset the triples were mapped and augmented, and then used to produce the desired RDFS. This was achieved using

Apache Jena. WordNet was employed to include class hierarchies (using the hypernyms of the class names) as an example. The chapter also presented the evaluation of the GATE RE model. The evaluation indicated that the GATE RE model produced an acceptable performance but not as good as the Stanford RE model. The next chapter describes the third proposed RDF dataset from Twitter data framework, the Regular Expression framework. This framework attempts to solve the main problem of the Stanford CoreNLP and GATE frameworks, namely the requirement for hand-crafted training data to support supervised learning for RE, by semi-automating the RE training data generation process.

Chapter 5

The Regular Expression RDF Dataset From Twitter Data Framework

5.1 Introduction

This chapter presents the third RDF dataset generation framework considered in this thesis, the Regular Expression RDF dataset from twitter data framework. This is another hybrid approach in the context of the categorisation presented in Chapter 2 in that it combines statistical, machine learning and linguistic approaches. The work presented in this chapter is directed at the following subsidiary research question, Subsidiary Question two from Chapter 1:

2. *Given the overhead associated with supervised learning is there an alternative semi-supervised approach to RE that can be adopted that does not adversely affect the quality of any generated RDF dataset?*

The Stanford CoreNLP and GATE RDF dataset from Twitter data frameworks presented in Chapters 3 and 4, both used supervised learning to generate a RE model. A significant disadvantage of supervised learning, as noted earlier, is the need for the creation of labelled training data. The resource required to produce this training data is a frequently encountered "bottle neck" with respect to RDF dataset generation, and thus the motivation for the work presented in this chapter.

The idea presented in this chapter is to use the concept of regular expressions [79] to assist in the process of training data generation. A regular expression, in the context of NLP, is a sequence of characters that define a search pattern. In more detail, the idea presented in this chapter is to use a semi-automated process to identify regular expressions that are indicative of the relationships between pairs of entities, and then use these search patterns to prepare an appropriate RE training set. This training set

can then be used to generate a relation extraction model in much the same way as used with respect to the Stanford CoreNLP and GATE frameworks described earlier.

The rest of this chapter is structured as follows. Section 5.2 describes the proposed, two-stage, Regular Expression RDF dataset from Twitter data framework, followed by further detail concerning the various processes that make-up the two stages of the framework. The proposed framework requires support from a range of NLP tools. As in the case of the previous frameworks described in Chapters 3 and 4, the framework includes an RDFS enrichment stage. The evaluation of the proposed regular expression framework is reported on in Section 5.3. For the evaluation the Stanford toolkit was used. The chapter is concluded with a short summary in Section 5.4.

5.2 The Regular Expression RDF dataset generation Framework

This section provides details concerning the third RDF dataset from Twitter data framework presented in this thesis, the Regular Expression RDF dataset from Twitter data framework. A schematic of the proposed framework is given in Figure 5.1. In Figure 5.1, the rectangle boxes indicate a process, task, action, or operation; whilst the rounded boxes are used to show the models that are trained an ready to use. A cylinder shape represents a data file or database. From the figure it can be seen that the proposed framework comprises two stages: (i) NER Model and Regular Expression Generation and (ii) RE Model Generation and RDF dataset generation.

The input to the first stage was a seed set of labelled tweets; a small number of tweets, less than that required with respect to the two previously proposed frameworks. The tweets were first cleaned in a manner similar to that described with respect to the previous two frameworks. Each tweet was then tokenized into individual words and the entities labelled manually. There are a variety of ways in which labels can be acquired. The recommendation made in this thesis is that some existing repository is used for this purpose, such as WordNet or the Schema.org knowledge bases. For the evaluation presented later in this Chapter, in Section 5.3, WordNet was used. The processed seed set had to be formatted in two ways: (i) to support Stanford NER model generation as described in Chapter 3, and (ii) to support to proposed regular expression mechanism. Two exampled of regular expression labelled seed set tweets are given below:

Tweet 1: <Location>Norway</Location> to completely ban <Fuel_V>petrol </Fuel_V> cars by <Date>2025</Date>

Tweet 2: Why did <Location>Oslo</Location> ban <Fuel_V>diesel</Fuel_V> in <Date>2017</Date>

These two examples were taken from the "motor vehicle pollution" domain of discourse used for evaluation purposes (see section 5.3), and will be returned to later in this section. Returning to Figure 5.1 it can be seen that Stage One of the proposed framework comprises two processes that may be applied in parallel: (i) NER model generation and (ii) regular expression pattern generation; each is discussed in further detail in Sub-sections 5.2.1 and 5.2.2 respectively.

The input to the second stage, Stage 2, is a larger collection (larger than the seed set) of tweets, but in this case unlabelled. From Figure 5.1, it can be seen that Stage two of the proposed framework comprises five process applied in sequence (indicated in the figure by rectangles): (i) tweet annotation using the NER model that was generated in Stage 1; (ii) the application of the regular expression patterns, from Stage One, to the annotated tweets to produce a set of entity-relation-entity triples; (iii) generation of an RE training set using the annotated tweets and the generated triples from the previous two processes; (iv) application of the training set to generate a RE model, (v) RDF dataset generation. Two more steps, not shown in the figure, were then applied: (i) class mapping and augmentation and (ii) RDF generation. Further details concerning these processes are presented in the following sub-sections.

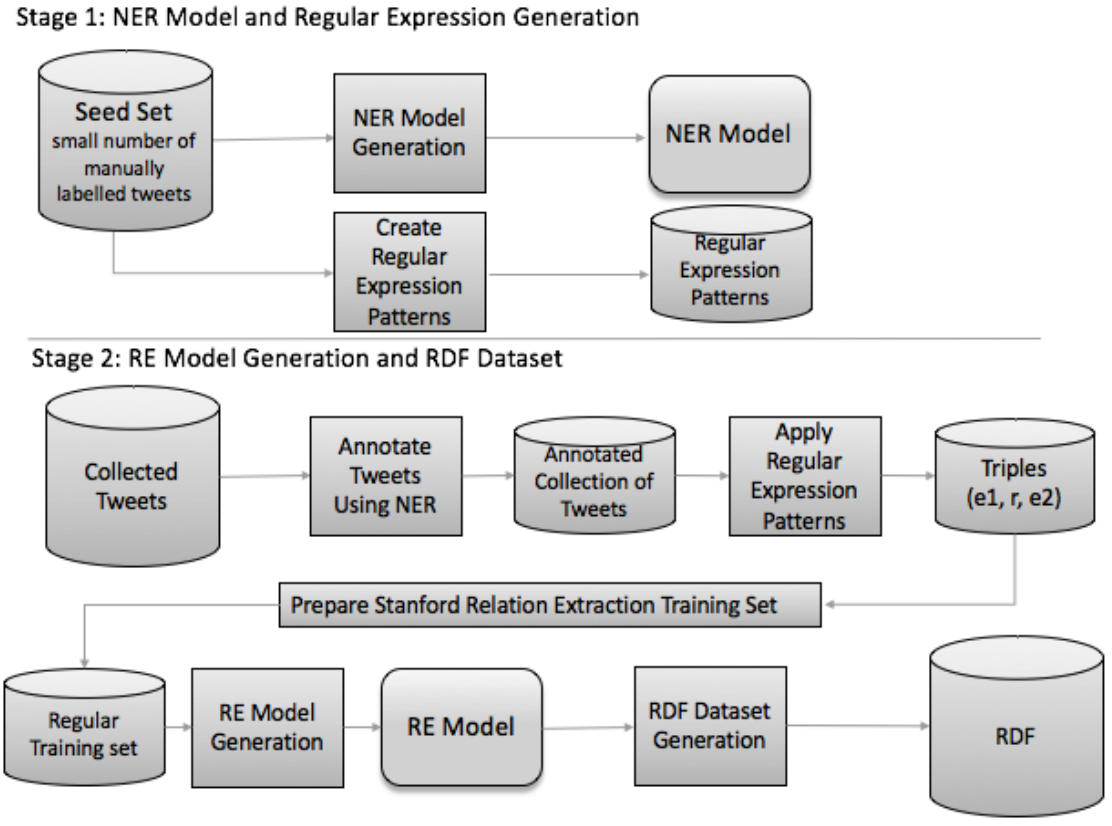

FIGURE 5.1: Schematic of the RDF Dataset Generation Framework Using Regular expression along with Stanford CoreNLP

5.2.1 NER Model Generation

The generation of a NER model was the first process in Stage 1 of the proposed Regular Expression framework. The idea was to train the model using the labelled seed set provided as input. For the evaluation presented later in this Chapter the Stanford CoreNLP toolkit was used to produce the desired NER model for reasons presented later in this chapter (Sub-section 5.3.4); other NER model generation mechanisms would be equally applicable. The generated model was then used to:

1. Annotate the larger collection of tweets that formed the input to Stage 2, and
2. Map entities to classes (the first step of enriching the RDF dataset) prior to RDFS generation.

5.2.2 Regular Expression Pattern Generation

As noted earlier, the biggest challenge associated with the two previously proposed frameworks (Chapters 3 and 4) is that they require manual labelling of significant amounts of training data to generate a RE model. A potential solution to this challenge was to use rule-based methods, as described in Chapter 2. Such methods use the concept of regular expressions. There have been previous examples where regular expression patterns have been used to extract knowledge from free text. For example for the purpose of extracting phone numbers, email addresses, software names, credit card numbers, social security numbers or gene and protein names [79]; this previous work motivated the use of regular expressions as presented in this chapter.

A regular expression, in the context of the proposed framework, is a string containing a combination of normal characters and special meta-characters or meta-sequences [129]. The string is used for matching purposes and hence to trigger a rule whenever a match is found; once triggered the rule will extract literals from the text that the pattern has been matched to. To illustrate the process a number of example regular expressions are given below. Note that with respect to the proposed Regular Expression framework the regular expressions were defined using a bespoke syntax founded, in part, on that presented in [72]. This syntax is one of the additional contributions of this thesis. The syntax also required a dedicated parser which is another of the additional contributions of this thesis. In the remainder of this chapter the notion of regular expressions is further discussed in terms of a sequence of examples.

Example 1, Email Address Extraction:

```
[a-z,A-Z,0-9,.,_,%,+,-]+@[a-z,A-Z,0-9,.,-]+.[a-z,A-Z]{2, }
```

This first example describes a regular expression to identify email addresses, written in upper case or lower case or a mixture of the two, and is defined using the syntax given by the grammar presented in Backus–Naur Form (BNF) in Table 5.1. The syntax states

TABLE 5.1: Basic BNF syntax for email regular expression

<emailAddress> ::=	<mailbox> "@" <domain>;
<mailbox> ::=	<characterSet>;
<domain> ::=	<characterSet> "." <characterSet>;
<characterSet> ::=	"[" <charSetList> "]" <quantifier> | "[" <charSetList> "]" <quantifier> <characterSet> ;
<charSetList> ::=	<charSet> | <charSet> "," <charSetList> ;
<charSet> =	"a-z" | "A-Z" | "0-9" | "." | "-" | "%" | "+" | "-" | "(.)";
<quantifier> =	"+" | "*" | "{" <integer> "}" | "{" <integer> "," <integer> "}" | "|" ;
<integer> =	<numChar> <integer> | <numChar> ;
<numChar> =	"0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9" ;

that an email address comprises an "@" character separating a mailbox and domain. The mailbox comprises a set of characters and a quantifier. The domain also comprises a set of characters and a quantifier, but followed by "." and a second set of characters and a quantifier. Looking at the email regular expression example the mailbox part can consist only of the listed set of predefined characters, and there can be any number of these as indicated by the "+" quantifier. The first part of the domain can also be any number of predefined characters, the last part can only comprise alphabetic characters, and there can be between two or more of these as indicated by the {2, } quantifier.

TABLE 5.2: BNF syntax for phone number regular expression extending the syntax given in Table 5.1

<phoneNumber>::=	<areaCode> "." <customerCode> | <areaCode> "-" <customerCode> ;
<areaCode> ::=	"0" <chartacterSet> ;
<customerCode> ::=	<chartacterSet> ;

Example 2, Phone Number Extraction:

0[0-9]{3}[.-][0-9]{3}[.-]{1}[0-9]{4}

This second example describes a regular expression pattern for extracting a phone number that starts with a zero followed by three digits followed by a dot or a hyphen followed by three digits, a dot or a hyphen, and then four digits. The additional grammar required with respect to the BNF given in Table 5.1 is given in Table 5.2.

TABLE 5.3: BNF syntax for social security number regular expression extending the syntax given in Tables 5.1 and 5.2

<socialSecurityNumber>::=	<areaNumber> <groupNumber> <Serial Number>
<areaNumber> ::=	<ChartacterSet> ;
<groupNumber> ::=	<ChartacterSet> ;
<Serial Number> ::=	<ChartacterSet>;

Example 3, Social Security Number Extraction:

[0-9]{3}[-]?[0-9]{2}[-]?[0-9]{4}

This third example describes a regular expression that can be used for identifying an US social security numbers. This regular expression pattern includes three possible formats for the social security number, hyphens between digits, spaces between digits or a set of digits without any spaces. The associate additional BNF is given in Table 5.3. In the above examples various quantifiers are used in connection with *character set lists*. A more comprehensive set of quantifiers, than those included in Table 5.1 is presented in Table 5.4 (based on [72]).

To identify relation expressions the BNF syntax given in Tables 5.1 to 5.3 was extended with some additional grammar as given in Table 5.5. The syntax allows regular expressions that feature a relation between two entity expressions, or following an entity expressions, where an entity expression comprises one or more entities.

Two example regular expressions for identifying entity-relation-entity triples, using the grammar presented in Table 5.5 and the pollution application domain considered later in this thesis, are given below:

Pattern 1: <Location> ([a-z,A-Z,]+) < /Location> .*(ban).* <Fuel_V> ([a-z,A-Z,]+) < / Fuel_V>

Table 5.4: The most common quantifiers and metacharacters used in regular expressions [72].

Metacharacter	Name	Description
.	Period	Matches any characters
?	Question mark	Matches zero or one of the items in the preceding character set list character
*	Asterisk	Matches zero or more of the items in the preceding character set list character
+	Plus	Matches one or more of the items in the preceding character set list character
|	Alternation	One of the items in the preceding character set list
{X}	Quantity	Match exactly X of the items in the preceding character set list
{X,Y}	Specified range	Match at least X and no more Y of the items in the preceding character set list

Table 5.5: BNF syntax for regular expression used in the Regular Expression RDF dataset generation framework, extending the syntax given in Tables 5.1 and 5.3.

<regularExpression> ::= <entityExp> <characterSet> <relationExp>
<characterSet> <entityExp>
| <entityExp> <characterSet> <entityExp>
<characterSet> <relationExp> ;

<relationExp> ::= <characterSet> ;

<entityExp> ::= <entity>;
| <entity> <characterSet> <entityExp>

<entity> ::= “<” <label> “>” <characterSet> “</” <label> “>” ;

<label> ::= <characterSet>;

Pattern 2: <Location> ([a-z,A-Z,]+) < /Location> .*(ban).* <(Fuel_V)> .* < / Fuel_V > .* <(Date)> ([a-z,A-Z,0-9,]+) < /Date>

The first of the above regular expression is intended to find triples the feature the relation “ban” linking two entities belonging to the class “Location” and the class “Fuel_V” respectively (Pattern 1). The second is intended to find triples that feature the relation “ban Fuel_V Date” linking two entities belonging to the class “Location” and the class “Date” respectively (Pattern 2). Note that the “.*” (dot-star) character sequence indicates that the character sequence of interest may be preceded and/or succeeded by additional text, as can the relation expression. Applying Patterns 1 and 2 to the two seed tweets given at the start of this section, and listed again below, will result in the

entity-relation-entity triples shown in Table 5.6.

Tweet 1: <Location>Norway</Location> to completely ban <Fuel_V>petrol </Fuel_V> cars by <Date>2025</Date>

Tweet 2: Why did <Location>Oslo</Location> ban <Fuel_V>diesel</Fuel_V> <Date>2017</Date>

TABLE 5.6: The result of applying example regular expression patterns 1 and 2 over the two example tweets, Tweet 1 and Tweet 2

Tweet number	Entity 1	Relation	Entity 2
Tweet 1	Norway	ban	petrol
Tweet 1	Norway	ban_Fuel_V_Date	2025
Tweet 2	Oslo	ban	diesel
Tweet 2	Oslo	ban_Fuel_V_Date	2017

Two more example regular expression patterns are presented below to describe the situations where the relation comes after two entity expressions:

Pattern 3: <Location>([a-z,A-Z,]+)</Location>.*<Fuel_V>([a-z,A-Z,]+) </Fuel_V>.* (ban) .*

Pattern 4: <Fuel_V>([a-z,A-Z,]+)</Fuel_V>.*(ban).*<Location> ([a-z,A-Z,]+)</Location>

For the evaluation presented later in this Chapter three categories of regular expression pattern were considered: (i) two entity expressions, (ii) three entity expressions and (iii) four entity expressions. The general form of these patterns are: e1,r,e2, e1,e2,r,e3 and e1,e2,e3,r,e4 respectively. Once a set of regular expression, for the seed set, have been created, the second stage of the proposed Regular Expression RDF dataset generation from Twitter data framework could be commenced, namely RE model generation.

5.2.3 RE Model Training Set Generation

Stage 2 of the proposed framework takes as input a set of unlabelled tweets that has been cleaned and tokenised. The aim is then to use the NER model and regular expressions from Stage 1 to create a training set with which to train an RE model which can then be used for RDF dataset generation. The first process in Stage 2 (see Figure 5.1) is thus the annotation of the input data using the NER model from Stage 1. This results in a set of tweets with entities labelled according to class. The next process is the application of the regular expressions from Stage 1 to the entity labelled tweets to produce a set

of triples of the form ⟨e1,r,e2⟩[1] (⟨subject, predicate, object ⟩). A dedicated parser was constructed to match regular expressions, defined using the syntax given in the foregoing, to entity-labelled free text. Note that it is entirely possible, given a specific tweet, that none of the generated regular expressions can be matched to the tweet, which means the relation could not be extracted. This might be considered to be a limitation of the proposed regular expression framework. The triples were then used, the third process in Stage 2, to automatically generate a training set for the generation of a RE model, which will be described in the next sub-section.

The reason for generating a RE model is to create a model that can extract relations that link pairs of entities. There are a number of tools that can be used to generate a RE model. For the evaluation of the proposed Regular Expression RDF dataset from Twitter data framework, reported on in Section 5.3 of this Chapter, the Stanford relation extraction tool was used as previously described in subsection 3.3.2 of Chapter 3. As discussed in Chapter 3, the Entity Mention Detection (EMD) and the Relation Mention Detection (RMD) components were therefore used to build the Stanford RE training set. The format of Stanford training records was presented previously (see Figure 3.4 in Chapter 3)

5.2.4 Relation Extraction and RDF Dataset Generation

The fourth process in Stage 2 of the Regular Expression framework workflow is the generation of a RE model using the training set generated in the foregoing processes. To gain confidence in the model, its performance can be validated using a test set. This needs to be in the same format as the training set. A typical expedient is to separate part of the training set and use this as a test set. Once a degree of confidence in the model has been confirmed the model can be applied. From experiments reported on later in this Chapter it was found that a training set of 222 records could be generated given a seed set of 100 records.

The output from the RE model was a set of triples of the form $\langle subject, predicate, object\rangle$, similar to the output generated using the Stanford CoreNLP and GATE frameworks presented earlier. Thus, given the tweet "Norway to completely ban petrol cars by 2025" this would produce the triples: (i) $\langle Norway, ban, petrol\rangle$, and (ii)) $\langle Norway, banFuel\_VDate, 2025\rangle$.

Apache Jena was used to generate the RDF dataset using the extracted triples. The process of RDF dataset generation followed the same process as described in Chapters 3 and 4. The process is therefore not discussed further here.

5.2.5 Class Mapping and Augmentation

The process of enriching the RDF dataset was starting commenced with a class mapping and augmentation process. This was where the entities within the extracted triples were

[1] A pattern of the form e1,r,e2,e3 might produce triples of the form ⟨e1,r,e2⟩, ⟨e1,r,e3⟩ and/or ⟨e2,r,e3⟩ similarly a pattern of the form e1,e2,e3,r,e4 might provide a number of triples.

mapped to their classes which were used thereafter to construct the final RDFS. This was achieved using the NER model as trained in Stage 1 of the proposed framework. Recall that with respect to the evaluation presented later in this chapter the Stanford NER model was adopted. Continuing with the above example, the entity $Norway$ would be mapped to the class *Location*, the entity *petrol* to the class $Fuel\_V$ and the entity 2025 to the class *Date*.

The identified classes and properties were augmented with super-classes and super-properties in the same manner as described previously in Chapters 3 and 4. The idea was to enhance the RDF dataset with super-classes so as to make the dataset more integrable with other dataset that have the same upper level ontology. As before, hypernyms extracted from WordNet were used with respect to the identified classes and properties. As mentioned in Chapters 3 and 4, WordNet was used as an example (an alternative might have been Schema.org).

5.2.6 RDF Schema Generation

The second and final process of enriching the RDF dataset was RDFS generation. This process was identical to the processes described in Chapters 3 and 4 using Apache Jena to build the RDFS. The process is therefore not discussed further here, other than to note that Apache Jena generates a RDFS by linking all the classes and properties extracted previously and build a class/property hierarchy.

5.3 Evaluation

The framework presented in this chapter was directed at RDF Dataset generation from Twitter data using the concept of regular expression to automate the process of generating an RE training set. In this section, the evaluation of the framework is presented. For the evaluation the motor vehicle pollution evaluation data set, also used with respect to the frameworks presented in Chapters 3 and 4, was again used. The objectives of the evaluation were:

1. To determine the effectiveness of the NER model generated using a seed set.
2. To determine the effectiveness of the RE model generated using the proposed regular expression technique.

5.3.1 Named Entity Recognition (NER) Evaluation

The proposed relation extraction framework was directed at reducing the degree of end user involvement required to generate training data with which to generate a RE model. The idea presented here was to use a seed set to build a NER model and then use regular expression to automatically build an RE training set. It was conjectured that by using a smaller seed, smaller than that required for the Stanford framework, an adequate NER model could still be constructed. To analyse this conjecture a series of experiments was

conducted using Stanford NER and kCV with k set to 3. The evaluation metrics used were again Precision, Recall and F1-score as used in the evaluations reported earlier.

For the evaluation a labelled motor vehicle pollution dataset was used comprised of 100 records. With only 100 records TCV no longer made sense; testing the NER model using test sets comprised of 10 records was considered to be insufficiently robust. Although, as described in Chapter 3, there is evidence in the literature where TCV has been used for smaller numbers of Tweets, for example in [3] only 290 Tweets were used. Thus, for the evaluation of the regular expression NER model, as described here, using only 100 tweets, three-fold Cross Validation was used. As a consequence the test sets comprised around 33 records, roughly the same as in the case of the NER model evaluation presented for the Stanford framework described in Chapter 3 where TCV was used. The results are shown in Table 5.7. The average F1-score was found to be 52.3%. Referring back to the F1-score obtained using 270 tweets to train the Stanford NER model, given in Table 3.1, an average F1-score of 78% was obtained. Thus, as expected, using fewer training records produced a worse performance, but it seems that the NER model presents a good result that helps to generate RDF datasets that can then be used to generate an RDFS where entities are mapped onto classes.

Table 5.7: 3 Fold Cross Validation results for the Regular Expression Stanford NER model evaluation

Fold Number	F
Fold 1	49.0%
Fold 2	53.0%
Fold 3	55.0%
Average	52.33%
Standard Deviation (SD)	3.06

5.3.2 Relation Extraction Evaluation

For evaluation of the RE model generated using the proposed regular expression technique the evaluation was conducted using TCV and Precision, Recall and F1-score as the evaluation metrics. The results are presented in Table 5.8. From the Table, it can be seen that the F1-scores range from between 56.9 and 88.5, the precision between 47.5% to 89.3%, and the recall between 63.8% to 87.7%. The Standard Deviation for F1-score, Recall and precision were 8.3, 7.0 and 11.5 respectively, again because of the imbalanced nature of the input data. The worst fold in Table 5.8 was Fold 9. The reason was that the precision of the relation "use_Environment_friendly_V_Date" was 0, the RE model failed to predict any relations correctly from the extracted relations in Fold 9. The number of occurrences of this relation in the data set was only 87 records, when that of the relation "ban" was 241 records which had an associated precision of 70.0%. What is important to note, however, is that the average RE precision, recall, and F1-score acquired using the Regular Expression RE model was better than that

obtained using GATE RE, but not as good as that obtained using Stanford RE with a larger hand-crafted training data set, whilst using a much smaller training set of 200 records. The average precision, recall, and F1-score values were 74.2%, 75.6% and 74.5% respectively. Further detail of the results obtained are included in Appendix C

Table 5.8: TCV results for the Regular Expression Stanford Relation Extraction model evaluation

Fold Number	Precision	Recall	F-score
Fold 1	75.9%	80.4 %	78.1%
Fold 2	89.3%	87.7%	88.5%
Fold 3	75.9%	63.8%	69.3 %
Fold 4	73.8%	77.5%	75.6%
Fold 5	70.0%	77.8%	73.7%
Fold 6	68.4%	72.2%	70.3%
Fold 7	79.6%	68.3%	73.5%
Fold 8	75.0%	82.4%	78.5%
Fold 9	47.5%	70.7%	56.9%
Fold 10	87.0%	75.8%	81.0%
Average	74.2%	75.6%	74.5%
Standard Deviation (SD)	11.5	7.0	8.3

5.3.3 RDF Schema Generation Using Motor Vehicle Pollution Dataset

The final RDFS generation process included in the Regular Expression framework was the same as that used with respect to the Stanford framework as described in Chapter 3, sub-section 3.4.4. The same classes and relations were extracted from the Twitter data. Four classes label, are (i) Location, (ii) Fuel_V, (iii) Environment_friendly_V, and (iv) Date; and four relations labels: (i) ban, (ii) use, (iii) ban_Fuel_V_Date, and (iv) use_Environment_friendly_V_Date. Apache Jena was used to create the RDFS file. The hierarchy structure was constructed using WordNet by using the hypernyms of the classes and relations extracted from the Twitter input data. In the case of the motor vehicle pollution domain the RDFS graph generated was exactly the same as the graph presented in Figure 3.9 in Chapter 3.

5.3.4 Stanford CoreNLP vs GATE

The proposed Regular Expression framework can operate using a range of NLP tools. As noted in the foregoing, with respect to the evaluation presented here the Stanford CoreNLP toolkit was used as also considered previously with respect to the proposed Stanford CoreNLP RDF dataset from Twitter Data Framework (see Chapter 3). An obvious alternative might have been the GATE toolkit as used with respect to the proposed GATE RDF Dataset from Twitter Data Framework (see Chapter 4). It therefore seemed appropriate to complete this evaluation section with an argument as to why the Stanford toolkit was adopted with respect to the Regular Expression framework, and

not the GATE toolkit, by considering the relative advantages and disadvantages of the toolkits under a number of headings as follows:

Programming language: The Stanford CoreNLP is available in several programming languages, including Python, Ruby, Perl, Scala, Clojure, Javascript (node.js), and several "dot" NET languages including C# and F# [14, 84]. On the other hand, the GATE tool is only available in Java**??**.

Usability: GATE features a bespoke Graphical User Interface (GUI), whereas Stanford, by default, uses a command line interface. The author has found GATE to be less "user-friendly" compared to the Stanford toolkit. The Stanford CoreNLP is easy to use than the GATE because of its straightforward nature unlike the GATE which entails a steep "learning curve". This has also been observed by other researchers, see for example [84].

RE Training set preparation: The Stanford RE tool has a clearly laid out set of rules for preparing a training set (as described in chapter 3). On the contrary, from the author's point of view, the rules for using the GATE RE tool are often ambiguous, hence more time is required to prepare an RE training set using GATE.

Named Entity Recognition (NER): Both the Stanford and GATE toolkits have a NER tool. For most Twitter application domains, it can be anticipated that the NER tools will need to be refined. In the case of the Stanford toolkit this means retraining using a bespoke training set. In the case of GATE this means defining bespoke entity lists to be included in the GATE gazetteer. Both the Stanford and GATE NER tools achieve their goal. However, the Stanford NER tool is more useful than the GATE gazetteer tool because the Stanford NER model uses a machine learning technique that provides the ability to predict entities that have not been mentioned explicitly in the training set.

Relation Extraction (RE): The Stanford RE tool requires a NER model, whilst GATE does not required a NER model. The most significant difference between the Stanford and GATE RE models is model performance as presented in Chapters 3 and 4. In the context of the evaluations presented earlier it can be seen that the Stanford RE model outperformed the GATE RE model by a significant margin.

Robustness and quality of linguistic analysis component: Both Stanford CoreNLP and GATE are robust and have high quality linguistical components which could be easily used for common linguistical scenarios [84].

