Dependency grammar feature based noun phrase extraction for text summarization.

Mrs. Dipti Sakhare

Department of E and Tc,

Bharati Veedyapeeth deemed University College of engineering,

Pune,Maharashtra, India.

Dr. Rajkumar

DRDO Scientist 'E',

Defense Institute of advanced Technology, Girinagar, Khadakvasala, Pune,Maharashtra, India.

Abstract- Now a days there is great amount of information available due to the development of Internet technologies. Every time when someone searches something on the Internet, the response obtained is a huge one with lots of information, which is impossible for a person to read completely. Hence one needs means of producing summaries of this information. Summarization is a very interesting and useful task which gives support to many other tasks like information extraction. It takes advantage of the techniques developed for Natural Language Processing tasks. In automatic text summarization most of the times the standard N gram model is used to develop the language model. The N gram models are unable to learn the grammatical relations of the sentences. Hence we propose to use the dependency grammar based noun phrase retrieval as a part of text preprocessing. This can be useful to learn the grammatical rules and thereby may be helpful to extract fundamental semantic units from the natural language text.

Keywords: Text Summarization, Natural Language Processing, language model, dependency grammar.

I. INTRODUCTION

With the continuing fast growth of information and data today, it has become more and more urgent and important to find proper information efficiently, with some improved mechanisms. The existence of the World Wide Web has caused an information explosion. Readers are overloaded with lengthy text documents for which a shorter version may also suffice. Therefore an automatic and accurate summarization system is essential which will be beneficial to help people comprehend all incoming information in a reasonable amount of time. There are even certain cases where automatic summarization can turn an intractable problem into a tractable one. Let us take the example of a old book repository. Someone searching the texts in a repository may have no means of determining the information in a given book beyond what is contained in the catalog description other than actually reading the entire book. If that person intends on searching through many books, this process might be so time consuming that it becomes infeasible! Hence a mean of automatic summarization would be very helpful to resolve such sort of problem.

Document usually consists of various topics. Some topics are typically described in detail by more sentences than other topics and hence may be inferred to comprise the major content of the document. But the topics that may be briefly mentioned to supplement or support the major topics are equally important. The stages of Text Mining incorporate the most trivial step of representing sentential components meaningfully. Since many researchers are exploring the semantic representations upon texts. This corpus- based computational linguistics includes Natural Language Processing tools viz. Part-Of-Speech (POS) taggers [1], Noun-Phrase chunkers [2], Semantic Parsers, Machine Translators and Text Summarizers. Moreover, the computational linguists need the Part-Of-Speech Tagging Tools and Noun Phrase (NP) Chunker modules to extract the typical noun-phrase fragments that are significantly essential thematic portion in natural language representations.

Basically summarization deals with selection of important sentences from the text. The importance of the sentence is based on statistical and linguistic features. To extract the statistical and linguistic feature ATS has to go through various phases. The important phases are 1) preprocessing 2) processing. Preprocessing is the structural representation of text. This paper concentrates on preprocessing phase. Specifically the emphasis is given on noun phrase retrieval of the sentences using dependency grammar. The important part of text summarization is identifying the essential content, understanding it clearly and generating a short text. In these steps a system has to understand the point of the text, which makes summarization a very hard problem. The summarization task involves many techniques as 1) semantic analysis 2) discourse processing and 3) inferential interpretation. The grammar chosen to build the language model plays an important role in semantic analysis. Here we concentrate on dependency grammar to build the language model for text summarization system.

II. N -gram Language models

In automatic text summarization (ATS), the standard choice for a language model (LM) is the n-gram model. The n-gram model is used to predict the probability of a word given its context of n − 1 preceding words

$$p(w_i|w_1^{i-1}) \approx p(w_i|w_{i-n+1}^{i-1}). \qquad \text{…………..1}$$

The sentence probability is then obtained as the product of these conditional probabilities

$$p_0(s) = \prod_{i=1}^{N} p(w_i|w_{i-n+1}^{i-1}) \qquad \text{……………2}$$

here n is the number of words in the sentences.

As the data is sparse, the n in equation 1 is most of the times set to 3 or 4.this type of modeling considers only the local dependencies of English language. As the choice of n is small the grammatical regularities can be captured implicitly. Consequently our point of interest is to model the grammatical knowledge explicitly. Due to the intrinsic sparseness of the data, n in equation 1 is usually set to 3 or 4 (for English). Therefore, the modeling is based on local dependencies of the language only; the grammatical regularities learned by the model will be captured implicitly within these short word windows. Consequently, we are interested in explicit modeling of grammati- cal knowledge.

