Text Entailment in Semantic Web World

Shubham Gautam
Computer Science & Engineering,
IIT Bombay
shubhamg@cse.iitb.ac.in

Pushpak Bhattacharyya
Computer Science & Engineering,
IIT Bombay
pb@cse.iitb.ac.in

Abstract

Text Entailment is the concept of determining whether the message conveyed by one sentence is exactly captured by another sentence or not. There have been various approaches discovered to determine the entailment between a pair of sentences. We present a system using RDF (Resource Description Framework) approach that uses online free data available for recognizing textual entailment. There are situations where an entity name in one sentence can be different from the entity name of another sentence, but they may actually belong to each other. We develop an approach for identifying entailment between two sentences using named entity resolution with the help of RDF, verb entailment and coreference resolution. We present our results on the Recognizing Textual Entailment dataset (RTE-1 to RTE-3) and show that where our approach stands amongst the approaches already developed in this direction.

1 Introduction

In the last few years, there has been a great interest arising in the field of "Text Entailment". There have been many approaches that have been developed for recognizing the entailment between two sentences ranging from lexical, logic based (Natural Logic) to semantic based approaches. Each approach has its own unique features. There are some approaches which are based on logic such as: Natural Logic (MacCartney and Manning, 2007), logic based on lexical resources (Glickman and Dagan, 2005) *etc.* But, there are situations that may involve different names of a person in different sentences. It can be any relation also. In this paper, we present an approach for recognizing textual entailment between a pair of sentences using RDF which is used for *named entity resolution*, stemming for *verb entailment* and dependency matching for getting the final result. This approach mainly deals with the real world examples which may consists of different names of an entity and for which a wikipedia[1] page exists because RDF freebase is the inherent part of any wikipedia page.

Our first subtask introduces about the named entities which for a particular person (or organization) can vary for different sentences. So, the task is to identify these named entities and then extracting out the matching named entities. In our second subtask, the main focus is to identify the related verbs which can be used in place of another verb. Finally, we matched the dependencies of *text* and *hypothesis* to each other to reach towards a final conclusion. Let us consider an example:

T : *Barack Obama came to Delhi.*

H : *U.S. president visited India.*

In the above example, both 'Barack Obama' and 'U.S. president' referring to the same person but the question arises here, how to identify this kind of situation that both the named entities in T and H are same? As far as current approaches are concerned, these approaches only focusses on the syntactic and semantic part of the sentences, but the situations such as given in the example above also play an important role. Current approaches in *text entailment* will return as "*No Entailment*" even both the sentences are same. The result is contradictory because none of the approaches focussed towards this thing.

The solution for this problem is to use a large amount of data which has access to the information of every entity present in this world. In the present scenario, wikipedia is the most commonly used website which keeps track of all the data related to an entity. But, again one question arises

[1] http://en.wikipedia.org/

that how can we use this data for getting this kind of information? The solution is that wikipedia data is stored in the form of RDF, which is also called as RDF freebase. How it can be used, we will see it in the further sections. Also, how to identify "Delhi is a part of India?" this concept is somewhat related to that proposed in (MacCartney and Manning, 2007). This is the concept of monotonicity. Our interest lies in RDF because it stores the data in the proper format which we shall discuss in the further sections.

2 Insight about Semantic Web

Consider a "wikipedia" page that consists of lots of information for a particular entity whether its a person, an organization or something else. The word 'semantic' signifies that those small pieces of information can be manipulated by a human in the manner that can be utilized further. According to the W3C[2], "The Semantic Web provides a common framework that allows data to be shared and reused across application, enterprise, and community boundaries." The main objective of *semantic web* is driving the evolution of the current Web by enabling users to find, share, and combine information more easily[3]. There are tasks which a human can do very easily such as to find the relation between two entities but how can a computer do such kind of tasks?

