PBML

The Prague Bulletin of Mathematical Linguistics

NUMBER 106 OCTOBER 2016 45–68

RealText-lex: A Lexicalization Framework for RDF Triples

Rivindu Perera, Parma Nand, Gisela Klette

Auckland University of Technology

Abstract

The online era has made available almost cosmic amounts of information in the public and semi-restricted domains, prompting development of corresponding host of technologies to organize and navigate this information. One of these developing technologies deals with encoding information from free form natural language into a structured form as RDF triples. This representation enables machine processing of the data, however the processed information can not be directly converted back to human language. This has created a need to be able to lexicalize machine processed data existing as triples into a natural language, so that there is seamless transition between machine representation of information and information meant for human consumption. This paper presents a framework to lexicalize RDF triples extracted from DBpedia, a central interlinking hub for the emerging Web of Data. The framework comprises of four pattern mining modules which generate lexicalization patterns to transform triples to natural language sentences. Among these modules, three are based on lexicons and the other works on extracting relations by exploiting unstructured text to generate lexicalization patterns. A linguistic accuracy evaluation and a human evaluation on a sub-sample showed that the framework can produce patterns which are accurate and emanate human generated qualities.

1. Introduction

Central to the entire discipline of Web of Data, is the concept of data representation using Resource Description Framework (RDF) triple form (Auer et al., 2007). An RDF triple is a subject, predicate, and object construct which makes data easily interlinked. The subject must always be a Uniform Resource Identifier (URI), so that it can be linked. The predicates are formed based on a clearly specified ontology. According to W3C recommendation on Linked Data (Lassila et al., 1998), an object can be a literal

Corresponding author: rivindu.perera@aut.ac.nz

instead of a URI. However, where possible, the objects must be represented in URI form as well.

As RDF triples open a lot of possibilities in Web data representation, it opens up opportunities to utilize this data in a wide range of application areas. One such area is Natural Language Generation (NLG). In essence, given RDF triples for an entity, there should be a possibility of generating a readable and meaningful description for that entity which has a wide variety of applications in diverse domains. For instance, an information kiosk in a museum can retrieve information from an open domain Linked Data resource (e.g., DBpedia) and transform this information to natural text to present to a user as a description with the possibility to expand or limit the amount of presented information according to the user needs. Such flexibility in amount of content presented is possible only because of the existence of the middle tier framework to transform the information to natural text, so that the information selection process is completely independent of the presentation.

The aim of the RealText project[1] is to generate readable, accurate, and informative descriptions from Web of Data (i.e., RDF triples) for Question Answering over Linked Data (QALD). RealText project consists of four major frameworks; namely,

- RealText$_{\text{lex}}$: lexicalization framework
- RealText$_{\text{agg}}$: aggregation framework
- RealText$_{\text{reg}}$: referring expression generation framework
- RealText$_{\text{rel}}$: surface realization framework

In this paper we limit the scope of our discussion to RealText$_{\text{lex}}$- lexicalization framework. This framework utilizes the DBpedia as the Linked Open Data resource to generate lexicalization patterns to transform triples into natural language sentences. In following sections we discuss the framework in detail.

The rest of the paper is organized as follows. Section 2 discusses what we mean by lexicalization in the context of Linked Data framework. In Section 3, we provide an introduction to DBpedia (Bizer et al., 2009; Lehmann et al., 2014), the RDF store used for the project and motivation for utilizing it. Section 4 discusses the related works in the area and compares our approach with them. In Section 5, we discuss the framework in detail. Section 6 discusses the experimental framework and provides an analysis of the results including some comparisons. Section 7 concludes the paper with an outlook on future work.

2. Lexicalization in RealText

Before we provide algorithmic details on our lexicalization framework, we first define the lexicalization in terms as used in Linked Data context.

[1]http://people.aut.ac.nz/~rperera/projects/realtext.html

Triple	Lexicalization Pattern
$\langle$*Steve Jobs, founder, Apple Inc*$\rangle_T$	$\langle$*S?, is the founder of, O?*$\rangle_L$
$\langle$*Klaus Wowereit, party, Social Democratic Party*$\rangle_T$	$\langle$*S?, is a member of, O?*$\rangle_L$
$\langle$*Canada, currency, Canadian dollar*$\rangle_T$	$\langle$*O?, is the official currency of, S?*$\rangle_L$
$\langle$*Canada, capital, Ottawa*$\rangle_T$	$\langle$*O?, is the capital city of, S?*$\rangle_L$
$\langle$*Rick Perry, birth date, 1950-03-04*$\rangle_T$	$\langle$*S?, was born on, O?*$\rangle_L$

Table 1. Example lexicalization patterns

According to Reiter and Dale (2000), lexicalization is the process of choosing words to represent abstract data in natural a language. This essentially focuses on selecting the content word to represent the same meaning.

The way that lexicalization is considered in RealText project is more sophisticated than the aforementioned definition. We consider the lexicalization as a process of finding patterns that can transform a given triple to the basic natural language form. To explain this further we have provided some examples in Table 1.

As shown in Table 1, RealText$_{\text{lex}}$ module is simply not looking for a lexical choice; it is meant to construct a syntactically correct and semantically appropriate pattern which can transform the triple into a natural language segment.

3. DBpedia: an interlinking hub for Linked Data

We utilize DBpedia as our RDF store for retrieving triples. The experiments to demonstrate lexicalization in this paper are specific to DBpedia due to three main reasons:

- sheer of volume
- as an interlinking hub
- open access

DBpedia is currently the fastest growing Linked Data resource that is available freely. Table 2 depicts relevant statistics illustrating its growth over five major releases. In Table 3 we compare the DBpedia against two other leading RDF stores. The results clearly shows that DBpedia has become a crystallization point in the Linked Data area hosting a vast amount of knowledge in triple form.

The nature of Linked Data is that the data is essentially interlinked. The amount of links (both incoming and outgoing) from the Linked Data resource enables it to be referenced from other similar resources. Table 4 summarises the interlinking for both incoming and outgoing links. The numbers show that DBpedia has become a central interlinking hub for Linked Data. Due to this high interlinkage, a framework that is based on DBpedia triples also indirectly contributes to the rest of the Linked Data cloud as well. This is possible because of the knowledge representation nature

Release version	Entities (in millions)	Triples (in billions)	Ontology classes
2014	4.58	3.00	685
3.9	4.26	2.46	529
3.8	3.77	1.89	359
3.7	3.64	1.00	320
3.6	3.50	0.67	272

Table 2. DBpedia growth rate in last 5 releases. Number of entities, triples and ontology classes are considered.

Triple store	Entities (in millions)	Triples (in billions)	Ontology classes	Query language
DBpedia	4.58	3	685	SPARQL
Freebase	44	2.4	40616	MQL
YAGO	10	0.12	451708	SPARQL

Table 3. Comparison of DBpedia statistics with Freebase and Yago

of Linked Data which enabled it to be reused without significant redefinition. This was one of the main motivations that influenced us to employ DBpedia for our lexicalization framework.

