A Multi-Strategy Approach for Lexicalizing Linked Open Data

Rivindu Perera and Parma Nand

School of Computer and Mathematical Sciences,
Auckland University of Technology
Auckland 1010, New Zealand
{rivindu.perera,parma.nand}@aut.ac.nz

Abstract. This paper aims at exploiting Linked Data for generating natural text, often referred to as lexicalization. We propose a framework that can generate patterns which can be used to lexicalize Linked Data triples. Linked Data is structured knowledge organized in the form of triples consisting of a subject, a predicate and an object. We use DBpedia as the Linked Data source which is not only free but is currently the fastest growing data source organized as Linked Data. The proposed framework utilizes the Open Information Extraction (OpenIE) to extract relations from natural text and these relations are then aligned with triples to identify lexicalization patterns. We also exploit lexical semantic resources which encode knowledge on lexical, semantic and syntactic information about entities. Our framework uses VerbNet and WordNet as semantic resources. The extracted patterns are ranked and categorized based on the DBpedia ontology class hierarchy. The pattern collection is then sorted based on the score assigned and stored in an index embedded database for use in the framework as well as for future lexical resource. The framework was evaluated for syntactic accuracy and validity by measuring the Mean Reciprocal Rank (MRR) of the first correct pattern. The results indicated that framework can achieve 70.36% accuracy and a MRR value of 0.72 for five DBpedia ontology classes generating 101 accurate lexicalization patterns.

Keywords: Lexicalization, Linked Data, DBpedia, Natural Language Generation

1 Introduction

The concept of Linked Data introduces the process of publishing structured data which can be interlinked. This structured data is represented in the form of triples, a data structure composed of a subject, a predicate, and an object (e.g., ⟨Brooklyn Bridge, location, New York City⟩). A Linked Data resource contains millions of such triples representing diverse and generic knowledge.

Recently, there has been a growing need to employ knowledge encoded in Linked Data in Natural Language Processing (NLP) applications [1]. However, Linked Data must be represented in natural text to support NLP applications.

The process of creating Linked Data does not support to associate such information because the creation process is a template based Information Extraction (IE).

This paper explores the appropriateness of employing lexicalization to convert Linked Data triples back to the natural text form. Lexicalization is the process of converting an abstract representation into natural text. This is a widely studies area in Natural Language Generation (NLG), but early approaches cannot be used with Linked Data because of the rapidly growing nature and domain diversity of Linked Data resources. The proposed framework is a pipeline of processes to generate patterns which can be used to lexicalize a given triple into its natural text form. The patterns are evaluated based on two factors: syntactic correctness and re-usability. The syntactic correctness for a lexicalization patterns is not only the adherence to specific syntactic representation, but also the coherence of the pattern (no unimportant additional text is included in the pattern). The re-usability factor measures whether the pattern can be generalized to a range of triples with same knowledge.

A lexicalization pattern is defined in this context as a generic pattern that can be used to convert a Linked Data triple into natural language sentence. The resulting sentence will essentially contain the subject and the object of the triple being considered. In some cases it is also possible to include the predicate of a triple in a sentence to represent the triple in a sentence. Table 1 shows three example triples and expected patterns to lexicalize them. However, there can be multiple patterns to lexicalize a given triple, but among them only one will be the most suitable pattern that can generate a coherent and syntactically correct sentence.

Table 1. Example lexicalization patterns for three triples. The symbol `s?` and `o?` represent the subject and the object of the triple respectively

Triple			Lexicalization pattern
Subject (s)	Predicate (p)	Object (o)	
Steve Jobs	birth date	1955-02-24	*s?* was born on *o?*
Steve Jobs	birth place	San Francisco	*s?* was born in *o?*
Berlin	country	Germany	*s?* is situated in *o?*

We hypothesise that employing Information Extraction (IE) on sentence collection related to Linked Data resources together with lexical semantic resources can generate syntactically correct lexicalization patterns. Further, such a model can cover most of the patterns required to lexicalize a triple. The proposed model uses Open IE based relation extraction model to extract relations present in text which are then converted into lexicalization patterns. In parallel, the framework mines lexicalization patterns using verb frames utilizing two lexical semantic resources: VerbNet [2] and WordNet [3]. The identified patterns are then ranked and categorized based on the ontology class hierarchy specified by DBpedia. We

have released the version 1.0 of this pattern database which can be found in project web site[1] [4, 5].

The remainder of the paper is structured as follows. In Section 2 we describe the proposed framework in detail. Section 2 presents the experiments used to validate the framework. Section 4 compares the proposed model with other similar works. Section 5 concludes the paper with an outlook on future work.

