Project Report on

Content Selection from Semantic Web Data

Submitted by: Sabita Acharya

1. Introduction

In the past, there have been some work on extracting interesting contents from huge amount of documents [1]. On the other hand, several significant efforts have also been made in natural language generation from data available in semantic web formalisms [2]. However, not much research work has been oriented towards selecting contents from semantic web data. With an exponential increase in the amount of structured data available in the Linked Open Data Cloud, if we are able to automatically perform content selection from semantic web data, we can get access to extremely wide range of factual data, which can then be used to conduct various research works and experiments. With these prospects in mind, **Content Selection from Semantic Data Challenge** was proposed by Bouayad-Agha et al. [3] in 2012. Our project is based on this challenge. Our main objective can be stated as: **Given a set of RDF triples and text from Wikipedia, we want to identify triples that are reflected in the target text**.

2. Related works

2.1 Systems submitted for Content Selection Challenge

Since the Content Selection Challenge began recently at 2012, there were only 2 systems that were submitted in the first year [4, 5]. Kutlak et al. [4] used the internet as corpus of documents and used the Google search engine for counting the number of documents containing the properties. They assume that there is a correlation between the knowledge of the people and the number of hits returned by google and apply a heuristics that if many people affirm a particular thing, then it must be true. Their approach is completely different from what we have done because they have taken internet as a corpus of documents while we plan to use texts from Wikipedia as our corpus. Similarly, Venigalla et al. [5] firstly performed predicate clustering by removing the leaf from each predicate expression and then applied certain heuristic rules based on co-occurrences of predicates in the training data. We, on the other hand, extract semantic relations between the components of a sentence and then use machine learning techniques to remove incorrect predictions.

2.2 Other systems

Barzilay et al. [1] used content selection to generate summary from a database that contains all the events in a football game including the scores and activities of the players. Similarly, Cimiano et al. [6] developed and evaluated a natural language generation system that converts RDF data into natural language text based on an ontology and an associated ontology lexicon. These two approaches gave us a wider view of other works that have been done in content selection and natural language generation from semantic data respectively.

Walter et al. [7] tried to find various lexicalization for properties and classes. They take Wikipedia texts as domain corpus and triples are extracted from RDF dataset of DBpedia. For each property, they retrieve triples and extract all sentences that contain the subject and object of the selected triples. They then parse those sentences using Link Grammar Parser and generate a pattern by replacing the subject and object of the triple by some variables. Similarly, for each class, they extract its label and then perform synonym search using Wordnet. Most of our initial approach is inspired from their work. The only differences are that we only deal with properties while they deal with classes too. Also, they consider all the patterns for properties generation from above mentioned steps to be true while we try to apply machine learning techniques in order to remove incorrect patterns.

Suchanek et al.[8] have worked on extracting relations from web documents. They also generate patterns from statements by using similar approach as mentioned in [7]. Then they move a step ahead by trying to filter out incorrect patterns that have been extracted. Much of our work is based on their approach with only a few variations in the type of dependency parser used and the feature representation methods during filtering which will be described in more detail in the next section.

A high level comparison of several features of the above described papers (excluding [5] and [6] because they address a different task) have been shown in the table below:

Systems	Input data	Parser used	Filtering techniques
Kutlak et al. [4]	Content selection challenge dataset	-	Google API
Venigalla et al. [5]	Content selection challenge dataset	-	Clustering predicates, Use rules
Walter et al. [7]	DBpedia, Wikipedia	Malt dependency parser	-
Suchanek et al. [8]	Not mentioned	Link grammar parser	kNN, SNM

Table 1: Comparison of various features of approaches that are related to our work

3. Project details

3.1 Dataset

Firstly, we planned to extract highly frequent properties from DBpedia and the related texts from Wikipedia. But, since we did not have an already annotated dataset, it would be hard for us to evaluate the performance of our system. Then, we moved on to use the dataset provided by the organizers of the Content Selection from Semantic Web Data Challenge.

The dataset contained 645 folders named according to the person, each containing a text file with sentences and a turtle file with predicates. Our first work was to generate a .tsv file from the .ttl (turtle) files provided. Once that was done, we noticed that the .tsv file only contained the predicates and objects while the subject were missing. Hence we extracted the subject from the folder name and appended it along with the predicate and object, thereby forming triples. Once that was done, we extracted all the sentences and triples from the different folders and copied them over to two text files, one for the sentences and the other for the triples. From the values of the objects, it was evident that they needed to be preprocessed before they could be used because they contained large amount of unnecessary information like Freebase mid-id, several phrases and numbers etc. We performed preprocessing of the object values during predicate extraction process because objects related to different properties had different kinds of unwanted information and a single approach would not be applicable to all cases.

In addition to this, we observed that out of total 18307 triples, there are 613 distinct predicates. Out of these 613 predicates, only 11 were present in over 40 percent of the files and only 19 predicates were present in over 10 percent of the files. This means that a large number of predicates are present only in a few files. Conversely, nearly 40 percent of text files only contained one or two sentences, which compounds the sparsity problem. In order to deal with this problem of sparse data, we had to manually add more varying data so that the functioning of the system does not depend on the dataset.

