From Semi-Automated to Automated Methods of Ontology Learning from Twitter Data

Saad Alajlan[1,2], Frans Coenen[1], and Angrosh Mandya[1]

[1] Department of Computer Science, The University of Liverpool, Liverpool, UK
[2] College of Computer and Information Sciences, Al Imam Mohammad Ibn Saud Islamic University, Riyadh, Saudi Arabia
{s.alajlan,coenen,angrosh}@Liverpool.ac.uk

Abstract. This paper presents four different mechanisms for ontology learning from Twitter data. The learning process involves the identification of entities and relations from a specified Twitter data set, which is then used to produce an ontology. The initial two methods considered, the Stanford and GATE based ontology learning frameworks, are both semi-automated methods for identifying the relations in the desired ontology. Although the two frameworks effectively create an ontology supported knowledge resource, the frameworks feature a particular disadvantage; the time-consuming and cumbersome task of manually annotating a relation extraction training data sets. As a result two other ontology learning frameworks are proposed, one using regular expressions which reduces the required resource, and one that combines Shortest Path Dependency parsing and Word Mover's Distance to fully automate the process of creating relation extraction training data. All four are analysed and discussed in this paper.

Keywords: Ontology Learning · Resource Description Framework · RDF · Relation extraction · Name Entity Recognition · Twitter · Dependency Graphs · Stanford Relation Extraction · Regular Expression.

1 Introduction

It is widely acknowledged that social media features a wealth of user-contributed content of all kinds [28, 30, 32]. The dramatic increase, over recent times, in the usage of social media platforms, such as Facebook and Twitter, has resulted in a rapid increase in the availability of this data. Exploiting this data, by extracting useful information from it, has thus become a research focus in the field of data mining and knowledge discovery. 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 [1]. Another example can be found in [3], where Twitter data was used to investigate whether public sentiment factors can improve the forecasting of social, economic or commercial indicators.

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 un-structured data is by preprocessing 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 *ontology*.

In the context of computer science, an ontology is defined as an "explicit formal specification [concerning a given domain of discourse] of terms and the relationships that exist among them" [31]. A general all-encompassing ontology that covers all possible domains of discourse is beyond the means of computer science at present. The research focus has been on specific domains of discourse. Many specific domain ontologies have been proposed, especially to support the notion of the semantic web [15]. Examples include ontologies directed at e-commerce, artificial intelligence and bio-informatics [13]. The challenge of ontology generation is that it is a labour intensive and time consuming activity; the phrase "ontology generation bottleneck" is sometime encountered. The solution is to semi-automate or automate the ontology generation process, so called *ontology learning* [34]. Ontology learning, also known as ontology extraction, ontology generation, or ontology acquisition, is defined as the process of automatically or semi-automatically creating an ontology. The fundamental process is to extract the concepts included in a given domain, and the relationships between these concepts, and encode them using an ontology language so as to facilitate information retrieval [19]. Ontology learning is argued to be the most appropriate solution for providing meaningful structure to unstructured data. Although studies have been conducted to analyse Twitter data from different viewpoints, there are not many studies that have focused on *ontology learning* from Twitter data.

Against this background this paper presents several frameworks for ontology learning from Twitter data, founded on a range of tools and techniques: (a) using the Stanford NLP (Natural Language Processing) tool kit, (b) uisng the GATE (General Architecture for Text Engineering) tool kit, (c) using regular expressions and (d) combining Dependency Parse (DP) with text similarity measures such as Word Mover's Distance (WMD). The first three of these frameworks also present various challenges [2]. For example, the Stanford NLP and GATE frameworks require labelled training data to build a Relation Extraction model. Ontology learning using regular expressions mitigates against the problem of the manual creation of Relation Extraction model training data by defining regular expression patterns to provide a semi-automatic method of creating such data; however, the regular expressions still need to be defined and this remains a difficult, cumbersome and time-consuming task. Given these problems, the last proposed framework, the DPWMD framework, which entirely automates the process of generating Relation Extraction model training data.

The four proposed ontology learning frameworks were evaluated using example Twitter data collections. The entity and relation recognition and extraction models were evaluated using ten-fold cross-validation. The generated ontologies were further evaluated by directing pre-defined SPARQL queries at the populated ontologies. If the retrieved answers matched the expected answers, for a given set of queries, the generated ontology was deemed to be correct.

The rest of this paper is structured as follows. Section 2 gives an overview of previous work on ontology learning and relation extraction systems. Section 3 describes the proposed frameworks for ontology learning from Twitter data. Section 4 then considers the evaluation of the proposed frameworks. Finally, some conclusions are presented in Section 5.

2 Previous Work

Ontology learning from structured data such as tabular data, it is argued, is relatively straight forward as such data, by definition, is already structured in that it typically has a data schema associated with it, or at least table column headings; an example of ontology learning from relational databases can be found in [18]. Ontology learning from unstructured data, by definition, presents a greater challenge. There has been significant reported work on ontology learning from document collections. This is typically founded on the idea of relation extraction from text. An early example can be found in [14] where an automated relation labelling for ontology learning was presented. In comparison, there has been very little reported work on ontology learning from Twitter data which presents additional challenges over other forms of ontology learning from unstructured text in that Tweets are short and typically feature many grammatical errors and abbreviations. One example can be found in [2].

A review of relevant previous works on ontology learning from unstructured data is presented in this section. The section is divided into two parts. Work on ontology learning from free text is presented in Sub-section 2.1. Because of its relevance to ontology learning, a review of relation extraction for ontology learning is presented in Sub-section 2.2.

2.1 Ontology Learning

A recent exemplar approach to ontology learning from large collections of text data is presented in [20]. The proposed process starts by building a Word2Vec model [23] and defining some seed entities. The initial seed entities and Word2Vec model are then used to extract all terms representing entities (essentially a list of words) that represent domain concepts founded on those expressed in the seed entities. The list of words, and their Part Of Speech (PoS) tags, is then processed further to identify nouns (entities) and verbs (relations between entities). Finally, a hierarchical ontology is constructed using the Balanced Iterative Reducing and Clustering using Hierarchies (BIRCH) algorithm [33], which is an extended hierarchical clustering algorithm. The problem with this method is the need to manually define the initial seed set, if this is not done appropriately the correct entities and relations will not be identified and hence the ontology will be wrong. From the authors' own experience, a second issue is that using PoS tags is not an effective way to identify the relationship between entities. Further examples of ontology learning from free text can be found in [9] and [21].

2.2 Entity and Relation Extraction

The identification of the entities and relationships within a domain of discourse is an essential precursor for generating any ontology. There has been much work directed at the automated extraction of entities and relations from free text. This work is typically founded on the use of machine learning, particularly supervised learning, and NLP. A number of tools and techniques are available as a result of this work. Of particular relevance concerning the work presented in this paper is the Stanford CoreNLP which has been extensively used for entity identification and relation extraction, examples can be found in [2, 6]. In [6] evaluation was conducted using a corpus of 110 articles relating to the US National Football League. In [2] the evaluation domain used was a Twitter corpus of 300 tweets relating to car pollution; the same evaluation domain as considered in this paper. In [6] a relation extraction system, founded on Stanford CoreNLP, was introduced that could be customised. The Stanford CoreNLP tool kit is also used with respect to the first framework considered in this paper. An alternative NLP tool kit, the GATE (General Architecture for Text Engineering) tool kit [7] is used with respect to the second framework considered in this paper.