T : *Sachin Tendulkar made a century.*

H : *The Little Master made a century.*

In the above example also, the relation between '*Sachin Tendulkar*' and '*The Little Master*' can be easily extracted out of the freebase data stored in the form of RDF. So, these kind of sentences give rise to a better accuracy towards 'text entailment'. It is the place where the need of *semantic web* arises. In the next section, an introduction of RDF is given which helps us to know the concept of *semantic web* clearly.

3 Fundamentals of RDF

RDF is a standard model for data interchange on the web. RDF extends the linking structure of the web to use URIs (Uniform Resource Identifier) to name the relationship between things as well as the two ends of the link (this is usually referred to as a triple). There is a large amount of knowledge distributed throughout the web and some data is definitely related to some other data in a certain manner. So, to link that data, RDF (*Resource Description Framework*) is required.

Example: *Barack Obama is also known as U.S. president*

Subject	**Predicate**	**Object**
Barack Obama	known as/ same as	U.S. president

As stated in the introduction that it will be good to use *RDF freebase*. It stores all the data related to an entity. If it is a case of a person's name then it stores all the names by which that person is known in the world. Thus, its a good way to resolve the named entity dependencies in this manner.

In this way, we can recognize the relation between different entities. Data is available in an ample amount on web. Utilizing that data in our approach, we can resolve the dependencies of two different entities by matching whether they are same or different.

3.1 Need of RDF in Text Entailment

The basic concept of 'text entailment' is that if there is a large amount of information from which various relations can be extracted out between the entities provided in the given sentence, then the probability of entailment between two sentences increases.

$$T \wedge B \longrightarrow H$$

where, T is text, H is hypothesis and B is the background knowledge. The above formula states that the information present in H can be derived from T with the help of some background knowledge. So, B plays a prominent role while determining the entailment. The importance of RDF in 'text entailment' lies in the fact that it has come to be used as a general method for modeling of information that is implemented in web resources. Here, 'information' is referring to the overall knowledge about anything.

4 Approach

RDF logic provides a path to the data that is linked to each other in some manner as we saw in the previous section, *e.g.,* there is an entity which can have many relations to other entities in this world. So, how and from where can we get this information? There are some modules of the approach proposed in this paper, which are as follows:

[2]http://www.w3.org

[3]http://en.wikipedia.org

4.1 Coreference Resolution

Coreferences point to the same entity in a sentence. Co-reference occurs when multiple expressions in a sentence or document refer to the same thing, *e.g.,* in the sentence "John told Meera about his future plan and she liked", 'his' is referring to 'John' and 'she' is referring to 'Meera'. This should be resolved at the beginning of the implementation because it will help in parsing phase to extract out the actual dependencies. We used Stanford coreNLP[4] project for this phase.

Handling coreferences at the starting of the approach is a good idea because this helps in determining the actual dependency of an entity to another entity instead of using 'he', 'she' or 'it' *etc.* When we get actual names replaced in place of pronouns (*he, she etc.*) then comparison of dependencies of two sentences would become easy and clear.

T : *John told Meera about his future plan and she liked*

H : *Meera liked the future plan of John*

In the example stated above if coreferences are not handled in the starting of the approach then 'he' and 'she' will be used in the dependencies and relation between Meera and John would not be captured by text as it is in the hypothesis. In H, relation is *liked* while in T (without handling coreferences), relation is *told* but after handling coreferences, relation from H would be captured by T.

4.2 Named Entity Resolution

It is also known as *entity identification* or *entity extraction*. It helps to detect names of persons, organizations, locations, expressions of time, quantities, monetary values, percentages, *etc.* It is very necessary to determine named entities present in the sentences because the name of a person or an organization may be more than one word in length. So, to organize this thing, named entity recognizer is required. *e.g.*

Sachin Tendulkar plays cricket

Without using NER, 'plays' will attach only to 'Tendulkar' but after using NER, 'Sachin Tendulkar' will be recognized as an entity and finally, 'plays' will attach to the whole entity instead to a partial entity (as in the above example, matching to only 'Tendulkar' instead of 'Sachin Tendulkar').