Collecting a significant amount of dataset to work on was the most challenging part of the project and more than half of the effort put into the project was spent on this task.

3.2 Dependency Parser

Dependency parser takes in a sentence and shows us how different components of a sentence are related to each other. There are several kinds of parsers like Malt Dependency Parser, Link Grammar Parser, Stanford parser, etc. that can be used for parsing a sentence. For our purpose, we used Stanford parser because of familiarity and also because it gives us the index of the terms in the sentence, which is helpful while generating features for filtering.

Following is an example of the result returned by Stanford parser:

Sentence: This is an example of dependency grammar.

Dependencies: nsubj (example-4, This-1)

cop (example-4, is-2)
det(example-4, an-3)
root (ROOT-0, example-4)
amod(grammar-7, dependency-6)
prep_of(example-4, grammar-7)

3.3 Project steps

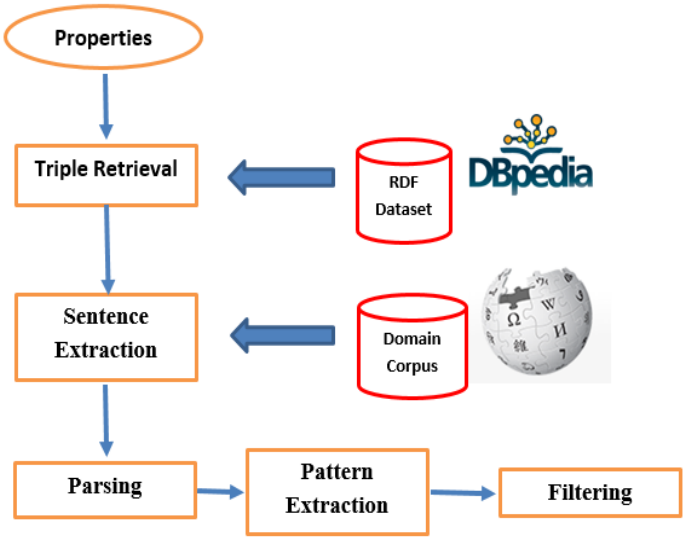

Fig 1: Brief overview of the various steps involved in the project

3.3.1. Triple retrieval

For each property, we extract all the triples where the property occurs. For example, for **nationality** property, all triples like <**Obama**> <**nationality**><**US**>, <**Christiano Ronaldo**> <**nationality**> <**Portugal**> will be extracted.

3.3.2. Sentence extraction

We extract all the sentences where the subject and the object that appear in the triples extracted above are present. For examples, sentences like **President Barack Obama was born in United States** are extracted in this step.

3.3.3. Parsing

All the sentences that were extracted from the above step are then passed through Stanford Dependency Parser. The parsed result helps in determining the relation between the components of the sentence including the subject and the object.

3.3.4. Pattern extraction

We then abstract over the subject and the object entities by replacing them with a variable. For the "nationality" property, we prepared a list of nations and their nationalities too because in most of the sentences, terms like "American", "Dutch" appeared instead of the country names.

For the "spouse" relation, identifying the correct representation of the subject and object in the sentences was hard. Since a person can be denoted by their full name, their first name only, their last name only or their nick names, we had a tough time in deciding what approach to use. We also had to do some manual correction replacement for "spouse" relation in those cases where our algorithm couldn't correctly identify the subject and object entities. We then use "XX" to represent subject and "YY" to represent object. Once this is done, we get sentences like: **XX was born in YY**.

3.3.5. Filtering

The patterns that are extracted in the above step may include correct sentences like **XX was born in YY** along with other incorrect sentences like **XX visited YY**. In order to remove such incorrect patterns, we will use **K-Nearest Neighbor algorithm**. But before applying the algorithm, we need to prepare a set of training and testing data and will also have to extract features upon which our algorithm will work. We begin by manually tagging the patterns that were extracted in the above step as positive or negative depending on whether they represent "nationality" relation or not.

3.3.5.1. Preparing training and testing data

Once we have extracted features from all the sentences, we perform 10 fold cross validation in order obtain non overlapping sets of training and testing data. From the data that has been collected for training purpose, we separate the positive and negative labelled data into two groups. So we finally have 3 sets of data. One is the test set, the other two are the positive training set and the negative training set. All the three sets of data are separately passed through the feature extraction module described below and are stored in three different data structures.

3.3.5.2. Feature extraction

We need features that are able to capture all the relations that exist between the subject and object and the other words in the sentence. Let us see how features are extracted by using an example:

Let **XX was born in YY##Pos** be a sentence we obtained after performing above steps.

- We pass the sentence through dependency parser, which gives us the relations between all the components of the sentence and also assigns part of speech to all the words.

Fig 2: Relations between all the component words.