A particular challenge of using supervised learning to extract entities and relations is preparing the training data. In many reported cases this is done manually [5, 26, 29]. In [2] a new method was suggested, founded on Stanford CoreNLP, that reduced the effort required to manually label a training set by using regular expression rules derived from a small number of tweets, a "seed set". This approach is also considered in this paper (the third framework considered). However, regular expressions only offer a partial solution as the expressions themselves have to be defined manually.

The desired solution is to automate the process of identifying entities, and the relations between them, and to then use theses entities and relations to prepare an appropriate training set, which will then help to predict the relations. One potential solutions is to use shortest path dependency parsing to find the most important information (relations) between two entities. There has been some reported work on using dependency parsing to extract relations from free text [4, 8, 25]. In [4] the authors presented a dependency parsing based approach to extracting relations from the ACE (Automated Content Extraction) newspaper corpus. The corpus consists of 422 documents, with a separate set of 97 documents for testing. The hypothesis was that the information required to captured the relation between two entities in a sentence is contained in the shortest path between the two entities as represented in a dependency graph. In [4] the authors used the Combinatory Categorial Grammar (CCG) and the Context Free Grammar (CFG) parsers to obtain dependencies. In [8] the focus was on biomedical texts. The idea was to identify whether there was any interaction between two protein names featured in a sentence by analysis the paths between the proteins within a given dependency parse tree and a set of pre-labelled parse trees. Two mechanisms for similarity measurement were considered: cosine similarity and modified distance similarity. Support Vector Machines (SVM) and k Nearest Neighbour (kNN) machine learning was used to classify every two proteins within a given dependency parse tree by looking at the label of the most similar pre-labelled parse trees. In [25] relations linking proteins were extracted using constituency syntactic parsing and the shortest dependency path between two proteins using Stanford dependency parsing. Constituent dependency tree nodes associated with the nodes on the shortest dependency path were preserved. The fourth framework presented in this paper considers dependency parsing.

3 Semi-automated and Automated Ontology Learning Frameworks for Twitter Data

The ontology learning frameworks, specifically directed at ontology learning from Twitter data, are presented in this section. As noted earlier, four frameworks are presented: (a) a Stanford NLP tool kit based framework, (b) a GATE based framework, (c) regular expression based framework and (d) a Dependency Parsing and Word Mover's Distance (DPWMD) based framework. The architecture of the Stanford and GATE based ontology learning frameworks are similar to each other, starting with the collecting of tweets and relation extraction. For this purpose, the Stanford NER (Named Entity Recognition) and Stanford Relation Extraction tools were used in the proposed Stanford ontology learning framework. The Gazetteer and GATE relation extraction tools were used with respect to the GATE framework. The regular expression-based ontology learning framework focused on preparing the relation extraction training set in a semi-automated manner by defining regular expression patterns, followed by subsequently using the training set to build a Relation Extraction model. The DPWMD framework is the most sophisticated framework. Further details are presented in the following sub-sections with the greatest emphasis on the fourth framework, the DPWMD framework.

3.1 Stanford NLP Tool based Ontology Learning Framework

The pipeline architecture for the Stanford NLP tool-based ontology learning framework is given in Figure 1. From the figure it can be seen that the process starts with a collection of tweets. The Twitter data is cleaned (not shown in the Figure) using various pre-processing steps. For example by deleting the hyperlinks present in a tweet. The next step is the *knowledge extraction* stage, which comprises two tasks, namely: (a) Named Entity Recognition (NER) and (b) Relation Extraction. This is followed by mapping of the identified entities to different classes. For example, entity mentions such as “UK”, “USA” and “China” (country names) would be mapped to the class "countries". The mapping entities step is followed by ontology generation. The result is a RDF represented ontology which, once populated, can be queried for information retrieval purposes. More details regarding the NER and Relation Extraction models, and the ontology generation step, are provided below.

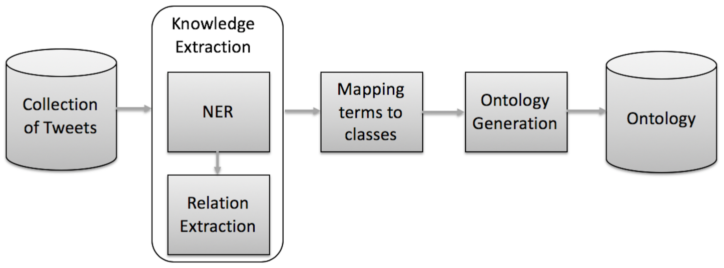
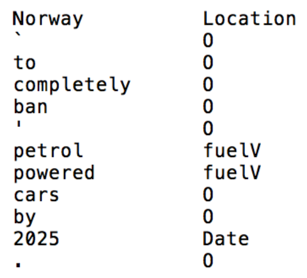


Fig. 1. Stanford Ontology Learning Framework [2].

Named Entity Recognition (NER) The Stanford NER tool was used to create a model to identify entities that featured within a tweet (after which they would be associated with classes). For example, the word "UK" belongs to the class *Location*. Currently, the Stanford NER tool supports the following seven classes: Location, Person, Organization, Money, Percent, Date and Time. Given this limitation, the Stanford NER tools is not able to identify entities belonging to other domains. For instance, in the context of the car pollution data used for the evaluation reported on later in this paper, the Stanford NER tool will not identify entity mentions belonging to classes such as "Fuel vehicles" and "Green vehicles". In order to identify such entities the Stanford NER tool has to be re-trained using an appropriate training set (corpus). The Stanford NER tool provides the facility to do this. Figure 2 shows an example of the training record format that is required to specify a NER training set to create a model to identify the entities belonging to the classes "Location", "Date" and "Fuel vehicles". The tweet shown in the Figure states: "Norway to completely ban petrol powered cars by 2025". The label "Location" indicates that the word "Norway" belongs to the class "Location", "O" indicates a *wild card*, "FuelV" is associated with the class "Fuel vehicles" and "Date" is associated with the class "Date". The re-trained NER model is then used to associate entities with classes, the relation extraction tool (described below) is then used to identify relations between the identified entities.

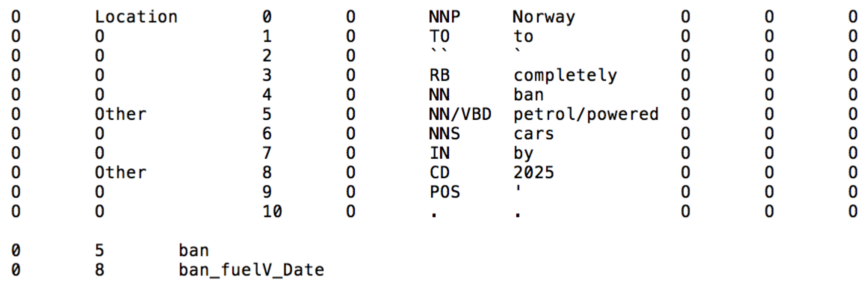
Relation Extraction Once a NER model has been created, the Stanford relation extraction tool was used to extract relations from tweets in a specific



```
Norway          Location
`               O
to              O
completely      O
ban             O
'               O
petrol          fuelV
powered         fuelV
cars            O
by              O
2025            Date
.               O
```

Fig. 2. Example Stanford NER training record [2].