A summary of the above is given in Table 5.9. From the foregoing, and from the table, it can be seen that both Stanford and GATE feature advantages and disadvantages. However, Stanford was found to be the most effective and efficient toolkit in the context of RDF dataset generation from Twitter data. For this reason the Stanford CoreNLP toolkit was used to evaluate the Regular Expression framework presented in this chapter.

This is also why the Stanford toolkit was used to evaluate the Dependency Parsing and Word Mover Distance framework presented in the next chapter, Chapter 6.

TABLE 5.9: Summarisation of the comparison between the Stanford CoreNLP and GATE toolkits

Features	Stanford CoreNLP	GATE
Programming language	Several programming languages	Java only
Interface	Command line interface	GUI
Usability	User friendly toolkit	Requiring effort to learn to use effectively
Training set preparation	Easy to prepare training set	Difficult to prepare training set
Named Entity Recognition	Stanford NER model can be used	Gazetteer can be used
Robust and quality linguistic analysis component	Robust and high quality linguistical components	Robust and high quality linguistical components
Model performance	Better performance than GATE	Good performance

5.4 Summary

In this chapter, the third of the four proposed RDF dataset from Twitter data Frameworks discussed in this thesis, the Regular Expression framework, has been presented. The motivation for the framework was to reduce the resource overhead required to generate a bespoke RE training set through the use of a small seed set which was then used to generate regular expressions. The framework comprised two stages. In the first stage, Stage 1, a NER model and a set of regular expressions were generated. In the second stage, Stage 2, the NER model and regular expressions from Stage 1 were used to create a RE training set which was then used to generate an RE model which was then applied to extract entity-relation-entity triples from which an RDF dataset could be derived. Then, two processes were used to enrich the RDF dataset before generating the desired RDFS file. The components of each stage were fully discussed, especially the process for defining and using the regular expressions. This was done using a bespoke syntax which in turn required a dedicated parser (two of the additional contributions of this thesis). The proposed Regular Expression framework required support from a range of tools; tools that are available within a range of NLP toolkits, but particularly the Stanford toolkit. The chapter was concluded with an evaluation of the proposed framework, including a comparison of the Stanford and GATE toolkits to justify the use of Stanford with respect to the evaluation of the proposed framework. The next chapter presents the final RDF dataset from Twitter data framework proposed in this thesis, the Shortest Path Dependency Parsing and Word Mover's Distance (SPDP-WND) framework.

Chapter 6

The Shortest Path Dependency Parsing and Word Mover's Distance RDF Dataset From Twitter Data Framework

6.1 Introduction

Chapters 3 and 4 presented the Stanford CoreNLP and GATE frameworks for RDF dataset from Twitter data; two frameworks founded on established NLP toolkits. Chapter 5, presented the Regular Expression RDF dataset from Twitter data framework, the third approach presented in this thesis designed to address the main limitation of the previous two frameworks, namely the need for substantial, hand-crafted, training datasets for RE model generation. To a large extent, the regular expressions framework addressed the disadvantage associated with the previous two frameworks. However, this framework still needed manual resource with respect to the creation of regular expression patterns. The framework presented in this chapter, the Shortest Path Dependency Parsing and Word Mover's Distance (SPDP-WMD) RDF dataset generation from Twitter Data Framework, which features a fully automated approach to RE model generation.

As in the case of the previous three chapters, the work presented in this chapter is designed to addresses the second subsidiary research questions from Chapter 1:

3. *Is it possible to devise an unsupervised approach to RE that can be adopted, that does not adversely affect the quality of any generated RDF dataset?*

The central idea underpinning the work presented in this chapter is to avoid the manual process of labelling a RE training set by using Shortest Path Dependency Parsing (SPDP) and Word Mover's Distance (WMD). As will be demonstrated in this chapter, SPDP and WMD, when used in combination, can be used to identify relationships

between pairs of entities, which can then be used to prepare an appropriate training set. The training set is used to create the relation extraction model, which in turn allows the identification of relations linking entities which in turn can be used to construct a RDF dataset.

The rest of the chapter is structured as follows. Section 6.2 presented the proposed SPDP-WMD RDF dataset from Twitter data framework. The section is divided into a number of sub-sections, each elaborating on the sub-processes that make-up the overall framework. Section 6.3 presents the evaluation process of the framework. At the end of the chapter some concluding remarks and discussion are presented in Section 6.4.

6.2 The Shortest Dependency Parsing and Word Mover's Distance (SPDP-WMD) RDF Dataset From Twitter Data Framework

A schematic of the proposed SPDP-WMD framework is given in Figure 6.1. From the Figure it can be seen that the framework comprises three stages: (i) Stage 1 - Named Entity Recognition (NER), (ii) Stage 2 - Relation Extraction (RE) and (iii) Stage 3 - RDF dataset generation. In Stage 1 a NER model is trained, using pre-labelled training data describing the domain of discourse, for use later in the overall process. In Stage 2 a RE model is trained using training data that is automatically generated, from a given input dataset. This is achieved as follows. SPDP is used to identify relation strings between the entities identified using the NER model from Stage 1. It was anticipated that many relation strings would be identified for each entity pair, hence the relation strings are clustered so that a unique label for groups of relations can be identified (one per entity pair). The clustering was conducted using WMD as the distance metric.

The relations are then used to create a training set to generate the desired RE model. The trained RE model can then be used to extract triples of the form $\langle subject, predicate, object \rangle$ (where the subject is the first entity, the object is the second entity, and the predicate is a relation between the entities) from the input data. In Stage 3, the RDF dataset generation stage, the triples from Stage 2 are used to generate a RDF dataset. As in the case of the previous frameworks presented in this thesis, to generate the final RDFS the entities in the triples were mapped onto their classes which were then augment to form a class hierarchy.

For the NER the Stanford NER tool was again adopted. This was for the same reasons as discussed in Chapter 5. The NER model, once generated, was used in Stage 2 to identify the entities of interest. The Stanford RE tool was adopted with respect to Stage 2, a decision driven by the adoption of the Stanford NER model for Stage 1. For the RDF dataset and RDFS generation, as in the case of the previous frameworks presented in this thesis, Apache Jena and WordNet were used. Also, Schema.org was used to build the final RDFS hierarchy so that a more integrable RDF datasets could be

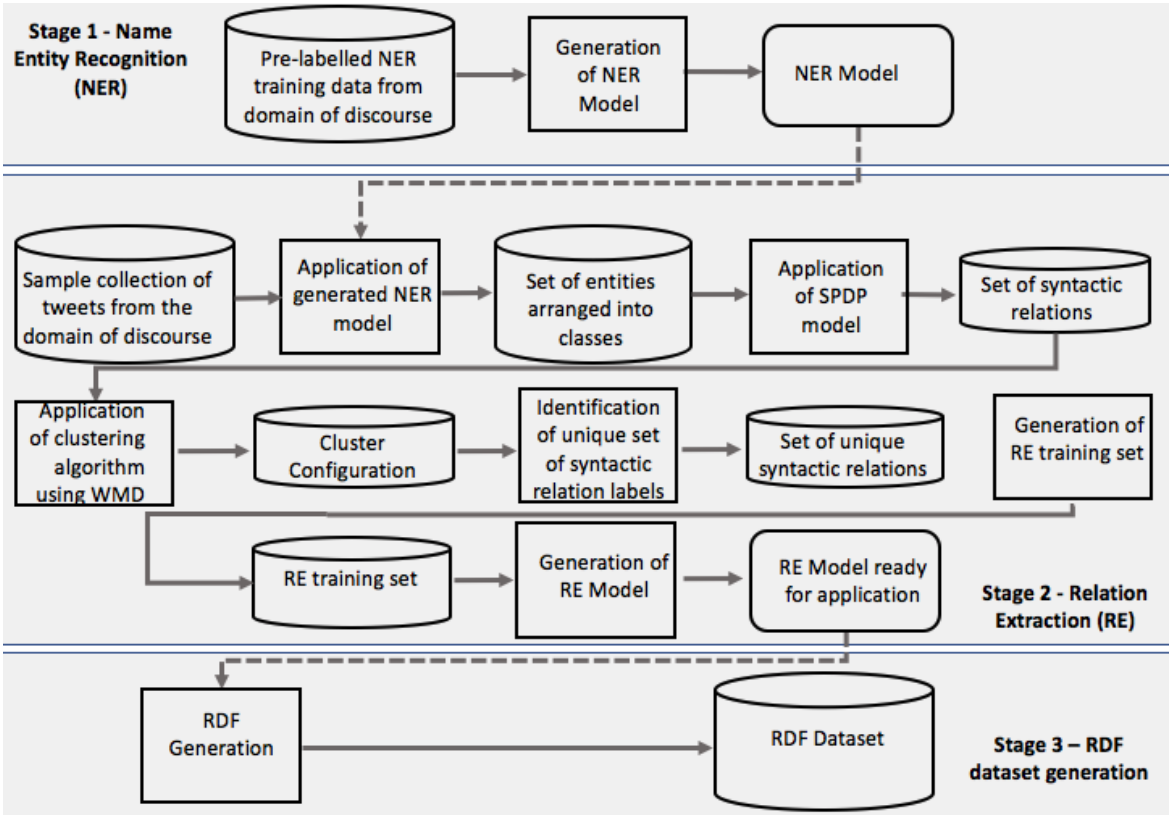

FIGURE 6.1: Schematic of the SPDP WMD RDF dataset generation from Twitter Data Framework

derived. Each of the above stages is described in further detail in then following three sub-sections, Sub-sections 6.2.1 to 6.2.3.

6.2.1 Stage 1 - Named Entity Recognition

The first stage in the proposed SPDP-WMD framework workflow (Figure 6.1) was NER. The input was a collection of training records (tweets) from the domain of discourse where the entities of interest have been pre-labelled according to their classes. As noted earlier in this thesis, there are number of NER tools that could have been adopted with respect to the proposed SPDP-WMD framework. As noted above the Stanford NER tool was selected. As described in chapter 3 and 5, the default Stanford NER tool was designed to identify seven default classes (Location, Person, Organization, Money, Percent, Date and Time). As also noted previously, in the context of Twitter domains it can be anticipated that many domain specific classes will not be supported given the Stanford NER default model. As such, the Stanford NER tool provides the ability to train a NER model using appropriate training data dedicated to a given target domain. The process for achieving this was described in Section 3.3.2 of Chapter 3. With respect to the proposed SPDP-WMD framework the identified entities have a role to play with respect to: (i) SPDP (ii) RE and (iii) the mapping of entities to their classes.

6.2.2 Stage 2 - Relation Extraction

Stage 2 of the proposed framework comprises a pipeline of six individual processes as shown on Figure 6.1. Namely: (i) NER model application, (ii) SPDP model application, (iii) clustering of relation strings, (iv) identification of unique relation labels, (v) generation of RE training data and (vi) RE model generation. Each is described in this sub-section as follows.

Stage 2 commences with the application of the NER model from Stage 1 to a collection of tweets that are representative of the domain of discourse of interest; the same domain of discourse as used in Stage 1. The input data could be the same as that used in Stage 1, but not necessarily so, hence two input sets are shown in Figure 6.1. The results will be a set of tweets with entities highlighted.

The next process applied in Stage 2 is SPDP. There are broadly two ways of imposing structure on sentences (tweets). One way of doing this is to use context free grammars; another, and that of interest here, is to use dependency parsing. The idea is the imposing a structure by considering which words are dependent on which others and presenting this as a *dependency tree* or *phrase structure tree*. Given knowledge of two entities that are known to appear in a sentence (tweet) we can find them in a dependency tree, representing the sentence (tweet), and then identify the "shortest" path across the tree between the two entities. The relationship between the two entities, it is argued, will then be held somewhere along this path. The advantage offered by SPDP, in the context of RDF dataset generation from twitter data, is that it can be used to circumvent the need for the hand-crafting of training data to build a RE model, as in the case of the frameworks presented in Chapters 3, 4 and 5; instead a dependency parse tree, together with knowledge of the relevant entities, can be used to generate the required training data. More specifically, with respect to the proposed framework, the idea was to use SPDP to automate the extraction of relations between specified entities and then to use these identified relations to prepare a RE training set that can be used to build a RE model using a supervised approach in a similar manner to that described in earlier chapters. This RE model can then be used to find additional (semantic) relations to those discovered using SPDP.

Figure 6.2 presents an example of a typed dependency parse tree, taken from [42], for an example sentence; "typed" because the links are labelled with grammatical relations. The grammatical relations between words that are presented in the figure are: (i) nominal subject (nsubj), (ii) direct object (dobj), (iii) referent (ref), (iv) determiner (det), (v) relative clause modifier (rcmod) and (vi) word introducing a rcmod (rel).

As in the case of NER, and as noted earlier in this thesis, there are a range of tools that can be used to generate parse trees. With respect to the proposed framework Stanford Enhanced Universal Dependency (SEUD) parsing was used. SEUD is founded on Stanford Universal Dependency (SUD) parsing which in turn is founded on Stanford Dependency (SD) parsing. SD parsing was originally developed in 2005, and quickly became the standard tool for the dependency parsing of English language texts. The

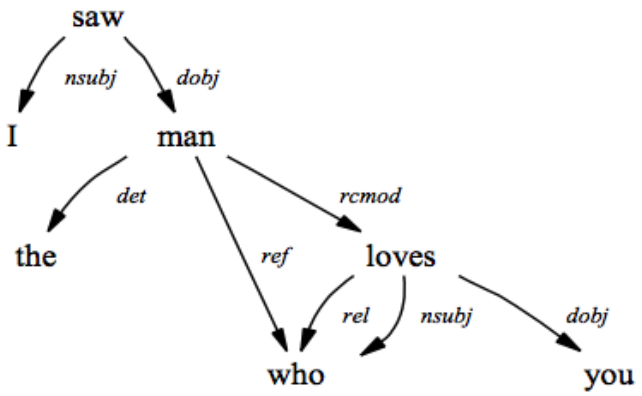

FIGURE 6.2: Example of a typed dependency parse for the sentence "I saw the man who loves you" [42].

basic SD parsing tool was improved in 2014 with the introduction of SUD [41]. SUD is an enhanced version of the basic SD parsing that includes a taxonomy for capturing grammatical relations, including a rich morphologically. Two layers of taxonomy are included in SUD: (i) a set of broadly attested universal grammatical relations, and (ii) emphasizing the lexicalist stance [41]. In 2016, work was conducted to enhanced SUD, this work resulted in SEUD [122]. SEUD made implicit relations between words to be more explicit by introducing further relations and augmenting relation names [122]. SEUD includes all the relations found in SUD and additional relations. Figures 6.3 and 6.4 illustrate the difference between SUD and SEUD. Figure 6.3 shows a dependency tree resulting from the application of SUD to the phrase "Sue and Paul are running". From the figure it can be seen that there is an indirect relation between the word "Paul" and the word "running". Figure 6.4 shows a dependency tree resulting from the application of SEUD to the same phrase, "Sue and Paul are running". From this figure it can be seen that a direct relation between the word "Paul" and the word "running" has been included. The direct relation between the words "Paul" and "running" is useful when Shortest Path Dependency Parsing is to be used. To give one more example, if we consider the tweet *"The UK to ban the sale of diesel and petrol cars by 2040"* (taken from the evaluation set reported on in section 6.3), using SUD the relation between "UK" and "diesel", is "ban sales cars petrol". Using SEUD the relation between "UK" and "diesel" is "ban sales cars". To sum up the distinction between the two is that relation extracted using SEUD is more meaningful than when using SUD.

Thus, from the foregoing and in the context of the proposed framework, SEUD parsing was used to extract relations between specific entity classes identified using the NER model from Stage 1; relations of the form $\langle e_1, r, e_2 \rangle$, where e_1 and e_2 are entities and r is a relation linking the two, SEUD parsing is preferable for achieving relation extraction for RDF dataset generation. The hypothesis was that the shortest path

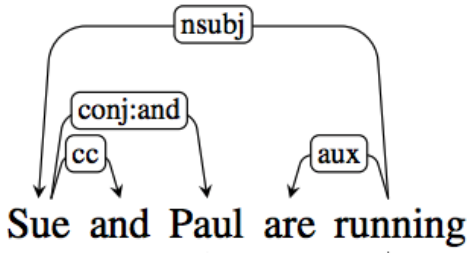

FIGURE 6.3: Example of SUD parsing applied to the sentence "Sue and Paul are running" [41].

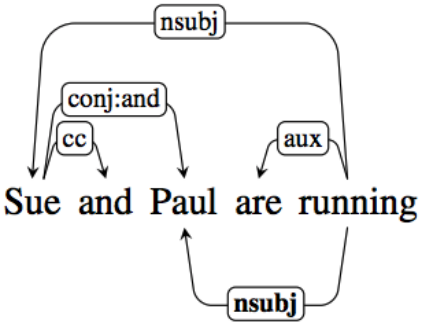

FIGURE 6.4: Example of SEUD parsing applied to the sentence "Sue and Paul are running" [122].

between pairs of entities describes the relation between e_1 and e_2, regardless of the number of words between entities and the relation [28].

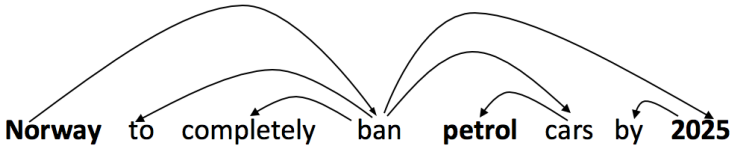

FIGURE 6.5: Example dependency parse 1.

Regarding the motor vehicle pollution example domain of discourse, Figure 6.5 shows the dependency parse for the example tweet: "Norway to completely ban petrol cars by 2025". The tweet features three entity classes "Location", "Fuel_V" and "Date". The desired relation r will then be based on the shortest path between entities. From the Figure, if the classes "Location" and "Fuel_V" were specified, the entities e_1 = "Norway" and e_2 = "petrol" would be found, and consequently the relation r = "ban cars". Alternatively, if "Location" and "Date" were chosen as the targeted classes, then e_1 = "Norway", e_2 = "2025" and r = "ban". Figure 6.6 presents the dependency parse for another example tweet: "The UK to ban the sale of diesel and petrol cars by 2040 to tackle". If the entity classes of interest are "Location" and "Fuel_V", e_1 = "UK",

e_2 = "diesel" and r = "ban sale cars". Given $e1$ = "UK" and e_2 = "petrol", then r = "ban sale cars". Otherwise, if "Location" and "Date" were chosen as the targeted classes, e_1 = "UK", e_2 = "2040" and r = "ban". Figure 6.7 summarises the foregoing two examples.

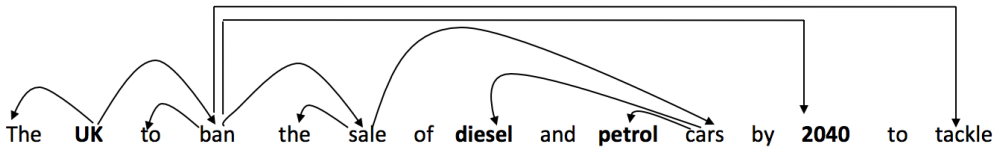

FIGURE 6.6: Example dependency parse 2.

Tweets	Entities and the relations based on the shortest path Dependency parsing
Norway to completely ban **petrol** cars by **2025**	**Norway** _____ ban cars_____ **petrol**
	Norway _____ ban _____ **2025**
The **UK** to ban the sale of **diesel** and **petrol** cars by **2040** to tackle	**UK**_____ ban sale cars_____**petrol**
	UK_____ ban sale cars_____**deiseal**
	UK_____ ban _____**2040**

FIGURE 6.7: Entities and relations based on shortest path dependency parsing found in two example tweets.

Returning to Figure 6.2, the third process in the Stage 2 workflow is the clustering of the relation labels identified from the SPDP process. The thinking here is as follows. Using the above SPDP technique, each extracted relation will comprise a set of words (with stop words), sometimes only one word. It can thus be anticipated that the sets of words extracted, each describing a relation, will feature significant dissimilarities. It is clearly desirable that generic relation labels are used to describe each relation, linking pairs of classes, that can exist in a given domain of discourse. The idea for achieving this is that the extracted relations, for each class pairing, are clustered and that labels are extracted according to the generated cluster configuration. With respect to the evaluation presented later in this thesis, a simple Nearest Neighbour Clustering (NNC) was used. For NNC to operate a similarity measure is required. With respect to the proposed SPDP-WND framework Word Mover's Distance (WMD) was used for this purpose [73, 106, 107] because of its ability to measure semantic similarity (dissimilarity).

WMD requires that the words to be compared are represented using a word embedding of some kind [73]. Word embedding is the process of learning semantically

meaningful representations of words in sentences and representing these as vectors; vectors that can be conceptualised as existing in a vector-space of some kind [117, 135]. In the reminder of this section such vectors will be indicated using the notation:

$$vec(w) \tag{6.1}$$

where w is a word of interest.

For the proposed SPDP-WMD framework, word2vec was used to generate the desired word embedding. Word2vec is a neural network based mechanism for learning word embeddings proposed by Mikolov et al. in 2013 [91, 92]. This model has attracted significant attentions in recent years and is frequently referenced in the literature [117], hence its utilisation with respect to the proposed SPDP-WMD framework. More specifically, a pretrained word2vec, skip-gram based model was used, which had been trained on 3 million words from Google news [92]. There are two fundamental models that can be used to generate embeddings using word2vec: (i) the Continuous Bag of Words (CBOW) model, and (ii) the Skip-gram model. The first operates according to frequency of context while the second operates by predicting context. The first is therefore limited by the number of examples for the usage of each word so does not work well with rarer words, unlike with the Skip-gram model. For the proposed framework the Skip-gram model was therefore used for generating word2ved embeddings.

Figure 6.8 presents an overview of WMD calculation using two example texts, Document 1 ($D1$) and Document 2 ($D2$). The non-stop words (i.e. "Obama", "Speaks", "media" and "Illinois" in $D1$ and "President", "greets", "press" and "Chicago" in $D2$) in the two documents are embedded into a word2vec space. The calculation of the distance between $D1$ and $D2$ is the minimum cumulative distance that all words in $D1$ need to travel to match the words in $D2$ in the word2vec space. Figure 6.9 illustrates the process of WMD calculation using the example documents from Figure 6.8 (documents $D1$ and $D2$, *the latter of which is considered to be the query document*) and a third document, $D3$. The distance of $D1$ and $D3$ to $D2$ are obtained by calculating the distance of each word vector in $D1$ and $D3$ with the closet semantic word in $D2$. For example, Illinois to Chicago is closer than Japan to Chicago. The reason is that vector $vec(Illinois)$ is closer to $vec(Chicago)$ than $vec(Japan)$. After calculating all the distances between words, the distance from $D1$ to $D2$ is 1.07, whilst the distance from $D2$ to $D3$ is 1.63, $D2$ is therefore said to be semantically closer to $D1$ than to $D3$.

The minimum cumulative cost of "travelling" between two documents, $\mathbf{d}$ to $\mathbf{d}'$, is calculated as follow:

$$\begin{aligned} &\underset{\mathbf{T} \geq 0}{min} \sum_{i,j=1}^{n} \mathbf{T}_{ij} c(i,j) \\ subject\,to: &\sum_{j=1}^{n} \mathbf{T}_{ij} = d_i \quad \forall i \in \{1,\ldots,n\} \\ &\sum_{i=1}^{n} \mathbf{T}_{ij} = d'_j \quad \forall i \in \{1,\ldots,n\}. \end{aligned} \tag{6.2}$$

Where: (i) we assume that we are given a matrix of word embedding $\mathbf{X} \in \mathbb{R}^{d \times n}$ for a vocabulary of n words, such that $x_i \in \mathcal{R}^d$ represents the embedding of the i^{th} word; (ii) $\mathbf{d}$ and $\mathbf{d}'$ are the normalised bag-of-words (nBOW) representations of the two documents in an n-dimensional space; (iii) $\mathbf{T}_{ij} \geq 0$ refers the extent that word i in $\mathbf{d}$ travels to match word j in $\mathbf{d}'$; and (iv) $c(i,j)$ is the cost associated with "traveling" from one word to another.

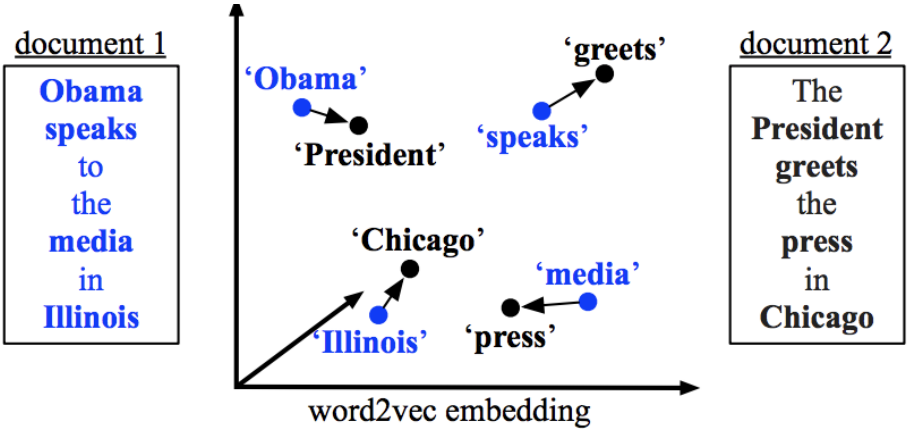

FIGURE 6.8: Overview of WMD calculation from [73].

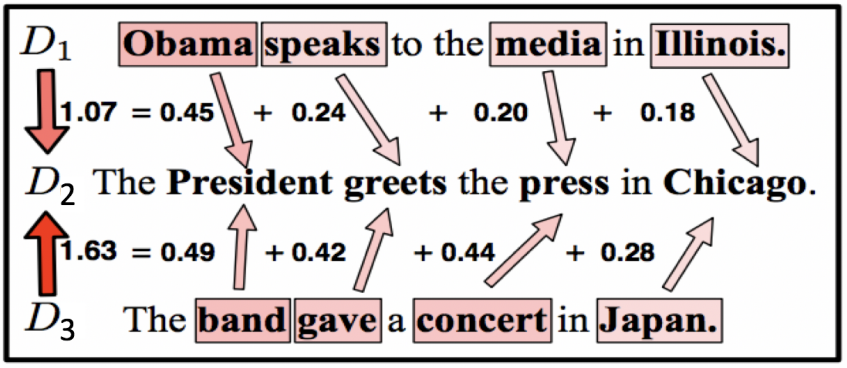

FIGURE 6.9: Example of example WMD calculation from [73].

For the proposed clustering WMD was not used directly. Instead the negative distance, d_{neg}, between two relations was used, calculated using equation 6.3. The result is a value on a scale between 0 and 1, where 1 is the most similar. For the evaluation presented later in this chapter, Nearest Neighbour Clustering (NNC) was used for the clustering. This requires a threshold σ to be set that defines "neighbourhood". Given that the purpose of the clustering is to support the identification of unique labels the specific value for this threshold is not significant, for the evaluation 0.6 was used. Figure 6.10 illustrates the clustering process with respect to a pair of entities. From the figure it can be seen that m clusters are produced. NNC is a well established clustering mechanism. In the context of the proposed framework it operates as follows. Given a set of static relation strings, $D = \{r_1, r_2, \ldots, r_n\}$, the first relation r_1 is placed in cluster 1. The similarity d_{neg} between the next relation, r_2, and cluster 1 is then calculated. If $d_{neg} > \sigma$, r_2 is added to cluster 1, otherwise a new cluster is created. The process continues until all n relations have been allocated to a class.

$$d_{neg} = \frac{1}{1 + WMD} \tag{6.3}$$

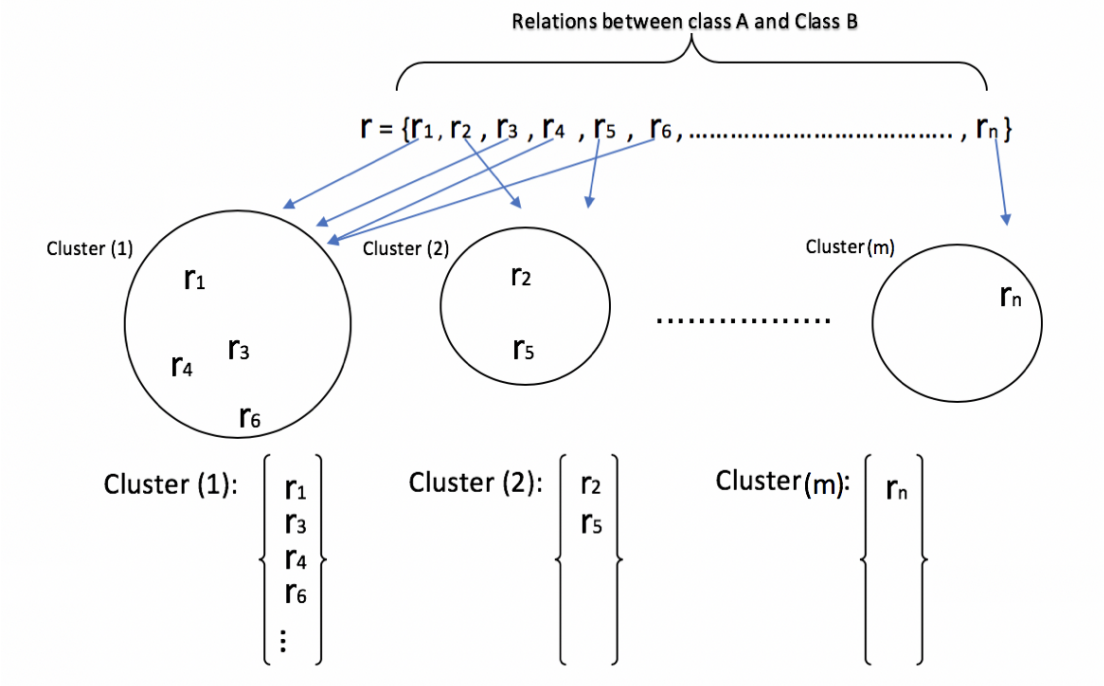

FIGURE 6.10: Cluster configuration, featuring m clusters, for relations r_1 to r_n for a given pair of entities.