The means of explicit grammatical modeling include structured LMs and syn- tactic triggers [3]. Essentially, these models preserve the conditional framework of equation 1; words are assigned syntactical classes based on some suitable parsing, and the current word is then conditioned on both its syntactical and plain word contexts. Alternatively, the grammatical information can be seen as an additional information source to be combined with the local dependency information captured by the n-gram model. For this approach, a natural framework is provided by maximum entropy (ME) modeling. In our experiments, we will apply the Whole Sentence Maximum Entropy language model (WSME LM) introduced in [4,5]. The WSME framework will enable us to merge arbitrary computational features into the n-gram model. Effectively, the sentence probabilities given by the n-gram model will be scaled according to the features present in the sentence.

In the present proposed work, the grammatical information will be captured by features extracted using dependency grammar; dependency grammars are discussed by Tapanainen in [6] and Nivre in [5]. Previously, grammatical features extracted with probabilistic context- free grammars (PCFG) have been used successfully in combination with the WSME models in [8]. Using grammatical features together with N -gram features in a WSME LM, a considerable reduction in perplexity over a baseline n-gram model can be reported. In the earlier work a small training set was used and no summarization experiments were performed [8]. The next section throws the light on dependency structure that we will use to build the language model.

III.REPRESENTATION OF DEPENDENCY STRUCTURE

The computational linguistics communities unanimously agree that grammatical dependencies are the most agreeable conceptual representation for any free text. Dependencies are exhibited among terms lying within or in the locality of adjacent sentences, revealing term-to-term associations for complete description of a topic being discussed in those sentences. The conceptualization of dependency parse trees was a milestone in encoding useful semantic information lying in the text. Initially the parse tree constructions needed training and revision phases, so as to identify term-dependencies in a tree and to replace incorrect dependencies with correct ones. In literature also developments of deterministic dependency parsers resulted with good accuracy as well as high performance in range of hundreds of sentences per second [9, 10].

There was the biggest breakthroughs in NLP in 1990s that works out with grammatical structure of sentential fragments [11]. These fragments when grouped together in multi-word phrases form the subject and object of the participating verb phrase. The Stanford group of probabilistic parsers use the knowledge gained from hand-parsed sentences in order to attempt generating the most-likely analysis of sentences in test-set. This package is a Java implementation available both in optimized PCFG and lexicalized dependency parsing versions.

The Stanford NLP Group has pioneered the concept of using typed dependencies. These are the simple descriptions of the grammatical relationships in a sentence and around its sentential neighborhood of relevant context. The salient feature is that typed dependencies are defined uniformly every pair of words, means 1-gram phrases in text-miming terminology, rather than directly hitting phrase-structure representations, that have long dominated the minds of computational linguistic community [9]. These dependencies can be well understood without thedeep linguistic expertise for carrying out mining tasks of Ontology learning, Machine translation of text, Text Summarization, etc. the intermediate step between the two is the contextual relation extraction. These semantic representations are in the form of causative relation patterns i.e. <Noun Phrase, Verb, and Noun Phrase>.

A.DEPENDENCY GRAMMAR FEATURE

We apply the WSME framework with features obtained using the dependency grammar [6,7]. Given a sentence s = (w_1 , w_2 , . . . , w_N), the dependency parsing results in head-modifier relations between pairs of words, together with the labels of the relation- ships. The labels describe the type of the relation, e.g. subject, object, negate. These asymmetric bilexical relations define a complete dependency structure for the sentence. An example of a parsed English sentence is shown in Figure 1. Here, we have followed the graphical notation convention used by Nivre in [7]; the dependency arrow points from the head to the modifier.

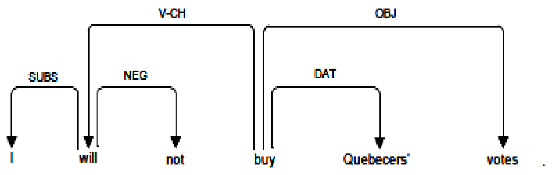

Figure 1. An example English sentence parsed with dependency grammar.

We will convert the dependencies into binary features to be used in the WSME model. We will experimente with dependency bigram and trigram features. Dependency bigram features contain a relationship between a head and a modifier, together with the name of the syntactic function. Which has dependency trigram features contain a modifier with its head and the head's head, comparable to a child-parent-grandparent relation. Examples of bigram and trigram features extracted from the sentence in Figure 1 are shown in Figure 2.