domain. The Stanford relation extraction tool also provides the ability to train a relation exaction model using an appropriate training set [27]. The Stanford relation extraction tool has been successfully used for extract relations across different domains. For example, in [6], the tool was used to identify relations in the domain of American football. In [2] it was used to predict relations from Twitter data related to car pollution (the same domain as used with respect to this paper). Figure 3 shows an example of a Stanford relation extraction training record. The example again uses the tweet “Norway to completely ban petrol powered cars by 2025”. The entity class is given in column 2, the Part Of Speech (PoS) tag is given in column 5 and the relevant content of the tweet in column 6. The last two lines of the example express the entities and the relations between them: (a) the relation “ban” that exists between word 0 and word 4 (entities “Norway” and “petrol powered”), and (b) the relation “Ban fuelV Date” that exists between words 0 and 6 (entities “Norway” and “2025”).



```
0       Location    0     0     NNP     Norway          0     0     0
0       O           1     0     TO      to              0     0     0
0       O           2     0     ``      `               0     0     0
0       O           3     0     RB      completely      0     0     0
0       O           4     0     NN      ban             0     0     0
0       Other       5     0     NN/VBD  petrol/powered  0     0     0
0       O           6     0     NNS     cars            0     0     0
0       O           7     0     IN      by              0     0     0
0       Other       8     0     CD      2025            0     0     0
0       O           9     0     POS     '               0     0     0
0       O           10    0     .       .               0     0     0

0       5       ban
0       8       ban_fuelV_Date
```

Fig. 3. Example of Stanford relation extraction training data record [2].

The Relation Extraction model, once trained, was used to extract relations from tweets (in the specified domain). Results were filtered to include only those relations that were pertinent to the ontology [2]. In other words, only the relations identified in the training set were used to generate the ontology (described further below). All results were saved in the form of triples of the form: $\langle entity1, relation, entity2\rangle$. For instance, for the example tweet "Norway to completely ban petrol powered cars by 2025", the following triples were saved: $\langle Norway, ban, petrol\rangle$ and $\langle Norway, ban\ fuelv\ Date, 2025\rangle$. The final step was to re-use the Stanford NER tool to map entities to their respective classes in order to generate the ontology. For example, the entity $Norway$ would be mapped to the class $Location$.

Ontology Generation The final step in the Stanford ontology learning framework presented in Figure 1 is ontology generation. To this end, the LODRefine tool, recommended by the the World Wide Web Consortium (W3C), was used to convert entity-relation-entity triples to an RDF represented ontology. The LODRefine tool is a combination of the OpenRefine tool and a RDF extension [12]. The ontology contained four classes "Location", "FuelV", "GreenV" and "Date"; and four relations: "ban", "use", "ban FuelV Date" and "Use GreenV Date".

3.2 GATE-based Ontology Learning Framework

The GATE-based framework for ontology learning is presented in this section. The architecture for the framework is given in Figure 4. Comparison of Figures 4 and Figure 1 indicate that the distinction between the two is in the *Knowledge Extraction* step. Note that for the purpose of Knowledge Extraction two GATE components were used, the Gazetteer and the GATE relation extraction tool. The gazetteer uses a prescribed lists of words, describing entity classes, and uses this list to identify entities within given texts. The relation extraction component is then used to extract relations that exists between the identified entities. This is followed by the ontology generation step, which is similar to the final step in the Stanford tool based framework described above.

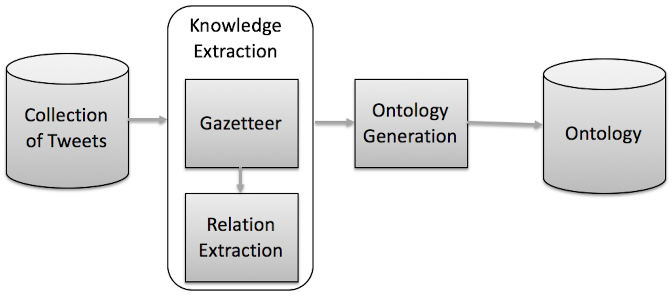


Fig. 4. GATE Ontology Learning Framework [2].

In further detail, the Knowledge Extraction process involves the following steps: (a) data pre-processing, (b) entity extraction using the gazetteer, (c) class pairing, (d) training set generation, (e) Relation Extraction model generation and (f) Relation Extraction model application. The data pre-processing involves word tokenisation and POS tagging. The A Nearly New Information Extraction (ANNIE) tool, available within GATE, was used for the pre-processing step. ANNIE assigns a sequential character ID number, c_i, to each character in a given Tweet T, $T = [c_1, c_2, \dots c_n]$. Each word is defined by a start (c_s) and end (c_e) id, based on the start and end character location within a tweet; a word id is thus expressed as $\langle c_s, c_e \rangle$; an example annotated tweet is given in Figure 5. In the next step, the gazetteer was used to annotate words in a tweet so as to identify entities and then assign class labels to those entities. The Gazetteer has predefined lists such as lists of locations, people and organisations. However, in the context of the vehicle pollution scenario used for evaluation purposes an appropriate lists was not available. As a result such a list had to be generated. JAPE [11] was used over the annotated tweets to pair classes. The Gazetteer assigns a unique Entity ID, e, to each entity.

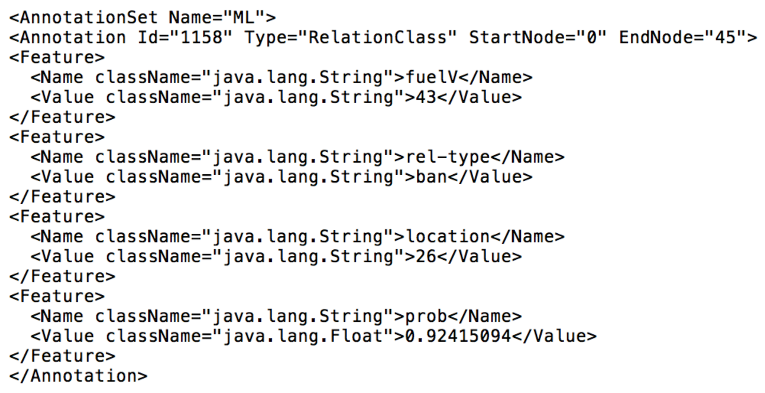
```
<TextWithNodes><Node id="0" />Norway<Node id="6" /> <Node id="7" />to<Node
id="9" /> <Node id="10" />'<Node id="11" />completely<Node id="21" /> <Node
id="22" />ban<Node id="25" /> <Node id="26" />petrol<Node id="32" /> <Node
id="33" />powered<Node id="40" /> <Node id="41" />cars<Node id="45" /> <Node
id="46" />by<Node id="48" /> <Node id="49" />2025<Node id="53" />'<Node
id="54" />.<Node id="55" /></TextWithNodes>


<Annotation Id="1111" Type="RelationClass" StartNode="0" EndNode="45">
<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">26</Value>
</Feature>
<Feature>
  <Name className="java.lang.String">fuelV</Name>
  <Value className="java.lang.String">43</Value>
</Feature>
</Annotation>
```

Fig. 5. Example annotated Tweet [2].

Similar to Stanford relation extraction, GATE follows a supervised learning approach. This means that GATE needs a training corpus. The training set was identified by assign relations manually to class pairs identified in the previews step. Once the training set had been identified, the GATE Relation Extraction model was generated. This model was then used to predicate classes and the relations between these classes.