The fourth process within Stage 3 (see Figure 6.2), once clustering is complete, is to select a label for each set of entity pair clusters in the generated cluster configuration

(each entity pair may be represented by more than one cluster). A variety of strategies can be considered: (i) select a label for each cluster so that the relation has a number of different labels possibly with slightly different nuances, (ii) select labels for only the largest clusters according to some threshold, or (iii) select only one label according to the largest cluster. For the proposed SPDP-WMD RDF dataset generation from Twitter data framework the third option was adopted, because it was considered desirable to have only one label per relation. The actual label used was arbitrarily selected according to one of the relation strings in the largest cluster.

Once a set of triples describing a static relation for each entity pair that exist in the input data had been established, the next process in Stage 2 was to create a training dataset with which to generate an RE model. Creation of the training data was done in an automated manner. This is important to note, because it is what distinguishes the SPDP-WMD framework from the other frameworks proposed in this thesis. The Stanford RE tool was used to generate the desired RE model because of its compatibility with the Stanford NER tool used earlier for reasons presented previously in this thesis in Chapter 5. The format for Stanford RE training records was presented earlier in Chapter 3. However, for completeness an example of a (SPDP-WMD framework) Stanford RE training record is given in Figure 6.11 using the same tweet as used as an example earlier in this section, namely: "Norway to completely ban petrol cars by 2025". Two relations are identified: (i) the relation "ban cars" between the entity "Norway" and the entity "petrol", and (ii) the relation "ban" between the entity "Norway" and the entity "2025". By comparing the training record in Figure 6.11 with the training record presented in Figure 3.4 in Chapter 3, it is clear that, as to be expected, the two records a very similar. The only difference is with respect to the relation tags used. The reason for this is that for the SPDP-WMD framework the process of **choosing** relation labels is completely automated, whilst previously labels were either identified manually (as described in Chapter 3) or using a semi-automated approach (the regular expression approach as described in Chapter 5). Some variation can therefore be expected. With respect to the evaluation reported on in Section 6.3 below, inspection of the generated relation names indicates that in all cases the relation names were all meaningful alternatives, providing more-or-less the same meaning, although it could be argue that further analysis might be appropriate here.

O	Location	0	O	NNP	Norway	O	O	O
O	O	1	O	TO	to	O	O	O
O	O	2	O	RB	completely	O	O	O
O	O	3	O	VB	ban	O	O	O
O	Other	4	O	NN	petrol	O	O	O
O	O	5	O	NNS	cars	O	O	O
O	O	6	O	IN	by	O	O	O
O	Other	7	O	CD	2025	O	O	O
0	4	ban_cars						
0	7	ban						

FIGURE 6.11: Example of a Stanford RE training data record

The final process in the Stage 2 workflow, for the proposed framework, was RE model generation. The operation of the Stanford RE tool for RE model generation was described earlier in Chapter 3 and is therefore not discussed further here. The output is a set of triple of the form $\langle subject, predicate, object \rangle$.

6.2.3 Stage 3 - RDF Dataset Generation

Once a set of triples have been extracted, the third stage of the SPDP-WMD framework was the RDF dataset generation stage. This is the same basic process as described with respect to the previous frameworks presented in Chapters 3, 4 and 5. As in the case of the previous frameworks, RDF dataset generation was conducted using Apache Jena. "rdf:Description" begins the definition of the resource being described. The "rdf:about" refers to the subject. The predicate, referred to as a property in the context of RDF, and the object were in the form: (<predicate rdf:resource="object" / >).

6.2.4 RDF Schema Generation

The same process for RDFS generation as described with respect to the previous frameworks presented in Chapters 3, 4 and 5 was adopted with respect to the proposed SPDP-WMD framework. The first stage, as in the case of the previous frameworks, was class mapping and augmentation. This involves the mapping of entities to their class and the building of class and property hierarchies by adding super-classes and super-properties (hypernyms) of the classes and properties that have been extracted. With respect to the evaluation presented in Section 6.3, two scenarios were considered, the motor vehicle pollution domain used to illustrate the operation of the various tools and techniques considered throughout this thesis, and the diabetes domain. The significance with respect to augmentation is that the diabetes domain is included in schema.org, whereas the pollution domain is not included. Thus, for the evaluation of the proposed SPDP-WMD framework both Wordnet and schema.org were used for the augmentation. The second stage was the actual the RDFS generation. The same mechanism for achieving as that used with respect to the frameworks described earlier in this thesis was adopted. Details were presented in Chapter 3; but, in brief, Apache Jena was used to constrict the RDFS by linking all the classes and proprieties.

6.3 Evaluation

In this section the evaluation of the proposed SPDP-WMD framework is presented. For the purposes of the evaluation the motor vehicle pollution Twitter datasets used with respect to earlier evaluations was again used, together with a diabetes dataset. Further details concerning diabetes dataset are presented in Sub-section 6.3.1. The objective of the evaluation was:

1. To determine the effectiveness of the RDF dataset generation process by evaluating the Stanford NER model and RE models generated using the proposed SPD-WMD.

. The results in the context of the above objective is discussed individually in Sub-section Named Entity Recognition (NER) Evaluation SPDP-WMD and Sub-section 6.3.3. Sub-section 6.3.4 present the structure of RDFS using motor vehicle pollution and diabetes datasets.

6.3.1 Diabetes Dataset

The diabetes evaluation dataset was designed to test the SPDP-WMD framework using a large unlabelled dataset. This dataset could not be used with respect to the evaluations of the three frameworks presented earlier in this thesis because these required manually labelled training data (or at least a seed set). The NER model incorporated into the SPDP-WMD framework did require training data hence the entities in 350 records of the diabetes dataset were hand labelled.

The guideline for collecting tweets regarding the diabetes domain was that the tweets had to include issues related to the risk factors, the symptoms and the treatment of diabetes using the diabetes hashtag. In total, 350 tweets were collected and labelled for NER, 1000 tweets were collected to be labelled automatically for the purposes of RE model generation, and a further 1000 tweets for populating the RDFS.

Four entity classes were identified within the diabetes dataset: (i) disease, (ii) treatment, (iii) cause, and (iv) symptom. When schema.org was used to construct the required RDFS hierarchy, four entity class names were changed to names referenced in schema.org, namely: (i) "MedicalCondition", (ii) "Treatmentindication" (iii) "MedicalCauses" and (iv) "MedicalSigneOrSymptom" respectively. Three relations were identified: (i) manage, (ii) thing_cause, and (iii) associates_symptoms. Figure 6.12 present the distribution of the classes across the NER diabetes dataset.

6.3.2 Named Entity Recognition (NER) Evaluation

Both the motor vehicle pollution and diabetes datasets were used to evaluate the NER model generation element of SPDP-WMD framework. TCV was used in both cases, the metrics used was F1-score as defined in Chapter 3. The NER results obtained using the motor vehicle pollution dataset were the same as those presented earlier in Chapter 3 (see Table 3.1). Note that the individual recall and precision values are not included in the table because the Stanford NER tool does not output this information. For the diabetes domain, the NER model was trained to identify the following four classes when WordNet was used to construct the RDFS hierarchy: (i) treatment, (ii) cause, (iii) symptom and (vi) disease; and the following four classes when schema.org was used to construct the RDFS hierarchy: (i) MedicalCondition, (ii) MedicalCauses, (iii) Treatmentindication and (iv) MedicalSigneOrSymptom.

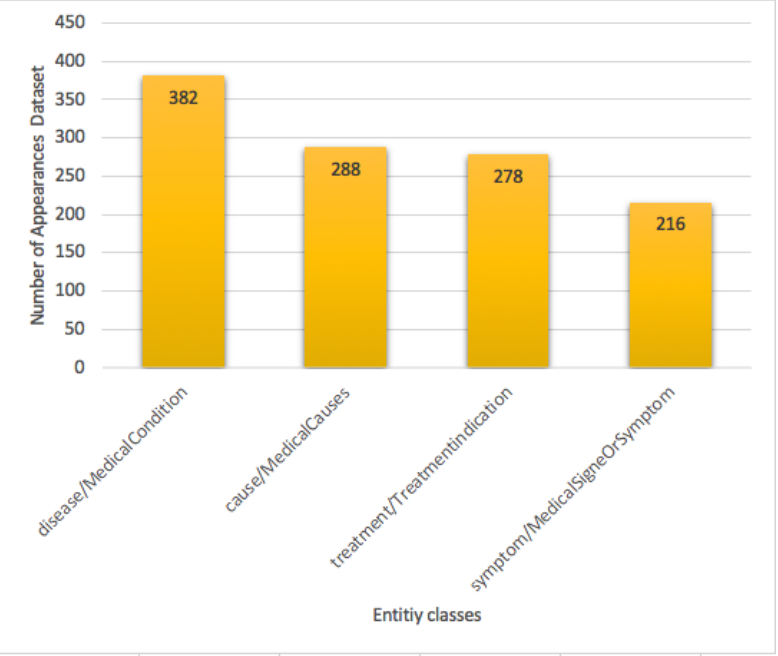

FIGURE 6.12: Distribution of the entities per class across the NER diabetes dataset.

Fold Number	F1-score
Fold 1	81.4%
Fold 2	80.4%
Fold 3	81.3%
Fold 4	81.9%
Fold 5	81.0%
Fold 6	81.7%
Fold 7	81.6%
Fold 8	80.9%
Fold 9	81.2%
Fold 10	81.4%
Average	81.3%
Standard Deviation (SD)	0.25

TABLE 6.1: TCV F1-score results for the Stanford NER model evaluation.

Table 6.1 presents the NER model F1-score evaluation results for the diabetes datasets. Again, the individual recall and precision values are not included in the table because the Stanford NER tool does not output this information. From Table 6.1 it can be observed that a small Standard Deviation was recorded, 0.25, unlike in the case of the motor vehicle pollution dataset. This was probably because the diabetes dataset was more balanced than the motor vehicle pollution dataset because it was larger. Thus, it was concluded that the NER models were consistent with respect to the larger diabetes dataset. The range of F1-scores was between 80.4% and 81.9%. The average F1-score of the ten folds was 81.3%.

6.3.3 Relation Extraction

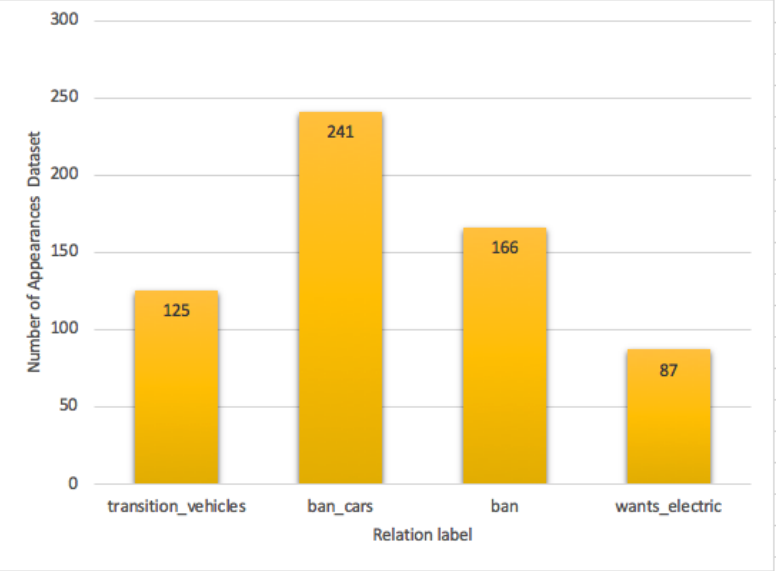

FIGURE 6.13: Distribution of the relations per class across the motor vehicle pollution dataset.

The evaluation of the RE model is presented in this sub-section. Recall that RDF dataset generation using the proposed SPDP-WMD framework featured four different relations to those described before with respect to motor vehicle pollution evaluation dataset, since an automated method of RE was utilised: (a) ban_cars, (b) transition_vehicles, (c) ban, and (d) wants_electric. Figure 6.13 illustrates the distribution of the relations across the dataset. For the evaluation ten cross validation was again used, with Precision, Recall and F1-score as the evaluation metrics. The results are presented in Tables 6.2 and 6.3. From the tables it can be observed that a wide spread of results was obtained, a high Standard Deviation, although less so for the diabetes dataset. The conjectured reason for this was due to the imbalanced nature of the dataset. The diabetes dataset was probably more balanced because it represented a larger dataset. The worst fold performance was recorded for Fold 10 of the motor vehicle pollution training data, which featured a F1-score of 52.9%. Closer inspection of this fold indicated that the recall of the relation "wants electric" was 25.0%, the RE model predicted 25% occurrences of this relation correctly, due to the small number of examples of this relation in the dataset (see Table D.1.10). The "wants electric" relation featured in the dataset only 87 times according to Figure 6.13, compared to the relation "ban cars" which featured 241 times. The overall averages precision, recall and F1-score values for the motor vehicle pollution domain were 75.8%, 74.2%, and 72.0% respectively, which seemed reasonable when compared to the Stanford approach (see Table 3.2 given in Chapter 3) because the process of preparing the training set was automated using the SPDP-WMD framework. Further detail concerning the evaluation are presented in Appendix D.

The results with respect to the diabetes evaluation training data are given in Table 6.3. From the table it can be seen that the worst fold performance was recorded for Fold

Table 6.2: TCV results for the SPDP-WMD framework RE model using the motor vehicle pollution domain

Fold Number	Precision	Recall	F1-score
Fold 1	69.5%	80.2%	74.5%
Fold 2	72.8%	72.0%	72.4%
Fold 3	91.1%	87.6%	89.3%
Fold 4	78.2%	66.3%	71.8%
Fold 5	75.0%	75.9%	75.4%
Fold 6	63.3%	67.9%	65.5%
Fold 7	66.7%	68.6%	67.6%
Fold 8	78.7%	70.0%	74.1%
Fold 9	76.1%	77.8%	76.9%
Fold 10	57.7%	48.8%	52.9%
Average	75.8%	74.2%	72.0%
Standard Deviation (Stand. Dev.)	8.5	6.5	9.2

2 of the diabetes training data, which featured a F1-score of 57.3%. Closer inspection of this fold showed that the recall of the relations "associates_symptoms" and "manage" were 40.0% and 34.0%, respectively, which meant that the RE model predicted 40% of the actual "associates_symptoms" relation and 34% of the actual "manage" relation. However, the precision scores were very high, 90.9% for "associates_symptoms" and 84.2% for "manage". This showed that most of the relations extracted by the RE system were correct. The average scores for the precision, recall and F1-score values were 81.9%, 64.4%, and 71.7% respectively. The overall result was deemed to be within an acceptable limit. The complete set of results for all ten fold are presented in Appendix D.

Table 6.3: TCV results for the SDP-WMD Stanford Relation Extraction model evaluation regarding diabetes domain

Fold Number	Precision	Recall	F1-score
Fold 1	80.9%	67.9%	73.8%
Fold 2	85.5%	43.1%	57.3%
Fold 3	78.4%	69.7%	73.8%
Fold 4	81.4%	69.1%	74.7%
Fold 5	77.8%	56.0%	65.1%
Fold 6	83.7%	64.2%	72.6%
Fold 7	79.4%	68.6%	73.6%
Fold 8	82.5%	66.4%	73.6%
Fold 9	87.9%	71.7%	79.0%
Fold 10	82.1%	67.6%	74.1%
Average	81.9%	64.4%	71.7%
Standard Deviation (Stand. Dev.)	3.1	8.6	6.1

6.3.4 RDF Schema Generation Using Motor Vehicle Pollution and Diabetes Datasets

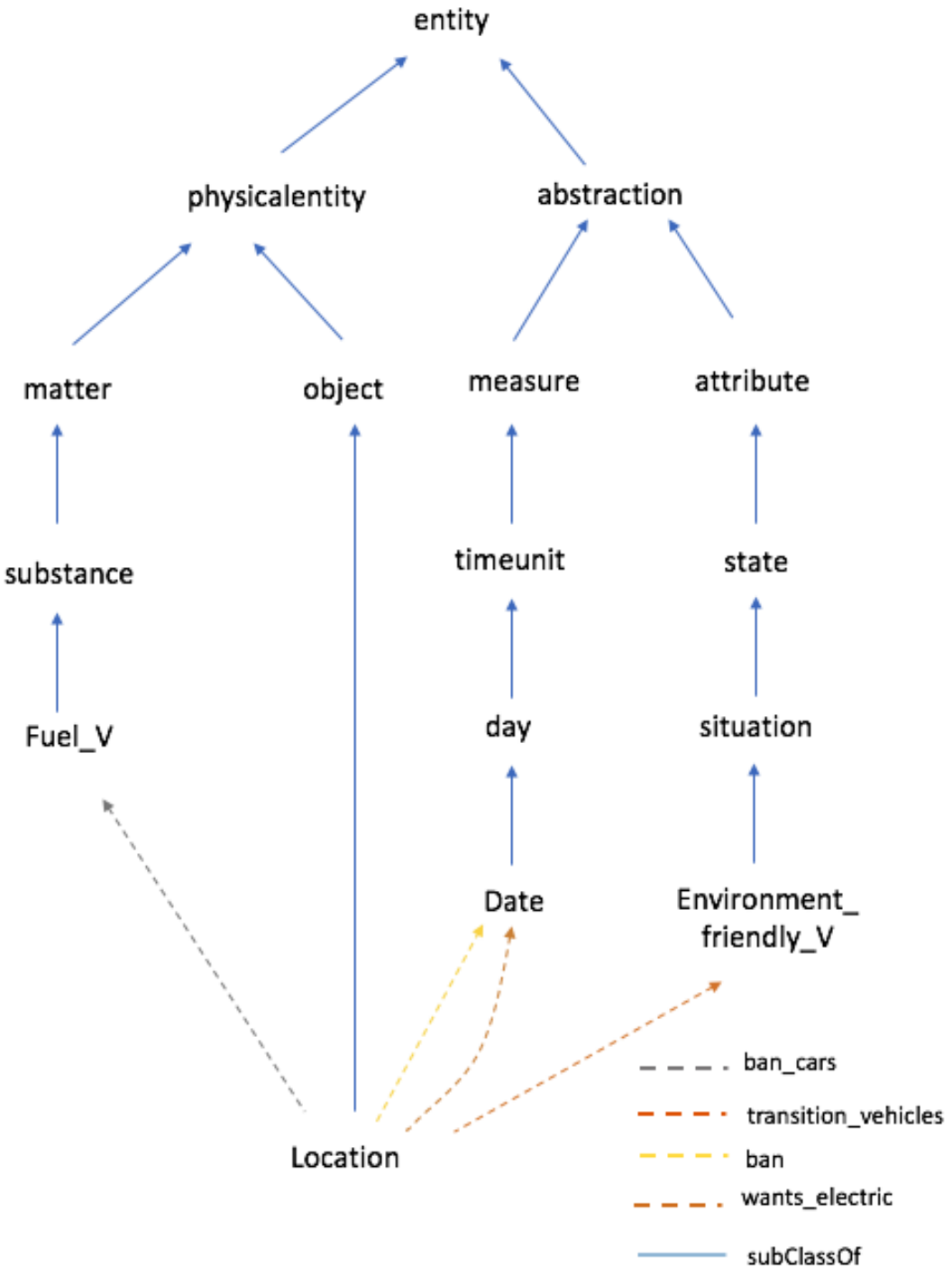

FIGURE 6.14: Node and link diagram illustrating the motor vehicle pollution RDFS and extended using WordNet.

The RDFS generated for the motor vehicle pollution dataset was the same as the RDFS presented in Figure 3.9 in Chapter 3 except that the property (relation) names

were different; however, as noted earlier, inspection of the relation names indicates that in all cases they convey the same meaning. Recall that the variation in relation names was because the process of preparing the training set for the RE model generation was automated and hence relation labels were not hand-crafted. The RDFS is presented in Figure 6.14. Of course, different property labels also impacted super-properties. Using WordNet the "Location" class had relationships with the classes "Fuel_V" and "Environment_friendly_V", namely "ban_cars" and "transition_vehicles", respectively. The "Location" class also had "ban" and "wants_electric" relationships with the "Date". The property "ban_cars" included more than one word, meaning that the first verb in the property was used to extract the super-properties. The super-properties of the property "ban_cars" were the same as property "ban" that are "outlaw", "forbid", "command", "order", "request", "ask", "communication", "convey", "transfer" and "move". Furthermore, the super-properties of "transition_vehicles" were "convert" and "change". Finally, the relation "wants_electric" did not have any super-property in WordNet.

The RDFS generated for the diabetes dataset and WordNet is given in Figure 6.15. Four classes were extracted from the tweets: (i) disease, (ii) cause, (iii) treatment, and (v) symptom; linked to each other using three properties: (i) associates_symptoms, (ii) thing_cause, and (iii) manage. "disease" class showed relationships with the classes "symptom" and "cause", namely "associates_symptoms" and "thing_cause", respectively. The "disease" class also had a "manage" relationship with the "treatment" class. From the figure it can be seen that the "disease" class had eight super-classes: "illness", "illhealth", "pathologicalstate", "physicalcondition", "condition", "state", "abstraction" and "entity". The "cause" class had seven super-classes: "origin", "beginning", "happening", "event", "psychologicalfeature", "abstraction" and "entity", and so on. Again, where a property name comprised more than one word the first verb was considered as the property label and WordNet was searched for hypernyms of this verb. As a result. the super-properties of the property "associates_symptoms", "thing_cause" were: "think" and "make", respectively. "mange" did not have any super-properties.

The RDFS generated for the diabetes dataset, and using Schema.org, is given in Figure 6.16. The reasons for including this alternative RDFS, constructed using Schema.org, were: (i) to present an example of using the process of generating RDFS using a schema repository, and (ii) Schema.org has a sector that focuses on the health field. Thus, the generate RDF will be integrable with other datasets that use the same upper level ontology. In more detail, "MedicalCondition" had two super-class: "MedicalEntity" and "Thing". "MedicalCauses" also had two super-classes: "MedicalEntity" and "Thing". "Treatmentindication" had three super-classes: "MedicalIndication", "MedicalEntity" and "Thing"."MedicalSigneOrSymptom" also had three super-classes: "MedicalCondition", "MedicalEntity" and "Thing". Note that the properties "associates_symptoms", "thing_cause" and "manage" did not have any super-properties in Schema.org.

6.4 Summary

In this chapter, the fourth and the last proposed RDF dataset from Twitter data framework has been presented, the Shortest Path Dependency Parsing and Word Mover's Distance (SPDP-WMD) RDF Dataset from Twitter Data Framework. This framework comprises three stages. The first stage consisted of using pre-labelled NER training data from the domain of discourse to create a NER model. The second stage comprised: (i) NER model application, (ii) SPDP model application, (iii) clustering of relation strings, (iv) identification of unique relation labels, (v) generation of RE training data and (vi) RE model generation. The third stage, the RDF dataset generation stage consisted of the desired RDF dataset generation. The Stanford NER and RE models were adopted with respect to the framework. Once all the desired entities and relations were extracted, the RDF dataset was generated. Then, the RDFS was generated by mapping entities into their classes. All classes and properties were augmented with a set of hypernyms. The RDF and RDFS generation was conducted using Apache Jena. The NRE and RE model generation process was evaluated using TCV applied to two evaluation datasets: the motor vehicle pollution dataset, also used in earlier chapters, and a diabetes domain dataset. The next chapter, the penultimate chapter of this thesis, presents the comparative evaluation of the four frameworks proposed in this thesis.

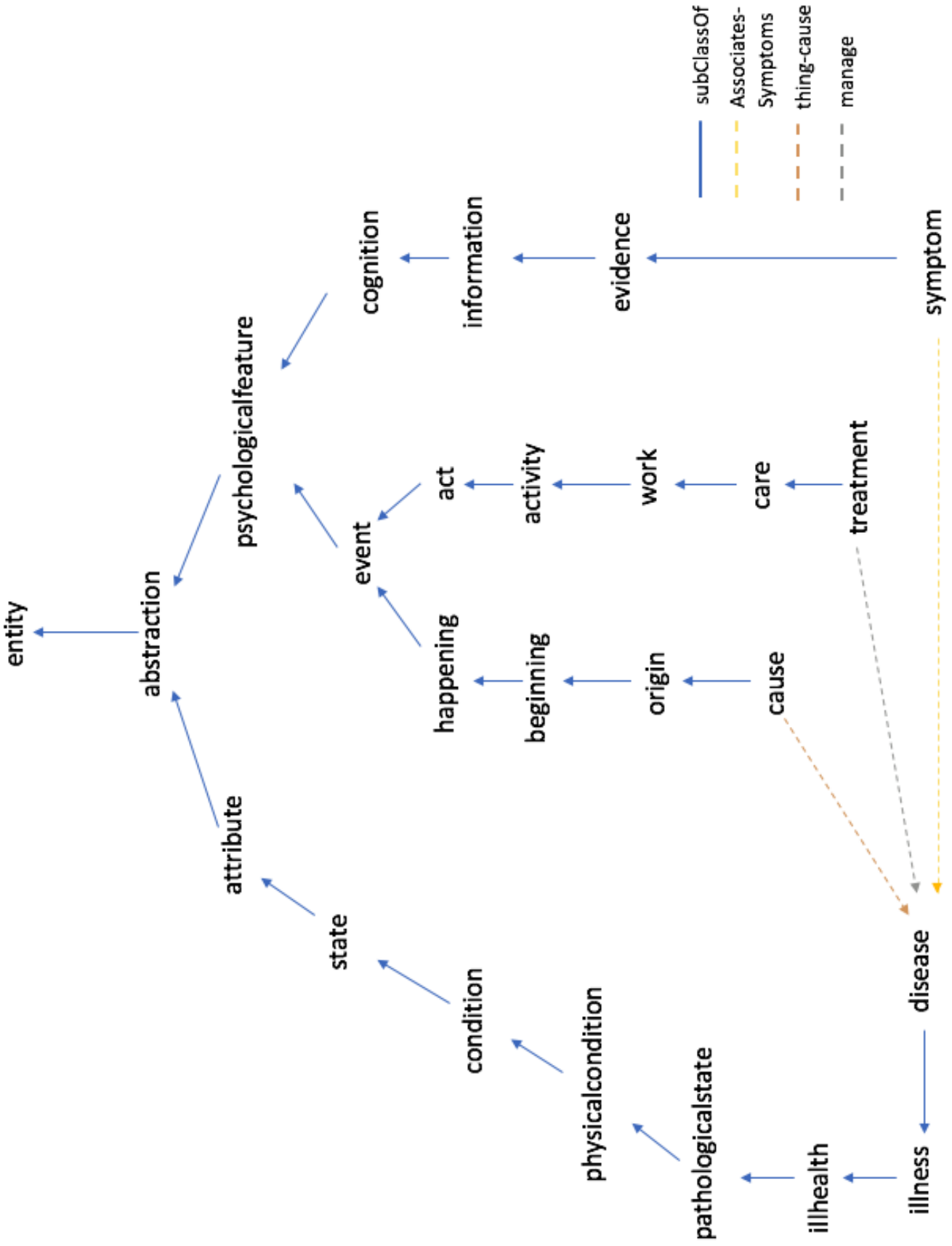

FIGURE 6.15: Node and link diagram illustrating the diabetes RDFS and extended using WordNet

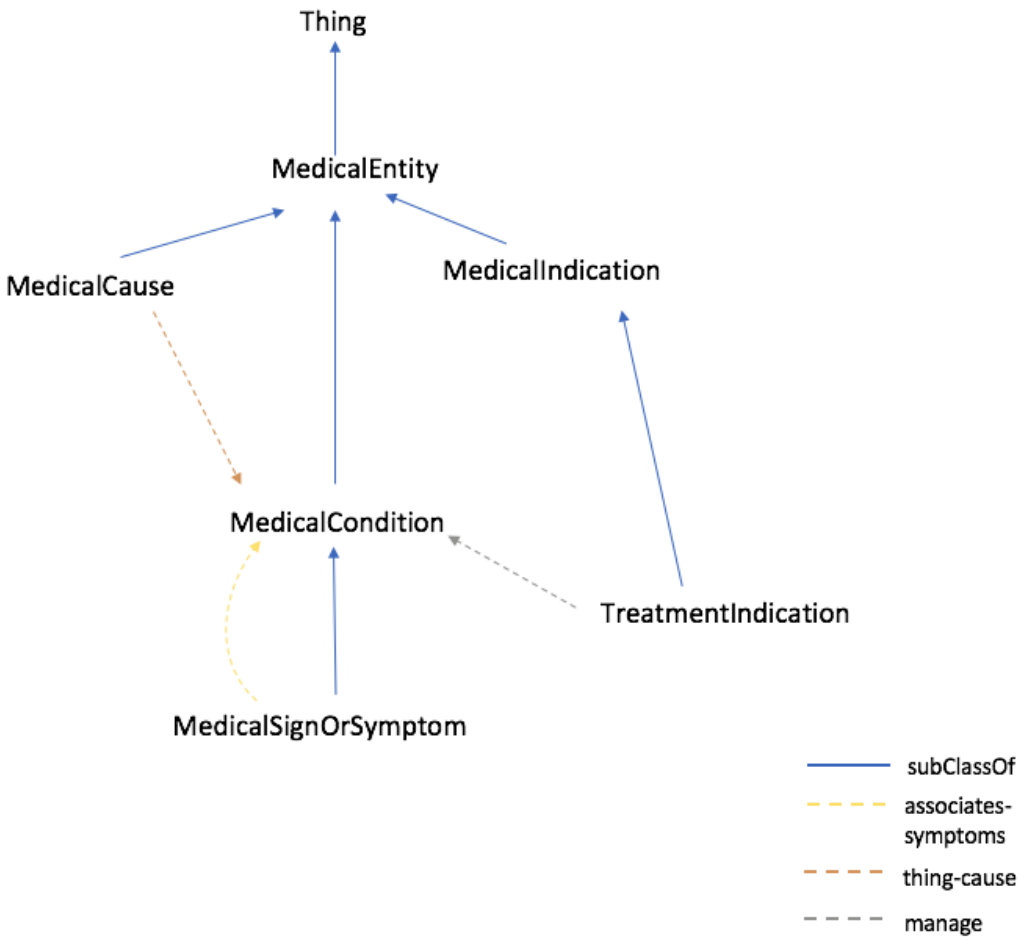

FIGURE 6.16: Node and link diagram illustrating the diabetes RDFS and extended using Schema.org.