[4]http://nlp.stanford.edu/software/corenlp.shtml

One person or organization may be known by its several names and these names can be used in the sentences also. So, its very difficult to compare two different words even if they are used for a particular person. Wikipedia's data is stored in freebase and freebase in turn stores them in the form of RDF. We used Stanford NER[5] for extracting out the dependencies and RDF freebase[6] for getting the information about those entities.

e.g., the aliases that are stored in RDF freebase for *Roger Federer* are as follows:

Federer Express
El reloj suizo
King Roger
JesusFed
NinjaFed
The Mighty Federer
The Swiss Meastro
Rog
Fed Express
Swiss Maestro

Its very hard to say that any of the above two entities are referring to the same entity without using any knowledge base. That means, if any two of the above names are used in two sentences then with the help of freebase, the machine can easily recognize the relation between the same named entities that look as different at first glance but are similar to each other.

4.3 Parsing

Parsing has been done to get the dependencies of the sentences. Here, our main intent is to find only those dependencies which occured with subject or object of the sentences so that rest of the dependencies can be ignored. We ignored the dependencies not consisting of subject or object of the sentence because these either contain prepositions or the articles used in the sentence. The output of the previous two modules will be utilized here. Suitable use of coreferences and named entity recognizer is the crucial part of parsing because this phase totally depends on the mentioned modules. For getting the dependencies, we used "Stanford dependency parser"[7], *e.g.,*

Ram went to school by bus.

The dependencies generated by "Stanford parser" are:

[5]http://nlp.stanford.edu/software/CRF-NER.shtml

[6]http://rdf.freebase.com

[7]http://nlp.stanford.edu/software/lex-parser.shtm

nsubj(went, Ram)
root(ROOT, went)
prep(went, to)
pobj(to, school)
prep(went, by)
pobj(by, bus)

Taking only the dependencies consisting of subject or object of the sentence:

nsubj(went, Ram)
pobj(to, school)
pobj(by, bus)

It can be seen from the above example that ruling out some of the dependencies keep intact the semantic of the sentence and it also reduces the complexity of the algorithm while implementation.

4.4 Verb Entailment

Parsing outputs dependencies and it consists of subject/object and the verbs used in the sentence. Till now, we have dealt with named entity resolution (subject/object comparison). Now, we would concentrate towards comparing verbs in the sentences, to know the relation between two verbs such as: synonymy, antonymy *etc.* Two sentences may contain different forms of the verb or there may be a synonymous verb in the second sentence from the verb of the first sentence. So, how can we compare these verbs? *e.g.,*

T : *Ram went to Delhi.*

H : *Ram visited Delhi.*

From the above example, we can see that the meaning of both the sentences is nearly the same. And, the main verbs *went* and *visited* are almost same. We, as a human can judge that both the verbs can be used interchangeably but how will computer understand?

For this matter, "WordNet"[8] was used but in the wordnet, only root form of the verbs can be compared, so in this case *went* and *visited* will never be matched and there will be no entailment in verbs. To handle this issue, a good quality stemmer is required so we used "MIT WordNet Stemmer"[9]. It reduces any form of the word to its actual root form so that comparison can be done easily and accurately.

[8]http://wordnet.princeton.edu

[9]http://projects.csail.mit.edu/jwi/api/edu/mit/jwi/morph/WordnetStemmer.html

4.5 Dependency Comparison

We got the dependencies from step 3 (parsing) of both the sentences. Now, the task is to compare the dependencies of hypothesis with those of text. Because we have found out the relation between the corresponding named entities and verbs from both the sentences so we would have to keep the relations found among NEs and verbs till the final output is calculated. Now, we can assign some common code to the matching NEs of T and H. Similarly, we can do so for verbs also so that it will be easier for comparing the dependencies of both T and H. If all the dependencies of hypothesis are present in text then there will be **Entailment** otherwise **No Entailment**. Here, we are not considering any pre-assigned threshold but we are checking fully matched dependency pairs.