An example of a GATE relation extraction training record is given in Figure 5. The Figure again shows the example tweet "Norway to completely ban petrol powered cars by 2025" used previously. The relation is "ban", which links entities



```
<AnnotationSet Name="ML">
<Annotation Id="1158" Type="RelationClass" StartNode="0" EndNode="45">
<Feature>
  <Name className="java.lang.String">fuelV</Name>
  <Value className="java.lang.String">43</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">location</Name>
  <Value className="java.lang.String">26</Value>
</Feature>
<Feature>
  <Name className="java.lang.String">prob</Name>
  <Value className="java.lang.Float">0.92415094</Value>
</Feature>
</Annotation>
```

Fig. 6. Example of a GATE Relation Extraction Result [2].

belonging to the class "Location" and the class "Fuelv". In the example, the character content of the tweet is indexed using a sequential numbering. The specific entities that were identified within the tweet have the IDs 0 to 6 for Norway belonging to the class "Location", and IDs 26 to 45 for petrol powered cars belonging to the class "Fuelv". The GATE Relation Extraction model, once trained, can be used to extract classes and associated relations from Twitter data. The results were of the form $\langle class1, class2, relation \rangle$. The result of the model prediction is shown in the Figure 6.

3.3 Regular expression-based Ontology learning framework

While the above described frameworks (using the Stanford coreNLP and GATE) provide useful mechanisms for supporting ontology learning, both involve significant end-user resource, particularly in the preparation of relation extraction training data. The entire process is therefore time consuming and does not generalise over all potential domains. The third mechanism considered in this paper was designed to address the training data preparation overhead by using regular expressions in order to limit the resource required with respect to the previous two frameworks. An overview of the ontology learning using regular expressions framework is given in Figure 7. Note that the framework interfaces with elements of Stanford CoreNLP, it could equally well be interfaced with GATE, however preliminary evaluation (reported on in Section 4 below) indicated that Stanford was a better option.

From Figure 7 it can be seen that the process starts with a collection of tweets. The first step is to generate a NER model using a "Seed Set"; in the context of the evaluation data set presented later in this paper 100 tweets were selected instead of the 300 used by the Stanford and Gate frameworks. The NER model is generated in a similar manner as described previously in Sub-section 3.1. The seed set was also used to generate a set of regular expression patterns. Three

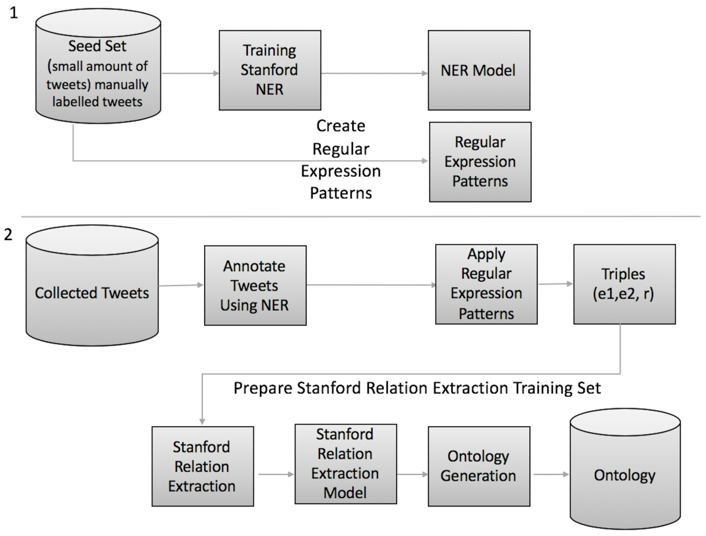

1
Seed Set (small amount of tweets) manually labelled tweets
Training Stanford NER
NER Model
Create Regular Expression Patterns
Regular Expression Patterns
2
Collected Tweets
Annotate Tweets Using NER
Apply Regular Expression Patterns
Triples (e1,e2, r)
Prepare Stanford Relation Extraction Training Set
Stanford Relation Extraction
Stanford Relation Extraction Model
Ontology Generation
Ontology

Fig. 7. Regular expressions ontology learning framework [2].

regular expression pattern categories were considered: (a) two entity expressions, (b) three entity expressions and (c) four entity expressions. The form of these patterns was: $\{e_1, ?, r, ?, e_2\}$, $\{e_1, ?, e_2, ?, r, ?, e_3\}$ and $\{e_1, ?, e_2, ?, e_3, ?, r, ?, e_4\}$ respectively, where ? was an arbitrary set of intervening words. In each case there were a number of variations, 6, 24 and 120 respectively. The advantage of the framework is that it can deal with more than two entities in a tweet, unlike comparable frameworks.

The input set of Tweets were annotated using the NER model. Then, the regular expression were applied to the annotated tweets to extract a set of triples of the form $\langle e_1, e_2, r\rangle$. Note that while, in certain cases, several such triples may be extracted from a single tweet, in some instances no triples were identified. The triples were then used to automatically generate a training set for the generation of the Relation Extraction model as described in sub-section 3.1 above. The remaining two steps are identical to those included in the previous two frameworks.

3.4 Dependency Parsing and Word Mover's Distance (DPWMD) based Ontology Learning Framework

The DPWMD ontology learning framework is presented in this section. The framework was designed to avoid the manual labelling of training data so as to reduce the resource required to prepare a training set. The framework is founded on the twin ideas of Dependency Parsing (DP) and Word Mover's Distance (WMD). The architecture of the proposed DPWMD framework is given in Figure 8. The figure shows a two-stage process: (a) creation of the desired NER model (top of the figure) and (b) creation of the Relation Extraction model (bottom of the figure).

The process starts with the creation of a NER model. With respect to the evaluation presented later in this paper Stanford CoreNLP was again used for this purpose. The input set of tweets were then pre-processed (not shown in the Figure) by deleting usernames of the tweet writers, filtering the non-ASCII characters and deleting the hyperlinks, after which shortest path dependency parsing [16] was used to extract the relations between pairs of entities. Word Mover's Distance (WMD) [17] was used to measure the similarity between relations that were extracted from the dependency parsing to create relation clusters. The next step was to produce a Relation Extraction model to facilitates relation prediction (Stanford was used for evaluation purposes). Following this, the identified entities were mapped into their respective classes (not shown in the Figure). Finally, the desired ontology was generated (using Apache Jena and WordNet to acquire all classes and property hypernyms). More detail concerning DP and WMD is provided in the following two sub-sections.