Chapter 7

Comparative Evaluation

7.1 Introduction

The previous four chapters presented four frameworks for RDF dataset generation from Twitter data designed to support the querying of social media knowledge sources. The frameworks were as follows:

1. The Stanford CoreNLP Framework (Chapter 3)
2. The GATE NLP Framework (Chapter 4)
3. The Regular Expression Framework (Chapter 5)
4. The Shortest Path Dependency Parsing and Word Mover's Distance (SPDP-WMD) Framework (Chapter 6).

The first was a benchmark. The second was an alternative to the benchmark (using a different toolkit). Both the first and the second were supervised approaches. The third was a semi-supervised approach, and the fourth a fully unsupervised approach to RE model generation.

Each of the frameworks was analysed individually using bespoke evaluation data as reported in the corresponding chapter for each framework. In this chapter, a comparative evaluation of these frameworks is presented. The work presented in this chapter is directed at two of the subsidiary research questions from Chapter 1:

4. How do we know when a generated RDF dataset is correct?
5. How can we best cause the generated RDF dataset to be integrable with another dataset that commits the same upper level ontology?

To provide an answer the first of the above subsidiary questions, the following were considered:

- The effectiveness, in terms of accuracy, of the Named Entity Recognition (NER) and Relation Extraction (RE) model generation processes used to create RDF datasets.

- The utility of the generated RDF datasets in terms of a set of derived Competency Questions (CQs) as judged by the author and/or the domain experts collaborating with the author of this thesis.

The idea underpinning the first of the above two considerations was that if the RE and NER models featured high accuracy (in other words they find the right entities and relations) then a good set of triples will be identified which will consequently result in representative RDF. The idea underpinning the second of the above two considerations was that by querying the populated RDFS using a set of CQs this would illustrate the kinds of answers that could be obtained from Twitter data in a given domain of discourse. This in turn would then be illustrative of the utility of the generated RDF datasets.

With respect to the second of the above subsidiary questions, for an RDF dataset to be (more) integrable with other RDF datasets, the idea proposed in this thesis was that the identified entities and properties should be extended into a class and a property hierarchy. This would then provide for the potential for generated RDF datasets to be integrable with other dataset that feature the same upper level ontology; thus further enhancing the utility of the RDF datasets generated using the proposed frameworks. Two mechanisms were explored whereby this could be achieved, one using WordNet and the other using Schema.org.

For the comparative evaluation the motor vehicle pollution and diabetes Twitter datasets, also used with respect to the evaluations presented earlier in this thesis, were used.

The remainder of this chapter is organised as follows. Comparative evaluations of the NER and RE models generated with respect to each framework are presented in Sections 7.2 and 7.3 respectively. The analysis of the generated RDF datasets and RDFS is then considered in Section 7.4. The querying of populated RDFS is considered in Section 7.5. Section 7.6 then reports on the evaluation of diabetes domain RDFS dataset by clinicians who have collaborated with the author with respect to the work presented in this thesis. The chapter concludes with a summary in Section 7.7.

7.2 Comparative Evaluation of Named Entity Recognition Models

The comparative evaluation of the NER model associated with each framework is considered in this section. Recall that the GATE RDF dataset from Twitter data framework, presented in Chapter 4, used the ANNIE gazetteer founded on user defined *entity lists* (also sometimes referred to as entity dictionaries) to identify entities. Thus, the ANNIE gazetteer is as good as the entity list it is provided with, and thus is not evaluated any further here. The remaining frameworks used the Stanford NER model, although in the case of the Regular Expression Framework this was trained using only a seed set. All three frameworks were tested using the motor vehicle pollution dataset, with 300

tweets for the Stanford and SPDP-WMD frameworks, and 100 tweets for the Regular Expression framework. k-fold cross-validation (kCV) was used throughout with $k = 10$ in the case the Stanford and SPDP-WMD frameworks, and $k = 3$ in the case of the Regular Expression framework. The latter because the seed set comprised only 100 tweets, which meant that TCV would be unsuitable because there would only be 10 records in the test set for each run, which would be insufficiently representative of the data as a whole. The evaluation metric used was F1-score.

A good F1-score means that the number false positives and false negatives is low. In other words that NER model under consideration identifies the majority of entities we are interested in and only a few items are incorrectly identifying as entities of interest. An F1-score of 1.00 is a perfect F1-score thus indicating a perfect NER model, while an F1-score 0.00 indicates an entirely imperfect NER model. In practice, regardless of the application domain, the models we generate typically have an F1-score of less than 1.00. However, just because a model does not feature an F1-score of 1.00 does not mean it is not useful. The threshold for what is a "useful" F1-score is of course difficult to define, and subjective according to the nature of the application under consideration. However, in the case of the NER models considered the "acid test" is whether the NER models result in appropriate RDF datasets reflecting the Twitter domain of discourse under consideration. As will be evidenced later in this chapter, it was found that NER models with F1-scores of 70.0% still resulted in usable RDF datasets.

The results of the NER model evaluation are presented in Table 7.1. It was conjectured that the greater the number of tweets used to generate a NER model, the better a NER model's performance. This conjecture is supported the results presented in Table 7.1. The average F1-score for the Stanford and SPDP-WMD frameworks, using a set of 300 tweets, was 78.0%; that for the Regular Expression framework, using only a seed set of 100 tweets, was 52.3%. Inspection of the table also indicates a small Standard Deviation (SD) of 0.71 for the Stanford and SPDP-WMD frameworks, and a larger 3.0 for the Regular Expression framework, indicating a greater degree of variance with respect to the latter. It cab thus be seen that NER model generation using only a small seed set, as in the case of the Regular Expression framework, does not produce as good a NER model than when a larger training set is used as in the case of Stanford and SPDP-WMD frameworks, although at the expense of the resource required to generate the training data. However, the Regular Expression framework still served to identify appropriate entities for the purpose of generating regular expression patterns as evidenced by effectiveness of the resulting RE model as discussed in the following Section 7.3.

The Stanford NER model used in the Standard and SPDP-WMD frameworks was also evaluated using 350 tweets taken from the Diabetes dataset. The results obtained were given in Table 6.1 of Chapter 6 and, for convenience are presented again in Table 7.2. From the table it can be seen that an average F1-score of 81.3% was obtained with a standard deviation of 0.25.

TABLE 7.1: TCV F1-score results for the Stanford NER models used in the Stanford and SPDP-WMD RDF frameworks described in Chapters 3 and 6; and 3CV F1-score results for the Regular Expression framework described in Chapter 5 with respect to the motor vehicle pollution dataset.

Fold Num.	Stanford and SPDP-WMD	Regular Expression
Fold 1	77.3%	49.0%
Fold 2	77.7%	53.0%
Fold 3	77.2%	55.0%
Fold 4	79.3%	
Fold 5	77.2%	
Fold 6	78.0%	
Fold 7	78.0%	
Fold 8	78.9%	
Fold 9	78.4%	
Fold 10	78.3%	
Average	78.0%	52.3%
Standard Deviation (Stand. Dev.)	0.71	3.0

TABLE 7.2: TCV F1-score results for the evaluation of the Stanford NER model used in the SPDP-WMD RDF framework described in Chapter 6 with respect to the diabetes dataset.

Fold Number	F1-score
Fold 1	81.4%
Fold 2	80.4%
Fold 3	81.3%
Fold 4	81.9%
Fold 5	81.0%
Fold 6	81.7%
Fold 7	81.6%
Fold 8	80.9%
Fold 9	81.2%
Fold 10	81.4%
Average	81.3%
Standard Deviation (Stand. Dev.)	0.25

7.3 Comparative Evaluation of Relation Extraction (RE)

The comparative evaluation of the RE model associated with each framework is considered in this section. TCV was used throughout. The results obtained using the motor vehicle pollution evaluation dataset are presented in Table 7.3. Recall that the Stanford, Regular Expression and SPDP-WMD frameworks all used the Stanford RE model although generated in different manners. Recall also that the GATE framework used the GATE RE model. From Table 7.3 it can be seen that all the frameworks acceptable F1-scores appropriate for RDF dataset generation, scores equal to or grater than 70.0%, with the possible exception of that produced using the GATE framework, as evidenced

by the example RDF datasets considered later in this chapter. The Stanford RE model produced the best average F1-score of 79.5%, but at the expense of considerable effort to hand label a RE training set. The proposed Regular Expression framework produced the second pest performance with an Average F1-score of 74.5%. This is interesting because the NER model used with this framework, trained using only a seed set produced the worst performing NER model (an average F1-score of 52.3% as discussed in Section 7.2), but this does not seen to have had a significant effect on the RE model generation as evidenced by the results presented in Table 7.3. The proposed SPDP-WMD framework produced an average F1-score of 72.0%, within acceptable limits, but with the benefit of automated RE model generation. The GATE framework produced the worst average F1-score of 68.2%, thus justifying the use of the Stanford RE model with respect to the Regular Expression and SPDP-WMD frameworks. From the table it can also be seen that the Standard Deviation values were obtained in each case were higher than in the case of the NER model evaluation, indicating a greater degree of variability. A possible explanation for this is the imbalance within the motor vehicle pollution dataset in that some relations appear much more frequently than others (see Chapters 3 and 6).

Table 7.3: TCV F1-Scores for the evaluation of the RE models used in the Stanford, GATE, Regular Expression and SPDP-WMD frameworks with respect to the motor vehicle pollution dataset.

Fold Num.	Stanford	GATE	Regular Expression	SPDP-WMD
Fold 1	76.7%	70.6%	78.1%	74.5%
Fold 2	85.0%	79.2%	88.5%	72.4%
Fold 3	78.4%	71.1%	69.3%	89.3%
Fold 4	96.1%	57.6%	75.6%	71.8%
Fold 5	56.1%	50.8%	73.7%	75.4%
Fold 6	79.3%	82.1%	70.3%	65.5%
Fold 7	75.3%	61.4%	73.5%	67.6%
Fold 8	72.7%	70.0%	78.5%	74.1%
Fold 9	87.9%	72.0%	56.9%	76.9%
Fold 10	88.1%	66.8%	81.0%	52.9%
Average	79.5%	68.2%	74.5%	72.0%
Standard Deviation (Stand. Dev.)	10.9	9.4	8.3	9.2

Stanford RE model generation with respect to proposed SPDP-WMD framework was also evaluated using the Diabetes dataset. The results obtained are presented in Table 7.4. From the table it can be seen that an average F1-score of 71.7% was obtained with a standard deviation of 6.1. A result similar to that obtained when using the motor vehicle pollution data set (see Table 7.3), indicating consistency in the performance of the Stanford RE model, at least in the case of the SPDP-WMD framework.

Table 7.4: TCV results for the SDP-WMD Stanford Relation Extraction model evaluation regarding diabetes domain

Fold Number	F1-score
Fold 1	73.8%
Fold 2	57.3%
Fold 3	73.8%
Fold 4	74.7%
Fold 5	65.1%
Fold 6	72.6%
Fold 7	73.6%
Fold 8	73.6%
Fold 9	79.0%
Fold 10	74.1%
Average	71.7%
Standard Deviation (Stand. Dev.)	6.1

7.4 Comparison of Generated RDF Datasets and RDF Schemas

As noted earlier, the frameworks described in this thesis were evaluated using two evaluation datasets drawn from the domains of motor vehicle pollution and diabetes. Details of these datasets were given previously in Chapters 3 and 6. In each case the desired RDF datasets were generated using apache Jena. The Stanford, GATE and Regular Expression frameworks, when used with respect to the motor vehicle pollution dataset, all produced the same RDF dataset. This was because:

1. **NER**: The same entity names were used in all three cases. The Stanford NER models were trained using these entity names, although only a seed set was used with respect to the Regular Expression framework. The same entity names were used to produce the entity list required by the ANNIE Gazetteer.

2. **RE**: The RE model used in Stanford and GATE frameworks were trained to produce the same set of relations, and these relation names were also used with respect to the regular expression patterns defined in the case of the Regular Expression framework.

In the case of the proposed SPDP-WMD framework, inspection of the generated RDF dataset using the motor vehicle pollution dataset (see Chapter 6) showed that the entity names were the same but the relation names were different. This was because, in the case of the SPDP-WMD framework, with respect to the motor vehicle pollution dataset, the Stanford NER model was trained using the same training dataset as used with respect to the evaluation of the previous frameworks, but the relation names were different because these were derived in an automated manner by the framework, although it should be noted that manual inspection indicated that they featured almost the same meaning in each case. Further inspection could be done regarding this matter.

It can be argued that, because of human intervention, the hand crafted relation names used with respect to the Stanford, GATE and Regular Expression frameworks are more meaningful than those produced using the SPDP-WMD framework. However, the SPDP-WMD framework provides the advantage of not requiring RE training data, and hence comes with significant efficiency advantages. The SPDP-WMD framework was also used to generate a Diabetes RDF dataset.

The RDF dataset for both the motor vehicle pollution and diabetes domains of discourse were enriched to allow the RDF datasets to be readily integrated with other RDF datasets that had the same upper level ontology. Initial experiments, using the motor vehicle pollution dataset, were conducted using WordNet to enrich the RDF. Later experiments conducted using the diabetes dataset, used both WordNet and Schema.org. The reasons for using Schema.org with the diabetes domain of discourse was to investigate alternatives with respect to how best to create the hierarchies designed to increase the integrability of the dataset with other datasets that had the same upper level ontology. Schema.org had a "sector" that focuses on the health domain, the intuition was therefore that this might provide for better integrability than WordNet which did not provide this focus. Note that Schema.org does not cover the pollution domain of discourse so could not be used to extend the motor vehicle pollution dataset. Node and link diagrams for the RDFS are given in Figures 7.1, 7.2, 7.3 and 7.4. Figures 7.1 presents the RDFS generated using the Stanford, GATE and Regular Expression frameworks, when applied to the motor vehicle pollution evaluation dataset, extended using WordNet; whilst Figure 7.2 presents the RDFS generated using the SPDP-WMD framework, when applied to the motor vehicle pollution dataset, extended using WordNet. Figure 7.3 presents the RDFS generated using the SPDP-WMD framework, when applied to the diabetes evaluation dataset, extended using WordNet; whilst Figure 7.4 presents the RDFS generated using the SPDP-WMD framework, when applied to the diabetes dataset, but extended using Schema.org.

From the Figures 7.1, 7.2, and 7.3 it is noticeable that the WordNet RDFS files are complex in terms of the hierarchy structure and general in meaning. The reason is that when using WordNet the focus is on linguistics hypernyms. With specific reference to the diabetes RDFS graphs given in Figures 7.3 and 7.4 it can be argued that the RDFS generated using Schema.org was more focused on the health domain than the WordNet RDFS. Also, it can be argued that the super-classes acquired using Schema.org achieve better the idea of sharing a common understanding of the structure of information. This is largely because the Schema.org is defined as an ontology while WordNet is a lexical dataset. Given the above it is concluded here that the generated RDFS using Schema.org seems to be more integrable than that generated using WordNet.

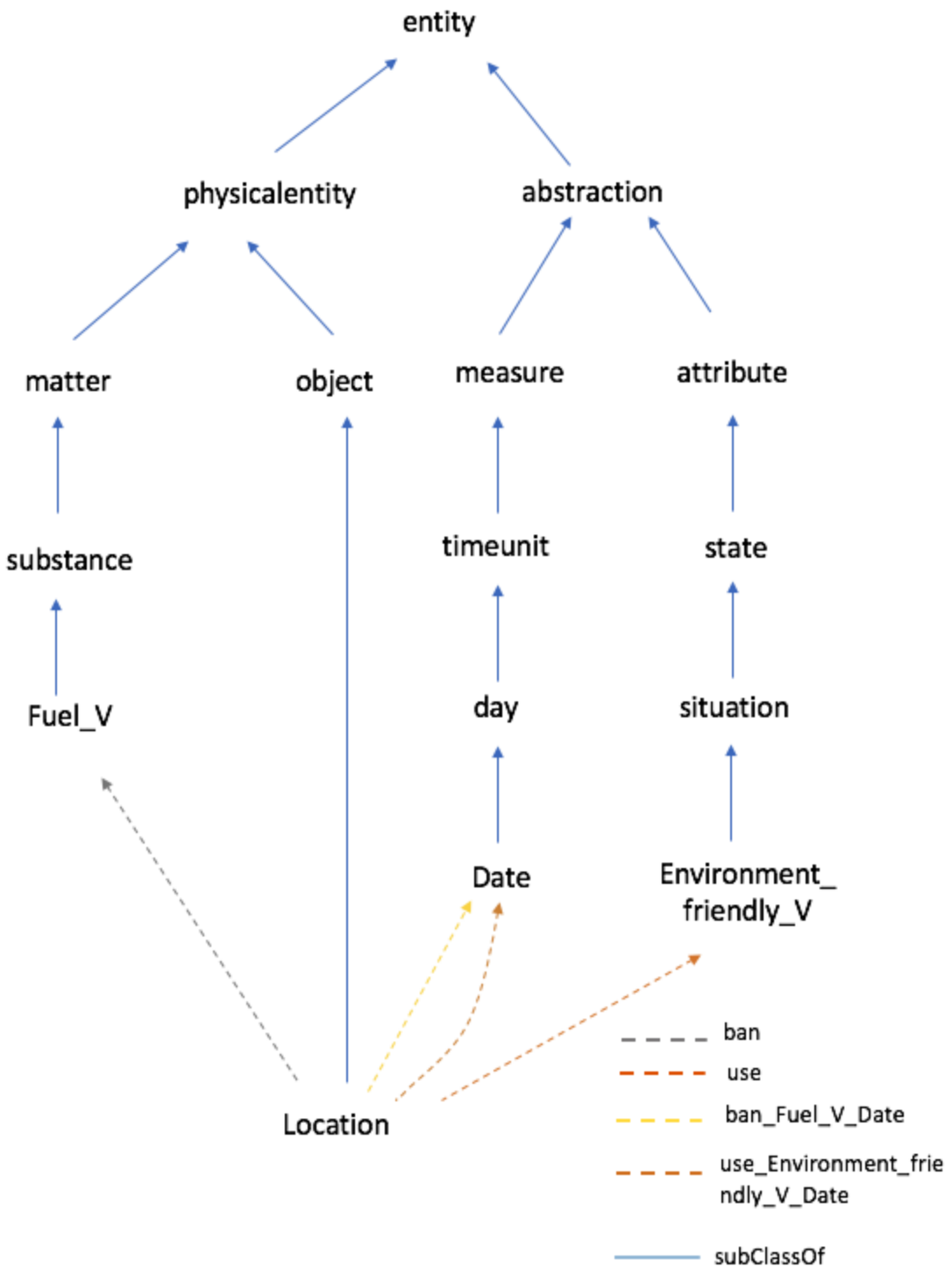

FIGURE 7.1: Node and link diagram illustrating the motor vehicle pollution RDFS, generated using the Stanford, GATE and Regular Expression frameworks and extended using WordNet.

7.5 Querying the populated RDF Schema

Recall that the motivation for the work presented in this thesis was to extract useful information (actionable knowledge) from social media data. The central idea postulated

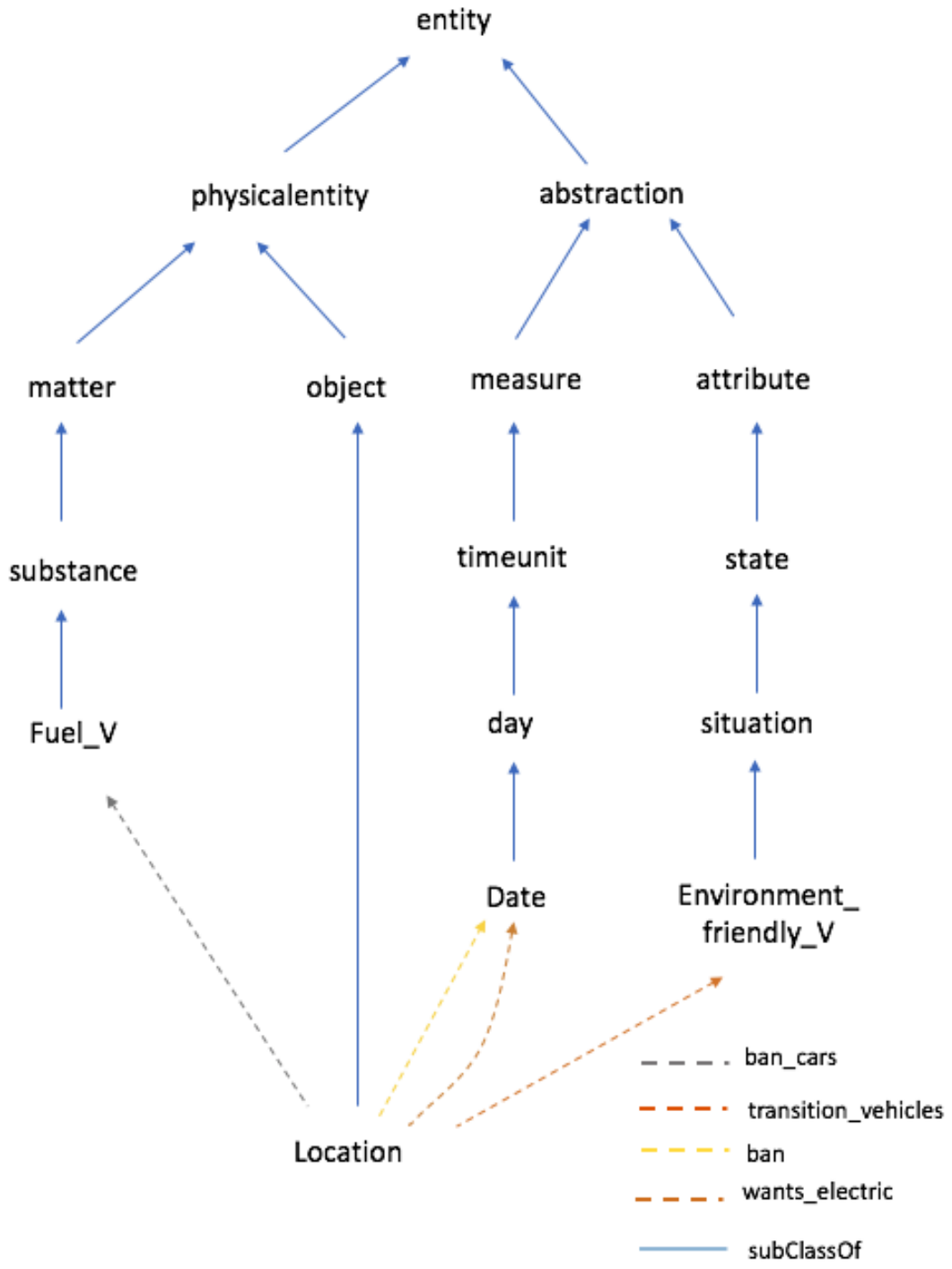

FIGURE 7.2: Node and link diagram illustrating the motor vehicle pollution RDFS, generated using the SPDP-WMD framework and extended using WordNet.

in this thesis is that this can best be done by imposing structure onto the social media data by translating it into an RDF format representative of a desired domain of discourse. Then, an RDFS generated to allow the the RDF dataset to be integrated into another dataset that commits the same upper level ontology. The RDFS was populated with further twitter data, 311 for motor vehicle pollution domain and and 1000

for the diabetes domain, in addition to that used to generate the original RDFS, and then queried for the purpose of extracting knowledge.

A set of Competency Questions (CQs) were then designed to give an indication of the kind of information that could be extracted from the populated RDFS. The CQs were expressed using the SPARQL query language; this was chosen because of its ability to express queries across diverse data sources, whether the data was stored natively as RDF or viewed as RDF via some form of middleware [111]. The SPARQL querying was facilitated using Apache Jena, one of the recommended tools from W3C. The idea of using CQs was influenced by the work of [15] where they were used to:

1. Identify the main classes of an RDFS, the relationships between these classes and the individuals (entities) belonging to the classes, and
2. Enable the developer to ensure that the populated RDFS was able to retrieve appropriate knowledge.

For the Motor Vehicle Pollution (MVP) domain, 6 CQs were generated as follows (MVP-CQs):

MVP-CQ1 What are the names of the locations mentioned in the given Twitter collection?

MVP-CQ2 What are the types of fuel motor vehicles that are mentioned in the given Twitter collection?

MVP-CQ3 What are the types of environmentally friendly motor vehicle that are mentioned in the given Twitter collection?

MVP-CQ4 What are the names of locations mentioned in the given Twitter collection that intend to ban fuel vehicles?

MVP-CQ5 When will California ban fuel vehicles?

MVP-CQ6 When will California require environment friendly Vehicle?

The results obtained are summarised in the following listings (the results in full are included in Appendix E). In each case the relevant SPARQL code is given on the left and the results obtained on the right. Note that the answers obtained reflect the content of the Twitter data and consequently the opinions of those posting the tweets. Experiments were conducted using the motor vehicle RDFS generated using the Stanford, GATE and Regular Expression frameworks (Figure 7.1) and that generated using the SPDP-WMS framework (Figure 7.2). The CQ results were the same in both cases.

Inspection of the listings for the MVP-CQs indicates that: (i) "location" was interpreted as entities belonging to the class "Location", (ii) "categories of motor vehicles" as entities belonging to the class "Fuel_V", (iii) "types of environmentally friendly motor vehicle" as entities belonging to the class "Environment_Friendly", etc. Inspection of the results indicated that the anticipated knowledge was extracted.

```
MVP-CQ1:                                                    | Result:
                                                            |
SELECT DISTINCT ?Location                                   | -Norway
WHERE {                                                     | -CA
?Location rdf:type NS:Location                              | -UK
      }                                                     | -California
```

```
MVP-CQ2:                                                    | Result:
                                                            |
SELECT DISTINCT ?fuelv                                      | -petrol
WHERE {                                                     | -diesel
?fuelv rdf:type NS:Fuel-V                                   | -fossil-
                                                            |  fuel
      }                                                     | -fossil-
                                                            |  fueled
```

```
MVP-CQ3:                                                    | Result:
                                                            |
SELECT DISTINCT ?EnvironmentFriendlyV                       | -hybrid
WHERE {                                                     | -Electric
?EnvironmentFriendlyV rdf:type NS:Environment-friendly-V    | -EVs
     }                                                      | -EV
```

```
MVP-CQ4:                                                    | Result:
                                                            |
SELECT DISTINCT ?Location                                   | -Norway
WHERE {                                                     | -CA
?Location rdf:type NS:Location.                             | -UK
?Location NS:ban ?fulev.                                    | -California
      }                                                     |
```

```
MVP-CQ5:                                                    | Result:
SELECT DISTINCT ?Date                                       | -2040
WHERE {                                                     |
NS:California NS:ban-Fuel-V-Date ?Date                      |
      }                                                     |
```

```
MVP-CQ6:                                                    | Result:
SELECT DISTINCT ?Date                                       | -2040
WHERE {                                                     | -2030
NS:California NS:use-Environment-friendly-V-Date  ?date     |
     }                                                      |
```

For the Diabetes domain, 9 CQs were derived as follows (D-CQs):

D-CQ1 What are the causes of diabetes?

