Dependency Syntactic Tree Supported Sentence Similarity Computing

Xiong Jing, Liu Yun-Tong and Yuan Dong
School of Computer and Information Engineering, Anyang Normal University, Anyang, 455000, Henan, China
High-Performance Server and Storage Technologies, State Key Laboratory, Jinan, 250100, Shandong, China

Abstract: In most fields of Natural Language Processing (NLP), sentence similarity computing plays an important role. At the same time, the syntactic similarity computing is the basis of sentence similarity computing. This study introduces a dependency syntactic tree similarity computation method based on multi-features. The method studies many features of dependency syntactic tree including word and word’s POS of each node and the dependency type between them. Then the similarity algorithm is proposed after comprehensively analyzing all the features. The experimental result is satisfied as this method describes dependency syntactic tree more comprehensively and accurately.

Key words: Similarity computing, syntactic dependency, multi-features, NLP, sentence processing

INTRODUCTION

Similarity is a complex concept and it is wildly discussed in semantics, philosophy and information science, etc. Now sentence similarity computing plays an increasingly important role in text-related research and applications in areas such as text mining (Li et al., 2006). It is the basis of many applications, such as snippet extraction, image retrieval, question-answer model and document retrieval (Gu et al., 2012). Syntactic similarity is the base of sentence similarity and the dependency-based representations in natural language parsing are increasing in recent years. The fundamental notion of dependency is based on the idea that the syntactic structure of a sentence consists of binary asymmetrical relations between the words of the sentence (Nivre, 2005).

There are some kinds of dependencies: Semantic dependencies (Melcuk, 2003), morphological dependencies (Ahonen et al., 1997), prosodic dependencies (Groch, 2011) and syntactic dependencies (Gibson, 1998). Dependency syntactic tree is made use of to find similar questions within the predetermined categories (Lian et al., 2013). There are few researches studying on syntactic similarity computation and most of them focus on words (Sagae and Gordon, 2009), not the whole structure.

In this study, we introduce a new method to compute the similarity between two dependency syntactic trees which considering not only word but also word’s POS, dependency type and correspondence between the two trees. This method can be used to measure the similarity between two dependency syntactic trees and to estimate dependency syntactic parsers by comparing the similarity between the output and manually annotated result.

DEPENDENCY SYNTACTIC

Dependency grammar is proposed by French linguist L. Tesniere in his book Éléments de la syntaxe structurale in 1959 which had a profound impact on Linguistics and is especially respected in Computational Linguistics. Dependency grammar reveals the syntactic structure by analyzing the dependency relationship among compositions of a sentence. In dependency grammar theory, verb is considered as the center or root of a sentence which dominate the other parts but verb itself is not dominated by any composition (Green and Dorr, 2004) and all dependency relationship can be described by one kind of dependency type.

The structure of a sentence can be represented as two forms in dependency grammar theory, one is syntactic tree and the other is relationship collection. As an example, the sentence “I use spring framework in my webapplication.” can be described as a tree showed in Fig. 1.

The relationship collection of sentence “I use spring framework in my webapplication.” is shown Table 1.
consider their five corresponding features, 1 for equal 
while 0 for not equal. Then order these five numbers 
by their weights, we can get a binary number (bbbbb), 
which is ranged from 0 to 31 and 0 means the two 
relationships are exactly the same while 31 means 
the two are completely different. Based on this binary 
number, we define the similarity between P₁ and P₂ as:

$$\text{SR}(P₁, P₂) = \frac{(bbbbb)_2}{(11111)_2}$$

(1)

Assumed within P₁ and P₂, there are C₁ = C₂, 
C₁-POS ≠ C₂-POS, H₁ ≠ H₂, H₁-POS ≠ H₂-POS, D₁ ≠ D₂, 
then the binary number is (10100), and the similarity 
between P₁ and P₂ is:

$$\text{SR}(P₁, P₂) = \frac{(10100)_2}{(11111)_2} = \frac{20}{31} = 0.6452$$

Let dependency relationship collections A = (a₁, 
a₂, ..., aₙ) and B = (b₁, b₂, ..., bₙ). Since the symmetry 
of similarity computation, we may assume that the size of A 
is smaller or equal than B that n = m.

For the similarity computation between A and B, we 
must decide how the relationships from A map the 
relationships from B. For each aᵢ ∈ A, 1 ≤ i ≤ n, we can find 
bᵢ ∈ B, 1 ≤ j ≤ m. We assumed that different bᵢ for different 
aᵢ and then there are n!/m!(m-n)! kinds of ways how 
collection A maps collection B.

Then we discuss a certain kind of way marked as Ωₖ, 
1 ≤ k ≤ n!/m!(m-n)! . In Ωₖ, for a certain relationship aᵢ, there 
is a relationship bⱼ which we can mark as bⱼ ∈ Ωₖ(aᵢ). Then 
we define the similarity of Ωₖ:

$$\text{Sim}(Ωₖ) = \sum_{aᵢ∈Ωₖ} [\text{SR}(aᵢ)]$$

(3)

Based on Sim(Ωₖ), we define the similarity of two 
relationship collections A and B as:

$$\text{SRC}(A, B) = \text{Max}(\text{Sim}(Ωₖ))$$

(4)

where, the value of k is.