4. Related work

Duma and Klein (2013) introduce the LOD-DEF system, which focuses on sentence template based verbalization for Linked Data. The approach is based on selecting a sentence where subjects and objects of triples are mentioned and then removes them from the sentence to make that sentence a template. These templates can be later reused given similar triples. However, this slot filling exercise shows a very naive approach towards the lexicalization of Linked Data and cannot be employed for individual triples. Duma and Klein (2013) do not take additional steps to further abstract the sentence template to generalize it. If the template contains certain adjectives and adverbs which were related to the training triples, then these are propagated to the test phase which ultimately makes the template inaccurate. Additionally, they do not employ preprocessing steps such as co-reference resolution. It is rather hard to find sentences which do not contain co-references to main subject and therefore we can confidently assume that when applied on a wide scale text collection, LOD-DEF

Property	Incoming links	Outgoing links
Total links	39 million	4.9 million
Number of datasets	181	14
Top 5 resources	Linked Open Colours	Freebase
	DBpedia Lite	Flickr Wrapper
	Flickr Wrapper	WordNet
	Freebase	GeoNames
	YAGO	UMBEL

Table 4. Statistics on DBpedia interlinking

can end up extracting patterns with unnecessary information corresponding to co-references.

An approach which closely resembles our framework that can be found in literature is the Lemon Model (Walter et al., 2013). In this approach a sentence collection is employed to extract patterns to lexicalize triples, which is similar to our approach. However, the pattern extraction process uses typed dependency paths between subject and object values to derive the pattern. In essence, given a typed dependency-parsed sentence which contain the subject and object, the shortest path between the subject and object is searched and the sub-string is extracted. This sub-string is considered as a template to lexicalize the triple. Although the approach is linguistically sound, this method has several challenges. Firstly, the sentence collection is used as a raw set without preprocessing. This means that sentences having co-references to an entity are not considered as candidate sentences. Furthermore, the extracted pattern is not further processed to make it cohesive by removing adverbs and adjectives which can make the pattern specific to a triple. The framework proposed in this paper, addresses these issues. Instead of dependency parsing, we use a state-of-the-art relation extraction mechanism to extract cohesive patterns from natural text followed by a series of alignment phases in an effort to improve the accuracy.

Ell and Harth (2014) propose yet another verbalization model based on maximal sub-graph patterns. The main focus of this study is the transformation of multiple triples represented in natural language into a graph form. In contrast, our framework is focused on how lexicalization patterns can be generated to transform individual triples to natural language sentences. We are more interested in this specific objective so that the framework is as widely generalizable as possible, hence would be able to support integration with rest of the modules in RealText framework as introduced in Section 1. In addition, Ell and Harth (2014) do not carry out any additional processing for further realization of the extracted pattern. The idiosyncrasy of any natural language including the English, means that there has to be additional post-processing of the noise within unstructured text. This is achieved by the post-processing realiza-

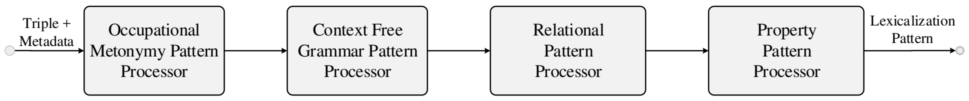
Triple +
Metadata
Occupational Metonymy Pattern Processor
Context Free Grammar Pattern Processor
Relational Pattern Processor
Property Pattern Processor
Lexicalization Pattern

Figure 1. Schematic representation of the lexicalization framework

tion operations which helps transform the noise text into accurate readable text which emanates human produced qualities.

5. RealText$_{\text{lex}}$ framework

The schematic representation of the RealText$_{\text{lex}}$ is shown in Fig. 1. The framework is composed of four main modules, all of them targeted towards generating lexicalization patterns. Although, they are placed in a pipeline, if one of the module finds a lexicalization pattern for a given triple, then the remaining modules will not be executed.

The following sections describe the modules shown in Fig. 1 in detail. Section 5.1 and Section 5.2 discuss the process of preparing input data (i.e., triples with metadata). Section 5.3 to Section 5.6 explain individual modules depicted in Fig. 1.

5.1. Triple retrieval

DBpedia SPARQL endpoint is available to extract triples through the published Web API[2]. However, due to availability issues our module uses the local cloned version of the DBpedia which provides uninterrupted resource access on demand. In particular, when requested for a particular RDF file of an entity, the retrieval module first checks the local repository and if not available downloads it from the DBpedia automatically and adds to the repository for any future use.

Currently, DBpedia RDF files contain two types of triples. The DBpedia properties which are extracted in raw form and provided under *dbprop schema*, do not contain a unique naming convention throughout the whole DBpedia entities. To address this drawback DBpedia *OWL property schema* was introduced by mapping *dbprop schema* to a consistent schema using community effort. This research utilizes the DBpedia *OWL property schema*.

We employed Jena RDF parser[3] to extract triples from the RDF files. However, the extracted triples contain some triples that are not appropriate for verbalization. These triples contain links to Wikipedia (e.g., Wikipedia page external link), identifiers (e.g.,

[2]http://dbpedia.org/sparql

[3]https://jena.apache.org/

viaf ID) and other properties (e.g., map caption, picture caption, and picture size) which were appropriately removed.

5.2. Metadata Embedding for Triples

The triples retrieved from the above step are associated with metadata, which pattern processing modules need to consult when generating lexicalization patterns. We provide below, a discussion on metadata and the methodology for deriving them if not available directly. Fig. 2 illustrates the proto-phrase representation of a triple to illustrate the use of the meta data.

Triple Verbalizations: The triples essentially do not contain verbal representation of data. In essence, the subject of a triple is a URI to an entity and predicates are represented as properties of a designed ontology. The only exception is that objects in a triple can contain literal values which are already verbalized, and in many occasions objects also contain URIs to other entities. The objective of triple verbalization is to transform the triple to derive the verbal form by addressing the aforementioned representations. Initially, we only address the above issues when verbalizing triples. Then in Section 5.5.3 we discuss further verbalization extensions for the triple object value, specific to relational pattern processing.

Ontology Class Hierarchy: The lexicalization patterns that are extracted for triples can be specific to the ontology class that they belong to. For an instance, consider two triples, $\langle$*Skype, author, Janus Friis*$\rangle_\mathrm{T}$ and $\langle$*The Scream, author, Edvard Munch*$\rangle_\mathrm{T}$, which are retrieved from DBpedia. Both of these triples contain the same predicate "*author*", however the entities described here belong to two different ontology classes, "*Software*" and "*Art Work*" respectively. The first triple can be lexicalized as "*Skype is developed by Janus Friis*", while the second triple will be generally lexicalized as "*The Scream is painted by Edvard Munch*". This differentiation is due to the fine ontology class that the subjects of the two entities belong to. This illustrates that associating the ontology hierarchy with the lexicalization pattern is critical when searching for a matching pattern for a new triple.