D-CQ2 What are the symptoms of diabetes?

D-CQ3 How can patients manage their diabetes?

D-CQ4 What kind of care is provided for an individual who has a disease?

D-CQ5 What kind of evidence is required to determine whether an individual has disease?

D-CQ6 What diseases are there mentioned in the given Twitter collection?

D-CQ7 What types of treatment are there mentioned in the given Twitter collection?

D-CQ8 What types of symptoms are there mentioned in the given Twitter collection?

D-CQ9 What kinds of physical condition does the person need a treatment for in the context of the given Twitter collection?

The translation into SPARQL for the above Diabetes CQs (D-CQs) is given in the following listings. As in the case of the earlier listings the SPARQL code is given on the left and the obtained results on the right, more detail of the results could be found in the Appendix E. Again, inspection of the results indicates that the answers to the queries were as anticipated indicating that appropriate knowledge was extracted.

```
D-CQ1:                                          | Result:
                                                |
SELECT DISTINCT ?causes                         | -Stress
WHERE {                                         | -Sugar
?causes NS:thing-cause NS:diabetes              | -carbohy-
      }                                         | drate
                                                | -Obesity
```

```
D-CQ2:                                          | Result:
                                                |
SELECT DISTINCT ?symptoms                       | -anxiety
WHERE {                                         | -hungry
?symptoms NS:associates-symptoms NS:diabetes    | -lupus
            }                                   | -Hypergly-
                                                |  cemia
                                                | -dizziness
```

```
D-CQ3:                                          | Result:
                                                |
SELECT DISTINCT ?treatment                      | -Exercise
WHERE {                                         | -Diet
?treatment NS:manage NS:diabetes                | -drugs
      }                                         | -insulin
                                                | -keto
```

D–CQ4:

```
SELECT DISTINCT ?care
WHERE {
?care rdfs:subClassOf       NS:care.
?relationclass rdfs:range       NS:disease.
?relationclass rdfs:domain  ?care
      }
```

Result:

-treatment

D–CQ5:

```
 SELECT DISTINCT ?evidence
 WHERE {
 ?evidence rdfs:subClassOf NS:evidence.
 ?relationclass rdfs:range NS:disease.
 ?relationclass rdfs:domain ?evidence
       }
```

Result:

-symptom

D–CQ6:

```
 SELECT DISTINCT ?disease
 WHERE {
 ?disease rdf:type NS:disease
       }
```

Result:

-diabetes
-diabetics
-diabetic
-diabetess

D–CQ7:

```
 SELECT DISTINCT ?treatment
 WHERE {
 ?treatment rdf:type NS:treatment
        }
```

Result:

-Exercise
-Diet
-drugs
-Insulin
-keto

D–CQ8:

```
 SELECT DISTINCT ?symptom
 WHERE {
 ?symptom rdf:type NS:symptom
       }
```

Result:

-anxiety
-hungry
-lupus
-Hypergly-
cemia
-dizziness

D–CQ9:

```
SELECT DISTINCT ?physicalcondition
```

Result:

-disease

```
WHERE {                                                    |
?physicalcondition rdfs:subClassOf* NS:physicalcondition   |
?relationclass rdfs:domain NS:treatment.                   |
?relationclass rdfs:range ?physicalcondition               |
      }                                                    |
```

Inspection of the results obtained using the D-CQs suggests that the proposed frameworks operated as intended. Some query results were incorrect. For example, "stress causes diabetes" is incorrect, based on consultation with clinicians collaborating with the author of this thesis as discussed in the following section. The reasons behind this was because the RDFS represented Twitter data and consequently the views if the authors of the individual tweets. In this case some of the tweet authors incorrectly believed that stress causes diabetes. The clinicians though the lack of knowledge of the nature of diabetes on behalf of Twitter users, as reflected by the tweets posted, was of interest in its own right.

7.6 Consultation with Human Experts

The final set of comparative evaluations conducted, and reported on in this chapter, was visual inspection by human users as suggested in [22]. The visual inspection of the motor vehicle pollution domain RDFS was conducted by the author of this thesis, because this was a domain that was readily understandable. This was the reason for choosing this domain of discourse as one of the evaluation domains to be utilised with respect to the work presented in this thesis. However, the author did not have the specialised knowledge required to judge the diabetes domain RDFS. Consequently the author collaborated with a number of clinicians so as to seek insight as to whether the class names and relationship names were meaningful given the diabetes domain. This was not a formal evaluation of any kind, but more a discussion with interested parties on how the proposed RDF dataset from Twitter data frameworks could be utilised in the context of the diabetes domain. The diabetes RDFS generated using the SPDP-WMD framework coupled with WordNet to build the hierarchy structure was presented to the human experts, drawn from the medical domain, who had agreed to collaborate with the author. The interaction with the collaborators was limited as the author did not want to overwhelm the experts who, at time of writing (2020), were very much engaged with the COVID-19 pandemic.

With respect to the motor vehicle pollution RDFS visual inspection by the author of this thesis indicated that the RDFS was appropriate, reflecting the tweets used to generate it. The RDFS in general was good. The classes and super-classes seemed to reflect the domain of discourse, and the relation names were considered to be appropriate in that they described the intended relations between classes. This was not surprising with respect to the Stanford, Gate and Regular Expression frameworks because the

relation names had been hand-crafted. However, this was also true for the SPDP-WMD framework where the relation names were generated automatically. Although it was noted that the relation "ban", linking the classes "Location" and "Date", could have been better expressed as "ban_Fuel_V_Date". Overall, the generated motor vehicle pollution RDFS files seemed appropriate.

With respect to the diabetes RDFS consultation was conducted using a template which was sent to five individuals who had agreed to collaborate with the author, to give insight as to whether the class names and relationship names were meaningful given the domain (diabetes). The individuals were all clinicians working in hospitals in the Merseyside region, as therefore domain experts, as follows:

1. A thyroid disease specialist.
2. An accident and Emergency (A&E) doctor and specialist in emergency medicine.
3. The president of the British Pharmacological Society.
4. A clinical lecturer and practitioner in clinical pharmacology and therapeutics.
5. A consultant in diabetes and general paediatrics.

The evaluation template is given in Appendix E. The template comprised four sections: (i) introduction, (ii) description of the RDFS, (iii) querying the populated RDFS, and (iv) RDFS relation names. The first section described the purpose of the template and the nature of the research. The second section provided a description of the diabetes RDFS. Examples of the entities belonging to each class (disease, cause, treatment and symptom), once the RDFS had been populated, were also provided. The third section listed the first five of the D-CQs listed above and the results obtained. For convenience these are listed again below together with the result obtained:

D-CQ1 What are the causes of diabetes?

Result Obesity, Stress, Sugar, carbohydrate.

D-CQ2 What are the symptoms of diabetes?

Result Hyperglycemia, anxiety, dizziness, hungry, hypos, lupus, blood pressure.

D-CQ3 How can patients manage their diabetes?

Result Exercise, Insulin, activity, diet, drugs, keto.

D-CQ4 What kind of care is provided for an individual who has a disease?

Result Treatment.

D-CQ5 What kind of evidence is required to determine whether an individual has disease?

Result Symptom.

The third section ended with the question: "Are the responses provided by the populated RDFS meaningful?". The fourth section was directed at relation names. The experts were asked three questions:

1. Do you think "manage" describes the relationship between the class treatment and the class disease?
2. Do you think "thing-cause" describes the relationship between the class cause and the class disease?
3. Do you think "associated-symptoms" describes the relationship between the class symptom and the class disease?

The fourth section ended with the question: "Are the generated relationship names appropriate?". Given the limited size of the dataset, a summary of the anecdotal responses from each expert is given below:

Expert 1, Thyroid disease specialist:

The generated RDFS was realistic. The "manage" and "associated_symptoms" were meaningful relation names. However, the relation name "thing_cause" was not considered to be easily understandable. Furthermore, some of the answers were not correct. For example "stress" is not one of the causes of "diabetes".

Expert 2, Accident and Emergency (A&E) doctor and a specialist in emergency medicine: The relationships and sub-classes seemed to map onto reality quite well. However, it would have been good to have more classes, such as treatment (medical) and treatment (lifestyle). Furthermore, although most of the answers in section three in the evaluation template were correct, some of the answers were incorrect. For example, "lupus" is not one of the symptoms of "diabetes".

Expert 3, The president of the British Pharmacological Society: The classes and the relations between classes were fine. A disease has a cause, the disease has treatments, and the diseases produces a number of symptoms. The relations could be considered along with more diseases (not only diabetes). Also, different symptoms of diabetes could be considered because new symptoms may develop as the diseases progressed. Moreover, the expert felt that the RDFS was realistic if it could take into account the complexity of the disease. The answers to the questions in Section three of the evaluation template were acceptable but it would be better if different disease sub-types, the management of different disease subtypes, and different symptoms at different disease stages was included.

Expert 4, Clinical lecturer and practitioner in clinical pharmacology and therapeutics: The RDFS, in general, was good. The symptom class could include

many more individual items as symptoms of hypoglycaemia, however, in general the classes worked. The relationship names "manage" and "associated_symptoms" were also deemed appropriate relation names. However, the "thing_cause" relationship between the class "cause" and class "disease" was not understandable.

Expert 5, consultant in diabetes and general paediatrics: The RDFS was broadly correct. The relation names were appropriate names that described the relations between classes. However, some CQ responses, such as "lupus", which was a response to the CQ "What are the symptoms of diabetes?", were incorrect.

To conclude, from the anecdotal responses, the most common issues observed with respect to the generated RDFS were that: (i) some responses to the CQ were incorrect, (ii) the relation "thing_cause" was not clear and (iii) the RDFS could be improved if it included a greater number of diseases. The first problem, incorrect responses, is a feature of the twitter collection used to train the RDFS rather than any inaccuracies within the RDFS, as discussed above in the previous sub-section, Sub-section 7.5. The second problem, that the relation "thing_cause" was not understandable was a function of the relation names being generated automatically. To solve this problem appropriate documentation could be generated to explain the provenance of each relation name so as to provide clarification. Involving end users is clearly undesirable. The final problem, that the RDFS just focused on one disease, will be addressed in future work by including more diseases and more tweets.

7.7 Summary

In this chapter a comparative evaluation of the four proposed RDF dataset from the Twitter data Frameworks was presented. For the evaluation two domains of discourse were used: (i) Motor vehicle pollution and (ii) diabetes. The motor vehicle pollution domain was used to evaluate all frameworks whilst the diabetes domain was used only with respect to the SPDP-WMD. The evaluation comprised five elements:

1. Comparative evaluation of the NER models used.
2. Comparative evaluation of the RE models used.
3. Comparison of generated RDF datasets and RDF Schemas
4. Querying populated RDFS
5. Consultation with Human Experts.

A number of CQs were generated as examples of how knowledge could be extracted from the populated RDFS. These CQs were converted to SPARQL queries which were applied to the populated RDFS; the results returned were then evaluated. For the consultation with human experts, in the context of the diabetes RDFS, five human experts were used

who had agreed to collaborate with the author of this thesis. The next chapter concludes the thesis with an overview of the main findings and some suggestions for future work.

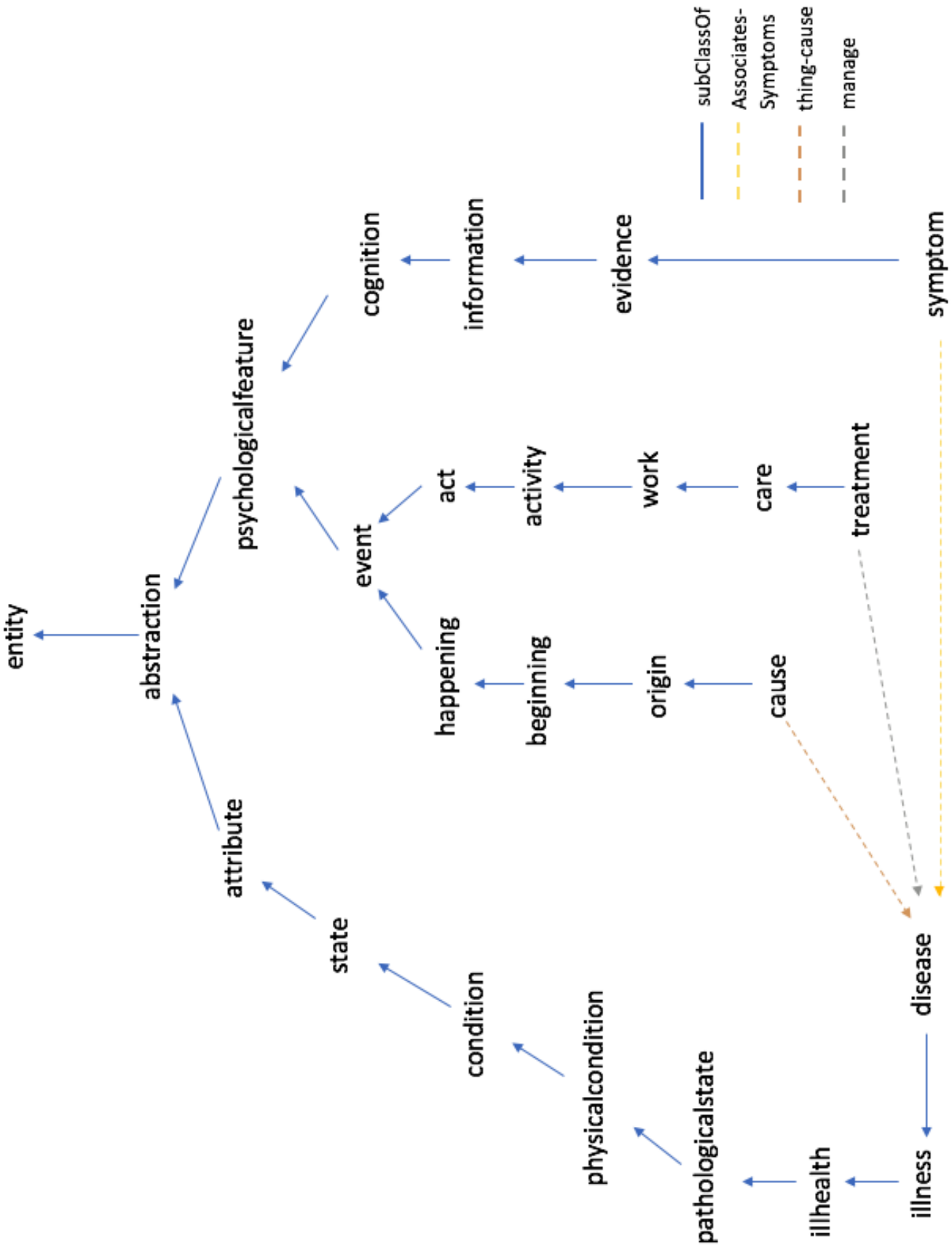

FIGURE 7.3: Node and link diagram illustrating the diabetes RDFS, generated using the SPDP-WMD framework and extended using WordNet.

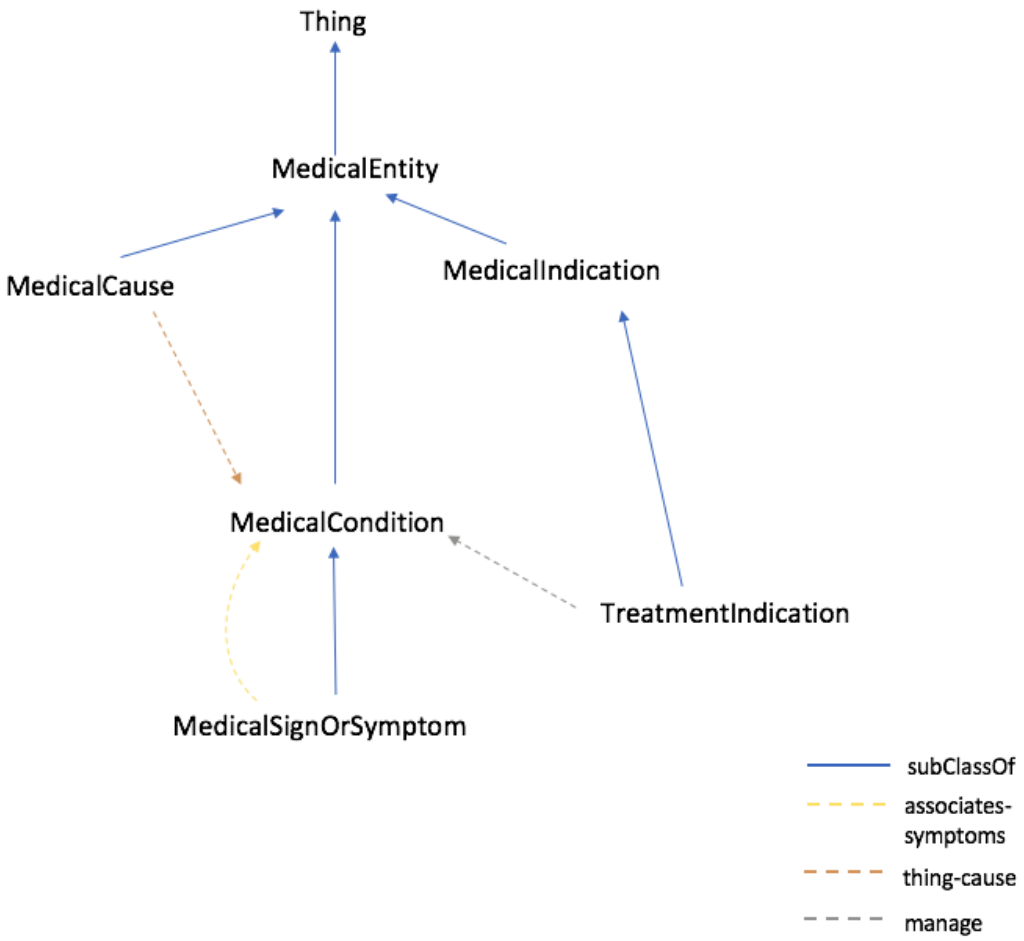

FIGURE 7.4: Node and link diagram illustrating the diabetes RDFS, generated using the SPDP-WMD framework and extended using Schema.org.

Chapter 8

Conclusion

8.1 Introduction

This chapter presents the conclusion to the research presented in this thesis. The chapter is structured as follows. In Section 8.2, a summary of the thesis is provided. Te main findings and contributions with respect to the main research question and the subsidiary research questions are considered in Section 8.3. Finally, some suggestions for potential future work, related to the work presented in this thesis, is presented in Section 8.4.

8.2 Summary of The Thesis

This thesis commenced with an introduction chapter, Chapter 1. This highlighted the motivation for the work presented in the thesis, namely the observation that there is a wealth of knowledge in social media data if only it could be extracted in structured manner. To impose a structure the generation of RDF datasets was proposed. However, it was also observed that the generation RDF datasets was a time-consuming endeavour. It was therefore proposed that the RDF dataset generation process should be automated. How best this could be done was therefore the main focus of the thesis (the overriding research question to be answered). To limit the scope of the research, the application domain was limited to Twitter data, although the work clearly has a much broader application.

In Chapter 2 the previous work relevant to the research reported on later in the thesis was presented. In the following four chapters four frameworks for RDF dataset from Twitter data were presented:

1. The Stanford CoreNLP RDF dataset from Twitter data framework.
2. The GATE RDF dataset from Twitter data framework.
3. The Regular Expression RDF dataset from Twitter data framework.
4. The Shortest Path Dependency Parsing and Word Mover's Distance (SPDP-WWD) RDF dataset from Twitter data framework.

An overview of the four frameworks is presented in Table 8.1.

Table 8.1: Frameworks Summary

RDF Dataset from Twitter Data Framework	NER	RE	RDF Dataset Generation (RDF)
Stanford CoreNLP	Supervised Learning using Stanford NER model	Supervised Learning using Stanford RE model	
GATE	GATE gazetteer (unsupervised but using entity lists)	Supervise Learning using GATE RE model	
Regular Expression	Supervised Learning using a seed set and Stanford NER ; model from which regular expression patterns are derived	Supervised Learning using Stanford RE generated, model and training data semi-automatically generated using regular expression patterns	Apache Jena
SPDP-WMD	Supervised Learning using Stanford NER model	Supervised Learning using Stanford RE model and training data automatically generated using SPDP and WMD	

The first RDF dataset from Twitter data framework, the Stanford CoreNLP Framework, was presented in Chapter 3. This framework, as the name implies, was founded on the use of the Stanford CoreNLP toolkit. The Stanford NER and RE models were trained to extract entities and relations from tweets, which were then used to construct an RDF dataset. The dataset was enriched to be more compatible with other datasets by building RDFS. The NER model was used to map entities to classes. For the construction of the RDFS hierarchy, the hypernyms of the classes and relations were acquired from WordNet. Apache Jena was used to generate the actual RDFS. This framework was a first "bench-mark" approach, any other framework would need to perform better than this approach to be worth while. The main disadvantage of the approach was the need to generate large amounts of training data. It was considered that the potential end users could be expected to generate some training data, but not be able to devote significant resource to this. The operation of the proposed approach was evaluated by using labelled testing data for the NER and RE models, and querying of the populated RDFS. To this end a bespoke evaluation Twitter dataset was generated covering the domain of motor vehicle pollution. The generated motor pollution RDF and RDFS

seemed appropriate and operated when used for querying purposes and humane expert evaluation. The average precision and recall values, obtained using TCV, were 77.74% and 82.26% respectively.

The second framework considered, the GATE framework (Chapter 4), attempted at addressing the Stanford CoreNLP framework training data generation overhead using the GATE (General Architecture for Text Engineering) NLP toolkit. This offered the advantage that NER were conducted using "entity lists" input to the GATE Gazetteer NER system, although these still had to be constructed by the user. RE was conducted using the GATE RE model which required training data in a similar manner to that required with respect to the Stanford RE model. WordNet and Apache Jena were again used to generate a RDFS. As in the case of the Stanford framework the GATE framework was evaluated using the motor vehicle pollution Twitter evaluation dataset and TCV. The average precision and recall values were 67.0 and 75.0 respectively. Comparing these values with the values obtained for the Stanford RE models indicated that the Stanford models provided a better performance. A further disadvantage of the GATE framework was the need to generate entity lists. Although the latter was considered to be less arduous than the need to create NER training data as in the case of the Stanford Framework. The performance of Stanford RE compared to GATE RE dictated that Stanford should be used for future frameworks.

The third framework, the Regular Expression framework (Chapter 5), partially addresses the problem of manually labelling training data associated with the Stanford and GATE frameworks. A semi-automated (semi-supervised) method using regular expressions, was proposed. A "seed set" comprised of a small number of manually labelled tweets was used to generate a NER model. A set of regular expression patterns were then defined, using a bespoke notation, which were then used to extracting relations, which were consequently used to generate an RE training set without further end user intervention. A specially designed regular expression parser was used to extract relations from tweets. WordNet and Apache Jena were again used to enrich the RDF dataset and generate the final RDFS. Evaluation was conducted using the same motor vehicle pollution dataset as used with respect to the Stanford and Gate frameworks. The average TCV precision and recall values obtained were 74.2% and 75.6% respectively. It was noticeable that the performance of the Regular Expression framework was better than GATE, and worse than Stanford with respect to precision and recall values. However, the advantage of the framework was the semi-automated method to prepare the RE training set. The perceived disadvantage of the Regular Expression framework was the need for a bespoke seed set and the need to define regular expressions manually.

The fourth and final framework (Chapter 6) was the SPDP-WMD framework. This featured an entirely automated process for generating a RE training set (although a NER training set was still required). The proposed process commenced with Shortest Path Dependency Parsing (SPDP) to extract relations that existed between entities. The extracted relations were clustered using Word Mover's Distance (WMD), which enable

selection of a relation class label for each entity pair. In this manner an RE training set was generated to train a Stanford RE model. This was used, together with a Stanford NER model, to extract entities and relations. As in the case of the previews frameworks considered, the RDF dataset was enrich to be more compatible with other datasets that had the same upper level ontology using RDFS. Experiments were conducted using both WordNet and Schema.org to build the RDFS hierarchies, only WordNet had been used previously. The RDFS was again constructed using Apache Jena. Using the motor vehicle pollution dataset average TCV precision and recall values of 75.8% and 74.2% were obtained respectively.

In Chapter 7, a comparative evaluation was presented using the motor vehicle pollution dataset. This indicated that the SPDP-WMD framework produced the best performance with no user involvement in the creation of RE training data. The SPDP-WMD framework was thus evaluated further using a larger diabetes training dataset. The populated RDFS result was presented to medical practitioners for comment. The feedback received was that the populated RDFS seemed mostly appropriate. Where the practitioners disagreed with the RDFS content it was found that this was a reflection of the views of Twitter users rather than any inaccuracies in the operation of the proposed frameworks. However, this was an interesting outcome. It was also observed that this might be exactly what an end user might require; the views of social media users rather than the "scientific truth".

8.3 Main findings

This section provides the main findings of the work presented in this thesis in term of the overriding research question and the subsidiary research questions. The section commences by considering the subsidiary research questions and then goes on to consider the main research question.

1. **Given that the central building blocks of RDF dataset are entities and relations can NER and RE be applied using the standard supervised learning tools and techniques available for natural language processing?**

   The first two frameworks that the thesis discuses, the Stanford CoreNLP and GATE frameworks, both used standard supervised learning tools, the Stanford CoreNLP toolkit and the GATE toolkit. Experiments conducted using both frameworks achieved good performance. Recall that GATE used "entity lists" as input to the GATE Gazetteer NER system. Thus it was concluded that the standard supervised learning tools and techniques available for NLP can be used for NER and RE to support RDF dataset generation, however, at the expense of requiring hand-crafted training data and/or entity lists.

2. **Given the overhead associated with supervised learning, is there an alternative semi-supervised approach to RE that can be adopted that does not adversely affect the quality of any generated RDF dataset?**

   The third framework discussed in the thesis, the Regular expression framework, implements a semi-supervised learning for RE approach. It uses a small dataset to create regular expression patterns which were then used to extract relations that existed between entities, which subsequently produced a comprehensive dataset. Despite being outperformed by the Stanford CoreNLP framework, the Regular expression framework outperformed the GATE framework. In conclusion, it is argued that the semi-supervised approach to RE is a good alternative to the Stanford and GATE frameworks considered earlier that requires less training data for RE model generation than the Stanford framework.

3. **Following on from the previous two subsidiary questions, is it possible to devise an unsupervised approach to RE that can be adopted, that does not adversely affect the quality of any generated RDF dataset?**

   The fourth framework that this thesis discusses, the SPDP-WMD framework, addressed this third subsidiary research question. This framework enabled automatic generation of a training dataset using Shortest Path Dependency Parsing (SPDP) and Word Mover's Distance (WMD) to construct RE models. SPDP was used to capture relations between entities and WMD to obtain generic relationship labels that connect pairs of entities. Using this framework, a RE model was trained to automatically extract relations between entities existing in a given dataset; in other words, there was no requirement for hand-crafted training data for RE model generation. The quality of the generated RDF dataset was found not to be effected when it was compared with those produced using the other frameworks proposed. Using the motor vehicle pollution domain of discourse the RDF dataset generation using the SPDP-WMD framework was largely the same as the RDF dataset that were generated using the Stanford, GATE and Regular Expression frameworks, with the exception that the relation names were the result of the unsupervised method of preparing the RE training data. Although the derived relation names had more-or-less the same meaning as the hand-crafted relation names used in the context of the Stanford, GATE and Regular Expression frameworks. Further inspection could be done to measure the similarity of the relation names among frameworks.

4. **How do we know when a generated RDF dataset is correct**

   The generated RDF dataset generation frameworks were evaluated using an evaluation strategy that comprised three strands: (i) evaluation of the NER and RE models using established machine learning approaches, namely k-fold Cross-Validation (kCV); (ii) querying the populated RDFS and (iii) consultation with human experts. All of these strands were designed to assess the correctness of the generated

RDF datsets; that they served their intended purpose. The results of the evaluation demonstrated that the proposed SPDP-WMD offered the greatest advantage in terms of the balance between the efficiency of the generation process and the utility of the generated RDF dataset. It is therefore argued here that the adopted evaluation strategy was well suited to measuring whether a generated RDF dataset was correct or not.

5. **How can we best cause the generated RDF dataset to be integrable with another dataset that commits the same upper level ontology?**

   The fundamental idea promoted in this thesis for making the generated RDF dataset more integrable with other datasets was to encapsulate the identified entities and relations in an RDFS so that a generated RDFS had the potential to be integrated with other datasets. For the motor vehicle pollution domain of discourse WordNet was used for this purpose. For the diabetes domain of discourse WordNet and Schema.org were used. A comparative evaluation of the two mechanisms indicated that both produced satisfactory hierarchies. The hierarchy generated using Schema.org (with respect to the diabetes domain of discourse) was argued to produce a slightly superior hierarchy, although not every domain of discourse is currently included in Schema.org.