2 RealText$_{lex}$ framework

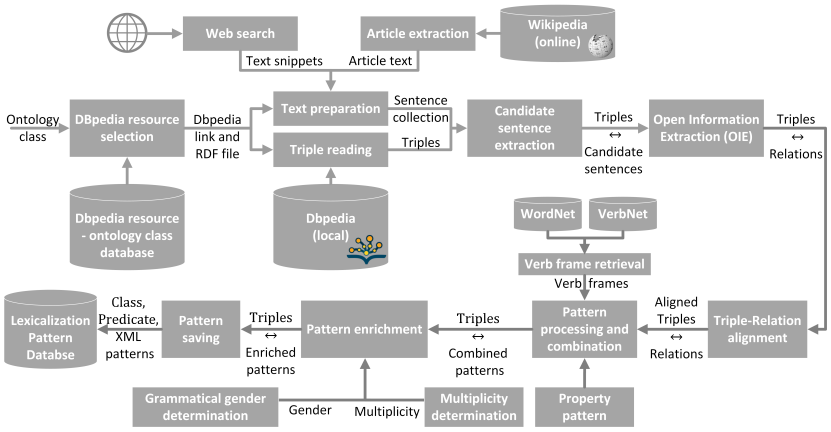

Fig. 1. Schematic representation of the complete framework

Fig. 1 depicts the complete process of generating lexicalization patterns in the proposed framework. The process starts with a given DBpedia ontology class (e.g., person, organization, etc.) and then selects two DBpedia resources for the entity (e.g., person ⇒ Steve Jobs, Bill Gates). In next step, the related Wikipedia text for the DBpedia resource together with text from other websites related to the DBpedia resource are extracted and prepares the text by applying co-reference resolution. The framework also reads the triples related to the DBpedia resource in parallel. Using both text collection and triples, the framework then extracts sentences which contain the triple elements. The selected sentences are then subjected to an Open Information Extraction (Open IE) based relation extraction. The output of this process is triples and relations related to each triple.

The triples and related relations extracted in aforementioned process are then used to determine lexicalization patterns. First, each relation is aligned with the related triple and relation patterns are extracted. At the same time the model

[1] `http://staff.elena.aut.ac.nz/Parma-Nand/projects/realtext.html`

retrieves the verb frames from VerbNet [2] and WordNet [3]. Both verb frames and relation patterns are then processed to identify basic patterns. The leftover triples without a pattern are associated with a predetermined pattern based on predicate of the triple. Next, a combined list of patterns are sent to the pattern enrichment process. This process adds additional features to the pattern such as multiplicity and grammatical gender. Finally, based on the enriched patterns, Extensible Markup Language (XML) patterns are built and stored in an indexed database.

Following sections describe the individual components of the framework in detail.

2.1 DBpedia resource selection

The function of this unit is to select two Dbpedia resources for the given ontology class. For an example, if the input is given as *Person* ontology class then the model should select two Dbpedia resources such as *http://dbpedia.org/resource/Walt_Disney* and *http://dbpedia.org/resource/Bill_Gates*.

Furthermore, to facilitate latter steps, this unit must also output RDF file names of these resources (e.g., *Walt Disney.rdf* and *Bill Gates.rdf*). This can be accomplished by executing a query in DBpedia online SPARQL endpoint[2]. However, executing SPARQL queries is a time consuming task and furthermore, if the endpoint is not functioning the whole lexicalization process is interrupted. To mitigate this, we have created a database containing DBpedia resource link, ontology class of the resource, and RDF file name. This database has 9010273 records and released with our framework for public usage. Since DBpedia does not offer dump which contains separate RDF file for each resource, we have downloaded partial set of RDF files related to records present in the database. If the required RDF file is not present in the repository, the model automatically downloads the resource for use as well as stores it in the repository for possible future usage.

2.2 Text preparation

The objective of the text preparation unit is to output co-reference resolved set of sentences. To achieve this objective, it first retrieves text from web and Wikipedia and then performs co-reference resolution. The resulting text is split to sentence level and passed as a text collection to the candidate sentence extraction unit. Following sections describe the process in detail.