5 Algorithmic Steps

In this section, an algorithm is given to capture all the aspects provided in the previous sections:

1. First, both the Text (T) and Hypothesis (H) are given as input.
2. For a sentence containing coreferences, there is a need to resolve that. So, Stanford coreNLP is used for Coreference Resolution.
3. From both the sentences, named entities are identified. So, Stanford NER system is used for getting the named entities.
4. Now, both the sentences are parsed using Stanford parser to get the dependencies.
5. Named entities identified in step 3 are termed as NE_T for named entities of text(T) and NE_H for those of hypothesis(H).
6. Now, take the Symmetric Difference of NE_T and NE_H as follows:

$$\text{symm-diff} = NE_T \triangle NE_H$$

7. The elements(named-entities) in the symm-diff are queried in the RDF database, which is freely available[10].
8. Till now, we were dealing with the subject or object of a dependency. Now, its time to play with verbs, so the verbs are searched in the Wordnet dictionary and the corresponding synsets are searched in that.

[10]http://www.freebase.com

9. Because, its difficult to obtain synset in the past and past participle form of the verb. So, MIT Wordnet Stemmer is used to stem the verbs of both the dependencies from T and H.

10. At this stage, we have both the named-entity resolution and verb entailment resolution. So, we should now compare the dependencies of both T and H that we got in step 4.

11. If all the dependencies from hypothesis are matched with the dependencies of text then there is ENTAILMENT otherwise NO ENTAILMENT.

6 Experiments with RTE data

We experimented with RTE 1 to RTE 3 development data set, both with and without coreference resolution. There are certain characteristics of RTE data set that our system is not able to handle. First, the difference between length of *text* and *hypothesis* is quite large but our system performs well for the pair of sentences of comparable lenghts. Second, we can do a query to the RDF freebase with a properly defined name of a person or organization but in many sentences only part of the name is used so in that case, the RDF freebase data is not fetched out and result could be reversed.

RTE Challenge	Bag of Words	RDF based
RTE 1	50	58.29
RTE 2	50.125	56.38
RTE 3	48.625	59.375

Table 1: Experimental Results (% Accuracy)

From table 1, we can see that our approach performs better in every case as compared to 'Bag of Words' approach. It performs better by a greater extent. We compared with 'Bag of Words' because it is the simplest approach so for getting the result it is a nice way to compare our approach with this one.

In the next table, we compared the approach proposed in this paper with the "NatLog" approach (MacCartney and Manning, 2007). This approach was based on 'Natural Logic' and it dealt with monotonicity of words used in the sentences. From table 2, we can see that our approach performs better to some accuracy as compared to the 'NatLog'. For RTE-3 development dataset, there is an improvement of 2.85% which is noticeable.

RTE Challenge	NatLog	RDF based Entailment
RTE 1	-	58.80
RTE 2	-	57.98
RTE 3	58	60.85

Table 2: Experimental Results after Coreference Resolution(% Accuracy)

Next, we have shown the graphical comparison so that it becomes easier by looking at graphs only.

System	%yes	precision	recall	accuracy
Stanford	50.25	68.66	66.99	67.25
NatLog	18.00	76.39	26.70	58.00
RDF based	**30.00**	49.23	**31.91**	**60.85**

Table 3: Performance on the RTE3 development set. % yes indicates the proportion of yes predictions made by the system. Precision and recall are shown for the 'yes' label.

From table 3[11], it can be seen that our approach performs better than NatLog system and it is slightly lesser than Stanford system in terms of accuracy. Our system gives a better number of 'yes' predictions compared to the 'NatLog' system.

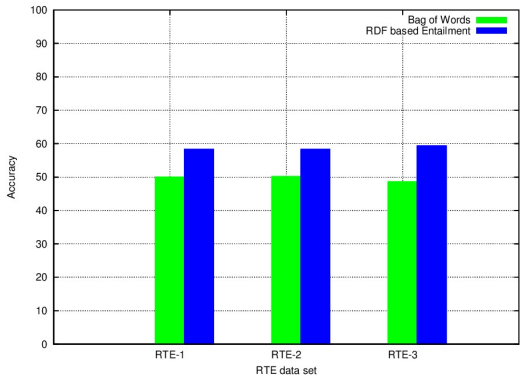

Figure 1: Comparison of Bag of Words and the proposed approach

7 Error Analysis

In this section, we present the analysis that was done manually by going through the RTE datasets with the GOLD standard value.