Returning to the overriding research question that this thesis sought to answer:

What is the most effective and efficient mechanism whereby an RDF dataset associated with a particular domain of discourse, described in the form of Twitter free text, can be generated?

The best NER and RE models, TCV average precision of 86.7% and recall of 89.7%, were generated using Stanford CoreNLP framework and the motor vehicle pollution domain of discourse. However at the expense of requiring a manually labelled training dataset; a notable limitation of this approach. The fourth framework, the SPDP-WMD framework, did not require a training set for RE model generation. The SPDP-WMD RE produced an average TCV precision and recall of 75.8% and 74.2% respectively, for the motor vehicle pollution domain of discourse. Not as good as that produced using the Stanford and Regular expression RE models. However, the SPDP-WMD framework featured much higher efficiency compared to all the other frameworks presented in this thesis. It seems that the most effective and efficient mechanism whereby an RDF dataset can be generated with respect to a particular domain of discourse, described in the form of free text (Twitter data), was the SPDP-WMD framework.

8.4 Future Work

The work presented in this thesis has proposed four frameworks for generating RDF datasets from Twitter data. The work has demonstrated that the frameworks can

effectively generate RDF from such data. The fourth framework, the SPDP-WMD framework, was found to be the most effective framework. This section suggests some opportunities for future work founded on the work presented in this thesis.

1. **Automatic named entity recognition (NER):** All of the frameworks considered in this thesis required NER model generated using training data of some kind. In the case of the Regular Expression framework (Chapter 5) this was a seed set; in the case of the other frameworks considered this was a substantial, hand labelled, training set (Chapters 3, 5 and 6). A mechanism for generating such training data would clearly be desirable. This would improve the efficiency of the frameworks presented in this thesis. Also, any generated RDF dataset could be readily expanded because a greater number of entities could be included.

2. **Additional evaluation using more sophisticated Twitter domains of discourse:** The evaluation presented in this thesis was conducted using only two domains of discourse, a more sophisticated evaluation would clearly be desirable.

3. **Extension to other forms of social media:** The work presented in this thesis was directed at Twitter data. Although it was conjectured that the frameworks considered could equally be applied to other sources of social media (for example facebook), no experiments were conducted using such alternative forms of social media. This would also clearly be worth investigating.

4. **Extension to other languages than English:** The work presented in this thesis has assumed social media data written in English. Of course, much of the world's social media data is not written in English. Another avenue for future work is to explore opportunities in the context of other world languages such as Mandarin Chinese, Hindi, Spanish and Arabic.

5. **Alternative Frameworks:** There remains scope for alternative RDF dataset generation from Twitter data (RDF dataset generation from social media data) frameworks with respect to NER and RE model generation. One possibility is to investigate the use of deep learning.

Overall the work presented in this thesis has established a sound foundation for RDF dataset generation from Twitter data; work that has merit in its own right and which provides a platform for future work.

Bibliography

[1] Eugene Agichtein and Luis Gravano, *Snowball: Extracting relations from large plain-text collections*, Proceedings of the fifth ACM conference on Digital libraries, 2000, pp. 85–94.

[2] Abeer Al-Arfaj and Abdulmalik Al-Salman, *Ontology construction from text: challenges and trends*, International Journal of Artificial Intelligence and Expert Systems (IJAE) **6** (2015), no. 2, 15–26.

[3] Dekha Anggareska and Ayu Purwarianti, *Information extraction of public complaints on twitter text for bandung government*, 2014 International Conference on Data and Software Engineering (ICODSE), IEEE, 2014, pp. 1–6.

[4] Davide Anguita, Alessandro Ghio, Sandro Ridella, and Dario Sterpi, *K-fold cross validation for error rate estimate in support vector machines.*, DMIN, 2009, pp. 291–297.

[5] G. Antoniou, Paul Groth, Frank Van Harmelen, and Rinke Hoekstra, *A semantic web primer.*, Cooperative information systems, MIT Press, 2012.

[6] Sinan Aral, *The hype machine: How social media disrupts our elections, our economy, and our health-and how we must adapt*, 2020.

[7] Eiji Aramaki, Sachiko Maskawa, and Mizuki Morita, *Twitter Catches The Flu: Detecting Influenza Epidemics using Twitter*, 2011 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, 2011, pp. 1568–1576.

[8] Muhammad Nabeel Asim, Muhammad Wasim, Muhammad Usman Ghani Khan, Waqar Mahmood, and Hafiza Mahnoor Abbasi, *A survey of ontology learning techniques and applications*, Database **2018** (2018).

[9] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives, *Dbpedia: A nucleus for a web of open data*, The semantic web, Springer, 2007, pp. 722–735.

[10] Nguyen Bach and Sameer Badaskar, *A review of relation extraction*, Literature review for Language and Statistics II **2** (2007), 1–15.

[11] Payam Barnaghi, Amit Sheth, and Cory Henson, *From data to actionable knowledge: Big data challenges in the web of things [guest editors' introduction]*, IEEE Intelligent Systems **28** (2013), no. 6, 6–11.

[12] David S Batista, Bruno Martins, and Mário J Silva, *Semi-supervised bootstrapping of relationship extractors with distributional semantics*, Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, 2015, pp. 499–504.

[13] Brahim Batouche, Claire Gardent, Anne Monceaux, and France Blagnac, *Parsing text into rdf graphs*, Proceedings of the XXXI Congress of the Spanish Society for the Processing of Natural Language, 2014.

[14] Branislav Bednár and Zuzana Bilanová, *Towards automated translating natural language sentences into intensional constructions*, 2019 IEEE 15th International Scientific Conference on Informatics, IEEE, 2019, pp. 000345–000350.

[15] Camila Bezerra, Fred Freitas, and Filipe Santana, *Evaluating ontologies with competency questions*, 2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT), vol. 3, IEEE, 2013, pp. 284–285.

[16] Daniel M Bikel, Scott Miller, Richard Schwartz, and Ralph Weischedel, *Nymble: a high-performance learning name-finder*, Proceedings of the fifth conference on Applied natural language processing, Association for Computational Linguistics, 1997, pp. 194–201.

[17] Steven Bird, *Nltk: the natural language toolkit*, Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions, 2006, pp. 69–72.

[18] Christopher M. Bishop, *Pattern recognition and machine learning (information science and statistics)*, 1 ed., Springer, 2007.

[19] Roi Blanco, Giuseppe Ottaviano, and Edgar Meij, *Fast and space-efficient entity linking for queries*, Proceedings of the Eighth ACM International Conference on Web Search and Data Mining, 2015, pp. 179–188.

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

[21] Kalina Bontcheva, Leon Derczynski, Adam Funk, Mark A Greenwood, Diana Maynard, and Niraj Aswani, *Twitie: An open-source information extraction pipeline for microblog text*, Proceedings of the international conference recent advances in natural language processing RANLP 2013, 2013, pp. 83–90.

[22] Janez Brank, Marko Grobelnik, and Dunja Mladenic, *A survey of ontology evaluation techniques*, Proceedings of the conference on data mining and data warehouses (SiKDD 2005), Citeseer Ljubljana, Slovenia, 2005, pp. 166–170.

[23] Dan Brickley, Ramanathan V Guha, and Andrew Layman, *Resource description framework (rdf) schema specification*, (1999).

[24] Dan Brickley, Ramanathan V Guha, and Brian McBride, *Rdf schema 1.1*, W3C recommendation (2014), 2004–2014.

[25] Sergey Brin, *Extracting patterns and relations from the world wide web*, International Workshop on The World Wide Web and Databases, Springer, 1998, pp. 172–183.

[26] Julian Brooke, Adam Hammond, and Timothy Baldwin, *Bootstrapped text-level named entity recognition for literature*, Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), 2016, pp. 344–350.

[27] Axel Bruns and Jean Burgess, *The use of twitter hashtags in the formation of ad hoc publics*, Proceedings of the 6th European consortium for political research (ECPR) general conference 2011, The European Consortium for Political Research (ECPR), 2011, pp. 1–9.

[28] Razvan C Bunescu and Raymond J Mooney, *A shortest path dependency kernel for relation extraction*, Proceedings of the conference on human language technology and empirical methods in natural language processing, Association for Computational Linguistics, 2005, pp. 724–731.

[29] Christopher JC Burges, *A tutorial on support vector machines for pattern recognition*, Data mining and knowledge discovery **2** (1998), no. 2, 121–167.

[30] Kate Byrne and Ewan Klein, *Automatic extraction of archaeological events from text*, Proceedings of the 37th International Conference, Williamsburg, Virginia, United States of America (2010), 1–16.

[31] Claire Cardie and Kiri Wagstaff, *Noun phrase coreference as clustering*, 1999 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora, 1999.

[32] Andrew Carlson, Justin Betteridge, Richard C. Wang, Estevam R. Hruschka, and Tom M. Mitchell, *Coupled semi-supervised learning for information extraction*, Proceedings of the third ACM international conference on Web search and data mining, ACM, 2010, p. 101.

[33] Anshuman Chhabra and Prasant Mohapatra, *Fair algorithms for hierarchical agglomerative clustering*, arXiv preprint arXiv:2005.03197 (2020).

[34] Jason PC Chiu and Eric Nichols, *Named entity recognition with bidirectional lstm-cnns*, Transactions of the Association for Computational Linguistics **4** (2016), 357–370.

[35] Wang Chunxiao, Liu Jingjing, Xiao Yire, Ding Min, Wang Zhaohui, Qi Gaofu, Shen Xiangchun, Wang Xuejun, Wu Jie, and Li Taiming, *Customizing an Information Extraction System to a New Domain*, Regulatory Peptides, vol. 141, Association for Computational Linguistics, 2007, pp. 35–43.

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

[37] Florângela Cunha Coelho, Thiago Christiano Silva, and Philipp Ehrl, *Internet access in brazilian households: Evaluating the effect of an economic recession*, World Conference on Information Systems and Technologies, Springer, 2019, pp. 716–725.

[38] Hamish Cunningham, *Gate, a general architecture for text engineering*, Computers and the Humanities **36** (2002), no. 2, 223–254.

[39] Hamish Cunningham, Diana Maynard, Kalina Bontcheva, Valentin Tablan, Cristian Ursu, Marin Dimitrov, Mike Dowman, Niraj Aswani, Ian Roberts, Yaoyong Li, et al., *Developing language processing components with gate version 5:(a user guide)*, University of Sheffield, 2009.

[40] Aurore De Amaral, *Rule-based named entity extraction for ontology population*, Proceedings of the Student Research Workshop associated with RANLP 2013, 2013, pp. 58–62.

[41] Marie-Catherine De Marneffe, Timothy Dozat, Natalia Silveira, Katri Haverinen, Filip Ginter, Joakim Nivre, and Christopher D Manning, *Universal stanford dependencies: A cross-linguistic typology.*, LREC, vol. 14, 2014, pp. 4585–4592.

[42] Marie-Catherine De Marneffe, Bill MacCartney, Christopher D Manning, et al., *Generating typed dependency parses from phrase structure parses.*, Lrec, vol. 6, 2006, pp. 449–454.

[43] Luciano Del Corro and Rainer Gemulla, *Clausie: clause-based open information extraction*, Proceedings of the 22nd international conference on World Wide Web, 2013, pp. 355–366.

[44] Leon Derczynski, Diana Maynard, Niraj Aswani, and Kalina Bontcheva, *Microblog-genre noise and impact on semantic annotation accuracy*, Proceedings of the 24th ACM Conference on Hypertext and Social Media, 2013, pp. 21–30.

[45] Dimitar Dimitrov, Erdal Baran, Pavlos Fafalios, Ran Yu, Xiaofei Zhu, Matthäus Zloch, and Stefan Dietze, *Tweetscov19-a knowledge base of semantically annotated*

tweets about the covid-19 pandemic, Proceedings of the 29th ACM International Conference on Information & Knowledge Management, 2020, pp. 2991–2998.

[46] Daniel Pinto dos Santos, Sebastian Brodehl, Bettina Baeßler, Gordon Arnhold, Thomas Dratsch, Seung-Hun Chon, Peter Mildenberger, and Florian Jungmann, *Structured report data can be used to develop deep learning algorithms: a proof of concept in ankle radiographs*, Insights into imaging **10** (2019), no. 1, 93.

[47] Micha Elsner, Eugene Charniak, and Mark Johnson, *Structured generative models for unsupervised named-entity clustering*, Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, 2009, pp. 164–172.

[48] Peter Exner and Pierre Nugues, *Entity Extraction: From Unstructured Text to DBpedia RDF Triples*, The Web of Linked Entities Workshop (WoLE 2012), CEUR, 2012, pp. 58–69.

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

[50] Pavlos Fafalios, Vasileios Iosifidis, Eirini Ntoutsi, and Stefan Dietze, *Tweetskb: A public and large-scale rdf corpus of annotated tweets*, European Semantic Web Conference, Springer, 2018, pp. 177–190.

[51] Christiane Fellbaum, *Wordnet*, Theory and applications of ontology: computer applications, Springer, 2010, pp. 231–243.

[52] David Ferrucci and Adam Lally, *Uima: an architectural approach to unstructured information processing in the corporate research environment*, Natural Language Engineering (2004), 1–26.

[53] Charles J Fillmore et al., *Frame semantics and the nature of language*, Annals of the New York Academy of Sciences: Conference on the origin and development of language and speech, vol. 280, 1976, pp. 20–32.

[54] Jenny Rose Finkel, Trond Grenager, and Christopher Manning, *Incorporating non-local information into information extraction systems by Gibbs sampling*, Proceedings of the 43rd annual meeting on association for computational linguistics, Association for Computational Linguistics, 2007, pp. 363–370.

[55] Angelin Florence and Vijaya Padmadas, *A summarizer system based on a semantic analysis of web documents*, 2015 International Conference on Technologies for Sustainable Development (ICTSD), IEEE, 2015, pp. 1–6.

[56] Klyne Graham and Jeremy Carroll, *Resource Description Framework (RDF): Concepts and Abstract Syntax*, W3C Recommendation **10** (2004), no. October, 1—-20.

[57] Ramanathan V Guha, Dan Brickley, and Steve Macbeth, *Schema. org: evolution of structured data on the web*, Communications of the ACM **59** (2016), no. 2, 44–51.

[58] Yaxiong Han, *A survey on chinese named entity recognition*, International Educational Applied Scientific Research Journal **3** (2018), no. 8.

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

[60] I Hemalatha, GP Saradhi Varma, and A Govardhan, *Preprocessing the informal text for efficient sentiment analysis*, International Journal of Emerging Trends & Technology in Computer Science (IJETTCS) **1** (2012), no. 2, 58–61.

[61] Jefferson Henrique, *Get old tweets*, 2017.

[62] Georgiana Ifrim, Bichen Shi, and Igor Brigadir, *Event detection in twitter using aggressive filtering and hierarchical tweet clustering*, Second Workshop on Social News on the Web (SNOW), Seoul, Korea, 8 April 2014, ACM, 2014.

[63] Clement J, *Number of monthly active facebook users worldwide as of 3rd quarter 2020.*

[64] Hamed Jelodar, Yongli Wang, Chi Yuan, Xia Feng, Xiahui Jiang, Yanchao Li, and Liang Zhao, *Latent dirichlet allocation (lda) and topic modeling: models, applications, a survey*, Multimedia Tools and Applications **78** (2019), no. 11, 15169–15211.

[65] Apache Jena, *A free and open source java framework for building semantic web and linked data applications*, Available online: jena. apache. org/(accessed on 28 April 2015) (2015).

[66] Guoliang Ji, Kang Liu, Shizhu He, and Jun Zhao, *Distant supervision for relation extraction with sentence-level attention and entity descriptions*, Proceedings of the AAAI Conference on Artificial Intelligence, vol. 31, 2017.

[67] Jing Jiang, *Information extraction from text*, Mining text data, Springer, 2012, pp. 11–41.

[68] Shrutika Kale and Sharvari Govilkar, *Survey of named entity recognition techniques for various indian regional languages*, International Journal of Computer Applications **164** (2017), no. 4, 37–43.

[69] Gjergji Kasneci, Fabian Suchanek, and Gerhard Weikum, *Yago-a core of semantic knowledge*, (2006).

[70] Arzoo Katiyar and Claire Cardie, *Nested named entity recognition revisited*, Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), 2018, pp. 861–871.

[71] Gregory Katsios, Svitlana Vakulenko, Anastasia Krithara, and Georgios Paliouras, *Towards open domain event extraction from twitter: Revealing entity relations.*, DeRiVE@ ESWC, 2015, pp. 35–46.

[72] Javed Ahmad Khan and Suresh Kumar, *Deep analysis for development of rdf, rdfs and owl ontologies with protege*, Proceedings of 3rd International Conference on Reliability, Infocom Technologies and Optimization, IEEE, 2014, pp. 1–6.

[73] Matt Kusner, Yu Sun, Nicholas Kolkin, and Kilian Weinberger, *From word embeddings to document distances*, International conference on machine learning, 2015, pp. 957–966.

[74] Ravendar Lal et al., *Information extraction of security related entities and concepts from unstructured text.*, (2013).

[75] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton, *Deep learning*, nature **521** (2015), no. 7553, 436–444.

[76] 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*, Proceedings of the fifteenth conference on computational natural language learning: Shared task, Association for Computational Linguistics, 2011, pp. 28–34.

[77] Fei Li, Meishan Zhang, Guohong Fu, and Donghong Ji, *A neural joint model for entity and relation extraction from biomedical text*, BMC bioinformatics **18** (2017), no. 1, 1–11.

[78] Kun Li, Junsheng Zhang, Changqing Yao, and Chongde Shi, *Automatic relation extraction from text: A survey*, 2016 International Conference on Identification, Information and Knowledge in the Internet of Things (IIKI), IEEE, 2016, pp. 83–86.

[79] Yunyao Li, Rajasekar Krishnamurthy, Sriram Raghavan, Shivakumar Vaithyanathan, and HV Jagadish, *Regular expression learning for information extraction*, Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing, 2008, pp. 21–30.

[80] Wenhui Liao and Sriharsha Veeramachaneni, *A simple semi-supervised algorithm for named entity recognition*, Proceedings of the NAACL HLT 2009 Workshop on Semi-Supervised Learning for Natural Language Processing, 2009, pp. 58–65.

[81] Nut Limsopatham and Nigel Collier, *Bidirectional lstm for named entity recognition in twitter messages*, Proceedings of the 2nd Workshop on Noisy User-generated Text (2016), 145–152.

[82] Ying Luo, Hai Zhao, and Junlang Zhan, *Named entity recognition only from word embeddings*, arXiv preprint arXiv:1909.00164 (2019).

[83] Yuanchao Ma, Bin Xu, Yin Bai, and Zonghui Li, *Building linked open university data: Tsinghua university open data as a showcase*, Joint International Semantic Technology Conference, Springer, 2011, pp. 385–393.

[84] Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky, *The stanford corenlp natural language processing toolkit*, Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations, 2014, pp. 55–60.

[85] Mohammad Masum, Hossain Shahriar, Hisham M Haddad, Sheikh Ahamed, Sweta Sneha, Mohammad Rahman, and Alfredo Cuzzocrea, *Actionable knowledge extraction framework for covid-19*, 2020 IEEE International Conference on Big Data (Big Data), IEEE, 2020, pp. 4036–4041.

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

[87] Brian McBride, *The resource description framework (rdf) and its vocabulary description language rdfs*, Handbook on ontologies, Springer, 2004, pp. 51–65.

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

[89] John McCrae, Christiane Fellbaum, and Philipp Cimiano, *Publishing and linking wordnet using lemon and rdf*, Proceedings of the 3rd Workshop on Linked Data in Linguistics, 2014.

[90] Deborah L McGuinness, Frank Van Harmelen, et al., *Owl web ontology language overview*, W3C recommendation **10** (2004), no. 10, 2004.

[91] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean, *Efficient estimation of word representations in vector space*, arXiv preprint arXiv:1301.3781 (2013).

[92] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean, *Distributed representations of words and phrases and their compositionality*, Advances in neural information processing systems, 2013, pp. 3111–3119.

[93] George A Miller, *Wordnet: An electronic lexical database*, MIT press, 1998.

[94] Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky, *Distant supervision for relation extraction without labeled data*, 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.

[95] Sudip Mittal, Prajit Kumar Das, Varish Mulwad, Anupam Joshi, and Tim Finin, *Cybertwitter: Using twitter to generate alerts for cybersecurity threats and vulnerabilities*, 2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), IEEE, 2016, pp. 860–867.

[96] Murali Mohanan and Philip Samuel, *Open nlp based refinement of software requirements*, International Journal of Computer Information Systems and Industrial Management Applications **8** (2016), 293–300.

[97] Guillermo Moncecchi, Jean-Luc Minel, and Dina Wonsever, *A survey of kernel methods for relation extraction*, 2010.

[98] Dhiraj Murthy, *Twitter and elections: are tweets, predictive, reactive, or a form of buzz?*, Information Communication and Society **18** (2015), no. 7, 816–831.

[99] David Nadeau and Satoshi Sekine, *A survey of named entity recognition and classification*, Lingvisticae Investigationes **30** (2007), no. 1, 3–26.

[100] Jay Nanavati and Yogesh Ghodasara, *A comparative study of stanford nlp and apache open nlp in the view of pos tagging*, International Journal of Soft Computing and Engineering **5** (2015), no. 5, 57–60.

[101] Mark H Needleman, *Rdf*, Serials Review **27** (2001), no. 1, 58–61.

[102] Truc-Vien T Nguyen and Alessandro Moschitti, *End-to-end relation extraction using distant supervision from external semantic repositories*, Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2, Association for Computational Linguistics, 2011, pp. 277–282.

[103] Ian Niles and Adam Pease, *Linking lixicons and ontologies: Mapping wordnet to the suggested upper merged ontology.*, Ike, 2003, pp. 412–416.

[104] Lucila Ohno-Machado, Susanna-Assunta Sansone, George Alter, Ian Fore, Jeffrey Grethe, Hua Xu, Alejandra Gonzalez-Beltran, Philippe Rocca-Serra, Anupama E

Gururaj, Elizabeth Bell, et al., *Finding useful data across multiple biomedical data repositories using datamed*, Nature genetics **49** (2017), no. 6, 816–819.

[105] Sachin Pawar, Girish K Palshikar, and Pushpak Bhattacharyya, *Relation Extraction: A Survey*, arXiv preprint arXiv:1712.05191 (2017).

[106] Ofir Pele and Michae Werman, *A linear time histogram metric for improved sift matching*, European conference on computer vision, Springer, 2008, pp. 495–508.

[107] Ofir Pele and Michael Werman, *Fast and robust earth mover's distances*, 2009 IEEE 12th International Conference on Computer Vision, IEEE, 2009, pp. 460–467.

[108] Andrew Perrin, *Social media usage*, Pew research center (2015), 52–68.

[109] Jakub Piskorski and Maud Ehrmann, *On named entity recognition in targeted twitter streams in polish.*, Proceedings of the 4th Biennial International Workshop on Balto-Slavic Natural Language Processing, 2013, pp. 84–93.

[110] Cecil Prescott, *Internet access – households and individuals, great britain: 2020 internet access – households and individuals, great britain: 2020*, 2020.

[111] E Prud'Hommeaux, Andy Seaborne, Eric Prud, and Hewlett-packard Laboratories, *SPARQL Query Language for RDF*, W3C working draftd (2008), 1–95.

[112] Daniel Ramage, David Hall, Ramesh Nallapati, and Christopher D Manning, *Labeled lda: A supervised topic model for credit attribution in multi-labeled corpora*, Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1, Association for Computational Linguistics, 2009, pp. 248–256.

[113] Bhavana Ramesh and Charles Weber, *Extracting and quantifying actionable knowledge using twitter data*, Portland International Center for Management of Engineering and Technology (2020).

[114] Alexander E Richman and Patrick Schone, *Mining wiki resources for multilingual named entity recognition*, Proceedings of ACL-08: HLT, 2008, pp. 1–9.

[115] Sebastian Riedel, Limin Yao, and Andrew McCallum, *Modeling Relations and Their Mentions without Labeled Text BT - Machine Learning and Knowledge Discovery in Databases*, Joint European Conference on Machine Learning and Knowledge Discovery in Databases, Springer, 2010, pp. 148–163.

[116] Alan Ritter, Sam Clark, Oren Etzioni, et al., *Named entity recognition in tweets: an experimental study*, Proceedings of the conference on empirical methods in natural language processing, Association for Computational Linguistics, 2011, pp. 1524–1534.

[117] Xin Rong, *word2vec parameter learning explained*, arXiv preprint arXiv:1411.2738 (2014).

[118] Dan Roth and Wen-tau Yih, *Probabilistic reasoning for entity & relation recognition*, Proceedings of the 19th international conference on Computational linguistics-Volume 1, Association for Computational Linguistics, 2002, pp. 1–7.

[119] Dan Roth and Wentau Yih, *A linear programming formulation for global inference in natural language tasks*, Tech. report, ILLINOIS UNIV AT URBANA-CHAMPAIGN DEPT OF COMPUTER SCIENCE, 2004.

[120] Camille L Ryan and Jamie M Lewis, *Computer and internet use in the united states: 2015*, US Department of Commerce, Economics and Statistics Administration, US . . . , 2017.

[121] Takeshi Sakaki, Makoto Okazaki, and Yutaka Matsuo, *Earthquake shakes twitter users: real-time event detection by social sensors*, Proceedings of the 19th international conference on World wide web, 2010, pp. 851–860.

[122] Sebastian Schuster and Christopher D Manning, *Enhanced english universal dependencies: An improved representation for natural language understanding tasks*, Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16), 2016, pp. 2371–2378.

[123] Surendra Sedhai and Aixin Sun, *Hspam14: A collection of 14 million tweets for hashtag-oriented spam research*, Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2015, pp. 223–232.

[124] Satoshi Sekine, *Nyu: Description of the japanese ne system used for met-2*, Proc. of the Seventh Message Understanding Conference (MUC-7, Citeseer, 1998.

[125] Mehrnoush Shamsfard and Ahmad Abdollahzadeh Barforoush, *The state of the art in ontology learning: a framework for comparison*, The Knowledge Engineering Review **18** (2003), no. 4, 293.

[126] Rahul Sharnagat, *Named entity recognition: A literature survey*, Center For Indian Language Technology (2014).

[127] Alisa Smirnova and Philippe Cudré-Mauroux, *Relation extraction using distant supervision: A survey*, ACM Computing Surveys (CSUR) **51** (2018), no. 5, 1–35.

[128] Stephen Soderland, *Learning information extraction rules for semi-structured and free text*, Machine learning **34** (1999), no. 1-3, 233–272.

[129] Tony Stubblebine, *Regular expression pocket reference: Regular expressions for perl, ruby, php, python, c, java and. net*, " O'Reilly Media, Inc.", 2007.

[130] Sudha Subramani and Manjula O'Connor, *Extracting actionable knowledge from domestic violence discourses on social media*, arXiv preprint arXiv:1807.02391 (2018).

[131] Peng Sun, Xuezhen Yang, Xiaobing Zhao, and Zhijuan Wang, *An overview of named entity recognition*, 2018 International Conference on Asian Language Processing (IALP), IEEE, 2018, pp. 273–278.

[132] Valentin Tablan, Diana Maynard, Kalina Bontcheva, Hamish Cunningham, V Tablan, D Maynard, and K Bontcheva, *Gate, an application developer's guide*, Department of Computer Science, University of Sheffield, UK **19** (2004).

[133] Yuan Tian and David Lo, *A comparative study on the effectiveness of part-of-speech tagging techniques on bug reports*, 2015 IEEE 22nd International Conference on Software Analysis, Evolution, and Reengineering (SANER), IEEE, 2015, pp. 570–574.

[134] Edmund Tsui, Rajesh C Rao, Andrew R Carey, Matthew T Feng, and Lorraine M Provencher, *Using social media to disseminate ophthalmic information during the# covid19 pandemic*, Ophthalmology (2020).

[135] Joseph Turian, Lev Ratinov, and Yoshua Bengio, *Word representations: a simple and general method for semi-supervised learning*, Proceedings of the 48th annual meeting of the association for computational linguistics, 2010, pp. 384–394.

[136] Ting Wang, Yaoyong Li, Kalina Bontcheva, Hamish Cunningham, and Ji Wang, *Automatic Extraction of Hierarchical Relations from Text*, European Semantic Web Conference, Springer, 2006, pp. 215–229.

[137] Wilson Wong, Wei Liu, and Mohammed Bennamoun, *Ontology learning from text: A look back and into the future*, ACM Computing Surveys (CSUR) **44** (2012), no. 4, 1–36.

[138] Fei Wu and Daniel S Weld, *Open information extraction using wikipedia*, Proceedings of the 48th annual meeting of the association for computational linguistics, 2010, pp. 118–127.