Text retrieval Text related to triples are retrieved from web as well as Wikipedia. Since, DBpedia is based on Wikipedia, the Wikipedia text contains natural language sentences corresponding to the DBpedia triples. However, this assumption is not always true. For example, in Wikipedia birth date of a person is mentioned in the format of *person name (birth date)* (e.g., "Walt Disney (December

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

5, 1901 - December 15, 1966) was an American business magnate...”). Such sentences cannot be used to identify a pattern to lexicalize the birth date of a person, because property birth date is not represented in natural text. Due to this we use text snippet retrieved from other web sites in conjunction with the Wikipedia text. For example, a sentence like “Walt Disney was born on December 5, 1901,...” appeared in *biography.com* has a natural language representation for the birth date property. Therefore, text retrieval is accomplished using both text extracted from Wikipedia text and snippets of text acquired through a web search.

Co-reference resolution The retrieved text in Section 2.2 can contain co-references to entities. For an example, consider following sentences extracted from Wikipedia page for *Walt Disney*.

– “*Walt Disney* (December 5, 1901 - December 15, 1966) was an American business magnate. *He* left behind a vast legacy, including numerous animated shorts and feature films produced during his lifetime. *Disney* also won seven Emmy Awards and gave his name to the Disneyland.”

In above text, *He* and *Disney* both refer to the entity *Walt Disney*. Existence of such mentions which does not reflect the original entity when taken as individual sentences, can have a negative effect when finding sentences to extract lexicalization patterns. Therefore, as a preprocessing step all such mentions are replaced by actual entity that it referenced. The framework uses Stanford CoreNLP [6] to identify co-reference mentions and replaces them with the actual entity. The resulting co-reference resolved sentence will look like follows:

– “*Walt Disney* (December 5, 1901 - December 15, 1966) was an American business magnate. *Walt Disney* left behind a vast legacy, including numerous animated shorts and feature films produced during his lifetime. *Walt Disney* also won seven Emmy Awards and gave his name to the Disneyland.”

The resulting co-reference resolved text is then passed to the sentence splitter.

Sentence splitting The sentence splitting is responsible for splitting the co-reference resolved text into sentences. This is accomplished using Stanford CoreNLP tokenizer [6]. The tokenizer is based on Penn Tree Bank guidelines and uses deterministic process to split sentences using set of heuristics.

2.3 Triple reading

Parallel to text preparation, the framework retrieves triples related to the DBpedia resources selected in Section 2.1. These triples are read from a local repository of DBpedia RDF files. The repository currently contains 163336 RDF files and can be automatically download from DBpedia on demand.

Currently, DBpedia contains two types of triples, DBpedia properties and DBpedia ontology mapped properties. The DBpedia properties are those which are extracted using the DBpedia Information Extraction framework. Recently,

the DBpedia team added an ontology property list to organize DBpedia properties by mapping them with human involvement. However, the mapping process is still in progress and many DBpedia properties still remain unmapped. Therefore, the proposed framework used DBpedia properties to read triples employing Jena RDF parser[3].

Fig. 2 depicts a sample set of triples that can be retrieved by the triple reader from DBpedia RDF file for the resource *Walt Disney*.

- ⟨Walt Disney, birth date, 1901-12-05⟩
- ⟨Walt Disney, birth place, Hermosa⟩

Fig. 2. Sample set of triples retrieved from triple reader

2.4 Candidate sentence extraction

The objective of candidate sentence extractor is to identify potential sentences that can lexicalize a given triple. The input is taken as a collection of co-reference resolved sentences (from text preparation unit described in Section 2.2) and a set of triples (from triple reader described in Section 2.3). This unit firstly verbalizes the triples using a set of rules. Following list shows the set of rules used to verbalize triples.

- Dates are converted to normal date format. (e.g., 1901-12-05 ⇒ December 05 1901)
- Object values pointing to another DBpedia resource are replaced with actual resource names. (e.g., dbpedia:Chicago ⇒ Chicago)

Then each sentence is analyzed to check either complete subject (s), the object (o) or the predicate (p) are mentioned in the sentence (S). The only exception is that the object is analysed as a complete phrase as well as set of terms (t_o). For example, the object value like "Walt Disney Company" will be split to three terms (e.g, Walt Disney Company ⇒ Walt, Disney, Company) and will check whether a subset of the terms appears in the sentence. This sentence analysis uses a set of cases which assigns a score to each sentence based on presence of a triple. The scores were determined based on preliminary experiments. The sentences which do not belong to any of the cases are not considered as candidate sentences. The cases with the respective score assigned

