Semi-automatic Lexicalized Tree Adjoining Grammar Extraction towards Natural Language Generation

Wei QIU[1,a], Tianfang YAO[2,b]

[1]Dept. of Computer Science and Engineering, Shanghai Jiao Tong University 800 Dong Chuan Road, Shanghai 200240, China

[2]Dept. of Computer Science and Engineering, Shanghai Jiao Tong University 800 Dong Chuan Road, Shanghai 200240, China

[a]qiuwei1987@gmail.com, [b]yao-tf@cs.sjtu.edu.cn

Keywords: Lexicalized Tree Adjoining Grammar, Grammar Extraction

Abstract. Deep grammar Formalism such as Tree Adjoining Grammar(TAG) is widely used in Natural Language Generation(NLG). However, crafting the grammar not only requires expert knowledge of both the grammar formalism and the target application domain of NLG, but also needs a lot of human labor. In this paper, we propose a semi-automatic approach to extract well-formed Lexicalized Tree Adjoining Grammar(LTAG) anchored with proper semantics from a parallel corpus. The parallel cor- pus consists of sentences and the correspondent semantics. This approach requires much less human effort. The result shows that our approach achieves comparable performance with the state-of-the-art. Meanwhile, the grammar extracted in this paper has better linguistic interpretation than previous work.

Introduction

As computer technology advances and gets more complicated, natural language can serve as a more natural and efficient interface between machines and humans. Natural language generation is the pro- cess to generate coherent natural language from machine oriented knowledge representations. NLG systems can be roughly categorized into three types: template-based approaches, grammar-based ap- proaches and corpus-driven approaches. Grammar-based approaches provide more flexibility and gen- eralization ability than template-based approaches. Meanwhile, most corpus-driven approaches either extend or enhance grammar-based approaches. Thus, grammar plays a crucial role in NLG systems. However, grammar crafting for NLG systems not only requires a lot of expert knowledge, but also is te- dious. In this paper, we will describe a semi-automatic approach to extract Lexicalized Tree Adjoining Grammar(LTAG) from a parallel corpus which consists of knowledge representation and correspon- dent generated natural language sentences. This approach dramatically reduces the expert knowledge and human effort involved.

This paper is organized as follows: firstly we briefly introduce the related work. Afterwords, we will describe our approach which includes two steps: the annotation step which requires human effort and the fully automatic grammar extraction step. The results will be discussed and some conclusions will be made.

Related Work

The idea using domain specific grammar extracted from parsed corpus for NLG system is proposed by [3] and [4]. Extracting grammar from treebanks has already attracted a lot of attention from re- searchers. Several proposals for Tree Adjoining Grammar(TAG) extraction are made by [8, 9, 10, 2]. For other grammar formalisms, see [5] for a detailed review . Our approach can be described as a similar approach to [3]'s method, but we payed more attention to semantic alignment annotation to make sure the grammar extracted has good linguistic

interpretation.

Tree adjoining Grammar(TAG) can be seen as an extension of traditional Context Free Gram- mar(CFG). It uses two types of trees, i.e., initial trees and auxiliary trees as the basic rewriting units. See [6] for a detailed description.

The rate of detoxification in the liver cell is directly proportional to the quantity of smooth endoplasmic reticulum in the liver cell.

```
(KBGEN-INPUT
    :TRIPLES (
        (|Detoxification19144| |base| |Liver-Cell19145|)
        (|Detoxification19144| |rate| |Rate-Value19132|)
        (|Rate-Value19132| |directly-proportional| |Quantity-Value19135|)
        (|Liver-Cell19145| |has-part| |Smooth-Endoplasmic-Reticulum19149|)
        (|Smooth-Endoplasmic-Reticulum19149| |quantity| |Quantity-Value19135|))
    :INSTANCES-TYPES (
        (|Detoxification19144| |instance-of| |Detoxification|)
        (|Rate-Value19132| |instance-of| |Rate-Value|)
        (|Liver-Cell19145| |instance-of| |Liver-Cell|)
        (|Smooth-Endoplasmic-Reticulum19149| |instance-of| |Smooth-Endoplasmic-Reticulum|)
        (|Quantity-Value19135| |instance-of| |Quantity-Value|))
    :ROOT-TYPES (
        (|Detoxification19144| |instance-of| |Event|)
        (|Liver-Cell19145| |instance-of| |Entity|)
        (|Rate-Value19132| |instance-of| |Property-Value|)
        (|Quantity-Value19135| |instance-of| |Property-Value|)
        (|Smooth-Endoplasmic-Reticulum19149| |instance-of| |Entity|)))
```