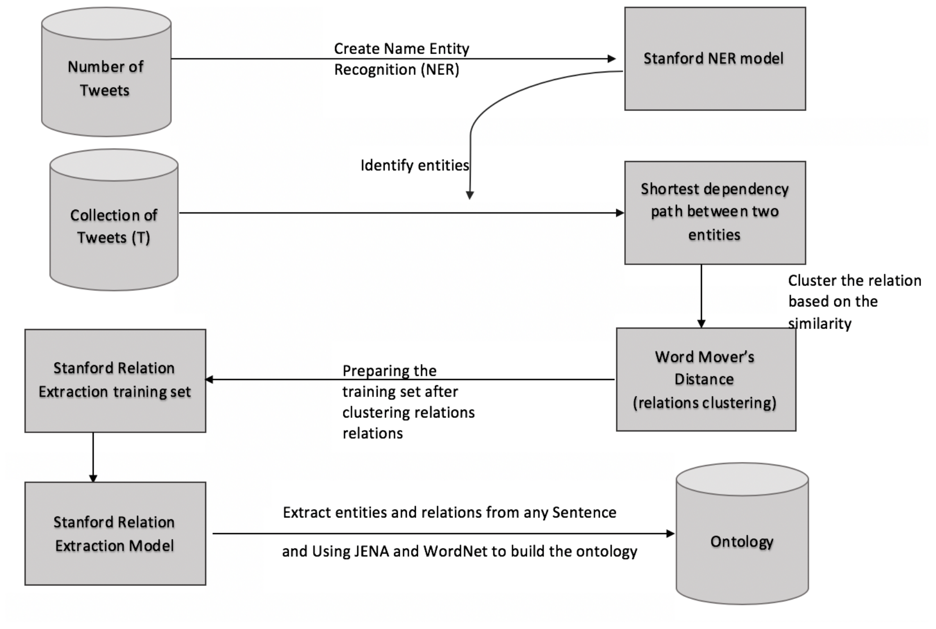

Number of Tweets
Create Name Entity Recognition (NER)
Stanford NER model
Identify entities
Collection of Tweets (T)
Shortest dependency path between two entities
Cluster the relation based on the similarity
Stanford Relation Extraction training set
Preparing the training set after clustering relations relations
Word Mover's Distance (relations clustering)
Stanford Relation Extraction Model
Extract entities and relations from any Sentence
and Using JENA and WordNet to build the ontology
Ontology

Fig. 8. Dependency Parsing and Word Mover's Distance (DPWMD) Ontology Learning Framework.

Dependency Parsing Dependency parsing is "the task of automatically analysing the dependency structure of a given input sentence" [16]. In Figure 8, the goal of the dependency parsing is to find the relationships that link the entities in a given tweet by identifying pairs of entities. The two tools utilized for this purpose were: (i) the Stanford NER tool and (ii) the Stanford Dependency Parsing tool. The Stanford NER tool was used to build a model to identify entities in a tweet by selecting two desired classes; the NER model was then used to capture all entities in the input data to, in turn, help the dependency parsing tool extract relations between the identified entities. The NER model was trained as described previously in Sub-section 3.1.

Once the NER model had been created, DP was used to extract relations between specific classes. For the evaluation presented later in this paper the pre-trained Stanford Dependency Parsing model was used. The hypothesis was that the shortest path between two entities described the relation between them [4]. The idea behind using Stanford NER and Stanford Dependency Parsing together is to specify the classes that the user wants to find relations between. The system therefore retrieved all the entities that belonged to the classes of interest and the relations between them. For example, Figure 9 shows the dependency parse for the tweet: "Norway to completely ban petrol cars by 2025". If the classes "Location" and "FuelV" were specified, the entities $Norway$ (e_1) and *petrol* (e_2) would be found, and consequently the relation (r) *ban cars* (according to the short path of the dependency graph shown in Figure 9). Alternatively, if "Location" and "Date" were selected as the target classes, e_1 would be $Norway$ and e_2 would be 2040, and the corresponding relation r would be *ban*. Figure 10 shows the dependency parse for another tweet: "The UK to ban the sale of diesel and petrol cars by 2040 to tackle". In this case e_1 is UK, belonging to the class "Location", e_2 is *diesel* belonging to the class "FuelV", and r is *ban sale cars*. Table 9 shows the targeted entities and the relations in these two example tweets. Note that the proposed approach has the advantage, unlike comparable frameworks, of being able to operate with more than two entities.

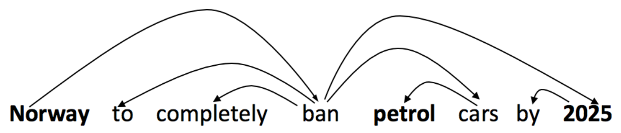


Fig. 9. Example dependency parse 1.

Word Mover's Distance Most of the relations between entities are sets of words. WMD is used to measure the similarity between relations obtained for different entity pairs belonging to similar classes. WMD is based on a word

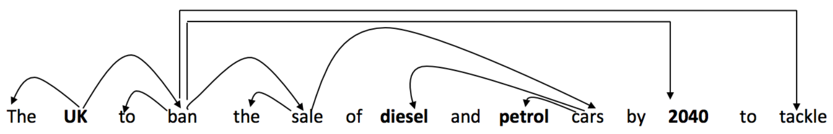
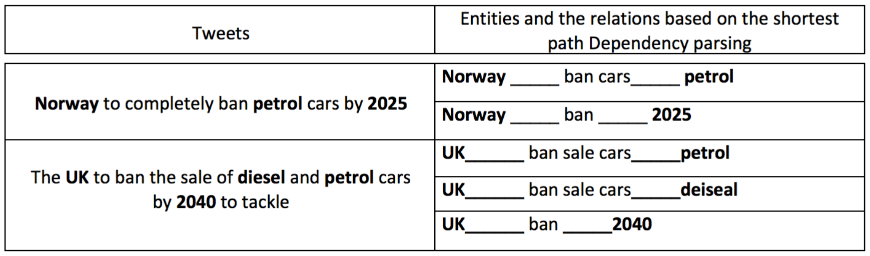


Fig. 10. Example dependency parse 2.



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**

Fig. 11. Entities and relations based on short path dependency parsing.

embedding method, which learns semantically meaningful representations for words. WMD "measures the dissimilarity between two text documents as the minimum distance that the embedded words of one document need to "travel" to reach the embedded words of another document" [17]. An example is given in Figure 12. WMD measures the distance between two text documents by calculating the minimum cumulative distance that all words in the first relation need to travel to match the second relation exactly.

Figure 13 shows an example where a list of relations has been extracted, using Stanford Dependency Parser, between two specific entity classes A and B. A Nearest Neighbour clustering (NNC) mechanism is used where by a cluster is created for the first relation. Then, the similarities between the first and second relations are measured using WMD. If the similarities between the two relations are equal to or higher than some threshold the second relation is added to the first cluster. Otherwise, a second cluster is created, and the second relation is added to the second cluster. Next, the third relation is considered and measured against the first cluster, and so on. Several clusters are therefore created, each containing a number of relations. One relation from the biggest cluster is selected to represent all the relations in the clusters. The next step was to automatically generate a training set for Relation Extraction model generation as described earlier in Sub-section 3.1.

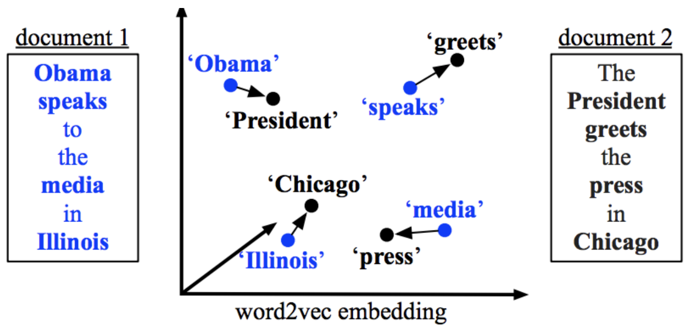


Fig. 12. Example of Word Mover's Distance captured from [17].