We choose the way which makes the similarity 
between A and B gets the maximum. The similarity 
computed in that way is considered as the similarity 
between A and B.

For example, sentence S₁: “I use spring framework 
in my web application.” whose syntactic structure showed 
in Table 1 and sentence S₂: “I create my web application 
based on EJB.” whose syntactic tree is shown in Fig. 2.

Table 2 shows the dependency relationship 
collections similarity computing between sentence S₁ 
and S₂.
Table 2: Dependency relationship similarity computing of S1 and S2

<table>
<thead>
<tr>
<th>Relationship collection of S1</th>
<th>Relationship collection of S2</th>
<th>SR (SL, S2) (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>a1: nsubj(use/VBP, I/PRP,nsubj)</td>
<td>b1:nsubj(create/VBP, I/PRP, nsubj)</td>
<td>a1b1 = 48.39</td>
</tr>
<tr>
<td>a2:root(ROOT-, use/VBP, root)</td>
<td>b2:root(ROOT-, create/VBP, root)</td>
<td>a2b2 = 74.19</td>
</tr>
<tr>
<td>a3: nsubj(webapplication/NN, spring/NN,nm)</td>
<td>b3:nsubj(webapplication/NN, my/PRPS pos)</td>
<td>a3b3 = 64.45</td>
</tr>
<tr>
<td>a4: dobj(use/VBP, framework/NN,dobj)</td>
<td>b4: dobj(create/VBP, webapplication/NN,dobj)</td>
<td>a4b4 = 22.98</td>
</tr>
<tr>
<td>a5: pos(webapplication/NN, my/PRPS pos)</td>
<td>b5: pos(create/VBP, on/IN, prep)</td>
<td>a5b5 = 100</td>
</tr>
<tr>
<td>a6: prep_in (framework/NN, webapplication/NN, prep)</td>
<td>b6: prep_in (create/VBP, EJB/NNP, prep)</td>
<td>a6b6 = 29.03</td>
</tr>
</tbody>
</table>

Fig. 2: Syntactic tree of sentence S2

Then based on Eq. 2-4, we can get the similarity between these two syntactic structures:

\[ \text{SRC}_{(S1,S2)} = \frac{0.4839 + 0.7419 + 0.0645 + 0.2258 + 1 + 0.2903}{6} \approx 46.77\% \]  

(5)

EXPERIMENTS AND ANALYSIS

**Experiment 1:** We choose 40 sentences which will be calculated to compare similarity with a specified sentence. There is one standard sentence and the others which are similar with the standard sentence in syntactic structure are manually ordered by the similarity. Then we order the sentences with our similarity computation algorithm. The top 10 most similar sentences are showed in Table 3.

The results show that our algorithm has almost the same order with the manual one in syntactic structure. Our analysis of the results shows that when the syntactic structure of test sentence is consistent with the standard one and the test sentence has lots of same words with the standard sentence simultaneously, their similarity is high. We analyzed the sentence syntactic tree and discovered that when the root node which is the central word of sentence in dependency grammar is incorrectly recognized or not the same, the similarity between the two structures is low or even zero. This algorithm could not handle polysemmous word well such as “spring” in standard sentence and No. 6 sentence.

**Experiment 2:** Given another 25 sentences, they are divided into 5 groups: computer, literature, philosophy, internet and industry. Each group includes 5 similar sentences. Put these 25 sentences into a test set including 200 noise sentences. To take each one of these 25 sentences as a standard sentence and compute the similarity with others. After order the sentence similarity, choosing the top 4 most similar sentences. If the checked sentences belong to the group of the standard sentence, it means they are right sentences. Comparing with the SVM, the precision, recall and F value are shown in Table 4.

Experimental results show that our approach compared VSM method have greatly improved the precision and recall. The main factors affecting the algorithm performance are sentence length, dependency parsing accuracy and center word differences. The more consistent of compared sentences in syntactic structure and the more common words they contain, the higher the similarity. And in this condition, the smaller the difference of dependencies numbers, the higher the similarity.

CONCLUSION

This study introduces a dependency syntactic tree similarity computation method based on multi-features.
This method starts from dependency relationship, matches relationships from two collections to get the maximum of the similarity and then finds the average of all similarity of relationships as the similarity of the two dependency syntactic trees. In this method, we consider five features of dependency syntactic tree including word and its POS of each node and the dependency type between them and then comprehensively analyze the similarity relationship of all the syntactic structures. However, this algorithm does not handle the semantic link within dependencies. Semantic-based dependency analysis is our future main research content.

ACKNOWLEDGMENTS

This research is supported by Development projects of Henan province science and technology (132102210264) and Henan Province Education Office Humanities and Social Science Project (2013-GH-383).

REFERENCES