Part of speech tagged sentence: [XX/NNP, was/VBD, born/VBN, in/IN, YY/NNP**]**

- From the results obtained from the above step, we generate features like:
  ({(nsubjpass-right-VBN) } { (prep_in-left-VBN)}) where, the components inside the first set of curly braces represents the relation between subject and the other words in the sentence and the components inside the second set of curly braces are obtained from the relation between object and the other words of the sentence.
  {(nsubjpass-right-VBN)} means that the **subject is related to a verb that falls on its right side through a relation "nsubj".**
  Since we also know the labels (positive or negative) of the sentence, we include that in our feature too. So finally, our feature looks like below:
  ({(nsubjpass-right-VBN)} {(prep_in-left-VBN)}{+1})

3.3.5.3 Applying kNN algorithm

Once the feature sets are ready, we extract one test feature and compare it with all the positive training features and negative training features. Let F1 be a feature from test set and F2 be a feature from the positive train set.

F1: ({(nsubjpass-right-VBN)} {(prep_in-left-VBN)})
F2: ({(dobj-left-NN)} {(advmod-right-RB)})

The components within the first set of curly braces of F1 show that the subject is related to a verb that is to its right by "nsubjpass" relation. Similarly, the object is related to a verb to its left by "prep_in" relation.Form the above features, we extract the following components:

C1: {(nsubjpass-right-VBN)} O1: {(prep_in-left-VBN)}
C2: {(dobj-left-NN)} O2:{(advmod-right-RB)}
Then, we find out the similarity between C1 and C2 and O1 andO2 by using the following formula:

$$sim(C_1, C_2) = \sum_{\substack{(con_1, dir_1, w_1) \in C_1 \\ (con_2, dir_2, w_2) \in C_2}} \frac{\alpha_1(con_1 \sim con_2) + \alpha_2(dir_1 \sim dir_2) + \alpha_3 sim(w_1, w_2)}{|C_1| \cdot |C_2|}$$

where con1, dir1,w1 are nsubjpass, right, VBN respectively from C1. Similarly, con2, dir2,w2 are dobj, left, NN from C2 respectively.

This gives us the similarity between the subjects and objects of F1 and F2 respectively. The final similarity value is the average of the similarity between subjects and objects.
sim(F1,F2)=(1/2)(sim(C1,C2)+sim(O1,O2))

We then evaluate the top ten similarity values between the test feature and positive training features and test feature and negative training features respectively. Then the one which gives the higher value has its class assigned to the test feature.

4. Results and Comparison

	Precision	**Recall**	**fscore**
Nationality	**81.26**	**76.36**	**78.14**
Spouse	71.49	87.14	77.98

Table 2: Results produced for two properties by the system

Since there are not many systems that have been submitted for the Content Selection Challenge, we could compare our result only with Venigalla et al. [5] and Kutlak et al. [4]. But since Kutlak et al. [4] have not reported their results, we referred only to Venigalla et al. [5] and found out that our system clearly outperforms the precision, recall and fscore obtained by their system by a wide margin .

5. Conclusion

This project gave us an opportunity to implement natural language techniques and evaluate their effectiveness in correctly selecting content from text. We adopted approaches suggested by authors of similar systems and found out that it provides significant results. For this project, we were not able to explore with many properties as we had wanted to because of the difficulty in extracting sufficient amount of expressive data. If proper dataset is available, this challenge could attract many more researchers to stride towards creating a symbiosis between natural language processing and semantic web data.

6. References

[1] R. Barzilay and M. Lapata, "Collective Content Selection for Concept-to-Text Generation," in *Proceedings of the Joint Human Language Technology and Empirical Methods in Natural Language Processing Conferences*, 2005.

[2] N. Bouayad-Agha, G. Casamayor, S. Mille, M. Rospocher, H. Saggion, L. Serafini, and L. Wanner, in *Proceedings of the 17th International Conference on Applications of Natural Language Processing to Information Systems*, 2012, pp. 216–221.

[3] N. Bouayad-Agha, G. Casamayor, L. Wanner and C. Mellish, "Content selection from semantic web data, " *in Proceedings of International Natural Language Generation Conference*, 2012, pp. 146–149.

[4] R. Kutlak, C. Mellish, K.V. Deemter, "Content Selection Challenge - University of Aberdeen Entry," in *Proceedings of the 14th European Workshop on Natural Language Generation*, 2013, pp. 208–209.

[5] H. Venigalla, B.D. Eugenio, "UIC-CSC: The Content Selection Challenge Entry from the University of Illinois at Chicago," in *Proceedings of the 14th European Workshop on Natural Language Generation*, 2013, pp. 210–211.

[6] P. Cimiano, J. Luker, D. Nagel and C. Unger, "Exploiting ontology lexica for generating natural language texts from rdf data," in *Proceedings of the 14th European Workshop on Natural Language Generation*, 2013, pp. 10–19.

[7] S. Walter, C. Unger, and P. Cimiano, "A corpus-based approach for the induction of ontology lexica," in *Proceedings Of the 18th International Conference on Applications of Natural Language to Information Systems*, 2013, pp. 102–113

[8] F.M. Suchanek, G. Ifrim, G. Weikum, "Combining linguistic and statistical analysis to extract relations from web documents," in *Proceedings of the 12th Association for Computing Machinery International conference on Knowledge discovery and data mining*, 2006.