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

by the candidate sentence extractor are shown in (1) below:

$$sc(T,S) = \begin{cases} 1, & \texttt{if } (\text{s}\in\text{S}) \ \&\ (\text{p}\in\text{S}) \ \&\ (\text{o}\in\text{S}) \\ 2/3, & \texttt{if } (\text{s}\in\text{S}) \ \&\ (\text{o}\in\text{S}) \\ 2/3, & \texttt{if } (\text{s}\in\text{S}) \ \&\ (\text{p}\in\text{S}) \\ 2/3, & \texttt{if } (\text{p}\in\text{S}) \ \&\ (\text{o}\in\text{S}) \\ 1/3, & \texttt{if } (\text{o}\in\text{S}) \\ 1/3 \times cr_{(t_o,S)}, & \texttt{if } (\text{cr}_{(t_o,S)} > 0.5) \end{cases} \tag{1}$$

where, $sc(T,S)$ represents candidate sentence score for triple (T) and sentence represented as set of terms (S). The ratio of terms contained in the sentence ($\text{c}_{(t_o,S)}$) is calculated as:

$$cr_{(t_o,S)} = \frac{\#(t_o \cap S)}{\#t_o} \tag{2}$$

where, t_o represents set of terms appearing in the object.

The output of this module is a collection of triples and sentences where each sentence is associated with one or more triple. Example scenario for triples taken from *Walt Disney* is shown in Fig. 3.

- ⟨Walt Disney, birth date, 1901-12-05⟩
  - `sentence:` Walt Disney was born on December 5, 1901.
    `score:0.667`
- ⟨Walt Disney, birth place, Hermosa⟩
  - `sentence:` Walt Disney was born in Hermosa.
    `score:0.667`

Fig. 3. Example output from candidate sentence extraction

2.5 Open Information Extraction

Once the candidate sentences are selected for each triple, we then extract relations from these candidate sentences in order to identify lexicalization patterns.

Extracting relations is a well known and heavily researched area in the IE domain. The approaches to relation extraction can be broadly divided into two categories: traditional closed IE based models and Open IE based models. The closed IE based models rely on rule based methods [7], kernel methods [8], and sequence labelling methods [9]. The burden associated with hand-crafted rules, the need for hand-tagged data for training, and domain adaptability are three key drawbacks associated with these traditional relation extraction models.

In recent times a new concept is arisen to address the drawbacks. The Open IE [10] essentially focuses on domain independent relation extraction and predominantly targets the web as a corpus for deriving the relations.

The framework proposed in this paper uses textual content extracted from the web which works with a diverse set of domains. Specifically, the framework uses Ollie Open IE system [11] for relation extraction. Each sentence from the web associated with a triple is analyzed for relations and all possible relations are determined. This module then associates each relation with the triple and outputs a triple-relations collection. A relation is composed of first argument (arg1), relation (rel), and second argument (arg2). The module also assigns a score to each relation which is defined as the average of confidence score given by Ollie and a score assigned for the specific sentence in Section 2.4.

Relations extracted for the example scenario shown in Section 2.4 are shown in Fig. 4.

- ⟨Walt Disney, birth date, 1901-12-05⟩
  - `arg1:` Walt Disney, `rel:` was born on, `arg2:` December 5, 1901 `score:0.934`
- ⟨Walt Disney, birth place, Hermosa⟩
  - `arg1:` Walt Disney, `rel:` was born in, `arg2:` Hermosa `score:0.785`

Fig. 4. Example output from Open Information Extraction module

2.6 Triple-relation alignment

The objective of this module is to determine how well each relation (arg1, rel, arg2) is aligned with the triple (s, p, o) and to generate a general pattern based on the alignment. This is accomplished by aligning verbalized triple (using triple verbalization rules described in Section 2.4) with the relation. Each relation is again scored in this step by assigning the final score which is average of score assigned by Open IE (in Section 2.5) module and score determined by the level of alignment. The alignment score is calculated as in (3):

$$sc(T, R_a) = \begin{cases} 1, & \texttt{if } (\text{s}\in \text{arg1} \cup \text{arg2}) \ \& \ (\text{p}\in\text{rel}) \ \& \ (\text{o}\in \text{arg1} \cup \text{arg2}) \\ 2/3, & \texttt{if } (\text{s}\in \text{arg1} \cup \text{arg2}) \ \& \ (\text{p}\in\text{rel}) \\ 2/3, & \texttt{if } (\text{s}\in \text{arg1} \cup \text{arg2}) \ \& \ (\text{o}\in \text{arg1} \cup \text{arg2}) \\ 2/3, & \texttt{if } (\text{p}\in\text{rel}) \ \& \ (\text{o}\in \text{arg1} \cup \text{arg2}) \\ 1/3, & \texttt{if } (\text{o}\in\text{arg2}) \\ 1/3 \times cr_{(t_o, arg2)}, & \texttt{if } (\text{cr}_{(t_o, arg2)} > 0.5) \end{cases} \tag{3}$$

where, $sc(T, R_a)$ represents triple-relation alignment score for triple (T) and aligned relation (R_a). The relation is represented in three components ($arg1, rel, arg2$).