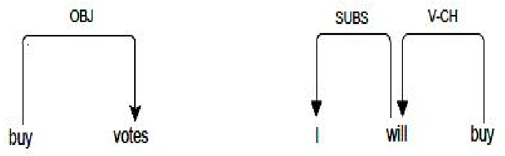

Figure 2. Example of dependency bigram (left) and trigram (right) feature

B.DEPENDENCY CONVERSION STRATEGY

The conversion strategy works on the finite set of dependencies containing fifty-five grammatical relation definitions pertaining to English Grammar, as stated by Marneffe, Manning [12]. Each grammatical relation is composed of two arguments, first the governor argument and second, the dependent argument, with parentheses pairs. The dependency definitions make use of Pen tree-bank Part-Of-Speech tags and phrasal labels. It was found upon detailed survey that only a few grammatical definitions appear in enormous amount upon parsing a sentential input, while the rest of relations pose a guest appearance, indicating the change in grammatical construction of the sentence. For instance, if the same fact is conveyed in different grammatical formats as enumerated below: the set of dependencies in Fig3, Fig 4 and Fig 5 for the following three natural language constructions consecutively.

'*Most artificial neural system models are made of individual computational elements.*'

'The individual computational elements make up the most artificial neural system models.'

'The individual computational elements that make up the most artificial neural system models are rarely called artificial neurons.'

advmod(models-5, Most-1)
amod(models-5, artificial-2)
nn(models-5, neural-3)
nn(models-5, system-4)
nsubjpass(made-7, models-5)
auxpass(made-7, are-6)
prep(made-7, of-8)
amod(elements-11, individual-9)
amod(elements-11, computational-10)
pobj(of-8, elements-11)

Figure 3. Typed-dependencies of sentence 1

det(elements-4, The-1)

```
amod(elements-4, individual-2)
amod(elements-4, computational-3)
nsubj(make-5, elements-4)
prt(make-5, up-6)
det(models-12, the-7)
advmod(artificial-9, most-8)
amod(models-12, artificial-9)
nn(models-12, neural-10)
nn(models-12, system-11)
dobj(make-5, models-12)
```

Figure 4. Typed-dependencies of sentence 2

```
det(elements-4, The-1)
amod(elements-4, individual-2)
 amod(elements-4, computational-3)
nsubj(neural-11, elements-4)
rel(make-6,tha5)
rcmod(elements-4, make-6)
prt(make-6, up-7)
 det(artificial-10, the-8)
advmod(artificial-10, most-9)
 dobj(make-6, artificial-10)
nn(models 13system-12)
nsubjpass(called-16, models-13)
auxpass(called-16, are-14)
advmod(called-16, rarely-15)
 ccomp(neural-11, called-16)
amod(neurons-18, artificial-17)
dobj(called-16, neurons-18)
```

Figure 5. Typed-dependencies of sentence 3

Note the Stanford dependencies generated for each of the above parsed sentences carry word-position numbers along with their arguments.

IV. REMOVAL OF NON-SEMANTIC CONSTITUENTS

The idea is to generate the noun phrases from the typed dependencies obtained from Stanford Parser. A noun phrase (NP) may contain specifiers and qualifiers and the specifiers in turn may contain determiners . In the different classes of determiners, the articles and demonstratives are not considered as important in finding text semantics which is shown as det(X, Y) in Fig. 1, 2 and 3. The authors propose the deletion of det dependencies as the first step in noun phrase extraction. But at the same time, the quantifying determiners (some, all etc) holds predet(X, Y) relations and these depend on the domain for removal or consideration in the noun phrase extraction. If the domain considered for semantics is textual question answering, then predet may play a significant role. On the other hand, the quantifying determiners can be neglected in the case of text summarization. Now, the filtered dependencies can be shown in figure 4 after the removal of determiner grammatical constituents of the sentential fragments illustrated in figure 3.

A.FORMATION OF NOUN PHRASES

The typed dependencies obtained from Stanford Parser very well semantically analyses the

sentences and relates the neighboring sentences. Taking these as input, the noun phrases can be extracted for processing any text document. For each noun phrase, its semantic head is marked. If the noun phrase is complex, the right most noun is identified as the head which holds for almost all noun phrases in English. Hence , the typed dependency noun compound modifier (nn) is first treated which depicts any noun that serves to modify the head noun. The typed dependency from figure 3, nn(models-14, neural-system-13) is joined to make the compound noun phrase “neural-system models”. Hence, all the existing head noun “models-14” in figure 3 will be replaced by “neural-system models -14“ and the nn dependency will be removed.