DPWMD Ontology Generation The ontology learning generation step is the final step in the proposed DPWMD ontology learning framework. There are many tools recommended by The World Wide Web Consortium (W3C) to use entity-relation-entity triples to generate an RDF represented ontology. In this paper, Apache Jena was used, which is an open-source Semantic Web framework based on Java. It provides an API to extract data and store a generated ontology in the form of a RDF graph. The resulting RDF data can be queried using SPARQL queries. WordNet was used with respect to the proposed DPWMD framework to enhance the ontology by adding super-classes and super-properties to the classes and relations that were identified. WordNet is an electronic lexical database for English nouns, verbs, adjectives and adverbs [10]. Uisng WordNet "Word forms are grouped into more than 117,000 sets of (roughly) synonymous word forms, so called synsets" and "These are interconnected by bidirectional arcs that stand for lexical (word-word) and semantic (synset-synset) relations, including hyper/hyponymy" [22]. Apache Jena was used with respect to the DPWMD framework instead of OpenRefine as used with respect to the three alternative frameworks described above. During implementation it was found that Apache Jena was easy to use and compatible with lexical databases such as WordNet (use to enrich the ontology in the context of the DPWMD Framework). An example of an RDF schema, generated using the DPWMD Framework, is given in Figure 14.

4 Evaluation

This section presents the evaluation of the four considered frameworks for ontology learning from Twitter data. For evaluation purposes, a specific Twitter data set was collected related to the car pollution domain. More detail concerning this data set is presented in Sub-section 4.1 below. The key objectives of the evaluation were as follows:

1. To evaluate the effectiveness, in terms of accuracy, of the Entity and Relation Extraction model generation process.

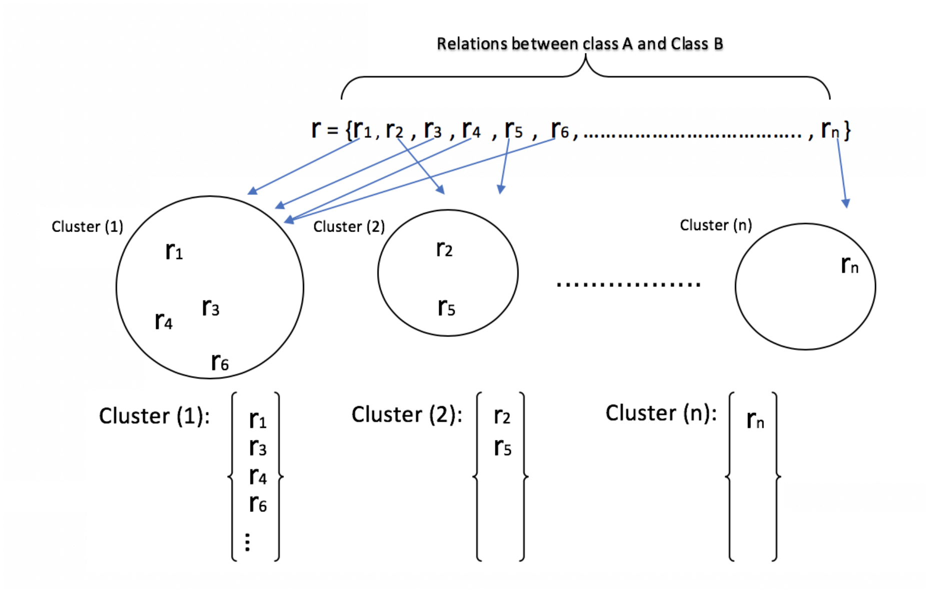

Relations between class A and Class B
r = {r1, r2, r3, r4, r5, r6, ..., rn}
Cluster (1)
Cluster (2)
Cluster (n)
r1
r3
r4
r6
r2
r5
rn
Cluster (1): r1 r3 r4 r6 ⋮
Cluster (2): r2 r5
Cluster (n): rn

Fig. 13. Relation list and clusters.

2. To investigate the effectiveness of the ontology generation process using the identified entities and relations.
3. To evaluate the utility of the generated ontology by querying it using sample queries.

Each objective is discussed in more details in Sub-sections 4.2 to 4.3 below.

4.1 Evaluation Data

This section briefly describes the car pollution domain evaluation data set. This dataset was also used in [2]. To create the data set tweets were collected using the Twitter API. The "car pollution" topic was chosen since it is easy to understand and hence any proposed mechanism using this data could be manually analysed, especially the generated ontology. The criteria for the collected tweets were that the tweets should contain content related to the banning of fuel vehicles or promoting the idea of using green vehicles in a certain location. Accordingly, three hundred tweets were collected and labelled manually, which then formed the evaluation data set. The tweets contained four entity classes: (a) *Location*, (a) *FuelV*, (c) *GreenV* and (d) *Date*; and four relation classes: (a) *ban*, (b) *Use*, (c) *ban fuelV Date* and (d) *use greenV Date*. The distribution of the classes and relations across the data set is shown in Figures 15 and 16; 768, 384, 198 and 1162 for the entity classes; and 241, 125, 166 and 87 for the relations class. Figures 15 and 16 indicate that, as expected, there were many more occurrences of entities than relations which meant that the data set was imbalanced. A typical

```
<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/greenV"/>
  <rdfs:Class rdf:about="http://example.com/physicalentity">
    <rdfs:subClassOf>
      <rdfs:Class rdf:about="http://example.com/entity"/>
    </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/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>
      <rdfs:Class rdf:about="http://example.com/abstraction"/>
    </rdfs:subClassOf>
  </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/Date">
    <rdfs:subClassOf rdf:resource="http://example.com/day"/>
  </rdfs:Class>
  <rdfs:Class rdf:about="http://example.com/abstraction">
    <rdfs:subClassOf rdf:resource="http://example.com/entity"/>
  </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/convert">
    <rdfs:subPropertyOf>
      <rdf:Property rdf:about="http://example.com/change"/>
    </rdfs:subPropertyOf>
  </rdf:Property>
  <rdf:Property rdf:about="http://example.com/transitionvehicles">
    <rdfs:range rdf:resource="http://example.com/greenV"/>
    <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/wantselectric">
    <rdfs:range rdf:resource="http://example.com/Date"/>
    <rdfs:domain rdf:resource="http://example.com/Location"/>
  </rdf:Property>

</rdf:RDF>
```

Fig. 14. Example of RDF File.

tweet features eight entities and two relations; not all entities were paired. The DPWMD ontology learning framework featured four different relations to those described above, because an automated method of relation extraction was used: (a) *ban cars*, (b) *transition vehicles*, (c) *ban* and (d) *wants electric*.

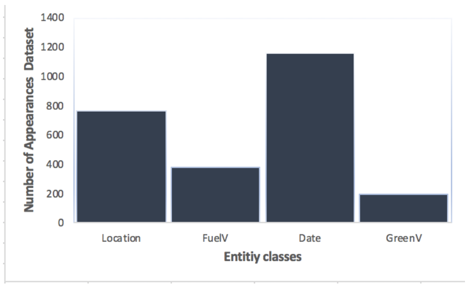


Fig. 15. Distribution of the entities per class across the training data set [2].

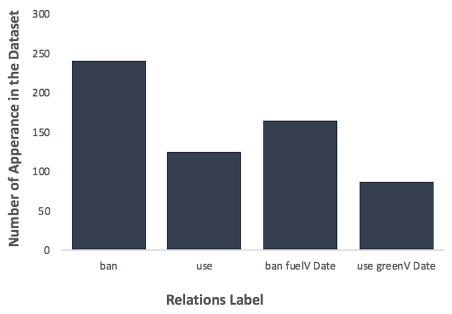


Fig. 16. Distribution of the relations per class across the training data set [2].