The ratio of object terms contained in the second argument of the aligned relation ($\mathrm{cr}_{(t_o,arg2)}$) is calculated as:

$$cr_{(t_o,arg2)} = \frac{\#(t_o \cap arg2)}{\#t_o} \tag{4}$$

This module outputs a collection of triples with aligned relations including the associated score determined by the level of alignment. The relations that are not aligned (based on the cases in (3)) are not included in the collection.

2.7 Pattern processing and combination

This module generates patterns from aligned relations in Section 2.6. In addition to these patterns, verb frame based patterns are also determined and added to the pattern list. Any other pattern that is not associated with these patterns are assigned a generic pattern based on the predicate. The following sections describe these processes in detail.

Relation based patterns Based on the aligned relations and triples, a string based pattern is generated. These string based patterns can get two forms as shown in Fig. 5 for two sample scenarios. The subject and object are denoted by symbols *s?* and *o?* respectively.

- ⟨Walt Disney, birth date, 1901-12-05⟩
  - `arg1:` Walt Disney, `rel:` was born on, `arg2:` December 5, 1901
    `pattern:` s? was born on o?
- ⟨Walt Disney, designer, Mickey Mouse⟩
  - `arg1:` Mickey Mouse, `rel:` is designed by, `arg2:` Walt Disney
    `pattern:` o? is designed by s?

Fig. 5. Basic patterns generated for two sample triples. *s?* and *o?* represent subject and object respectively.

Verb frame based patterns Verb frame based pattern generation process attempts to find patterns based on semantic frames. A semantic frame of a verb is a grammatical structure that shows the usage of the verb. For instance, the verb *create* has a semantic frame *Noun phrase, Verb, Noun phrase* which can be realized in an example as "Walt Disney created Mickey Mouse Clubhouse".

The framework utilizes two lexical semantic resources, VerbNet and WordNet to mine patterns. We have created a slightly moderated version of VerbNet which has all verbs and their associated frames merged into one indexed database and can be found in the RealText$_{lex}$ project website. Currently, the framework

generate only one type of pattern (`s? Verb o?`), if the predicate is a verb and if that verb has the frame {*Noun phrase, Verb, Noun phrase*} in either VerbNet or WordNet.

Property based patterns The predicates which cannot associate with a pattern in the above processes described in Section 2.7 and Section 2.7 are properties belonging to the DBpedia resources selected in Section 2.1. The left over predicates are assigned a generic pattern (`s? has` ⟨`predicate`⟩ `of o?`) based on the specific predicate. For example, a predicate like "`eye colour`" will be assigned with a pattern like "`s? has eye colour of o?`".

The patterns acquired from aforementioned modules are then passed to the enrichment module to add related information.

2.8 Pattern enrichment

Pattern enrichment adds two types of additional information; grammatical gender related to the pattern and multiplicity level associated with the determined pattern. When searching a pattern in the lexicalization pattern database, these additional information is also mined in the lexicalization patterns for a given predicate of an ontology class.

Grammatical gender determination The lexicalization patterns can be accurately reused later only if the grammatical gender is recorded with the pattern. For example, consider triple, ⟨`Walt Disney, spouse, Lillian Disney`⟩ and lexicalization pattern, "`s? is the husband of o?`". This pattern cannot be reused to lexicalize the triple ⟨`Lillian Disney, spouse, Walt Disney`⟩, because the grammatical gender of the subject is now different, even though the property (spouse) is same in both scenarios. The framework uses three types of grammatical gender types (male, female, neutral) based on the triple subject being considered in the lexicalization. The gender of the subject is determined by DBpedia grammatical gender dataset [1].

Multiplicity determination Some triples in DBpedia has same subject and predicate values for different object values. Fig. 6 shows set of triples taken from DBpedia resource for Nile river and for East river which have the same predicate (country).

According to Fig 6, Nile River has three countries listed as it does not belong to one country, but flows through these countries. However, East River belongs only to United States. The lexicalization patterns generated for these two scenarios will also be different and cannot be shared. For example, lexicalization pattern for Nile river will in the form of "`s? flows through o?`" and for East River it will be like "`s? is in o?`". To address this variation, our framework checks whether there are multiple object values for the same subject and predicate, then it adds the appropriate property value (multiple/single) to the pattern.