Predicate Information: We also tag each triple if the predicate requires a date value, measured numbers, or a normal numbers as the object. This is mainly to support the relational pattern extraction process and will be further explained in Section 5.5. To identify whether a predicate needs a date value, XML schema definitions associated (if any) with objects are consulted. The current data representation in DBpedia provides only the XML schema definition with the predicate representing numerical (e.g., double, integer) or temporal (e.g., date/time) properties. The predicates which require measurement unit in the real world are not associated with measurement unit information. This causes a severe issue when transforming these predicates into natural language. For example, to transform the triple $\langle$*Michael Jordan, height, 1.98*$\rangle_\mathrm{T}$ to natural language, we need the measurement unit for height. To address this , a measurement unit database was created which provides details on predicates which re-

Predicate	Ontology URI	Measurement Unit	
		Short Name	Long name
height	http://dbpedia.org/ontology/height	m	meter
budget	http://dbpedia.org/ontology/budget	USD	US Dollars
areaTotal	http://dbpedia.org/ontology/areaTotal	m^2	square meter
populationDensity	http://dbpedia.org/ontology/populationDensity	$ppkm^2$	persons per square kilometre

Table 5. Sample set of records from the measurement unit database

quire measurement units. Table 5 depicts sample set of selected records from this database.

Natural Gender: Natural gender of a subject is another property that affects lexicalization pattern not generalizable across all entities that associate a particular predicate. For instance consider the two triples, ⟨*Barack Obama, spouse, Michelle Obama*⟩$_{\mathrm{T}}$ and ⟨*Michelle Obama, spouse, Barack Obama*⟩$_{\mathrm{T}}$. Although they have the same predicate and both subjects belong to the same fine ontology class, a lexicalization pattern generated for the first triples such as ⟨*S?, is the husband of, O?*⟩$_{\mathrm{L}}$ cannot be used for the second triple as the natural gender of subjects are different. Due to this fact, the framework also associates the natural gender of the subject with the retrieved triple. To find natural gender we consult the DBpedia NLP (Natural Language Processing) dataset (Mendes et al., 2012) as a primary resource and missing records are added.

Object Multiplicity: Some triples contain the same subject and predicate with different objects. These triples with multiple objects require different natural language representation compared to another predicate with single object. For example consider triples related to *Nile River*, ⟨*Nile, country, Egypt*⟩$_{\mathrm{T}}$, ⟨*Nile, country, Rwanda*⟩$_{\mathrm{T}}$, and ⟨*Nile, country, Uganda*⟩$_{\mathrm{T}}$ which describe the countries that Nile River flows through. However, the same information is represented for *East River* as ⟨*East River, country, USA*⟩$_{\mathrm{T}}$ which describes that *East River* is located in USA. These two scenarios need two different lexicalization patterns such as ⟨*S?, flows through, O?*⟩$_{\mathrm{L}}$ and ⟨*S?, located in, O?*⟩$_{\mathrm{L}}$ respectively. This shows that object multiplicity plays a crucial role in deciding the most appropriate lexicalization pattern for a given triple. Therefore, each triple is associated with a property which describes the multiplicity computed by analysing the whole triple collection.

The triples with aforementioned metadata are passed to the pattern extraction modules (explained in Section 5.3 to Section 5.6).

$$\begin{bmatrix}
\text{Subject}_{Raw} & \text{Steve\_Jobs} \\
\text{Predicate}_{Raw} & \text{birthDate} \\
\text{Object}_{Raw} & \text{1955-02-24} \\
\text{Subject}_{Verbalized} & \text{Steve Jobs} \\
\text{Predicate}_{Verbalized} & \text{birth date} \\
\text{Object}_{Verbalized} & \begin{bmatrix} 1 & \text{February 24, 1955} \\ 2 & \text{24 February 1955} \\ 3 & \text{.....} \end{bmatrix} \\
\text{OntologyClasses} & \begin{bmatrix} 1 & \text{Agent} \\ 2 & \text{Person} \end{bmatrix} \\
\text{Predicate(RequireDate)} & \text{True} \\
\text{Predicate(DateInfo)} & \begin{bmatrix} \text{Type} & \text{Single} \\ \text{Format} & \text{YMD} \end{bmatrix} \\
\text{Predicate(RequireNormalNumber)} & \text{False} \\
\text{Predicate(RequireMeasuredNumber)} & \text{False} \\
\text{Predicate(MeasurementUnitInfo)} & \begin{bmatrix} \text{Short name} & \text{Null} \\ \text{Long name} & \text{Null} \end{bmatrix} \\
\text{NaturalGender} & \text{Male} \\
\text{Multiplicity} & \text{False}
\end{bmatrix}$$

Figure 2. Example proto-phrase specification of a triple

5.3. Occupational Metonym Patterns

Metonym is a single word or phrase which is referred to not by its own name, but by a name that is associated with the meaning of it (Kövecses and Radden, 1998). A well understood and highly used metonym is "Hollywood", which is used to denote the USA film industry. In the same way, there exist several metonyms which are created based on the occupations. Some of them are "commander", "owner", and "producer" which are used, respectively, to denote someone who gives commands to one or more people, someone who owns something, and someone who produces something.

5.3.1. Morphological Formation

Fig. 3 shows the classification hierarchy of English morphology and highlights under which category occupational metonyms are classified. Based on this classification, it is clear that occupational metonyms are nominalization of verbs.

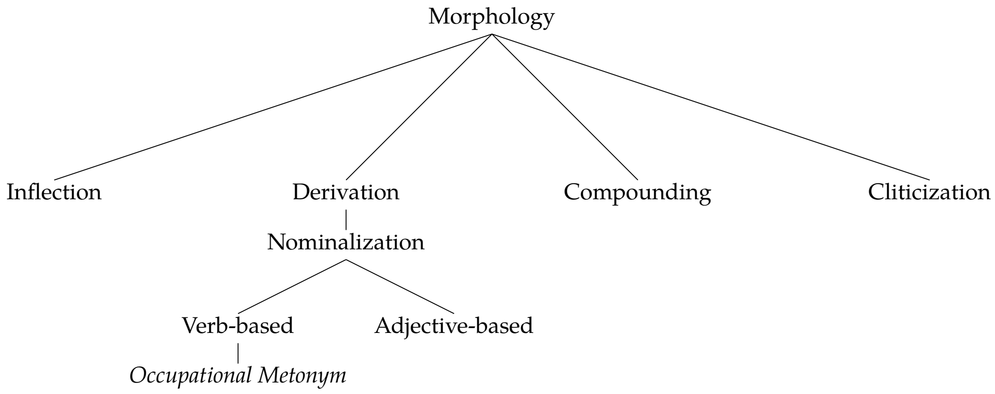

Figure 3. Classification hierarchy of English morphology

Two widely used forms of nominalization for occupation metonyms is the affixing of so-called agentive nominals; *-er* and *-or* nominals. These nominalizations can be directly applied on a base verb as well as can be applied on top of other morphological inflections. For example, Fig. 4(a) and Fig. 4(b) show two different occupational metonym forms in different granularity of applying nominalizations to form occupational metonyms.

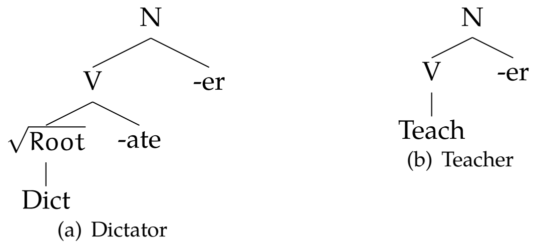

Figure 4. Two different occupational metonym formation applying -er nominals

Although it is possible to develop an unsupervised lexicon by nominalizing verbs, idiosyncrasy of English makes it rather hard. In some cases, the nominalized noun may also refer to non-agentive nominals (Schäfer, 2011) as in the two examples below.