[139] Jason Xu and Kenneth Lange, *Power k-means clustering*, International Conference on Machine Learning, PMLR, 2019, pp. 6921–6931.

[140] Jia Xue, Junxiang Chen, Chen Chen, Chengda Zheng, Sijia Li, and Tingshao Zhu, *Public discourse and sentiment during the covid 19 pandemic: Using latent dirichlet allocation for topic modeling on twitter*, PloS one **15** (2020), no. 9, e0239441.

[141] Vikas Yadav and Steven Bethard, *A survey on recent advances in named entity recognition from deep learning models*, arXiv preprint arXiv:1910.11470 (2019).

[142] Lim Cheng Yang, Ian KT Tan, Bhawani Selvaretnam, Ewe Kok Howg, and Lau Heng Kar, *Text: Traffic entity extraction from twitter*, Proceedings of the 2019 5th International Conference on Computing and Data Engineering, 2019, pp. 53–59.

[143] Yaosheng Yang, Wenliang Chen, Zhenghua Li, Zhengqiu He, and Min Zhang, *Distantly supervised ner with partial annotation learning and reinforcement learning*, Proceedings of the 27th International Conference on Computational Linguistics, 2018, pp. 2159–2169.

[144] Alexander Yates, Michele Banko, Matthew Broadhead, Michael J Cafarella, Oren Etzioni, and Stephen Soderland, *Textrunner: open information extraction on the web*, Proceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT), 2007, pp. 25–26.

[145] Wajdi Zaghouani, *Renar: A rule-based arabic named entity recognition system*, ACM Transactions on Asian Language Information Processing (TALIP) **11** (2012), no. 1, 1–13.

[146] Daojian Zeng, Kang Liu, Yubo Chen, and Jun Zhao, *Distant supervision for relation extraction via piecewise convolutional neural networks*, Proceedings of the 2015 conference on empirical methods in natural language processing, 2015, pp. 1753–1762.

[147] Min Zhang, Jie Zhang, Jian Su, and Guodong Zhou, *A composite kernel to extract relations between entities with both flat and structured features*, Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics, Association for Computational Linguistics, 2006, pp. 825–832.

[148] Yijia Zhang, Hongfei Lin, Zhihao Yang, Jian Wang, Shaowu Zhang, Yuanyuan Sun, and Liang Yang, *A hybrid model based on neural networks for biomedical relation extraction*, Journal of biomedical informatics **81** (2018), 83–92.

Appendix A

The Stanford CoreNLP RDF Dataset From Twitter Data Framework

A.1 Stanford Relation Extraction Evaluation

Detailed TVC results for evaluation of the Stanford relation extraction model that used in the Stanford CoreNLP framework are presented in detail below.

Table A.1: The results of Fold 1 for Stanford RE model

Relation Label	Precision	Recall	F-score
ban_Fuel_V_Date	70.6	92.3	80.0
use_Environment_friendly_V_Date	40.0	50.0	44.4
ban	80.8	100.0	89.4
use	60.0	90.0	72.0
Total	67.6	88.5	76.7

Table A.2: The results of Fold 2 for Stanford RE model

Relation Label	Precision	Recall	F-score
ban_Fuel_V_Date	66.7	87.5	75.7
use_Environment_friendly_V_Date	83.3	55.6	66.7
ban	83.3	100.0	90.9
use	100.0	93.8	96.8
Total	81.8	88.5	85.0

Table A.3: The results of Fold 3 for Stanford RE model

Relation Label	Precision	Recall	F-score
ban_Fuel_V_Date	77.8	58.3	66.7
use_Environment_friendly_V_Date	66.7	60.0	63.2
ban	84.2	94.1	88.9
use	91.7	78.6	84.6
Total	81.6	75.5	78.4

Table A.4: The results of Fold 4 for Stanford Relation Extraction model

Relation Label	Precision	Recall	F-score
ban_Fuel_V_Date	88.2	100.0	93.8
use_Environment_friendly_V_Date	100.0	75.0	85.7
ban	100.0	100.0	100.0
use	80.0	100.0	88.9
Total	94.2	98.0	96.1

Table A.5: The results of Fold 5 for Stanford RE model

Relation Label	Precision	Recall	F-score
ban_Fuel_V_Date	48.8	71.4	58.0
use_Environment_friendly_V_Date	5.6	22.2	8.9
ban	77.1	79.4	78.3
use	73.3	68.8	71.0
Total	47.2	69.0	56.1

Table A.6: The results of Fold 6 for Stanford RE model

Relation Label	Precision	Recall	F-score
ban_Fuel_V_Date	75.0	80.0	77.4
use_Environment_friendly_V_Date	71.4	45.5	55.6
ban	95.8	92.0	93.9
use	85.7	54.5	66.7
Total	85.2	74.2	79.3

Table A.7: The results of Fold 7 for Stanford RE model

Relation Label	Precision	Recall	F-score
ban_Fuel_V_Date	76.2	72.7	74.4
use_Environment_friendly_V_Date	36.4	40.0	38.1
ban	96.0	82.8	88.9
use	64.7	100.0	78.6
Total	74.3	76.4	75.3

Table A.8: The results of Fold 8 for Stanford RE model

Relation Label	Precision	Recall	F-score
ban_Fuel_V_Date	66.7	76.9	71.4
use_Environment_friendly_V_Date	50.0	37.5	42.9
ban	62.5	76.9	69.0
use	62.5	76.9	69.0
Total	70.2	75.5	72.7

Table A.9: The results of Fold 9 for Stanford RE model

Relation Label	Precision	Recall	F-score
ban_Fuel_V_Date	88.9	94.1	91.4
use_Environment_friendly_V_Date	100.0	81.8	90.0
ban	79.3	95.8	86.8
use	100.0	73.7	84.8
Total	88.6	87.3	87.9

Table A.10: The results of Fold 10 for Stanford RE model

Relation Label	Precision	Recall	F-score
ban_Fuel_V_Date	92.3	80.0	85.7
use_Environment_friendly_V_Date	66.7	85.7	75.0
ban	95.8	92.0	93.9
use	78.6	100.0	88.0
Total	86.7	89.7	88.1

A.2 RDF Schema Generation

The RDFS file generated using the motor vehicle pollution domain of discourse evaluation dataset and referenced in Chapters 3, 4 and 5 is presented in detail below.

```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF

    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:NS="http://example.com/"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
  <rdfs:Class rdf:about="http://example.com/state">
    <rdfs:subClassOf>
      <rdfs:Class rdf:about="http://example.com/attribute"/>
    </rdfs:subClassOf>
  </rdfs:Class>
  <rdfs:Class rdf:about="http://example.com/substance">
    <rdfs:subClassOf>
      <rdfs:Class rdf:about="http://example.com/matter"/>
    </rdfs:subClassOf>
  </rdfs:Class>
  <rdfs:Class rdf:about="http://example.com/day">
    <rdfs:subClassOf>
      <rdfs:Class rdf:about="http://example.com/timeunit"/>
    </rdfs:subClassOf>
  </rdfs:Class>
  <rdfs:Class rdf:about="http://example.com/Date">
    <rdfs:subClassOf rdf:resource="http://example.com/day"/>
  </rdfs:Class>
  <rdfs:Class rdf:about="http://example.com/matter">
    <rdfs:subClassOf>
      <rdfs:Class rdf:about="http://example.com/physicalentity"/>
    </rdfs:subClassOf>
  </rdfs:Class>
  <rdfs:Class rdf:about="http://example.com/abstraction">
    <rdfs:subClassOf>
      <rdfs:Class rdf:about="http://example.com/entity"/>
    </rdfs:subClassOf>
  </rdfs:Class>
  <rdfs:Class rdf:about="http://example.com/Fuel-V">
    <rdfs:subClassOf rdf:resource="http://example.com/substance"/>
  </rdfs:Class>
  <rdfs:Class rdf:about="http://example.com/physicalentity">
    <rdfs:subClassOf rdf:resource="http://example.com/entity"/>
  </rdfs:Class>
  <rdfs:Class rdf:about="http://example.com/attribute">
    <rdfs:subClassOf rdf:resource="http://example.com/abstraction"/>
  </rdfs:Class>
  <rdfs:Class rdf:about="http://example.com/timeunit">
    <rdfs:subClassOf>
```

```
      <rdfs:Class rdf:about="http://example.com/measure"/>
    </rdfs:subClassOf>
  </rdfs:Class>
  <rdfs:Class rdf:about="http://example.com/measure">
    <rdfs:subClassOf rdf:resource="http://example.com/abstraction"/>
  </rdfs:Class>
  <rdfs:Class rdf:about="http://example.com/Location">
    <rdfs:subClassOf>
      <rdfs:Class rdf:about="http://example.com/object"/>
    </rdfs:subClassOf>
  </rdfs:Class>
  <rdfs:Class rdf:about="http://example.com/situation">
    <rdfs:subClassOf rdf:resource="http://example.com/state"/>
  </rdfs:Class>
  <rdfs:Class rdf:about="http://example.com/Environment-friendly-V">
    <rdfs:subClassOf rdf:resource="http://example.com/situation"/>
  </rdfs:Class>
  <rdfs:Class rdf:about="http://example.com/object">
    <rdfs:subClassOf rdf:resource="http://example.com/physicalentity"/>
  </rdfs:Class>
  <rdf:Property rdf:about="http://example.com/move"/>
  <rdf:Property rdf:about="http://example.com/forbid">
    <rdfs:subPropertyOf>
      <rdf:Property rdf:about="http://example.com/command"/>
    </rdfs:subPropertyOf>
  </rdf:Property>
  <rdf:Property rdf:about="http://example.com/ask">
    <rdfs:subPropertyOf>
      <rdf:Property rdf:about="http://example.com/request"/>
    </rdfs:subPropertyOf>
  </rdf:Property>
  <rdf:Property rdf:about="http://example.com/communicate">
    <rdfs:subPropertyOf>
      <rdf:Property rdf:about="http://example.com/convey"/>
    </rdfs:subPropertyOf>
  </rdf:Property>
  <rdf:Property rdf:about="http://example.com/transfer">
    <rdfs:subPropertyOf rdf:resource="http://example.com/move"/>
  </rdf:Property>
  <rdf:Property rdf:about="http://example.com/use-Environment-friendly-V-Date">
    <rdfs:range rdf:resource="http://example.com/Date"/>
    <rdfs:domain rdf:resource="http://example.com/Location"/>
  </rdf:Property>
  <rdf:Property rdf:about="http://example.com/ban-Fuel-V-Date">
    <rdfs:range rdf:resource="http://example.com/Date"/>
    <rdfs:domain rdf:resource="http://example.com/Location"/>
    <rdfs:subPropertyOf>
```

```
      <rdf:Property rdf:about="http://example.com/outlaw"/>
    </rdfs:subPropertyOf>
  </rdf:Property>
  <rdf:Property rdf:about="http://example.com/use">
    <rdfs:range rdf:resource="http://example.com/Environment-friendly-V"/>
    <rdfs:domain rdf:resource="http://example.com/Location"/>
  </rdf:Property>
  <rdf:Property rdf:about="http://example.com/request">
    <rdfs:subPropertyOf rdf:resource="http://example.com/communicate"/>
    <rdfs:subPropertyOf rdf:resource="http://example.com/ask"/>
  </rdf:Property>
  <rdf:Property rdf:about="http://example.com/convey">
    <rdfs:subPropertyOf rdf:resource="http://example.com/transfer"/>
  </rdf:Property>
  <rdf:Property rdf:about="http://example.com/outlaw">
    <rdfs:subPropertyOf rdf:resource="http://example.com/forbid"/>
  </rdf:Property>
  <rdf:Property rdf:about="http://example.com/ban">
    <rdfs:range rdf:resource="http://example.com/Fuel-V"/>
    <rdfs:domain rdf:resource="http://example.com/Location"/>
    <rdfs:subPropertyOf rdf:resource="http://example.com/outlaw"/>
  </rdf:Property>
  <rdf:Property rdf:about="http://example.com/command">
    <rdfs:subPropertyOf>
      <rdf:Property rdf:about="http://example.com/order"/>
    </rdfs:subPropertyOf>
  </rdf:Property>
  <rdf:Property rdf:about="http://example.com/order">
    <rdfs:subPropertyOf rdf:resource="http://example.com/request"/>
  </rdf:Property>
</rdf:RDF>
```

Listing A.1: Motor Vehicle Pollution RDFS File generated using the Stanford CoreNLP, GATE and Regular Expression framework and extended using WordNet

Appendix B

The GATE RDF Dataset From Twitter Data Framework

As presented in Chapter 4, GATE uses a number of Configuration files with respect to RE which define a range of parameters and features which can be adjusted with respect to a variety of applications. The Configuration file used with respect to the evaluation of GATE RE model, using the motor vehicle pollution domain of discourse, in Chapter 4 is presented below.

B.1 Configuration File of GATE Relation Extraction

```
<DATASET>
    <INSTANCE–TYPE>RelationInstance</INSTANCE–TYPE>
    <INSTANCE–ARG1>Location</INSTANCE–ARG1>
    <INSTANCE–ARG2>Fuel_V</INSTANCE–ARG2>
    <FEATURES–ARG1>
      <ARG>
        <NAME>Location</NAME>
        <SEMTYPE>NOMINAL</SEMTYPE>
        <TYPE>Location</TYPE>
        <FEATURE>id</FEATURE>
      </ARG>
      <ATTRIBUTE>
        <NAME>Form</NAME>
        <SEMTYPE>NOMINAL</SEMTYPE>
        <TYPE>Token</TYPE>
        <FEATURE>string</FEATURE>
        <POSITION>0</POSITION>
      </ATTRIBUTE>
      <ATTRIBUTE>
        <NAME>POS</NAME>
        <SEMTYPE>NOMINAL</SEMTYPE>
```

```
        <TYPE>Token</TYPE>
        <FEATURE>category</FEATURE>
        <POSITION>0</POSITION>
      </ATTRIBUTE>
    </FEATURES-ARG1>
    <FEATURES-ARG2>
      <ARG>
        <NAME>Fuel_V</NAME>
        <SEMTYPE>NOMINAL</SEMTYPE>
        <TYPE>Fuel_V</TYPE>
        <FEATURE>id</FEATURE>
      </ARG>
      <ATTRIBUTE>
        <NAME>Form</NAME>
        <SEMTYPE>NOMINAL</SEMTYPE>
        <TYPE>Token</TYPE>
        <FEATURE>string</FEATURE>
        <POSITION>0</POSITION>
      </ATTRIBUTE>
      <ATTRIBUTE>
        <NAME>POS</NAME>
        <SEMTYPE>NOMINAL</SEMTYPE>
        <TYPE>Token</TYPE>
        <FEATURE>category</FEATURE>
        <POSITION>0</POSITION>
      </ATTRIBUTE>
    </FEATURES-ARG2>
    <ATTRIBUTE_REL>
        <NAME>distance</NAME>
        <SEMTYPE>NOMINAL</SEMTYPE>
        <TYPE>RelationInstance</TYPE>
        <ARG1>Location</ARG1>
        <ARG2>Fuel_V</ARG2>
        <FEATURE>distance</FEATURE>
    </ATTRIBUTE_REL>
    <ATTRIBUTE_REL>
        <NAME>poslist</NAME>
        <SEMTYPE>NOMINAL</SEMTYPE>
        <TYPE>RelationInstance</TYPE>
        <ARG1>Location</ARG1>
        <ARG2>Fuel_V</ARG2>
        <FEATURE>poslist</FEATURE>
    </ATTRIBUTE_REL>
    <ATTRIBUTE_REL>
        <NAME>Class</NAME>
        <SEMTYPE>NOMINAL</SEMTYPE>
        <TYPE>RelationClass</TYPE>
```

```
        <ARG1>Location</ARG1>
        <ARG2>Fuel_V</ARG2>
        <FEATURE>rel-type</FEATURE>
    </ATTRIBUTE_REL>
</DATASET>
```

Listing B.1: The Configuration File

B.2 GATE Relation Extraction

Detailed TCV results for the evaluation of the GATE RE model generated using the motor vehicle evaluation dataset are presented in detail below.

Table B.2.1: The results of Fold 1 for GATE RE model

Relation Label	Precision	Recall	F-score
ban_Fuel_V_Date	100.0	77.8	87.5
use_Environment_friendly_V_Date	0.0	0.0	0.0
ban	95.2	95.2	95.2
use	100.0	100.0	100.0
Total	73.8	68.2	70.0

Table B.2.2: The results of Fold 2 for GATE RE model

Relation Label	Precision	Recall	F-score
ban_Fuel_V_Date	87.5	73.6	80.0
use_Environment_friendly_V_Date	83.3	45.4	58.9
ban	96.0	89.0	92.3
use	75.0	100.0	85.7
Total	85.4	77.0	79.2

Table B.2.3: The results of Fold 3 for GATE RE model

Relation Label	Precision	Recall	F-score
ban_Fuel_V_Date	94.7	94.7	94.7
use_Environment_friendly_V_Date	11.0	80.0	17.8
ban	88.8	88.8	88.8
use	83.3	83.3	83.3
Total	69.2	86.7	71.1

Table B.2.4: The results of Fold 4 for GATE RE model

Relation Label	Precision	Recall	F-score
ban_Fuel_V_Date	30.5	68.7	42.3
use_Environment_friendly_V_Date	20.0	25.0	22.2
ban	85.0	87.1	86.1
use	66.7	100.0	80.0
Total	50.5	70.2	57.6

Table B.2.5: The results of Fold 5 for GATE RE model

Relation Label	Precision	Recall	F-score
ban_Fuel_V_Date	42.1	50.0	45.7
use_Environment_friendly_V_Date	60.0	22.2	42.8
ban	52.9	69.2	60.0
use	39.5	89.4	54.8
Total	48.6	60.5	50.8

Table B.2.6: The results of Fold 6 for GATE RE model

Relation Label	Precision	Recall	F-score
ban_Fuel_V_Date	50.0	86.6	63.4
use_Environment_friendly_V_Date	62.5	100.0	79.9
ban	88.4	88.4	88.4
use	100.0	100.0	100.0
Total	75.2	93.7	82.1

Table B.2.7: The results of Fold 7 for GATE RE model

Relation Label	Precision	Recall	F-score
ban_Fuel_V_Date	61.5	100.0	76.1
use_Environment_friendly_V_Date	0.0	0.0	0.0
ban	81.4	100.0	89.7
use	72.7	88.8	80.0
Total	53.9	72.2	61.4

Table B.2.8: The results of Fold 8 for GATE RE model

Relation Label	Precision	Recall	F-score
ban_Fuel_V_Date	47.3	75.1	58.1
use_Environment_friendly_V_Date	33.3	25.0	50.0
ban	100.0	100.0	100.0
use	80.0	100.0	88.8
Total	69.3	75.0	70.0

Table B.2.9: The results of Fold 9 for GATE RE model

Relation Label	Precision	Recall	F-score
ban_Fuel_V_Date	100.0	57.1	72.7
use_Environment_friendly_V_Date	26.6	57.1	36.3
ban	87.5	97.2	92.1
use	76.9	100.0	86.9
Total	72.7	77.8	72.0

Table B.2.10: The results of Fold 10 for GATE RE model

Relation Label	Precision	Recall	F-score
ban_Fuel_V_Date	43.7	58.3	50.0
use_Environment_friendly_V_Date	66.6	28.5	40.0
ban	77.3	93.1	84.5
use	95.2	90.9	93.0
Total	70.7	67.7	66.8

Appendix C

The Regular Expression RDF Dataset From Twitter Data Framework

C.1 Regular Expression Relation Extraction

Detailed TCV results for the evaluation of the Stanford RE model incorporated into the Regular Expression framework, and using the motor vehicle pollution dataset, are presented in detail below.

TABLE C.1.1: The results of Fold 1 for regular expression RE model

Relation Label	Precision	Recall	F-score
ban_Fuel_V_Date	76.5	92.9	83.9
use_Environment_friendly_V_Date	100.0	16.7	28.6
ban	85.2	100.0	92.0
use	44.4	50.0	47.1
Total	75.9	80.4	78.1

TABLE C.1.2: The results of Fold 2 for regular expression RE model

Relation Label	Precision	Recall	F-score
ban_Fuel_V_Date	81.0	100.0	89.5
use_Environment_friendly_V_Date	100.0	20.0	33.3
ban	96.4	93.1	94.7
use	83.3	83.3	83.3
Total	89.3	87.7	88.5

Table C.1.3: The results of Fold 3 for regular expression RE model

Relation Label	Precision	Recall	F-score
ban_Fuel_V_Date	76.2	59.3	66.7
use_Environment_friendly_V_Date	100.0	20.0	33.3
ban	83.9	81.2	82.5
use	40.0	33.3	36.4
Total	75.9	63.8	69.3

Table C.1.4: The results of Fold 4 for regular expression RE model

Relation Label	Precision	Recall	F-score
ban_Fuel_V_Date	69.2	90.0	78.3
use_Environment_friendly_V_Date	0.0	0.0	0.0
ban	77.3	89.5	82.9
use	83.3	71.4	76.9
Total	73.8	77.5	75.6

Table C.1.5: The results of Fold 5 for regular expression RE model

Relation Label	Precision	Recall	F-score
ban_Fuel_V_Date	40.0	54.5	46.2
use_Environment_friendly_V_Date	100.0	25.0	40.0
ban	91.3	100.0	95.5
use	65.0	92.9	76.5
Total	70.0	77.8	73.7

Table C.1.6: The results of Fold 6 for regular expression RE model

Relation Label	Precision	Recall	F-score
ban_Fuel_V_Date	50.0	50.0	50.0
use_Environment_friendly_V_Date	14.3	33.3	20.0
ban	89.5	94.4	91.9
use	100.0	57.1	72.7
Total	68.4	72.2	70.3

Table C.1.7: The results of Fold 7 for regular expression RE model

Relation Label	Precision	Recall	F-score
ban_Fuel_V_Date	68.4	76.5	72.2
use_Environment_friendly_V_Date	33.3	25.0	28.6
ban	96.0	88.9	92.3
use	71.4	41.7	52.6
Total	79.6	68.3	73.5

Table C.1.8: The results of Fold 8 for regular expression RE model

Relation Label	Precision	Recall	F-score
ban_Fuel_V_Date	57.1	66.7	61.5
use_Environment_friendly_V_Date	0.0	0.0	0.0
ban	87.5	100.0	93.3
use	85.7	66.7	75.0
Total	75.0	82.4	78.5

Table C.1.9: The results of Fold 9 for regular expression RE model

Relation Label	Precision	Recall	F-score
ban_Fuel_V_Date	27.3	90.0	41.9
use_Environment_friendly_V_Date	0.0	0.0	0.0
ban	70.0	87.5	77.8
use	47.5	70.7	56.9
Total	75.0	82.4	78.5

Table C.1.10: The results of Fold 10 for regular expression RE model

Relation Label	Precision	Recall	F-score
ban_Fuel_V_Date	69.2	69.2	69.2
use_Environment_friendly_V_Date	33.3	11.1	16.7
ban	96.8	96.8	96.8
use	100.0	77.8	87.5
Total	87.0	75.8	81.0

Appendix D

The Shortest Path Dependency Parsing and Word Mover's Distance RDF Dataset From Twitter Data Framework

D.1 SPDP-WMD Relation Extraction

D.1.1 Motor Vehicle Pollution

Detailed TVC results for the evaluation of the Stanford RE model, incorporated into the SPDP-WMD framework, using the motor vehicle pollution evaluation dataset are presented in detail below.

TABLE D.1.1: The results of Fold 1 for the SPDP-WMD Stanford RE model evaluation using the motor vehicle pollution domain evaluation dataset

Relation Label	Precision	Recall	F-score
ban	72.7	53.3	61.5
ban_cars	70.4	89.3	78.7
transition_vehicles	60.0	85.7	70.6
wants_electric	100.0	50.0	66.7
Total	69.5	80.2	74.5

Table D.1.2: The results of Fold 2 for the SPDP-WMD Stanford RE model evaluation using the motor vehicle pollution domain evaluation dataset

Relation Label	Precision	Recall	F-score
ban	60.0	56.2	58.1
ban_cars	72.7	84.2	78.0
transition_vehicles	88.9	47.1	61.5
wants_electric	100.0	66.7	80.0
Total	72.8	72.0	72.4

Table D.1.3: The results of Fold 3 for the SPDP-WMD Stanford RE model evaluation using the motor vehicle pollution domain evaluation dataset

Relation Label	Precision	Recall	F-score
ban	93.8	68.2	78.9
ban_cars	92.6	92.6	92.6
transition_vehicles	81.8	90.0	85.7
wants_electric	83.3	100.0	90.9
Total	91.1	87.6	89.3

Table D.1.4: The results of Fold 4 for the SPDP-WMD Stanford RE model evaluation using the motor vehicle pollution domain evaluation dataset

Relation Label	Precision	Recall	F-score
ban	60.0	60.0	60.0
ban_cars	84.4	74.5	79.2
transition_vehicles	76.5	72.2	74.3
wants_electric	66.7	30.8	42.1
Total	78.2	66.3	71.8

Table D.1.5: The results of Fold 5 for the SPDP-WMD Stanford RE model evaluation using the motor vehicle pollution domain evaluation dataset

Relation Label	Precision	Recall	F-score
ban	63.2	63.2	63.2
ban_cars	85.1	80.0	82.5
transition_vehicles	77.8	87.5	82.4
wants_electric	44.4	66.7	53.3
Total	75.0	75.9	75.4

TABLE D.1.6: The results of Fold 6 for the SPDP-WMD Stanford RE model evaluation using the motor vehicle pollution domain evaluation dataset

Relation Label	Precision	Recall	F-score
ban	85.7	31.6	46.2
ban_cars	92.7	80.9	86.4
transition_vehicles	62.5	76.9	69.0
wants_electric	11.5	60.0	19.4
Total	63.3	67.9	65.5

TABLE D.1.7: The results of Fold 7 for the SPDP-WMD Stanford RE model evaluation using the motor vehicle pollution domain evaluation dataset

Relation Label	Precision	Recall	F-score
ban	57.9	61.1	59.5
ban_cars	72.2	76.5	74.3
transition_vehicles	66.7	72.7	69.6
wants_electric	60.0	42.9	50.0
Total	66.7	68.6	67.6

TABLE D.1.8: The results of Fold 8 for the SPDP-WMD Stanford RE model evaluation using the motor vehicle pollution domain evaluation dataset

Relation Label	Precision	Recall	F-score
ban	60.0	78.9	68.2
ban_cars	93.0	90.9	92.0
transition_vehicles	73.3	50.0	59.5
wants_electric	66.7	26.7	38.1
Total	78.7	70.0	74.1

TABLE D.1.9: The results of Fold 9 for the SPDP-WMD Stanford RE model evaluation using the motor vehicle pollution domain evaluation dataset

Relation Label	Precision	Recall	F-score
ban	76.9	71.4	74.1
ban_cars	79.2	95.0	86.4
transition_vehicles	77.8	73.7	75.7
wants_electric	61.5	47.1	53.3
Total	76.1	77.8	76.9

Table D.1.10: The results of Fold 10 for the SPDP-WMD Stanford RE model evaluation using the motor vehicle pollution domain evaluation dataset

Relation Label	Precision	Recall	F-score
ban	44.4	42.1	43.2
ban_cars	66.0	67.4	66.7
transition_vehicles	54.8	40.5	46.6
wants_electric	50.0	25.0	33.3
Total	57.7	48.8	52.9

D.1.2 Diabetes

Detailed TCV results for the evaluation of the Stanford RE model, incorporated into the SPDP-WMD framework, using diabetes evaluation dataset are presented in detail below.