- In many sentences, there was a confusion to which of the two NEs, a coreference should be attached with. So, due to this confusion, many times result was contrary to the expectation.

[11] Accuracy shown is for the overall system which consists of 'no' labels also.

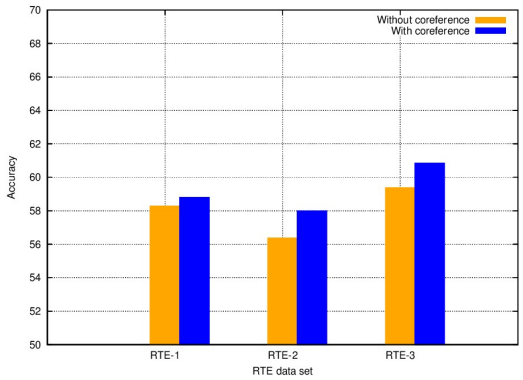

Figure 2: Comparison with coreference and without coreference in the proposed approach

- Sometimes named entities identified do not have corresponding RDF document with that name e.g. for a persons name, it should be a full name otherwise, it gets confused to which page to fetch as there can be more than one person with same first name.

- Negation words also play a crucial role in determining the entailment. In this approach, we checked that there should be equal number of negation words so this assumption sometimes gets wrong e.g. not go and stop are similar in meaning but there is an inequality of negation words. So, in many sentences, this problem arises.

- Sometimes there are more number of dependencies of hypothesis as compared to the text. In such cases, even if all the dependencies of text are matched to the hypothesis but there could be some dependencies remaining in the hypothesis that could result in a wrong result.

8 Conclusion and Future Work

Our RDF based approach towards *text entailment* fulfills the aim of resolving the named entity issue. A knowledge base is required which keeps track of the updated information of the world data. In this way, RDF freebase is appropriate for resolving these issues. For a sentence to be completed, a verb is also required with the subject or object. Verb entailment also plays a vital role when recognizing the entailment between two sentences. After all, the dependencies of hypothesis are compared with those of text and a final result is produced by the system.

From the example, it can be seen that the information is stored in the form of linked open data. Suppose we are going to extract information about some entity (of real world, in this example lets assume John, Mary, Steve and Richard are real world entites) in the form of *Question Answering*, *e.g.*

:john	:a	:person
:john	:hasMother	:mary
:john	:hasFather	:steve
:steve	:hasBrother	:richard

Question : ***Who is the uncle of John?***

Because, above data has information about John's father and Steve's (father) brother. So, if we can extract the relation from a lexical resource that 'uncle' is that person who is the brother of the given person's father then the above question can be answered easily.

RDF data is stored in the form of linked data, so there is an option to relate different entities and extracting out the relation from the stored data itself and then that relation can be compared with the one occuring in the given pair of test sentences. RDF stores the data in the form of Linked Open Data(LOD) that can be used efficiently in determining the entailment of the sentence in some other given sentence. By this way, this idea is a nice path finder for various areas such as: *Text Summarization*, *Question Answering*, *Information Extraction, etc.*.

References

Glickman, O. and Dagan, I. 2005 *A Probabilistic Setting and Lexical Cooccurrence Model for Textual Entailment*. EMSEE '05. Association for Computational Linguistics.

MacCartney, B. June, 2009. *Natural Language Inference, Ph.D. thesis*. Stanford University.

MacCartney, B. and Manning, C. D. 2007. *Natural logic for textual inference, pages 193-200*. RTE '07. Association for Computational Linguistics.

Dagan, I., Glickman, O., and Magnini, B. 2005. The pascal recognising textual entailment challenge.

Haghighi, A. D., Ng, A. Y., and Manning, C. D. 2005. Robust textual inference via graph matching. pages 387–394.

Google 2013 Freebase Data Dumps https://developers.google.com/freebase/data