4.2 Effectiveness of Ontology Learning Frameworks

Stanford, GATE, Regular Expression and DPWMD relation extraction, and Stanford NER are evaluated in the next two sub-section by providing some statistics that show the accuracy of the models. Further details of this evaluation is presented below.

Stanford Name Entity Recognition (NER) Evaluation. For the Stanford Name Entity Recognition (NER) evaluation the evaluation data set of 300 tweets was divided into a 270 tweet training set and a 30 tweet test set which were used to generate and evaluate the Stanford NER model generation tool used by the proposed Stanford and DPWMD frameworks. The NER model was trained to identify the four entity classes: *Location*, *Date*, *FuelV* and *GreenV*. Table 1 presents the Ten-fold Cross-Validation results of the evaluation of the generated models. Inspection of the table indicates a small Standard Deviation (Stand. Dev.) of 0.71. It can thus be concluded that the generated Stanford NER model was consistent and accurate. Using the regular expression ontology learning framework the Stanford NER model was trained using a seed training set of 100 records, a third of the available evaluation data set. Because of the size of the training set, three-fold Cross-Validation was applied to evaluate the model. The results are presented Table 2. From this table it can be seen that the average F-score was less than that obtained when using 270 records for the training (see Table 1), but within acceptable limits.

Table 1. TCV results for the Stanford NER model evaluation [2].

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

Table 2. Three-fold Cross Validation results for the Regular Expression Stanford NER model evaluation [2].

Fold	F-score
1	49.0
2	53.0
3	55.0
Average	52.3
Standard Deviation (SD)	3.0

Stanford, GATE, Regular Expression, DPWMD Relation Extraction Evaluation Ten-fold Cross-Validation (TCV) was also used to evaluate all the Relation Extraction models (Stanford, GATE, Regular Expression and DP-WMD). Table 3 presents the results obtained. Inspection of the table indicates a wide spread of results (high standard deviations). The conjectured reason for this was that the training data was imbalanced in nature. Fold 5 is the worst performing fold. Inspection of this fold revealed that nine examples of the *use greenV Date* relationship class were included, but that only two were correctly classified. From Figure 16 it can be seen that there were only 87 entities within the *use greenV Date* class, which means only nine examples per fold. However, the *ban* relation class appears 241 times. Comparing between the GATE and Stanford Relation Extraction models, the Stanford model produced a better average F-score than the GATE system. This is the reason why it was decided to utilise the Stanford NLP Regular Expression model, as opposed to the GATE model, with respect to the remaining two frameworks. The F-score values ranged from 56.9 to 88.5, again the conjectured reason behind this was the imbalanced nature of the training data. However, what is interesting to note is that the average relation extraction F-score obtained using the Regular Expression approach was better than the GATE approach although not as good at the Stanford approach; whilst using a smaller relation extraction training set.

Table 3. F-Scores using GATE (RE), Stanford (RE) and Regular expression (Stanford CoreNLP) and DPWMD (Stanford CoreNLP) relation extraction.

Fold Num.	GATE (RE)	Stanford (RE)	Regular expression (Stanford CoreNLP)	DPWMD (Stanford CoreNLP)
Fold 1	70.6	76.7.2	78.1	74.5
Fold 2	79.4	85.0	88.5	72.4
Fold 3	71.1	78.4	69.3	89.3
Fold 4	57.6	96.1	75.6	71.8
Fold 5	50.8	56.1	73.7	75.4
Fold 6	82.1	79.3	70.3	65.5
Fold 7	61.4	75.3	73.5	67.6
Fold 8	70.0	72.7	78.5	74.1
Fold 9	72.0	87.9	56.9	76.9
Fold 10	66.8	88.1	81.0	52.9
Average	68.2	79.5	74.5	72.0
Standard Deviation (SD)	9.5	10.9	8.3	9.2

Inspection of the result for the last model, DPWMD, in Table 3 shows that Fold 10 produced the worst performance. Inspection of this fold indicated that there were only 16 examples of *wants electric*, and only four of them were predicted correctly since the number of *wants electric* was so small. The *wants electric* entity featured in the training set only 93 time, unlike other relations such as *ban cars* which was mentioned 493 time. Overall, the average F-scorer seems reasonable, reaching 72.04%. The F-score was less than that obtained using the regular expression framework, however, the relation extraction process using the DPWMD framework was fully automated, unlike in the case of the regular expression framework where the extraction was only semi-automated.

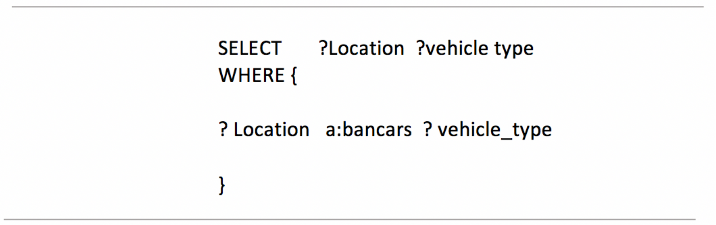
4.3 Evaluation of The Utility of The Generated Ontologies

From a visual inspection of the semantics of the ontologies generated using the Stanford, GATE and Regular expression ontology learning frameworks the ontologies seem correct. Other than visual confirmation, the generated ontologies were automatically evaluated by checking their syntactic integrity using the RDF

W3C Validation Tool. It was found that, in this simple Scenario, the generated ontologies were correct semantically and syntactically.

The ontology generated using the DPWMD ontology learning framework, presented in Figure 14, shows that the class *Location* was linked with three other classes (*Date*, *FuelV* and *GreenV*) and that the connection relations were *bancars*, *transition vehicles*, *ban* and *wants electric*, since Apache Jena was used to generate the ontology. The *Location* class super-classes were obtained from WordNet as discussed in sub-section 3.4. Also, the class *Date* had super-classes. Moreover, some properties (relations) had supper properties such as *transition vehicles* has *convert* and *change*. It is noticeable that some classes did not have a super-classes, such as *FuelV* and *GreenV*; this was because these were not defined in WordNet. Whatever the case, it was concluded that this obtained ontology was also semantically correct

To evaluate the utility of the generated ontologies, the ontologies were populated using an evaluation data set of 311 tweets, and then SPARQL was used to query the data. In [24] it was noted that "SPARQL could be used to express queries across diverse data sources, whether the data is stored natively as RDF or viewed as RDF via middleware". For the evaluation, Apache Jena was used to support the SPARQL querying of the RDF represented data. Figure 17 shows an example of a SPARQL query. The query asks for all locations that will ban any type of car. From the result of the query it can be concluded that the generated ontology was appropriate. To sum up, the ontology generated using the DPWMD ontology learning framework was considered to be better than that generated using the Stanford, GATE and Regular expression ontology learning frameworks, because it had more classes and properties generated by using WordNet hyponymys.



```
SELECT     ?Location  ?vehicle type
WHERE {

? Location   a:bancars  ? vehicle_type

}
```

Fig. 17. Example of SPARQL query [2].

