Pronoun Resolution and Summary Extraction From English Documents

Naman Kumar Gupta, Saurabh Garg and Ratna Sanyal.

Research and Development centre, Universal Digital Library
IIIT-Allahabad, India

Abstract

This paper presents an approach to generate a precise and meaningful summary of a Document in English Language. Here, we adopt a modified Sidner's Focusing algorithm to perform pronoun resolution. We devise an algorithm to divide any sort of compound and complex sentences into simple sentences. Anaphora resolution is performed on these simple sentences. Then we find the lexical cohesion between pairs of the Noun Phrases whose antecedents have already been found.

1 Introduction

Document Summarization is one of the most important areas of research in Natural Language Processing (NLP). This study examines various aspects involved in creating a semantically precise and meaningful summary. Section 2 briefly describes a method to find Noun Phrases (NP) present in the Document. The formation of NP's is very important as both anaphora resolution and LSI depends on the various NP's existing in the document. Section 3 is the main contribution of the present work. In this section we give a recursive rule based algorithm to divide complex and compound sentences into virtual simple sentences, on the basis of the conjunctions present in the sentences, without changing the meaning or the semantics of the sentences. Anaphoric Resolution is performed on these simple sentences using the Sidner's Focusing Algorithm which is explained in Section 4. Once antecedents of all the Noun Phrases have been found, a Latent semantic net is formed between the NP's and the original sentences of the document. The conclusion and scope of future work is mentioned in Section 5.

2 Finding Noun Phrases

A Sentence consists mainly of a Noun Phrases and Verb Phrases (VP). It is very important that we find out the noun phrases that occur in a document as accurately as possible. This is because:

- Anaphoric Resolution depends on the existence of the various NP's that are present in the document. We try to map a NP with a previously occurring NP by finding the antecedent or the co-reference of that NP.
- Our formation of Lexical Semantic Index also depends on the same NP's. Once the antecedents of the NP's have been found using the focusing algorithm, we create a list of all the NP's that occur in the document and form an LSI between the important NP's and important sentences depending upon the number of times the NP's have occurred in the document.

Any Part of Speech (POS) tagger can be used to find out the part of speech of a word. As soon as we find a Noun or a Proper Noun, we look at the words in the near vicinity. Since the articles and the Adjectives describe a Noun, they should be included in the Noun Phrase. The following "rules" describe the method used for identifying the NP's:

- We need to concatenate Pronoun Nouns which occur one after another without any conjunction. For Ex.: Sachin Ramesh Tendulkar or Lord Mountbatten. Here, in the first NP, all the three words are Proper Nouns which belong to the same entity. So they form a single NP. Any articles or adjectives before the NP are also included in the NP.

- We also check for infinitives in a sentence. They are also considered as NP's, as they too can be co-referred. A verb which is preceded by "TO" is known as an infinitive as is used by a Noun.
  Ex. Jack, a small town boy, yearned *to live* in big city like New York.

- Gerunds are also included as Noun Phrases. If verbs in present continuous forms are not preceded by an auxiliary verb, then they are known as gerunds. And are used as noun phrases.
  Ex. Considering the widespread interest in the election, only a handful of voters turned up.

- If two NP's are separated by "of", then they are clubbed into a single Noun Phrase.
  Ex. The Queen of England, The President of the United States of America.
  In the above example, NP's "The President" and "the Unites States of America" are concatenated into a single NP using the "of" preposition to form "The President of the United States of America" as one NP.

We also maintain an identifier to keep track of the important nouns in a NP. After removing the associated adjectives and articles, nouns or Proper Nouns present in the NP are marked as the identifier of that NP. This means that this identifier can be used in the document as co-reference to the complete NP.
Ex. Jack bought a red car. The car has many modern features like cruise control.
In the above example, "car" is the identifier for both the NP's "a red car" and "The car". By comparing the identifiers, we mark them as co-referents. Here obviously we cannot distinguish between two noun phrases on the basis of their adjectives.

In case of Noun Phrases containing Proper Nouns, each Proper Noun is an identifier.
Ex. Ram Kishor Aggarwal. Here the NP "Ram Kishor Aggarwal" can be referred by "Ram", "Kishor" or "Aggarwal", so all the three are taken as identifiers of the NP. We try to match the largest possible string of words.

3 Sentence Splitting

What is a Sentence?
A Sentence is primarily said to consist of a Noun Phrase and Verb Phrase. And these NP's and VP's are further divided into Noun Phrases and Verb Phrases to form complex sentences.

Simple Sentence
A simple sentence is a sentence which contains only a subject and a predicate. Subject is the doer of the verb and Predicate consists of the verb and the object.

Complex Sentence
A complex sentence is a sentence which has more than one verb phrase.

Now, in Sidner's focusing algorithm, emphasis is given on the "theme" or object of the verb. This algorithm works well for simple sentences. However, if the sentence has more than one verb, then this algorithm does not produce satisfactory results since only one verb phrase can be focused upon. The rest will have to be ignored in Sidner's method as one sentence is considered at a time during the focusing algorithm. This necessitates the need for division of a complex sentence into simple sentences in which all the Noun Phrases associated with different verbs can be focused upon. The number of simple sentences will depend upon the number of verbs in the original sentence.