- scratcher – a scratched lottery ticket
- broiler – a special part of a stove for cooking meat or fish

The occupational metonym lexicon used for this research is built under supervision by carefully considering accurate occupational metonyms.

There are multiple occasions where the aforementioned occupational metonyms appear as predicates of the triple. For example, the triple ⟨*Batman Begins, publisher, Christopher Nolan*⟩$_{\mathrm{T}}$ contains the "publisher" as the predicate which is an *-er* nominalized form the verb "publish". Since the base verb of the nominalization indicates the verb related to the profession, we can specify a straightforward lexicalization as ⟨*Christopher Nolan, published, Batman Begins*⟩$_{\mathrm{LT}}$. However, a further realization of the pattern can be formed by a passivized version as ⟨*Batman Begins, is published by, Christopher Nolan*⟩$_{\mathrm{LT}}$.

5.4. Context Free Grammar Patterns

Context Free Grammar (CFG) is considered dual purpose in NLP. This means that it can be used to understand the language as well as to generate language, based on a given grammar rules. For instance, Busemann (2005) describes the TG/2 shallow NLG system, which uses CFG rules and associated templates to generate natural language text. Furthermore, Stribling et al. (2005) demonstrated the SCiGen program which generates scientific papers using handwritten CFG rules. However, a burden associated with CFG is that the grammar rules need to be specified in advance, either as handwritten rules or as phrase structure trees derived from a seed corpus.

Due to the burdens associated with CFG based language production, our system does not use CFG as the main source. Only certain predicates which satisfy a predetermined constraint are associated with a CFG pattern. The constraint is that the predicate must either be a verb in past tense (e.g., influenced) or a predicate that is provided in passive form (e.g., maintained by). The CFG basic grammar form ($\mathcal{L}_0$) for single sentence level construction can be illustrated as follows:.

S → NP VP
NP → NNP
VP → VBD NP

where *S* denotes a sentence. *NP*, *NNP*, *VP*, and *VBD* represent a noun phrase, proper noun, verb phrase, and verb in past tense, respectively.

The CFG patterns are applied to the triples with predicates which are identified as verbs in past tense and if the identified verb has a frame NP↔VP↔NP. For an example, the triple ⟨*Socrates, influenced, Plato*⟩$_{\mathrm{T}}$ can be lexicalized as its predicate satisfies the above CFG rule (i.e., NP↔VP↔NP); in essence the verb "*influence*" has the required frame. In addition, to these types of triples, CFG pattern processing module also covers the predicates which are passive form verbs (e.g., ⟨*Aristotle, influencedBy, Parmenides*⟩$_{\mathrm{T}}$). Besides the methodology, CFG pattern processing also needs a verb frame database to identify whether verb contains the required frame. To accomplish this, we have built a verb frame database based on the VerbNet (Kipper et al., 2008), and this database also provides all the forms of the verb (past, present, and past participle).

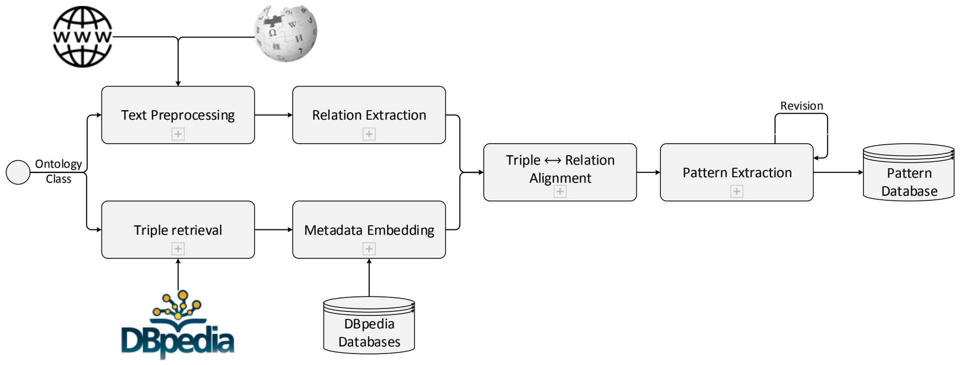

Figure 5. Relational pattern extraction process

Ontology class hierarchy	Agent, Person	Agent, Organisation, Company
Entities	Jimmy Wales Larry Sanger Natalie Portman	Google Intel Microsoft

Table 6. Sample input to the relational pattern extraction module. The example shows two ontology class hierarchies and associated entities. The actual input contains a series of class hierarchies and their associated entities.

5.5. Relational Patterns

Relational patterns are lexicalization patterns which are derived from the unstructured text using relation extraction. In brief, we process large number of unstructured text resources related to the triples and extract relations using Open Information Extraction (OpenIE) (Etzioni et al., 2008). These relations are then aligned with the triples to extract lexicalization patterns. Fig. 5 depicts the schematic representation of the relational pattern extraction process.

The module is initiated with ontology class hierarchy and associated entity collection. Table 6 depicts a sample input to the framework.

The module takes the aforementioned input and then moves to a parallel process of relation extraction and triple retrieval. During this process, it collects text related to each of the entities provided and then extract relations from the collected text. On the other hand triples related to these entities are retrieved from the DBpedia and enriched with metadata which is needed for the latter processes. The relations are then aligned with triples to extract relational patterns. We explain the process in detail in the following sections.

5.5.1. Text Preprocessing

We first retrieve unstructured text related to the entities from Wikipedia as well as from web based text resources. Since DBpedia contains information extracted from Wikipedia (i.e., Wikipedia Infoboxes which contain the unstructured data are converted to Linked Data), it is considered as a primary resource for text to be extracted. Articles extracted from Wikipedia are wrapped in a HTML boilerplate and this causes a serious issue when extracting pure text representation of the article. To address this the module employs the Boilerpipe (Kohlschütter et al., 2010), a shallow text feature based boilerplate removal algorithm.

However, Wikipedia itself is not sufficient to build a text corpus to extract wide range of relations. Therefore, we extract text from other web resources when building the text corpus.

What we expect from this text is a description related to a particular entity. Also sentences in the description should discuss information related to the entity. However, the text extracted from this step can contain co-references to already mentioned entities. Such conferences cannot be resolved once the relation extraction is performed. Therefore, as a preprocessing task we resolve the co-references by applying the entity full name. For example a paragraph like,

"*Abraham Lincoln* is regarded as one of America's greatest heroes. *He* is a remarkable story of the rise from humble beginnings to achieve the highest office in the land."

will be converted to,

"*Abraham Lincoln* is regarded as one of America's greatest heroes. *Abraham Lincoln* is a remarkable story of the rise from humble beginnings to achieve the highest office in the land."

We utilized the Stanford CoreNLP (Manning et al., 2014) for this task. However, manual corrections are done where necessary to stop propagating preprocessing errors to the latter modules which perform relation extraction and triple-relation alignment.