Table 1.subject object role of noun phrases

Sentences / Contextual role	Text (fig.1)	Text (fig.2)	Text (fig.3)
Subject Role	Individual computational elements	Individual computationa l elements	Individual Computat-ional elements, Artificial
Object Role	Most artificial neural system models	Artificial neural system models	Most artificial, System models

V. THE PROPOSED ALGORITHM

The textual training corpus will consists of articles from the DUC 2002(English Corpus)·

To train the language model, we first parse the corpus and the sample (with inserted commas) using the functional dependency grammar parser

Next, we extract the dependency bigram and trigram features from the parse trees

The grammatical features are then pruned to only include those that occur at least five times in the generated sample.

Excluding features that do not occur in the sample frequently enough is needed in order to avoid infinite gradients for such feature weights during learning, as also observed by Schofield in [12].

In addition to the grammatical features, sentence length features will also be used, corresponding to lengths 1 to 20 which activate when sentence length is at least l. The sentence length features were chosen to constrain the marginal distribution of sentence lengths under the model to equal that of the corpus. This is motivated by the fact that an n-gram model lacks explicit modeling of sentence lengths as shown by Schofield.

6. CONCLUSIONS

We will perform our experiments with WSME LM using binary features extracted with a dependency grammar parser. The dependency features will be in the form of labeled asymmetric bilexical relations; experiments will be done on dependency bigram and trigram features corresponding to child-parent and child-parent-grandparent relations, respectively.

WSME LMs provide an elegant way to combine statistical models with linguistic information. The main shortcoming of the method is the high memory consumption requirement during training of the model .the experiments will be carried out on relatively large training set of sentences. We will try to show that an increase in modeling accuracy can be achieved using dependency grammar.

RREFERENCES

[1] Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003.Feature-Rich Part-of-Speech Tagging with a Cycli Dependency Network. In Proceedings of HLT-NAACL 2003 pages 252-259.

[2] Philip Brooks, "SCP: Simple Chunk Parser", Artificial Intelligence Center, University of Georgia, Athens, Georgia, USA, May 2003,
www.ai.uga.edu/mc/ProNTo/Brooks.pdf.

[3] J. Bellegarda, "Statistical language model adaptation: review and perspectives," Speech Communication, vol. 42, no. 1, pp. 93–108, 2004.

[4] R. Rosenfeld, "A whole sentence maximum entropy language model," in Proceedings of the IEEE Work- shop on Speech Recognition and Understanding, 1997.

[5] R. Rosenfeld, S. Chen, and X. Zhu, "Whole-sentence exponential language models: A vehicle for linguistic-statistical integration," Computers, Speech and Language, vol. 15, pp. 55–73, 2001.

[6] P. Tapanainen and T. Järvinen, "A non-projective dependency parser," in Proceedings of the fifth con- ference on Applied natural language processing, pp. 64–71, Association for Computational Linguistics, 1997.

[7] J. Nivre, "An efficient algorithm for projective dependency parsing," in Proceedings of the 8th Interna- tional Workshop on Parsing Technologies (IWPT), pp. 149–160, Citeseer, 2003.

[8] F. Amaya and J. Benedí, "Improvement of a Whole Sentence Maximum Entropy Language Model Using Grammatical Features," in Proceedings of the 39th Annual Meeting on Association for Computational Linguistics, p. 17, Association for Computational Linguistics, 2001.

[9] Giuseppe Attardi, Massimiliano Ciaramita: Tree Revision Learning for Dependency Parsing. Proceedings of HLT-NAACL 2007, Rochester, 2007.pp 388-395.

[10] Aoife Cahill, Michael Burke, Ruth O'Donovan, Stefan Riezler, Josef van Genabith and Andy Way (2008) Wide-Coverage Deep Statistical Parsing using Automatic Dependency Structure Annotation, Computational Linguistics, Vol. 34, No. 1, pages 81-124. September 2008.

[11] Katrin Fundel, Robert Küffner, Ralf Zimmer. RelEx - Relation extraction using dependency parse trees. Bioinformatics, vol 23, no. 3, pp. 365-371, 2007.

[12] Marie-Catherine de Marne_e and Christopher D. Manning, "Stanford typed dependencies manua