- Nile River
  - ⟨Nile River, country, Burundi⟩
  - ⟨Nile River, country, Egypt⟩
  - ⟨Nile River, country, Kenya⟩
- East River
  - ⟨East River, country, United States⟩

Fig. 6. Object Multiplicity scenario for predicate type *country*

These enriched patterns with grammatical gender and multiplicity are then passed to the pattern saving module to store these in a database.

2.9 Pattern saving and lexicalization pattern database

The pattern saving module first converts the patterns to an XML format and then it saves these to the lexicalization pattern database with the final score calculated as described in Section 2.6. A sample XML pattern generated using triple ⟨Walt Disney, birth date, 1901-12-05⟩ is shown in Fig. 7.

```
<lexpat>
<pat> s? was born on o? </pat>
<gender> male </gender>
<multiplicity> false </multiplicity>
</lextpat>
```

Fig. 7. Sample XML pattern

The XML patterns are written to the lexicalization database with three other attributes, ontology class, predicate and the final score. Sample set of records are shown in Table 2.

Table 2. Sample set of entries from lexicalization pattern database

Ontology class	Predicate	Pattern	Score
Person	birthDate	<lexpat>...</lexpat>	0.984
Person	birthPlace	<lexpat>...</lexpat>	0.791
River	country	<lexpat>...</lexpat>	0.454

3 Experimental framework

The experimental framework analysed both lexicalization patterns as well as the underlying process to generate them. We used a subset of randomly selected DBpedia ontology classes to evaluate the accuracy of the framework. The following sections describe the results and evaluation details for the selected ontology classes.

3.1 DBpedia ontology class selection

Table 3 shows the ontology classes selected for evaluation of the framework including various other related statistics. For the experiment, 5 ontology classes were randomly selected which gave us 254 triples in total. A predetermined rule set was used to filter out invalid triples such as "Wikipedia URL", "WordNet type", and "photo collection URL" which cannot be lexicalized as natural language representations. The further removed duplicate predicates, leaving a total of 132 predicates that were used in the experiment.

Table 3. Results of the ontology class selection

Ontology class	Valid triples	Invalid triples	Unique predicates to lexicalize
Bridge	61	10	39
Actor	60	12	37
Publisher	15	5	9
River	49	9	35
Radio Host	29	4	12

3.2 Experimental settings and results

This section presents results collected from each module contributing towards pattern mining as well as the overall evaluation results.

Table 4 shows the results for the individual modules in framework. The table gives the numbers for the sentences processed, the candidate sentences and relations extracted from the candidate sentences. Table 5 shows the summary if the breakdown if the results for pattern extraction. The last 5 columns of the table also shows the results for the pattern enrichment modules. To get a clear idea about on the accuracy of the framework, we checked how many syntactically correct lexicalization patterns appear as the highest ranked pattern for the given predicate. In this context syntactic correctness was considered as being both grammatically accurate and coherent. The results of this evaluation is shown in Fig. 8 for each of the ontology classes.

Since, the framework ranks lexicalization patterns using a scoring system, we considered it as a method that provides a set of possible outputs. We decided to

Table 4. Results of the sentence and relations processed

Ontology class	All sentences	Candidate sentences	Extracted relations
Bridge	202	146	445
Actor	129	126	422
Publisher	212	25	99
River	88	69	158
Radio Host	38	27	87

Table 5. Results of the pattern extraction module

Ontology class	Relational patterns	Verb frame patterns	Property based patterns	Pattern enrichment				
				Multiplicity		Grammatical gender		
				Multiple	Single	Male	Female	Neutral
Bridge	272	8	9	163	126	0	0	289
Actor	422	0	16	369	69	400	22	16
Publisher	39	1	4	32	12	0	0	44
River	157	2	10	158	11	0	0	169
Radio Host	30	1	1	14	18	0	0	32

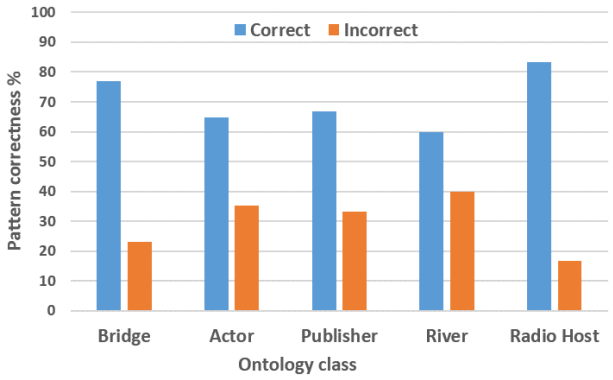

Fig. 8. Analysis of syntactic correctness of the extracted patterns

get a statistical measurement incorporating Mean Reciprocal Rank (MRR) as shown below to compute the rank of the first correct pattern of each predicate in each ontology class.