The result of this process, co-reference resolved set of sentences, is passed to the relation extraction process.

5.5.2. Relation Extraction

The task of relation extraction is to extract relation triples from the co-reference resolved text. The approaches towards relation extraction can be categorized into two camps, Closed Information Extraction (ClosedIE) (Moens, 2006) and Open Information Extraction (OpenIE) (Etzioni et al., 2008).

The ClosedIE, which is the traditional approach towards the relation extraction, attempts to extract natural language relations between two mentioned entities. This approach relies on rule based methods, kernel methods and sequence labelling methods. These methods have several key drawbacks compared to ClosedIE, such as, the need for hand-crafted rules, the need for hand-tagged data, and difficulties in domain adaptability.

For the purpose of applying relation extraction in this project, we looked at a domain independent technique, which looks at the linguistic structure of the sentence to extract relations. The recently proposed OpenIE was chosen for this purpose because it can handle a large scale open domain corpus such as the web (web as a corpus). OpenIE approach for relation extraction deviates significantly from the traditional relation extraction process. OpenIE identifies relations using relational phrases. A relational phrase is a natural language phrase that denotes a relation in a particular language. The identification of such relational phrases makes the system scalable by extracting arbitrary number of relations without tagged data. Furthermore, as relational phrases are based on linguistic knowledge and do not involve domain knowledge, OpenIE can work in multiple domains with minimum configurations.

We used Ollie (Mausam et al., 2012) OpenIE system in this module. Ollie has several advantages over the other two analysed systems, ClauseIE (Del Corro and Gemulla, 2013) and Reverb (Fader et al., 2011). ClasueIE is a clause based OpenIE module which performs on a pre-specified set of clauses derived from dependency parsing. Due to this specification, ClasueIE is unable to find many linguistic structures outside its scope. As Ollie is trained on large number of instances, it can extract several relations which are not covered by ClauseIE. On the other hand, Ollie is the successor of Reverb, and hence Ollie has significant improvements over Reverb.

5.5.3. Triple Relation Alignment

Once the relations are extracted using the OpenIE, we then align each relation with the triple to identify candidate relations which can be considered as lexicalization patterns. The aligner is mainly focused on mapping the subject and object of a triple with the arguments of a relation. To accomplish this mapping we employ the word overlapping measure. In particular, we employ the Phrasal Overlap Measure (POM) calculated according to (1).

$$sim_{overlap,phrase}(s_1, s_2) = \tanh\left(\frac{overlap_{phrase}\,(s_1,s_2)}{|s_1| + |s_2|}\right) \tag{1}$$

where, $s_1$and s_2 are two text strings and $overlap_{phrase}\,(s_1,s_2)$ is calculated using (2).

$$overlap_{phrase}(s_1, s_2) = \sum_{i=1}^{n} \sum_{m} i^2 \tag{2}$$

where, m is a number of i-word phrases that appear in two text strings.

The overlapping is calculated based on the exact textual representation. However, there can be scenarios where the object of a triple has more than one representation. For example, a date can be represented by multiple formats in natural language. Therefore, when calculating the overlap between the triple object and the relational argument phrase, all possible formats and verbalizations of the triple object must be consulted. The list below shows the verbalizations carried out to support phrasal overlap matching.

Date: The dates are verbalized for phrase matching by converting the date form to 7 different formats.

Measured Values: Triple objects which are measured values can be represented in multiple ways by associating them with different measurement units. However, the challenge here is that DBpedia does not provide the measurement unit of the original triple object value. To overcome this, a database is created which maps triple objects (only measured ones) to the measurement units.

Normal Numbers: Normal numbers are transformed to different scales as well as to verbal representation.

5.5.4. Pattern Extraction

The pattern extraction process elicits a lexicalization pattern from the aligned relation by substituting them with expressions. In essence we represent the subject as S? and object as O?.

A naive replacement of subject and object cannot be accomplished here due to several reasons.

Firstly, relation arguments can be mapped with one of the verbalizations instead of a triple object. If the relation object is aligned with one of the verbalizations of the object value, then direct replacement can cause information loss of unnecessary information being included in the pattern. To avoid this, the module searches for each verbalization in the triple argument and then replace them with required token.

Secondly, triple object can be mapped with a compound token from a relation argument. Consider the below example where a triple and an argument are provided, which has an acceptable alignment score.

Triple: $\langle$*Barack Obama, spouse, Michelle Obama*$\rangle_{\mathrm{T}}$

Relation: $\langle$*Barack Obama, was married to, Michelle LaVaughn Robinson Obama*$\rangle_{\mathrm{R}}$

In the above scenario, the triple object is mapped to the relation arg_2, which is expressive. A partial substitution of the triple object is possible in such scenarios, but they result in inaccurate data leaving some tokens unaffected. To solve this issue we introduce the dependency tree based compound token substitution. We first aggregate the relation segments, so that it is transferred to a natural language sentence. This sentence is then dependency parsed and universal typed dependencies (de Marneffe et al., 2014) are extracted for the relation argument. An example scenario of dependency parsed aggregated sentence for the relation $\langle$*Barack Obama, is married to, Michelle*

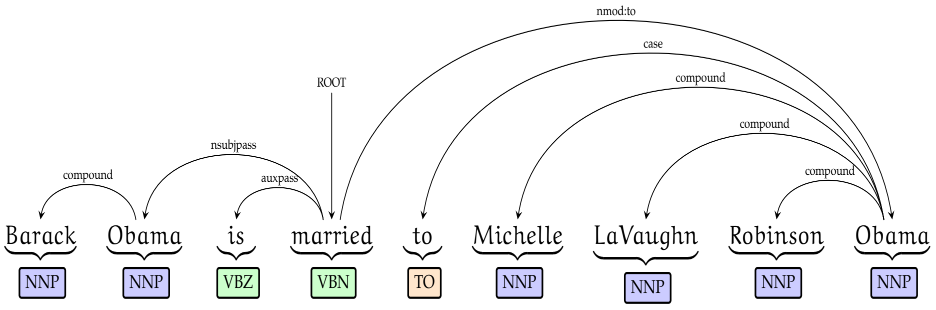

Figure 6. Compound noun identification based on the compound dependency relation

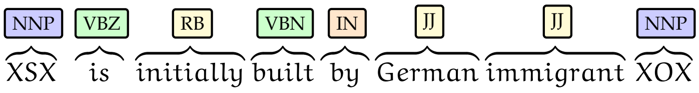

Figure 7. POS tagged transformed sentence

LaVaughn Robinson Obama$\rangle_{\mathrm{R}}$ is shown in Fig. 6. Typed dependencies which represent the compound relations are transformed back to multi-word phrases and other tokens are kept as separate. Next, each multi-word phrase is checked whether it contains the triple object tokens in full. In the occasion of such a scenario, the multi-word phrase is substituted with the triple object value (e.g., $\langle$*S?, is married to, O?*$\rangle_{\mathrm{L}}$).