Table D.1.2.1: The results of Fold 1 for the SPDP-WMD Stanford RE model evaluation using the diabetes evaluation dataset

Relation Label	Precision	Recall	F-score
Associates_Symptoms	100.0	50.0	66.7
manage	86.4	51.4	64.4
things_cause	77.3	85.0	81.0
Total	80.9	67.9	73.8

Table D.1.2.2: The results of Fold 2 for the SPDP-WMD Stanford RE model evaluation using the diabetes evaluation dataset

Relation Label	Precision	Recall	F-score
Associates_Symptoms	90.9	40.0	55.6
manage	84.2	34.0	48.54
things_cause	84.4	52.9	65.1
Total	85.5	43.1	57.3

Table D.1.2.3: The results of Fold 3 for the SPDP-WMD Stanford RE model evaluation using the diabetes evaluation dataset

Relation Label	Precision	Recall	F-score
Associates_Symptoms	100.0	71.4	83.3
manage	79.5	72.9	76.1
things_cause	68.4	65.0	66.7
Total	78.4	69.7	73.8

Table D.1.2.4: The results of Fold 4 for the SPDP-WMD Stanford RE model evaluation using the diabetes evaluation dataset

Relation Label	Precision	Recall	F-score
Associates_Symptoms	90.9	66.7	76.9
manage	87.0	66.7	75.5
things_cause	75.4	71.9	73.6
Total	81.4	69.1	74.7

Table D.1.2.5: The results of Fold 5 for the SPDP-WMD Stanford RE model evaluation using the diabetes evaluation dataset

Relation Label	Precision	Recall	F-score
Associates_Symptoms	83.3	35.7	50.0
manage	92.6	55.6	69.4
things_cause	70.2	60.6	65.0
Total	77.8	56.0	65.1

Table D.1.2.6: The results of Fold 6 for the SPDP-WMD Stanford RE model evaluation using the diabetes evaluation dataset

Relation Label	Precision	Recall	F-score
Associates_Symptoms	90.0	50.0	64.3
manage	81.4	68.6	74.5
things_cause	84.6	64.7	73.3
Total	83.7	64.2	72.6

Table D.1.2.7: The results of Fold 7 for the SPDP-WMD Stanford RE model evaluation using the diabetes evaluation dataset

Relation Label	Precision	Recall	F-score
Associates_Symptoms	80.0	53.3	64.0
manage	81.4	71.4	76.1
things_cause	77.6	70.4	73.8
Total	79.4	68.6	73.6

Table D.1.2.8: The results of Fold 8 for the SPDP-WMD Stanford RE model evaluation using the diabetes evaluation dataset

Relation Label	Precision	Recall	F-score
Associates_Symptoms	80.0	19.0	30.8
manage	77.6	76.3	76.9
things_cause	90.0	75.0	81.8
Total	82.5	66.4	73.6

Table D.1.2.9: The results of Fold 9 for the SPDP-WMD Stanford RE model evaluation using the diabetes evaluation dataset

Relation Label	Precision	Recall	F-score
Associates_Symptoms	88.9	47.1	61.5
manage	82.0	67.2	73.9
things_cause	92.3	81.1	86.3
Total	87.9	71.7	79.0

Table D.1.2.10: The results of Fold 10 for the SPDP-WMD Stanford RE model evaluation using the diabetes evaluation dataset

Relation Label	Precision	Recall	F-score
Associates_Symptoms	92.9	50.0	65.0
manage	73.1	67.9	70.4
things_cause	88.2	75.0	81.1
Total	82.1	67.6	74.10

D.2 RDF Schema Generation

The RDFS files presented in Chapter 6 with respect to the motor vehicle pollution and diabetes domains are presented in detail below.

```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:NS="http://example.com/"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
  <rdfs:Class rdf:about="http://example.com/state">
    <rdfs:subClassOf>
      <rdfs:Class rdf:about="http://example.com/attribute"/>
    </rdfs:subClassOf>
  </rdfs:Class>
  <rdfs:Class rdf:about="http://example.com/substance">
```

```
    <rdfs:subClassOf>
      <rdfs:Class rdf:about="http://example.com/matter"/>
    </rdfs:subClassOf>
  </rdfs:Class>
  <rdfs:Class rdf:about="http://example.com/day">
    <rdfs:subClassOf>
      <rdfs:Class rdf:about="http://example.com/timeunit"/>
    </rdfs:subClassOf>
  </rdfs:Class>
  <rdfs:Class rdf:about="http://example.com/Date">
    <rdfs:subClassOf rdf:resource="http://example.com/day"/>
  </rdfs:Class>
  <rdfs:Class rdf:about="http://example.com/matter">
    <rdfs:subClassOf>
      <rdfs:Class rdf:about="http://example.com/physicalentity"/>
    </rdfs:subClassOf>
  </rdfs:Class>
  <rdfs:Class rdf:about="http://example.com/abstraction">
    <rdfs:subClassOf>
      <rdfs:Class rdf:about="http://example.com/entity"/>
    </rdfs:subClassOf>
  </rdfs:Class>
  <rdfs:Class rdf:about="http://example.com/fuel-V">
    <rdfs:subClassOf rdf:resource="http://example.com/substance"/>
  </rdfs:Class>
  <rdfs:Class rdf:about="http://example.com/physicalentity">
    <rdfs:subClassOf rdf:resource="http://example.com/entity"/>
  </rdfs:Class>
  <rdfs:Class rdf:about="http://example.com/attribute">
    <rdfs:subClassOf rdf:resource="http://example.com/abstraction"/>
  </rdfs:Class>
  <rdfs:Class rdf:about="http://example.com/timeunit">
    <rdfs:subClassOf>
      <rdfs:Class rdf:about="http://example.com/measure"/>
    </rdfs:subClassOf>
  </rdfs:Class>
  <rdfs:Class rdf:about="http://example.com/measure">
    <rdfs:subClassOf rdf:resource="http://example.com/abstraction"/>
  </rdfs:Class>
  <rdfs:Class rdf:about="http://example.com/Location">
    <rdfs:subClassOf>
      <rdfs:Class rdf:about="http://example.com/object"/>
    </rdfs:subClassOf>
  </rdfs:Class>
  <rdfs:Class rdf:about="http://example.com/situation">
    <rdfs:subClassOf rdf:resource="http://example.com/state"/>
  </rdfs:Class>
```

```
<rdfs:Class rdf:about="http://example.com/Environment-friendly-V">
  <rdfs:subClassOf rdf:resource="http://example.com/situation"/>
</rdfs:Class>
<rdfs:Class rdf:about="http://example.com/object">
  <rdfs:subClassOf rdf:resource="http://example.com/physicalentity"/>
</rdfs:Class>
<rdf:Property rdf:about="http://example.com/move"/>
<rdf:Property rdf:about="http://example.com/forbid">
  <rdfs:subPropertyOf>
    <rdf:Property rdf:about="http://example.com/command"/>
  </rdfs:subPropertyOf>
</rdf:Property>
<rdf:Property rdf:about="http://example.com/convert">
  <rdfs:subPropertyOf>
    <rdf:Property rdf:about="http://example.com/change"/>
  </rdfs:subPropertyOf>
</rdf:Property>
<rdf:Property rdf:about="http://example.com/ask">
  <rdfs:subPropertyOf>
    <rdf:Property rdf:about="http://example.com/request"/>
  </rdfs:subPropertyOf>
</rdf:Property>
<rdf:Property rdf:about="http://example.com/communicate">
  <rdfs:subPropertyOf>
    <rdf:Property rdf:about="http://example.com/convey"/>
  </rdfs:subPropertyOf>
</rdf:Property>
<rdf:Property rdf:about="http://example.com/ban-cars">
  <rdfs:range rdf:resource="http://example.com/fuel-V"/>
  <rdfs:domain rdf:resource="http://example.com/Location"/>
  <rdfs:subPropertyOf>
    <rdf:Property rdf:about="http://example.com/outlaw"/>
  </rdfs:subPropertyOf>
</rdf:Property>
<rdf:Property rdf:about="http://example.com/transfer">
  <rdfs:subPropertyOf rdf:resource="http://example.com/move"/>
</rdf:Property>
<rdf:Property rdf:about="http://example.com/wants-electric">
  <rdfs:range rdf:resource="http://example.com/Date"/>
  <rdfs:domain rdf:resource="http://example.com/Location"/>
</rdf:Property>
<rdf:Property rdf:about="http://example.com/request">
  <rdfs:subPropertyOf rdf:resource="http://example.com/communicate"/>
  <rdfs:subPropertyOf rdf:resource="http://example.com/ask"/>
</rdf:Property>
<rdf:Property rdf:about="http://example.com/transition-vehicles">
  <rdfs:range rdf:resource="http://example.com/Environment-friendly-V"/>
```

```
    <rdfs:domain rdf:resource="http://example.com/Location"/>
    <rdfs:subPropertyOf rdf:resource="http://example.com/convert"/>
  </rdf:Property>
  <rdf:Property rdf:about="http://example.com/convey">
    <rdfs:subPropertyOf rdf:resource="http://example.com/transfer"/>
  </rdf:Property>
  <rdf:Property rdf:about="http://example.com/outlaw">
    <rdfs:subPropertyOf rdf:resource="http://example.com/forbid"/>
  </rdf:Property>
  <rdf:Property rdf:about="http://example.com/ban">
    <rdfs:range rdf:resource="http://example.com/Date"/>
    <rdfs:domain rdf:resource="http://example.com/Location"/>
    <rdfs:subPropertyOf rdf:resource="http://example.com/outlaw"/>
  </rdf:Property>
  <rdf:Property rdf:about="http://example.com/command">
    <rdfs:subPropertyOf>
      <rdf:Property rdf:about="http://example.com/order"/>
    </rdfs:subPropertyOf>
  </rdf:Property>
  <rdf:Property rdf:about="http://example.com/order">
    <rdfs:subPropertyOf rdf:resource="http://example.com/request"/>
  </rdf:Property>
</rdf:RDF>
```

LISTING D.1: Motor Vehicle Pollution RDFS File generated using the SPDP-WMD framework and extended using WordNet.

```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF

    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:NS="http://example.com/"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">

<rdfs:Class rdf:about="http://example.com/disease">
    <rdfs:subClassOf>
      <rdfs:Class rdf:about="http://example.com/illness"/>
    </rdfs:subClassOf>
  </rdfs:Class>
  <rdfs:Class rdf:about="http://example.com/activity">
    <rdfs:subClassOf>
      <rdfs:Class rdf:about="http://example.com/act"/>
    </rdfs:subClassOf>
  </rdfs:Class>
  <rdfs:Class rdf:about="http://example.com/illhealth">
    <rdfs:subClassOf>
      <rdfs:Class rdf:about="http://example.com/pathologicalstate"/>
```

```
    </rdfs:subClassOf>
  </rdfs:Class>
  <rdfs:Class rdf:about="http://example.com/abstraction">
    <rdfs:subClassOf>
      <rdfs:Class rdf:about="http://example.com/entity"/>
    </rdfs:subClassOf>
  </rdfs:Class>
  <rdfs:Class rdf:about="http://example.com/beginning">
    <rdfs:subClassOf>
      <rdfs:Class rdf:about="http://example.com/happening"/>
    </rdfs:subClassOf>
  </rdfs:Class>
  <rdfs:Class rdf:about="http://example.com/happening">
    <rdfs:subClassOf>
      <rdfs:Class rdf:about="http://example.com/event"/>
    </rdfs:subClassOf>
  </rdfs:Class>
  <rdfs:Class rdf:about="http://example.com/psychologicalfeature">
    <rdfs:subClassOf rdf:resource="http://example.com/abstraction"/>
  </rdfs:Class>
  <rdfs:Class rdf:about="http://example.com/event">
    <rdfs:subClassOf rdf:resource="http://example.com/psychologicalfeature"/>
  </rdfs:Class>
  <rdfs:Class rdf:about="http://example.com/cause">
    <rdfs:subClassOf>
      <rdfs:Class rdf:about="http://example.com/origin"/>
    </rdfs:subClassOf>
  </rdfs:Class>
  <rdfs:Class rdf:about="http://example.com/care">
    <rdfs:subClassOf>
      <rdfs:Class rdf:about="http://example.com/work"/>
    </rdfs:subClassOf>
  </rdfs:Class>
  <rdfs:Class rdf:about="http://example.com/act">
    <rdfs:subClassOf rdf:resource="http://example.com/event"/>
  </rdfs:Class>
  <rdfs:Class rdf:about="http://example.com/state">
    <rdfs:subClassOf>
      <rdfs:Class rdf:about="http://example.com/attribute"/>
    </rdfs:subClassOf>
  </rdfs:Class>
  <rdfs:Class rdf:about="http://example.com/treatment">
    <rdfs:subClassOf rdf:resource="http://example.com/care"/>
  </rdfs:Class>
  <rdfs:Class rdf:about="http://example.com/illness">
    <rdfs:subClassOf rdf:resource="http://example.com/illhealth"/>
  </rdfs:Class>
```

```
<rdfs:Class rdf:about="http://example.com/cognition">
  <rdfs:subClassOf rdf:resource="http://example.com/psychologicalfeature"/>
</rdfs:Class>
<rdfs:Class rdf:about="http://example.com/origin">
  <rdfs:subClassOf rdf:resource="http://example.com/beginning"/>
</rdfs:Class>
<rdfs:Class rdf:about="http://example.com/physicalcondition">
  <rdfs:subClassOf>
    <rdfs:Class rdf:about="http://example.com/condition"/>
  </rdfs:subClassOf>
</rdfs:Class>
<rdfs:Class rdf:about="http://example.com/work">
  <rdfs:subClassOf rdf:resource="http://example.com/activity"/>
</rdfs:Class>
<rdfs:Class rdf:about="http://example.com/attribute">
  <rdfs:subClassOf rdf:resource="http://example.com/abstraction"/>
</rdfs:Class>
<rdfs:Class rdf:about="http://example.com/pathologicalstate">
  <rdfs:subClassOf rdf:resource="http://example.com/physicalcondition"/>
</rdfs:Class>
<rdfs:Class rdf:about="http://example.com/condition">
  <rdfs:subClassOf rdf:resource="http://example.com/state"/>
</rdfs:Class>
<rdfs:Class rdf:about="http://example.com/information">
  <rdfs:subClassOf rdf:resource="http://example.com/cognition"/>
</rdfs:Class>
<rdfs:Class rdf:about="http://example.com/evidence">
  <rdfs:subClassOf rdf:resource="http://example.com/information"/>
</rdfs:Class>
<rdfs:Class rdf:about="http://example.com/symptom">
  <rdfs:subClassOf rdf:resource="http://example.com/evidence"/>
</rdfs:Class>
<rdf:Property rdf:about="http://example.com/thing-cause">
  <rdfs:range rdf:resource="http://example.com/disease"/>
  <rdfs:domain rdf:resource="http://example.com/cause"/>
  <rdfs:subPropertyOf>
    <rdf:Property rdf:about="http://example.com/make"/>
  </rdfs:subPropertyOf>
</rdf:Property>
<rdf:Property rdf:about="http://example.com/think"/>
<rdf:Property rdf:about="http://example.com/Associates-Symptoms">
  <rdfs:range rdf:resource="http://example.com/disease"/>
  <rdfs:domain rdf:resource="http://example.com/symptom"/>
  <rdfs:subPropertyOf rdf:resource="http://example.com/think"/>
</rdf:Property>
<rdf:Property rdf:about="http://example.com/succeed"/>
<rdf:Property rdf:about="http://example.com/manage">
```

```
    <rdfs:range rdf:resource="http://example.com/disease"/>
    <rdfs:domain rdf:resource="http://example.com/treatment"/>
    <rdfs:subPropertyOf rdf:resource="http://example.com/succeed"/>
  </rdf:Property>
</rdf:RDF>
```

LISTING D.2: Diabetes RDFS File generated using the SPDP-WMD framework and extended using WorNet.

```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF


    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:NS="http://example.com/"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
  <rdfs:Class rdf:about="http://example.com/MedicalCause">
    <rdfs:subClassOf>
      <rdfs:Class rdf:about="http://example.com/MedicalEntity"/>
    </rdfs:subClassOf>
  </rdfs:Class>
  <rdfs:Class rdf:about="http://example.com/MedicalIndication">
    <rdfs:subClassOf>
      <rdfs:Class rdf:about="http://example.com/MedicalEntity"/>
    </rdfs:subClassOf>
  </rdfs:Class>
  <rdfs:Class rdf:about="http://example.com/MedicalSignOrSymptom">
    <rdfs:subClassOf>
      <rdfs:Class rdf:about="http://example.com/MedicalCondition"/>
    </rdfs:subClassOf>
  </rdfs:Class>
  <rdfs:Class rdf:about="http://example.com/MedicalEntity">
    <rdfs:subClassOf>
      <rdfs:Class rdf:about="http://example.com/Thing"/>
    </rdfs:subClassOf>
  </rdfs:Class>
  <rdfs:Class rdf:about="http://example.com/TreatmentIndication">
    <rdfs:subClassOf rdf:resource="http://example.com/MedicalIndication"/>
  </rdfs:Class>
  <rdfs:Class rdf:about="http://example.com/MedicalCondition">
    <rdfs:subClassOf rdf:resource="http://example.com/MedicalEntity"/>
  </rdfs:Class>
  <rdf:Property rdf:about="http://example.com/thing-cause">
    <rdfs:range rdf:resource="http://example.com/MedicalCondition"/>
    <rdfs:domain rdf:resource="http://example.com/MedicalCause"/>
  </rdf:Property>
  <rdf:Property rdf:about="http://example.com/Associates-Symptoms">
```

```
    <rdfs:range rdf:resource="http://example.com/MedicalCondition"/>
    <rdfs:domain rdf:resource="http://example.com/MedicalSignOrSymptom"/>
  </rdf:Property>
  <rdf:Property rdf:about="http://example.com/manage">
    <rdfs:range rdf:resource="http://example.com/MedicalCondition"/>
    <rdfs:domain rdf:resource="http://example.com/TreatmentIndication"/>
  </rdf:Property>
</rdf:RDF>
```

LISTING D.3: Diabetes RDFS File generated using the SPDP-WMD framework and extended using Schema.org.

Appendix E

Evaluation

E.1 Competency Questions Results

More results of the Competency Questions (CQs) for both pollution motor vehicle and diabetes datasets.

Pollution Motor Vehicle

MVP-CQ1 What are the names of the locations mentioned in the given Twitter collection?

Norway
CA
UK
California
Paris
Island
Services
European
Toyota
Charter
Jupiter
Madrid
norway
Ireland
Councilor
Speak
Mexico
Eastbourne
Uk
city
France
Piaggio
Tesla
Sweden
Cambridge

Beyond
Paris
China
Balearics
GER
Den
FIN
NeD
China
Spain
Sweden Denmark
Mexico
Segregated
News–Paris
Africa
Porter
Pakistan
Chile
October
Costa Rica
Oxford
British
Classic
Nigeria
france
France
Scotland
Electricity
Islington
Cambridge
Britain
Sunday
uk
Amsterdam
Conservative
Oslo
Finland
Paris
Imagine
Norway
Austria
France
Euromyths
Nobody
Honda
USA
Senate

```
Patanjali
Refund
London
India
Hackney
EU
Tower Hamlets
Theresa
US
Carpool
Campaign
Volewica
France
UK
Carscoops
Denmark
New Zealand
Oxford
Ivanishvili
Wales
cities
Oslo
Castles
```

MVP-CQ2 What are the types of fuel motor vehicles that are mentioned in the given Twitter collection?

```
petrol
diesel
fossil-fuel
fossil-fueled
Petrol
Gas
Diesel
Gasoline
gasoline
gas
diesel burning
diesel-fueled
electric
diesel-fuelled
PETROL
petrol/diesel
diesel-powered
Diesel Fueled
combustion engine
DIEsel-fuelled
```

```
Petro/Diesel
Combustion Engine
petrol-powered
fuel-driven
fossil fuels
```

MVP-CQ3 What are the types of environmentally friendly motor vehicle that are mentioned in the given Twitter collection?

```
hybrid
Electric
EVs
EV
zero-emission
electric
hybrids
emission-free
FuelCellVehicles
hybrid/electric
Hybrid
electricvehicles
ElectricVehicles
```

MVP-CQ4 What are the names of locations mentioned in the given Twitter collection that intend to ban fuel vehicles?

```
Norway
CA
UK
California
Paris
Island
Services
European
Toyota
Charter
Jupiter
Madrid
norway
Ireland
Councilor
Speak
Mexico
Eastbourne
Uk
city
France
```

Piaggio
Tesla
Sweden
Cambridge
Beyond
Paris
China
Balearics
GER
Den
FIN
NeD
China
Spain
Sweden Denmark
Mexico
Segregated
News–Paris
Africa
Porter
Pakistan
Chile
October
Costa Rica
Oxford
British
Classic
Nigeria
france
France
Scotland
Electricity
Islington
Cambridge
Britain
Sunday
uk
Amsterdam
Conservative
Oslo
Finland
Paris
Imagine
Norway
Austria
France
Euromyths

Nobody
Honda
USA
Senate
Patanjali
Refund
London
India
Hackney
EU
Tower Hamlets
Theresa
US
Carpool
Campaign
Volewica
France
UK
Carscoops
Denmark
New Zealand
Oxford
Ivanishvili
Wales
cities
Oslo
Castles

E.1.1 Diabetes

D-CQ1 What are the causes of diabetes?

stress
sugar
carbohydrate
obesity
sweets
Glucose
caffeine
Hypertension
Weight
Obesity
fat Obesity
carbs
carb
sleep
Sleep

```
overweight
sugary
obese
fat
fatty
sweet
sugars
sugarlevel
carbohydrates Sugar
carbohydrates
Carbs
weight
hypertension
Sugar
glucose
```

D-CQ2 What are the symptoms of diabetes?

```
anxiety
hungry
lupus
Hyperglycemia
dizziness
hypoglycemia Hyperglycemia
hypoglycaemic
polyuria
blood pressure
polycystic
thirst
hypos
stomach pain
Hypoglycemia
pancreatic
hypoglycemia
hyperglycemia
Depression Anxiety
diets
```

D-CQ3 How can patients manage their diabetes?

```
Exercise
Diet
drugs
insulin
keto
low-carb diets
lifestyle
```

```
activity
CBD
medication
physical activity
diets
exercise
Physical activity
education
Keto diet
physical
healthy
exercising
keto diet
Insulin
diet
healthy diet
ketogenic diet
walking
medications
```

D-CQ7 What types of treatment are there mentioned in the given Twitter collection?

```
Exercise
Diet
drugs
insulin
keto
low-carb diets
lifestyle
activity
CBD
medication
physical activity
diets
exercise
Physical activity
education
Keto diet
physical
healthy
exercising
keto diet
Insulin
diet
healthy diet
ketogenic diet
walking
```

```
medications
```

D-CQ8 What types of symptoms are there mentioned in the given Twitter collection?

```
anxiety
hungry
lupus
Hyperglycemia
dizziness
hypoglycemia Hyperglycemia
hypoglycaemic
polyuria
blood pressure
polycystic
thirst
hypos
stomach pain
Hypoglycemia
pancreatic
hypoglycemia
hyperglycemia
Depression Anxiety
diets
```

E.2 Human Expert Evaluation Template

The evaluation template used for the analysis of the diabetes RDFS by human experts is presented below:

Questions:

- Is the RDFS realistic?
- Are the responses provided by the populated RDFS meaningful?
- Are the generated relationship names appropriate?

1. **Introduction**

   It is widely acknowledged that social media features a wealth of user contributed content of all kinds. Twitter data has been analysed from many different perspectives to extract information that is both meaningful and useful. For example, tweets on Ebola disease have been analysed to examine how users communicate, on social media platforms, regarding disease outbreaks.

The key challenge in extracting meaningful content from Twitter data arises from its unstructured format, rendering it difficult to query. One possible mechanism whereby some form of structure can be imposed on unstructured data is by pre-processing the data and labelling potential items, for example, by identifying entities and relationships between entities. However, it is not enough to simply identify entities and relations to allow the querying of unstructured data. A shared understanding of the entities, and the corresponding relationships between them, is required. In other words, what is required is an "RDFS" - a formal shared understanding of a domain of discourse.

An example RDFS, generated using the proposed mechanism and directed at the domain of diabetes, is given in Figure 1. What we would like is for medical experts to look at the RDFS and provide an opinion as to its validity or otherwise. The RDFS is discussed in further detail in Section 1 below. The questions that might be directed at it in Section 2, and some further discussion in Section 3.

2. **Description of the RDFS Presented in Figure 1**

   With reference to figure 1 the colour encoding is as follows:

   Blue arrows indicate a "subclass" relationship, a word of more specific meaning than a general or superordinate term applicable to it. For example, spoon is a subclass of cutlery. In the RDFS, the class disease is a subclass of illness, and the class illness is a subclass of ill-health.

   Yellow, red and orange arrows indicate that there are relations between the classes indicated. Thus, the treatment class has a relationship with the disease and that relation is "manage". Similarly the cause and symptom classes have relationships with the disease class, the relations "thing-cause" and "associated-symptoms" respectively.

   An RDFS, once construction, can be populated with specific data. Populating the RDFS given in Figure 1 with Twitter data related to the domain of diabetes the values of the classes (not shown in the figure) become:

   (a) Disease class: diabetes.

   (b) Cause class cause: Obesity, Stress, Sugar, carbohydrate.

   (c) Treatment class treatment: Exercise, Insulin, activity, diet, drugs, keto.

   (d) Symptom class: Hyperglycemia, anxiety, dizziness, hungry, hypos, lupus, blood pressure.

   The first question we are interested in is:

   - Is the generated RDFS realistic?

3. **Querying the Populate RDFS**

One way of testing the proposed data generation mechanism is the query the populated RDFS using a database style query language. Some example queries and returned answers are given below.

(a) What are the causes of diabetes?

- Obesity, Stress, Sugar, carbohydrate.

(b) What are the symptoms of diabetes?

- Hyperglycemia, anxiety, dizziness, hungry, hypos, lupus, blood pressure.

(c) How can the patients manage their diabetes?

- Exercise, Insulin, activity, diet, drugs, keto.

(d) What kind of care is provided for n individual who has a disease?

- Treatment.

(e) What kind of evidence is required to determine whether an individual has disease?

- Symptom.

The second question we are interested in is:

Are the responses provided by the populated RDFS meaningful?

4. **RDFS Relation Names**

The third question we are interested in is:

- Are the generated relationship names appropriate?

More specifically:

(a) Do you think “manage” describes the relationship between the class treatment and the class disease?

(b) Do you think “thing-cause” describes the relationship between the class cause and the class disease?

(c) Do you think “associated-symptoms” describes the relationship between the class symptom and the class disease?

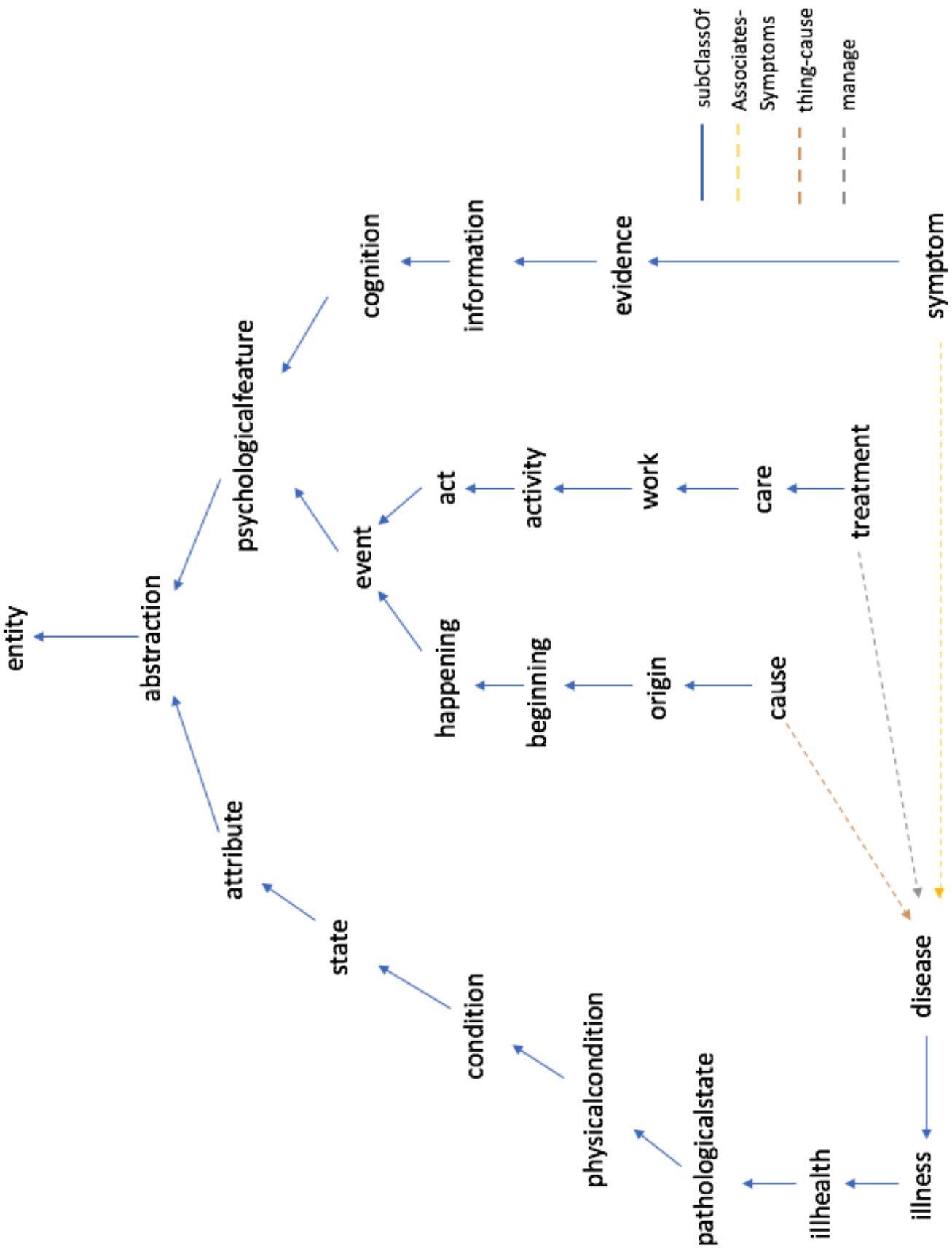

FIGURE E.1: Node and link diagram illustrating the diabetes RDFS