$$MRR = \frac{1}{|P|} \sum_{i=1}^{|P|} \frac{1}{rank_i} \tag{5}$$

where P and $rank_i$ represent predicates and the rank of the correct lexicalization for the i^{th} predicate respectively. Table 6 depicts the MRR results for the 5 ontology classes being considered.

Table 6. Mean Reciprocal Rank analysis for ranked lexicalization patterns

Ontology class	**MRR**
Bridge	0.77
Actor	0.69
Publisher	0.72
River	0.61
Radio Host	0.83

Table 7 shows a statistical summary of proposed approach.

Table 7. Statistics of evaluation of proposed approach

	Candidate templates	**Lexicalizations**	**Accuracy**
Proposed approach	393	101	70.36%

3.3 Observations and discussions

The following observations can be made based on the results of the experiment. Firstly, the triple filtering step was effective in being able to filter out all the invalid triples as shown in Table 3. The filtering process was able to filter out 40 triples from total of 254. All invalid filtered out triples were identified as true negatives.

Apart from the *Publisher* ontology class (Table 4) more than 50% of the sentences were able to be selected as candidates. This was mainly due to the variation of the sentence patterns appeared in the two DBpedia resources which were selected by resource selection module (Section 2.1). The two resources selected (*Marvel Comics* and *Elsevier*) had sentences which were hard to align with triples. This caused a dramatic decrease in selection score thus eliminating 88% of the sentences.

The number of extracted relations in Table 4 shows that 1211 relations were able to be extracted from a 393 candidate sentences. This emphasizes an interesting point that on average 3 relations are associated with each candidate sentence. This is one of the significant benefit that our framework received by incorporating Open IE model which can extract large number of accurate relations compared to traditional closed IE relation extraction approaches.

Fig. 8 shows that our framework has achieved 70.36% average accuracy for 5 ontology classes where the lowest accuracy was reported as 60%. This evaluation does not take into account the rank of the correct lexicalization patterns and measures the number of correct patterns present in the extracted set of patterns. On the other hand, MRR based evaluation provides an detailed look at ranking of the first correct lexicalization. Average MRR value of 0.724 achieved for 5 ontology classes. The level of alignment between triples under consideration and the relations had a positive effect on the accuracy. For example, the ontology class *Radio Host* has achieved good accuracy and a MRR value (accuracy: 83.34%, MRR: 0.83) with a smallest set of relations because of the alignment between relations and triples.

The verb frame and property patterns have not significant impact on pattern extraction, however, they are still crucial for identifying patterns that cannot be represented as semantic relations.

Finally, based on the comparison in Table 7, it is clear that proposed approach in this paper has advanced the way of deriving lexicalizations by generating reasonable number of valid patterns and with a higher accuracy.

4 Related work

This section discusses the related works in which have used Linked Data for lexicalization and also compares our framework and concept with them.

Unsupervised template extraction model presented by Duma and Klein [12] is similar in goal to our framework. Duma and Klein [12] attempt to extract sentences from Wikipedia which have DBpedia triples mentioned and then extract templates based on a heuristic. This model differ from our framework in many ways. Firstly, it does not consider extracting more textual content to generate candidate sentences. Furthermore, Wikipedia data is taken in raw format without further processing.

Walter et al. [13] introduces the Lemon model which extracts lexicalizations for DBpedia using dependency patterns extracted from Wikipedia sentences. They argue that dependency parsing can generate patterns out of sentences to lexicalize a given triple. However, the strategy of traversing through a dependency parse is not explained well. In our initial experiment the use of dependency parsing for Wikipedia sentences showed that determining a pattern is hard to accomplish by sole usage of a dependency parse. The framework proposed in this paper introduces Open IE to address this issue. Furthermore, our framework is equipped with additional functions to enrich the sentence collection (e.g, coreference resolution). Gerber and Ngomo [14] discuss the string based model to extract patterns. This simple model limits it generating more coherent patterns for triples being under consideration. Furthermore, it is not possible to expect all different lexicalization patterns from text. In our model this is achieved by integrating two lexical semantic resources VerbNet and WordNet which in parallel contributing to the pattern extraction process.