Additionally, a post-processing revision step is designed to extract cohesive patterns which can be generalized regardless of the entity it is actually associated with. This cohesion is focused on filtering out adjectives and adverbs from the text. The extracted pattern is first transformed to natural language sentences by aggregating them and replacing subject and object expressions (*S*? and O?) with proper nouns (XSX and XOX) to avoid parser misclassification by taking the punctuations of the expressions into account. Fig. 7 depicts an example scenario where presence of adjectives make the patterns specific to a single entity. The example pattern is extracted from the sentence "Brooklyn Bridge is initially built by German immigrant John A. Roebling" for the triple $\langle$*Brooklyn Bridge, architect, John A. Roebling*$\rangle_{\mathrm{T}}$. However, this pattern cannot be generalized due to the adjectives and adverbs. Therefore, a further cleaning is done on patterns to remove adjectives and adverbs.

In addition to the aforementioned tasks, relational pattern extraction needs a threshold point to select a lexicalization pattern. This is because relational patterns come with different alignment scores. In the research we set this value to 0.21 as this value is corresponds to the single token matching in the alignment. In essence, if subject

Pattern	Predicate	Triple	Resulting lexicalized triple
$\langle$*S?'s P?, is, O?*$\rangle_{\mathrm{L}}$	height	$\langle$*Michael Jordan, height, 1.98*$\rangle_{\mathrm{T}}$	$\langle$*Michael Jordan's height, is, 1.98*$\rangle_{\mathrm{LT}}$
$\langle$*S?, has, O? P?*$\rangle_{\mathrm{L}}$	championships	$\langle$*Rubens Barrichello, championships, 0*$\rangle_{\mathrm{T}}$	$\langle$*Rubens Barrichello, has, 0 championships*$\rangle_{\mathrm{LT}}$
$\langle$*S?, is, O?*$\rangle_{\mathrm{L}}$	occupation	$\langle$*Natalie Portman, occupation, actress*$\rangle_{\mathrm{T}}$	$\langle$*Natalie Portman, is, an actress*$\rangle_{\mathrm{LT}}$
$\langle$*P? in S?, is, O?*$\rangle_{\mathrm{L}}$	largestCity	$\langle$*Australia, largesCity, Sydney*$\rangle_{\mathrm{T}}$	$\langle$*Largest city in Australia, is, Sydney*$\rangle_{\mathrm{LT}}$
$\langle$*S?, P?, O?*$\rangle_{\mathrm{L}}$	isPartOf	$\langle$*Delft, isPartOf, South Holland*$\rangle_{\mathrm{T}}$	$\langle$*Delft, is part of, South Holland*$\rangle_{\mathrm{LT}}$

Table 7. Property patterns with examples

and object are composed of one token each the normalized overlap measure of both subject and object equals to the 0.5 and hyperbolic tangent value of this is 0.45. Therefore, the multiplication of subject and object alignment equals to 0.2116. Since all other alignments are greater than this value of alignment, the single token alignment is considered as a cut-off point for relational lexicalization patterns.

5.6. Property Patterns

Property patterns specify a limited lexicon where certain predicates are associated with a pre-specified list of templates as lexicalization patterns. Five such patterns are specified, which will be applied only to the predetermined list of predicates. Table 7 list the 5 patterns with some examples of lexicalization when applied to triples with predetermined predicates. As shown in Table 7, the pattern may contain all three triple expressions which will be replaced by their verbalized form during the lexicalization. The module is designed in such a way that it can be scaled with newly defined property patterns without additional effort.

Since property pattern module is at the end of the pattern processing sequence, some of the triples may still use the pattern determined in a previous module instead of the property pattern thus making the property pattern to be ignored. This setting is arranged if majority of the triples are lexicalized with the property patterns, then the linguistic variation is negatively affected by having more similar sentences throughout a passage. Since language variety is one of the fact that make language naturalize, the framework attempts to maintain the variety to a level that it can achieve with the current settings.

Another important factor to notice in property patterns is that they are not associated with the ontology class of the subject. This is intentionally left in order to generalize the property patterns and apply them in a wide scale thus providing at least a basic lexicalization for majority of the triples.

6. Evaluation

In this section we discuss the evaluation of the lexicalization framework. Section 6.1 introduces the evaluation settings and present the acquired results. In Section 6.2, we discuss these results in detail and explain the limitations of the proposed framework.

6.1. Evaluation settings and results

Table 8 depicts a sample set of triples and some of the lexicalization patterns generated by the framework that can be associated with those triples. The table also depicts the pattern source of each lexicalization pattern and in case if the source is a relational pattern, the alignment score is also provided.

The evaluation of the framework is two fold. We first carried out an author evaluation on the linguistic accuracy of the extracted patterns and appropriateness to triples. The second evaluation was based on a survey where a group of participants were asked to rate the lexicalization patterns for their linguistic accuracy and appropriateness. Since human evaluation is resource expensive, the second evaluation considered only a randomly selected set of triples and associated lexicalization patterns from a pool.

6.1.1. Linguistic accuracy evaluation

This evaluation phase analysed lexicalization patterns selected for 400 triples from 28 entities categorized under 25 ontology classes. Since the framework described here is part a of a larger project which utilizes the Linked Data in Question Answering (QA), the source of triples is a collection of triples associated with entities in a Linked Data based QA dataset, QALD-2 (Unger, 2011) test dataset.

We evaluated each lexicalization pattern for their syntactic and semantic accuracy and the results are shown in Fig. 8. According to the figure it is clear that framework was able to generate grammatically correct patterns for 283 triples from the complete 400 triple data collection. The framework was unable to associate any pattern for 70 triples and generated incorrect patterns for 47 triples. Except for one entity (E-3), for all other entities, the framework was able to associate more than 50% of the triples with accurate lexicalization patterns.

During the analysis, we found several factors that affect a triple to be left without a lexicalization pattern since most of them need a relational pattern if the other modules

Triple	Pattern	Source	Score
$\langle$*Marlon Fernandez, birth place, London*$\rangle_{\mathrm{T}}$	$\langle$*S?, was born in, O?*$\rangle_{\mathrm{L}}$	Relational	0.8192
$\langle$*Marlon Fernandez, birth date, 2001-11-09*$\rangle_{\mathrm{T}}$	$\langle$*S?, was born on, O?*$\rangle_{\mathrm{L}}$	Relational	0.9028
$\langle$*K2, first ascent person, Achille Compagnoni*$\rangle_{\mathrm{T}}$	$\langle$*S?, was climbed by, O?*$\rangle_{\mathrm{L}}$	Relational	0.4182
$\langle$*Battlestar Galactica, network, Syfy*$\rangle_{\mathrm{T}}$	$\langle$*S?, is aired on, O?*$\rangle_{\mathrm{L}}$	Relational	0.2910
$\langle$*United Kingdom, currency, Pound sterling*$\rangle_{\mathrm{T}}$	$\langle$*O?, is the official currency of, S?*$\rangle_{\mathrm{L}}$	Relational	0.3852
$\langle$*Battlestar Galactica, creator, Glen Larson*$\rangle_{\mathrm{T}}$	$\langle$*S?, was created by, O?*$\rangle_{\mathrm{L}}$	Metonym	-
$\langle$*Rick Perry, successor, Bill Ratliff*$\rangle_{\mathrm{T}}$	$\langle$*O?, succeeded, S?*$\rangle_{\mathrm{L}}$	Metonym	-
$\langle$*Ottawa, population total, 883391*$\rangle_{\mathrm{T}}$	$\langle$*S?'s population total, is, O?*$\rangle_{\mathrm{L}}$	Property	-
$\langle$*Lisbon, highest region, Benfica*$\rangle_{\mathrm{T}}$	$\langle$*highest region in S?, is, O?*$\rangle_{\mathrm{L}}$	Property	-
$\langle$*Aristotle, influenced, Jewis Philosophy*$\rangle_{\mathrm{T}}$	$\langle$*S?, influenced, O?*$\rangle_{\mathrm{L}}$	CFG	-
$\langle$*Microsoft, founded by, Bill Gates*$\rangle_{\mathrm{T}}$	$\langle$*S?, is founded by, O?*$\rangle_{\mathrm{L}}$	CFG	-