We now describe a method for splitting a complex sentence into simple sentences. The division of sentences is done on the basis of an exhaustive list of conjunctions. We follow a simple recursive call upon encounter with a conjunction. We maintain a data structure "clause" for each clause which is as follows:

1. Subject of the sentence (Vector of Strings)
2. Verb of the sentence (Vector of Strings)
3. The actual clause(Vector of Strings)
4. Last noun phrase(String)
5. Conjunction between this clause and next clause (String)
6. Auxiliary verb in the clause(String)
7. Gerund in the clause(String)

Virtually, we divide a sentence into clauses depending upon the conjunctions in the sentence. We maintain the above data structure for each clause. Initially, starting from the beginning of the sentence, we wait till the next conjunction. If the next clause contains no verb or auxiliary verb, then we concatenate the two clauses into a single clause and update its data structure accordingly.

If it does contain a verb or an auxiliary verb, then we send the remaining sentence from the previous conjunction to a recursive call. There the process is repeated until the end of sentence is reached. Then we start backtracking in the recursive call, checking the data structure of the clauses. When the last sentence is reached, its data structure is sent back to source function. There we compare the data structures of the two clauses:

```
Form_sentence(clause Cur_sent, next_sent)
{
        /* Here we compare the data
        structures of the two clauses and
        form new sentences on the basis of
        the rules mentioned below.*/
        /* A clause containing a subject and
        a verb is considered as a complete
        sentence*/
```

Rule 1:
When next sentence is complete and current sentence contains only subject: Both clauses are concatenated with the conjunction and the subject is updated and returned to source.

Rule 2:
When next sentence is complete and current sentence does not contain subject and verb or contains only a verb: Next sentence is taken as a complete sentence and stored in a temporary array. The current sentence is sent to the source.

Rule 3:
When both next sentence and current sentence are complete: Next sentence is taken as a complete sentence and stored in a temporary array. The current sentence is sent to the source.

Rule 4:
When next sentence contains only verb and auxiliary verb, and if next sentence doesn't contain an

auxiliary verb: The auxiliary verb is copied to that of the next sentence and both sentences are returned to the source.

Rule 5:
Only when the next sentence is complete and no changes are made to it in the form_sentence: The next sentence is taken as a new sentence and the current sentence, whatever it may be, is sent to the source. This is due to a missing compound sentence which may be present in the source clause.

Ex. Ram and Mohan went to the market and bought a computer.

The two sentences formed will be:
Ram and Mohan went to the market.
Ram and Mohan bought a computer.

Rule 6:
Next sentence has only subject: This means that it will act as an object of the verb of the current sentence. It is concatenated with the current sentence along with the conjunction and the last noun phrase is updated.

Rule 7:
The last noun phrase is checked when the conjunctions
{
which|who|whose|whom|that|,which|
,who } are the separating conjunction between the current and next sentence: In this case, the last noun phrase acts as the subject of the next sentence.

Rule 8:
Whenever any changes are made to a clause, the data structure of the respective clause is updated immediately. The actual clause string is changed whenever a subject or an object is added. Other respective changes are also made.

In the end, all the simple sentences are stored in a temporary vector. This vector is then passed to the focusing algorithm for resolution of the pronouns is done. This is explained in the next section.

4 Focusing Algorithm

At this stage, we have simple sentences as the input to the focusing algorithm. Before that we calculate the attributes of the noun phrases which were formed earlier. We find out its gender, number, animacy. The gender is found using word net and certain database entries. The animacy of the noun phrase is also found out using the word net by classifying certain sys-sets as animate or inanimate (Evans and Or˘asan, 2000).

Gender and number of certain named entities were found by the database provided by CoNLL-2003. Organizations, locations, dates and names were found and given appropriate gender, number and animacy values.

Then the current focus, alternate current focus list, actor focus, actor focus list, actor stack, focus stack, actor set and theme set are initialized and maintained (Ebru Ersan and Varol Akman, 1994, Focusing for Pronoun Resolution for English Discourse: An implementation).

The co-reference for all the noun phrases are stored in their respective structures. They are then passed to LSI and a list of noun phrases is made.

5 Conclusion and Future Work

This work proposes a new set of rules for sentence splitting. The simple “virtual” sentences are then used to find the “focus” of the given text. Significant success has

been achieved for a fairly good percentage of sentences. Splitting of sentences into smaller sentences has improved upon Sidner's Focusing Algorithm greatly.

There is scope of improvement and lot of work needs to be done. Rules for interrogative sentences, for the purpose of sentence splitting still needs to be worked out. Pronoun resolution needs to be improved. Also, other types of anaphora resolution like One, Bound still have to be considered. Addition of an Inference Engine will ensure certain world knowledge in our resolution process. Work in these areas are in progress and will be reported soon.

Acknowledgements: The authors would like to express their gratitude to the UDL project for financial assistance and infrastructural support.

6 References

[1] B.J.Grosz and C.L.Sidner. "Attentions, intentions and the structure of discourse", Computational Linguistics,12(3):175-204,1986.

[2] J.Gruber. "Lexical Structure in Syntax and Sematics", New York, NY, 1976

[3] Ebru Ersan and Varol Akman. "Focusing for Pronoun Resolution for English Discourse: An implementation", 1994.

[4] Elke Teich1 and Peter Fankhauser2. WordNet for Lexical Cohesion Analysis.

[5] Fellbaum, C., ed.: WordNet: An electronic lexical database. MIT Press, Cambridge (1998).

[6] Brunn, M., Chali, Y., and Pinchak. C. "Text Summarization Using Lexical Chains". Work. on Text Summarization. 2001.

[7] Evans and Or˘asan. "Learning to identify animate references", 2001.

[8] Richard Evans, Constantin Orasan. "Improving anaphora resolution by identifying animate entities in texts", 2000.

[9] Kristina Toutanova and Christopher D. Manning. Stanford Log-linear Model Tagger v1.0 - June 7 2004.