Ell and Harth [15] present the language independent approach to extract RDF verbalization templates. The core of this model is the process of aligning a data graph with the extracted sentence. To extract patterns they utilize maximal sub-graph pattern extraction model. However, there are key differences in our framework compared to one introduced by Ell and Harth [15]. Our framework starts the process with a text preparation unit which provides a potential text to extract patterns. Furthermore, we have studied the effectiveness of Open IE in pattern extraction which is specifically targeting on extracting relational patterns using a web as the corpus.

5 Conclusion and future work

This paper presented a framework to generate lexicalization patterns for DBpedia triples using a pipeline of processes. The pipeline starts with ontology classes which is then used to mine patterns aligning triples with relations extracted from sentence collections from the web. Furthermore, two additional pattern identification modules, verb frame based and property based were used to add additional patterns. The framework also incorporates an enrichment process for the further alignment of the patterns to the human generated text. The framework generated patterns were human-evaluated and showed an accuracy of 70.36% and a MRR of 0.72 on test dataset.

In future, we aim to target on expanding the test collection to build a reasonable sized lexicalization pattern database for DBpedia. The results also showed that the verb frame based pattern mining can be further improved by incorporating additional lexical semantic resources. The individual composite modules can also be improved (especially the relation-triple alignment module) to generate high quality lexicalization patterns for DBpedia as well as to generalize the process for other Linked Data resources. This will allow us to introduce a generalizable lexicalization pattern generation framework for evolving Linked Data cloud.

References

1. Mendes, P.N., Jakob, M., Bizer, C.: DBpedia for NLP: A Multilingual Cross-domain Knowledge Base. In: International Conference on Language Resources and Evaluation, Istanbul, LREC (2012)
2. Kipper, K., Dang, H.T., Palmer, M.: Class-Based Construction of a Verb Lexicon. In: Seventeenth National Conference on Artificial Intelligence, Austin, AAAI Press (July 2000) 691–696
3. Miller, G.A.: WordNet: A Lexical Database for English. Communications of the ACM **38**(11) (1995) 39–41
4. Perera, R., Nand, P.: Real text-cs - corpus based domain independent content selection model. In: Tools with Artificial Intelligence (ICTAI), 2014 IEEE 26th International Conference on. (Nov 2014) 599–606
5. Perera, R., Nand, P.: The role of linked data in content selection. In: The 13th Pacific Rim International Conference on Artificial Intelligence. (Dec 2014) 573–586

6. Manning, C., Bauer, J., Finkel, J., Bethard, S.J., McClosky, D.: The Stanford CoreNLP Natural Language Processing Toolkit. In: The 52nd Annual Meeting of the Association for Computational Linguistics, ACL (2014)
7. Brin, S.: Extracting Patterns and Relations from the World Wide Web. In: International Workshop on The World Wide Web and Databases, Valencia, Springer-Verlag (March 1998) 172–183
8. Zelenko, D., Aone, C., Richardella, A.: Kernel methods for relation extraction. In: Empirical methods in natural language processing - EMNLP '02. Volume 10., Morristown, NJ, USA, Association for Computational Linguistics (July 2002) 71–78
9. Kambhatla, N.: Combining lexical, syntactic, and semantic features with maximum entropy models for extracting relations. In: ACL 2004 on Interactive poster and demonstration sessions, Morristown, ACL (July 2004)
10. Etzioni, O., Banko, M., Soderland, S., Weld, D.S.: Open information extraction from the web. Communications of the ACM **51**(12) (December 2008) 68
11. Mausam, Schmitz, M., Bart, R., Soderland, S., Etzioni, O.: Open language learning for information extraction. In: Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, Jeju Island, ACL (July 2012) 523–534
12. Duma, D., Klein, E.: Generating Natural Language from Linked Data: Unsupervised template extraction. In: 10th International Conference on Computational Semantics (IWCS 2013), Potsdam, ACL (2013)
13. Walter, S., Unger, C., Cimiano, P.: A Corpus-Based Approach for the Induction of Ontology Lexica. In: 18th International Conference on Applications of Natural Language to Information Systems, Salford, Springer-Verlag (2013) 102–113
14. Gerber, D., Ngomo, A.C.N.: Bootstrapping the Linked Data Web. In: The 10th International Semantic Web Conference, Bonn, Springer-Verlag (2011)
15. Ell, B., Harth, A.: A language-independent method for the extraction of RDF verbalization templates. In: 8th International Natural Language Generation Conference, Philadelphia, ACL (2014)