Table 8. Sample set of triples, lexicalization patterns, and the pattern source. S? and O? denote subject and object respectively.

are incapable of assigning predefined lexicalization pattern. One of the main reasons for the relational pattern processing not being able to capture all possible scenarios is due to the lack of text available for entities used to extract patterns. This is mainly due to two reasons: firstly, some entities (e.g., Rubens Barrichello, Marlon Fenandez) do not have enough information recorded on the web, and secondly, the information is available but cannot be extracted due to technical limitations in the presentation layer (e.g., dynamic content). These two limitations will be further investigated and expanded as our future work.

Another aspect of lexicalization is that some entities used for relational pattern mining might have acronyms which are frequently used. For example, the entity Secret Intelligence Service is called MI6 in text collection. This affects the alignment of relations with the triples since triples use the full name while the relation which is extracted from text which uses the acronym. At this level of research, we did not focus on acronym resolution, however, it is one of the obvious future tasks.

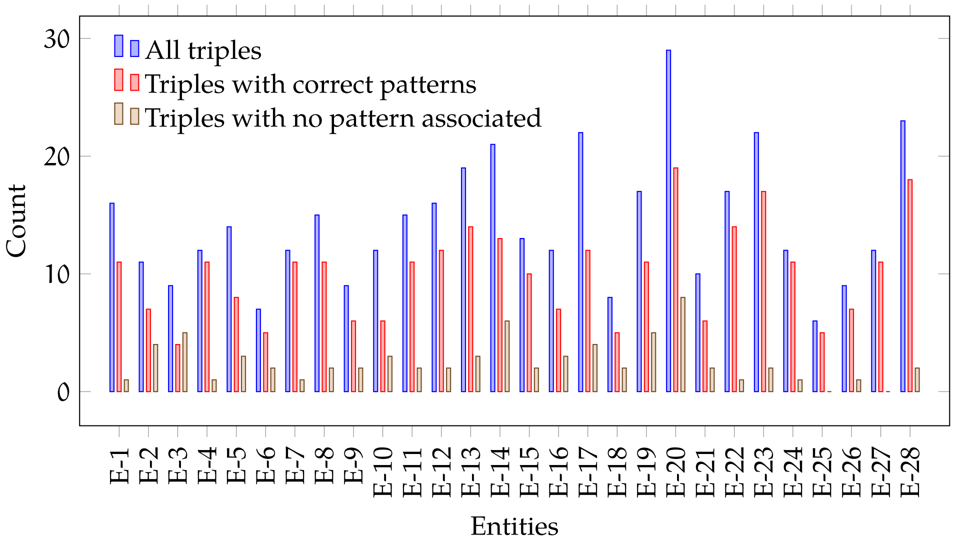

Figure 8. Lexicalization pattern evaluation for linguistic accuracy. Entities are denoted as E-1 to E-28.

6.1.2. Human evaluation with ratings

We hired 5 postgraduate students to evaluate a 40 randomly selected lexicalization patterns from the whole pattern pool which contained 400 lexicalization patterns generated for the QALD-2 test dataset. The participants were briefed with the task by providing them with three examples showing how each lexicalization pattern should be ranked according to the criteria provided. The inter-annotator agreement measured in Cronbach Alpha resulted with values 0.866 and 0.807 for readability and accuracy respectively.

Table 9 shows the results of this evaluation phase where lexicalization patterns are classified into the weighted average of rating values. In addition, a shallow analysis revealed a possible existence of a correlation between the readability and accuracy rating values. To further study this two tailed Spearman correlation analysis was performed which resulted in 0.828 correlation coefficient (p<0.001). This strong positive correlation reveals that accuracy of a lexicalization patterns and its readability are closely related.

6.1.3. Comparison with Lemon model

Among the three related works described in Section 4, the Lemon model (Walter et al., 2013) which has similar objective to ours, focussing on generating lexicaliza-

Weighted average of ratings	Accuracy	Readability
1.0 - 2.0	1	1
2.1 - 3.0	0	0
3.1 - 4.0	11	10
4.1 - 5.0	28	29

Table 9. Human evaluation results of the lexicalization patterns. The weighted average of ratings are categorized into four classes.

	RealText	Lemon
Accuracy (Full Automatic)	70.75%	37%
Accuracy (Semi automatic)	-	76%
DBpedia classes	25	30

Table 10. A comparison between RealText and Lemon. Note that semi automatic accuracy is not mentioned for our framework (RealText) as it is fully automatic.

tion patterns for individual triples rather than the whole graph. However, as Lemon model is not available for evaluation and has not released the evaluation dataset, this comparison limited to the results shown in (Walter et al., 2013).

According to the results shown in Table 10, it is clear that RealText has performed with a much higher accuracy than Lemon model in full automatic mode. Furthermore, human intervention between the process has boosted the Lemon model accuracy by 105.4%. Using human intervention in triple databases with millions of triples is not feasible as it may need excessive human resources. In this paper we showed a cost effective and scalable lexicalization framework. The framework is scalable in two ways. Firstly, the pattern mining modules connected through a loosely coupled architecture makes it possible to plug additional pattern mining modules. Secondly, utilizing OpenIE and universal typed dependencies make it possible to apply our framework in another language with minimum redesign.

6.2. Observations and discussions

The linguistic accuracy evaluation revealed that the framework was able to generate 283 accurate lexicalization patterns for 400 triples. This means that the framework achieved an accuracy level of 70.75%. The most similar system available for comparison, Lemon model, was able to achieve only 37% accuracy in its full automatic model. This shows that our approach has produced lexicalization patterns with much higher accuracy compared to the latest state-of-the-art model. This was further attested by the human evaluation where more than 70% of the lexicalization patterns are rated between values 4.1 and 5 for both accuracy and readability. In addition, more than 90% of the lexicalization patterns were rated above the average rating values for both accuracy and readability. This again confirms the quality of the lexicalization patterns achieved by our framework.

Our post analysis on human evaluation by calculating the correlation between readability and accuracy revealed that the two have a strong positive correlation. Similar evidence can be found in a previous research carried out by Reiter and Belz (2009) in a different domain.