5 Conclusion

This paper has presented four frameworks for learning ontologies from Twitter data: (a) Stanford, (b) GATE, (c) Regular expression and (d) Dependency Parsing and Word Mover's Distance. The output from these frameworks was an RDF represented ontology generated utilising either LODRefine or Apache Jena. A disadvantage of the Stanford and GATE relation extraction frameworks was that they needed pre-labelling relation extraction training data in a specified format. This was a significant disadvantage since the preparation of this training data required considerable end-user resource. The regular expression ontology learning framework was proposed to solve this problem. However, the regular expression ontology learning framework only provided a partial solution as it was still necessary to manually create the required regular expression patterns. The DPWMD ontology learning framework was therefore proposed which addressed the relation extraction training data problem using dependency parsing and Word Mover's Distance. All four proposed frameworks were evaluated using a car pollution evaluation data set comprised of 300 tweets. The generated ontology was evaluated by populating the generated ontologies with a further set of 311 tweets, and querying the data using SPARQL querying. The results were very encouraging. For future work the authors intend to consider a much larger evaluation data collection. It is anticipated that this will required further modification of the best performing proposed DPWMD ontology learning from Twitter data framework.

References

1. Ahmed, W., Demaerini, G., Bath, P.A.: Topics discussed on twitter at the beginning of the 2014 ebola epidemic in united states. iConference 2017 Proceedings (2017)
2. Alajlan., S., Coenen., F., Konev., B., Mandya., A.: Ontology learning from twitter data. In: Proceedings of the 11th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management - Volume 2: KEOD,. pp. 94–103. INSTICC, SciTePress (2019)
3. Arias, M., Arratia, A., Xuriguera, R.: Forecasting with twitter data. ACM Transactions on Intelligent Systems and Technology (TIST) **5**(1), 1–24 (2014)
4. Bunescu, R.C., Mooney, R.J.: A shortest path dependency kernel for relation extraction. In: Proceedings of the conference on human language technology and empirical methods in natural language processing. pp. 724–731. Association for Computational Linguistics (2005)
5. Carlson, A., Betteridge, J., Wang, R.C., Hruschka, E.R., Mitchell, T.M.: Coupled semi-supervised learning for information extraction. In: Proceedings of the third ACM international conference on Web search and data mining. p. 101. ACM (2010)
6. Chunxiao, W., Jingjing, L., Yire, X., Min, D., Zhaohui, W., Gaofu, Q., Xiangchun, S., Xuejun, W., Jie, W., Taiming, L.: Customizing an Information Extraction System to a New Domain. In: Regulatory Peptides. vol. 141, pp. 35–43. Association for Computational Linguistics (2007)
7. Cunningham, H.: Gate, a general architecture for text engineering. Computers and the Humanities **36**(2), 223–254 (2002)
8. Erkan, G., Ozgur, A., Radev, D.R.: Semi-supervised classification for extracting protein interaction sentences using dependency parsing. In: Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL) (2007)

9. Exner, P., Nugues, P.: Entity Extraction: From Unstructured Text to DBpedia RDF Triples. In: The Web of Linked Entities Workshop (WoLE 2012). pp. 58–69. CEUR (2012)
10. Fellbaum, C.: Wordnet. In: Theory and applications of ontology: computer applications, pp. 231–243. Springer (2010)
11. H. Cunningham D. Maynard, V. Tablan: JAPE: a Java Annotation Patterns Engine (Second Edition). Department of Computer Science, University of Sheffield (2000)
12. Harlow, C.: Data Munging Tools in Preparation for RDF: Catmandu and LODRefine. The Code4Lib Journal **30**(30), 1–30 (2015)
13. Iqbal, R., Murad, M.A.A., Mustapha, A., Sharef, N.M.: An analysis of ontology engineering methodologies: A literature review. Research Journal of Applied Sciences, Engineering and Technology **6**(16), 2993–3000 (2013)
14. Kavalec, M., Svaték, V.: A study on automated relation labelling in ontology learning. Ontology Learning from Text: Methods, evaluation and applications pp. 44–58 (2005)
15. Klusch, M., Kapahnke, P., Schulte, S., Lecue, F., Bernstein, A.: Semantic Web Service Search: A Brief Survey. KI - Künstliche Intelligenz **30**(2), 139–147 (2016)
16. Kübler, S., McDonald, R., Nivre, J.: Dependency parsing. Synthesis Lectures on Human Language Technologies **1**(1), 1–127 (2009)
17. Kusner, M., Sun, Y., Kolkin, N., Weinberger, K.: From word embeddings to document distances. In: International conference on machine learning. pp. 957–966 (2015)
18. Li, M., Du, X.Y., Wang, S.: Learning ontology from relational database. In: 2005 International Conference on Machine Learning and Cybernetics. vol. 6, pp. 3410–3415. IEEE (2005)
19. Maedche, A., Staab, S.: Ontology learning for the semantic web. IEEE Intelligent systems **16**(2), 72–79 (2001)
20. Mahmoud, N., Elbeh, H., Abdlkader, H.M.: Ontology learning based on word embeddings for text big data extraction. In: 2018 14th International Computer Engineering Conference (ICENCO). pp. 183–188. IEEE (2018)
21. Mazari, A.C., Aliane, H., Alimazighi, Z.: Automatic construction of ontology from arabic texts. In: ICWIT. pp. 193–202 (2012)
22. McCrae, J., Fellbaum, C., Cimiano, P.: Publishing and linking wordnet using lemon and rdf. In: Proceedings of the 3rd Workshop on Linked Data in Linguistics (2014)
23. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)
24. Prud'Hommeaux, E., Seaborne, A., Prud, E., Laboratories, H.p.: SPARQL Query Language for RDF. W3C working draftd pp. 1–95 (2008)
25. Qian, L., Zhou, G.: Tree kernel-based protein–protein interaction extraction from biomedical literature. Journal of biomedical informatics **45**(3), 535–543 (2012)
26. Riedel, S., Mccallum, A.: Relation Extraction with Matrix Factorization. In: Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. pp. 74–84 (2013)
27. Roth, D., Yih, W.t.: Global Inference for Entity and Relation Identification via a Linear Programming Formulation. Introduction to Statistical Relational Learning pp. 553–580 (2019)
28. Stieglitz, S., Dang-Xuan, L.: Social media and political communication: a social media analytics framework. Social network analysis and mining **3**(4), 1277–1291 (2013)

29. Takamatsu, S., Sato, I., Nakagawa, H.: Reducing Wrong Labels in Distant Supervision for Relation Extraction. In: Acl. pp. 721–729. Association for Computational Linguistics (2012)
30. Tanwar, M., Duggal, R., Khatri, S.K.: Unravelling unstructured data: A wealth of information in big data. In: 2015 4th International Conference on Reliability, Infocom Technologies and Optimization (ICRITO)(Trends and Future Directions). pp. 1–6. IEEE (2015)
31. Thomas Gruber: A translation approach to portable ontology specifications. Knowledge acquisition **5**(2), 199–220 (1993)
32. Xiang, Z., Gretzel, U.: Role of social media in online travel information search. Tourism management **31**(2), 179–188 (2010)
33. Zhang, T., Ramakrishnan, R., Livny, M.: Birch: an efficient data clustering method for very large databases. In: ACM Sigmod Record. vol. 25, pp. 103–114. ACM (1996)
34. Zhou, L.: Ontology learning: State of the art and open issues. Information Technology and Management **8**(3), 241–252 (2007)