7. Conclusion and future work

This paper presented a lexicalization framework for RDF triples. The framework centred on mining patterns to transform RDF triples using four pattern mining modules. The evaluation of the framework concentrated on both linguistic accuracy evaluation and human evaluation. Both evaluations showed that the framework can generate accurate and readable lexicalization patterns and the results are far better compared to the most similar existing lexicalization module, Lemon model.

In future we plan to expand the framework to other Linked Data resources and well to show the scalability of the framework. In addition we will also be applying the framework in practical applications to assess the applicability of the designed framework. Much of the background work for this had already taken place. As the first application we have planned to integrate a biography generator which selects triples from DBpedia and employ the lexicalization framework to generate a textual biography.

Acknowledgements

The work reported in this paper is part of the RealText project funded by Auckland University of Technology.

Bibliography

Auer, S, C Bizer, G Kobilarov, and J Lehmann. Dbpedia: A nucleus for a web of open data. In *6th international The semantic web and 2nd Asian conference on Asian semantic web conference*, pages 722–735, Busan, Korea, 2007. Springer-Verlag. URL http://link.springer.com/chapter/10.1007/978-3-540-76298-0{_}52.

Bizer, C, J Lehmann, and G Kobilarov. DBpedia-A crystallization point for the Web of Data. *Web Semantics: science …*, 2009. URL http://www.sciencedirect.com/science/article/pii/S1570826809000225.

Busemann, Stephan. Ten Years After : An Update on TG/2 (and Friends). *Proceedings 10th European Workshop on Natural Language Generation*, 2, 2005.

de Marneffe, Marie-Catherine, Timothy Dozat, Natalia Silveira, Katri Haverinen, Filip Ginter, Joakim Nivre, and Christopher D Manning. Universal Stanford Dependencies: A cross-linguistic typology. In *9th International Conference on Language Resources and Evaluation (LREC'14)*, pages 4585–4592, 2014. ISBN 978-2-9517408-8-4. URL papers3://publication/uuid/D4B7BB39-4FFB-4AA6-B21E-701A91F27739.

Del Corro, Luciano and Rainer Gemulla. ClausIE: clause-based open information extraction. pages 355–366, may 2013. URL http://dl.acm.org/citation.cfm?id=2488388.2488420.

Duma, Daniel and Ewan Klein. Generating Natural Language from Linked Data: Unsupervised template extraction. In *10th International Conference on Computational Semantics (IWCS 2013)*, Potsdam, 2013. Association for Computational Linguistics.

Ell, Basil and Andreas Harth. A language-independent method for the extraction of RDF verbalization templates. In *8th International Natural Language Generation Conference*, Philadelphia, 2014. Association for Computational Linguistics.

Etzioni, Oren, Michele Banko, Stephen Soderland, and Daniel S. Weld. Open information extraction from the web. *Communications of the ACM*, 51(12):68–74, dec 2008. ISSN 00010782. doi: 10.1145/1409360.1409378. URL http://dl.acm.org/ft{_}gateway.cfm?id=1409378{&}type=html.

Fader, Anthony, Stephen Soderland, and Oren Etzioni. Identifying relations for open information extraction. In *Empirical methods in Natural Language Processing*, pages 1535–1545, 2011. ISBN 978-1-937284-11-4. doi: 10.1234/12345678. URL http://dl.acm.org/citation.cfm?id=2145432.2145596$\delimiter"056E30F$nhttp://dl.acm.org/citation.cfm?id=2145596.

Kipper, Karin, Anna Korhonen, Neville Ryant, and Martha Palmer. A large-scale classification of English verbs. *Language Resources and Evaluation*, 42(1):21–40, 2008. ISSN 1574020X. doi: 10.1007/s10579-007-9048-2.

Kohlschütter, Christian, Peter Fankhauser, and Wolfgang Nejdl. Boilerplate Detection using Shallow Text Features. In *ACM International Conference on Web Search and Data Mining*, pages 441–450, 2010. ISBN 9781605588896. doi: 10.1145/1718487.1718542. URL http://portal.acm.org/citation.cfm?id=1718542.

Kövecses, Zoltán and Günter Radden. Metonymy: Developing a cognitive linguistic view. *Cognitive Linguistics (includes Cognitive Linguistic Bibliography)*, 9(1):37–78, 1998.

Lassila, Ora, Ralph R Swick, et al. Resource Description Framework (RDF) model and syntax specification. 1998.

Lehmann, Jens, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Soren Auer, and Christian Bizer. DBpedia - A Large-scale , Multilingual Knowledge Base Extracted from Wikipedia. *Semantic Web journal*, 5(1):1–29, 2014.

Manning, Christopher, John Bauer, Jenny Finkel, Steven J Bethard, and David McClosky. The Stanford CoreNLP Natural Language Processing Toolkit. In *The 52nd Annual Meeting of the Association for Computational Linguistics*, Baltimore, 2014. Association for Computational Linguistics.

Mausam, Michael Schmitz, Robert Bart, Stephen Soderland, and Oren Etzioni. Open language learning for information extraction. In *Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 523–534, Jeju Island, jul 2012. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=2390948.2391009.

Mendes, Pablo N., Max Jakob, and Christian Bizer. DBpedia for NLP: A Multilingual Cross-domain Knowledge Base. In *International Conference on Language Resources and Evaluation*, Istanbul, Turkey, 2012.

Moens, Marie Francine. *Information extraction: Algorithms and prospects in a retrieval context*, volume 21. 2006. ISBN 1402049870. doi: 10.1007/978-1-4020-4993-4.

Reiter, Ehud and Anja Belz. An Investigation into the Validity of Some Metrics for Automatically Evaluating Natural Language Generation Systems. *Computational Linguistics*, 35(4):529–558, dec 2009. ISSN 0891-2017. doi: 10.1162/coli.2009.35.4.35405. URL http://dl.acm.org/citation.cfm?id=1667988.1667994.

Reiter, Ehud and Robert Dale. *Building Natural Language Generation Systems*. Cambridge University Press, Cambridge, United Kingdom, jan 2000. ISBN 9780511551154. URL http://www.cambridge.org/us/academic/subjects/languages-linguistics/computational-linguistics/building-natural-language-generation-systems.

Schäfer, Florian. Naturally atomic er-nominalizations. *Recherches linguistiques de Vincennes*, 40 (1):27–42, 2011. ISSN 0986-6124. doi: 10.4000/rlv.1303. URL http://rlv.revues.org/351.

Stribling, Jeremy, Max Krohn, and Dan Aguayo. SciGen, 2005. URL https://pdos.csail.mit.edu/archive/scigen/.

Unger, Christina. Question Answering over Linked Data: QALD-1 Open Challenge. Technical report, Bielefeld University, Bielefeld, 2011.

Walter, Sebastian, Christina Unger, and Philipp Cimiano. A Corpus-Based Approach for the Induction of Ontology Lexica. In *18th International Conference on Applications of Natural Language to Information Systems*, pages 102–113, Salford, 2013. Springer-Verlag.

Address for correspondence:
Rivindu Perera
rivindu.perera@aut.ac.nz
Software Engineering Research Laboratory (WT-704),
School of Engineering, Computer and Mathematical Sciences,
Auckland University of Technology,
Private Bag 92006,
Auckland 1142, New Zealand