[1 The rate of] [2 detoxification] [3 in] [4 the liver cell] [5 is directly proportional to] [6 the quantity of] [7 smooth endoplasmic reticulum] [8 in] [9 the liver cell] .

Fig. 1: An example sentence and correspondent semantic triples from the KBGen Corpus. Semantics alignment annotation is listed as well.

To ground our discussion, we will first introduce the KBGen task [1] and the related corpus. The goal of KBgen task is to produce a coherent description of biological entities, processes and connec- tions between them. The task involves aggregation and the generation of intra-sentential, syntactically bound pronouns but it does not require the generation of any discourse anaphora of referring expres- sions. The corpus we used from KBgen task contains 107 instances. An example sentence and the correspondent semantic triples are given in Fig. 1.

LTAG Extraction from a Sentence-Semantics Parallel Corpus

Semantics Alignment Annotation. The grammar that we want to have consists of elementary trees with semantics(triples in our case) anchored. Hence, we need to align the internal structure of the sentence with the triples to establish the link between the elementary trees and the the triples. To reduce the expert knowledge required, our annotation is carried out on the surface form of the sentences. Hence, only little knowledge about TAG formalism will be needed. The alignment in our annotation schema has several properties: firstly, every word is aligned to some set of triples, i.e., no word in the sentence is redundant. Secondly, similarly, every triple is aligned to some set of words. Thirdly, the relation between the sets of words and the sets of triples should be one-to-one map. Lastly, the alignment should be as fine-grained as possible, i.e., any alignment which is a refinement of the above alignment must lack one of the above properties. These requirements are simplified and idealized to ensure the automatic extraction can be made deterministic and efficient, and the extracted grammar will be composable from both syntactic

and semantic perspective. We will show in the result section that this annotation schema is sufficient for extracting well-formed and linguistic interpretable LTAG.

To ensure the annotated corpus satisfies the properties above, we propose several annotation guide- lines as follows:

1. NPs are aligned to their entity declarations.
2. For simple intransitive or transitive verbs, the event entity declaration and the triples which specify the agent(as wells as the patient) are aligned to the verb.
3. In an n-ary conjuction, all of the $n - 1$ interposed conjuctions are aligned to specialized *coor- dinates* relations which group all the coordinated entities into a simple artificial entity.
4. A prepostition, e.g., `'X *of* Y' is aligned to a single triple which links some two entities (X and Y). Hence, the preposition would be aligned to an auxiliary tree where one of the semantic links will correspond to a substitution and the other corresponds to a adjunction in the resulting grammar.

An example of the annotation is given in Fig. 1.

Preprocessing We first run the Stanford parser([7]) to get the phrase structure of generated sen- tences. Stanford parser also gives the head information of each subtree structure, which is a strong hint in the heuristics in our grammar extraction algorithm. Secondly, we noticed there are inconsis- tencies in the output of Stanford parser. For example, in the output of sentence *The plasma membrane moves away from the cell wall during plasmolysis in a walled cell inside hypertonic solution, resulting in a damaged cell.*, the PP phrase *during …* and the S *, resulting in …* are both modifiers. Hence, both of them and the ADVP *away …* should be put on different levels if they are composed in an adjuction manner. However, the adjectives are sisters to the nouns that they modify. To resolve this asymmetry in the parser output, we deflatten the parse tree by lifting up the modifier and combine it with the orig- inal head node into an artificial father node. By deflattening the tree, we can also make the extracted trees smaller and therefore more general. To determine if a node is an adjuct(modifier), we check if:

- A is a PP
- A is an S which does not have the form [S [Vp=H] [TO=H] …][3]
- A is a comma (we treat commas as adjuncts, so that, along with the ``real" adjunct, they will form a two-level auxiliary tree)
- A is an SBAR of the form [SBAR [WHNP=H [WDT=H which]]…] (appositive relative clauses)

To deal with the flat conjunction structure in NPs, we use a specialized rule. For every noun occurs at the end of a NP, or after a comma or a conjuction, we check whether its left sister is permissible to be a modifier. If so, the modifier will be marked as an adjunct and be put with the noun under a new noun tag.

Algorithm 1: extract conjunction subpart

```
for all subtree in Tree do
  if subtree has a conjunction structure then
    cut the conjunction tree off the subtree
    remove all arguments in the conjunction structure
    copy group information as candidate information
    {This information is for semantic alignment in the
    following step}
    change the empty node'group to empty set
    update the group information of all separated parts
    put subtree, conjunction tree, argument trees into
    agenda
  end if
end for
```

Algorithm 2: extract substituition subpart

```
headgroup ⇐ current_tree'sheadgroup
for all subtree in current_tree do
  if subtree's group is disjoint from
   headgroup then
    cut subtree off currentree
    copy the relative group
    information to candidate
    update all group information
    put substitution mark
  end if
end for
```

LTAG extraction In this section we present the algorithm to extract LTAG trees from the prepro- cessed parallel corpus. The main routine is described in Alg. 3. It calls other three

subroutines Alg. 1,

Alg. 2 and Alg. 4 respectively. Some extracted grammar fragments for the example sentence in Fig. 1 are listed in Fig. 2 and Fig. 3.

Algorithm 3: LTAG extraction algorithm

```
Require: Tree, alignment
for all node in Tree do
    add semantic group information according to alignment
end for
agenda ⇐ [Tree]
well_formed_tree ⇐ []
while agenda is not empty do
    current_tree = agenda.pop()
    if current_tree only contains one group then
        well_formed_tree.append(.current_tree)
    end if
    extract conjunction subpart
    extract adjunction subpart
    if adjunction found then
        continue
    else
        extract substitution subpart
    end if
end while
return well_formed_tree
```

Algorithm 4: extract adjunction subpart

```
for all subtree in Tree do
    for all leftmost child or rightmost child of subtree do
        if child has the same pos as subtree then
            if child is well separated from other parts then
                cut off subtree from its parent
                copy the group information as candidate information
                update the hole's group as empty set
                cut off child from subtree
                copy the group information of child hole as candidate information
                update child hole's group as empty set
                update group information of all subparts
                put adjunction mark
                put all subparts into agenda
            end if
        end if
    end for
end for
```

Automatic Semantic Alignment The last step is to align the semantic variables to the syntax holes in the extracted grammar tree. We use the Alg. 5.

Algorithm 5: Align semantic variables to syntax

```
for all instance in alignment do
    if it also occurs in triples then
        continue
    else
        align the tree which has the same group to it
    end if
end for
for all triple in alignment do
    align the tree which has the same group to it
    for all hole in tree do
        assign the instance which is in the hole candidate set to it
    end for
end for
```

Fig. 2: Grammar fragment 1

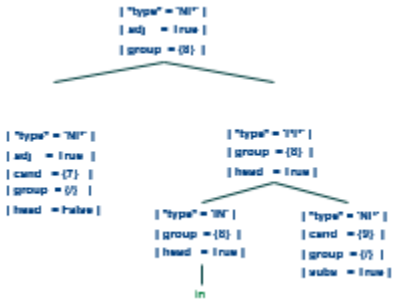

Fig. 3: Grammar fragment 2

Results and Discussion

We evaluate our approach by examining both the precision and the recall. They are defined as in Eq. 1.

$$\text{Precision} = \frac{\#\text{correct grammar trees}}{\#\text{all grammar trees}} \qquad \text{Recall} = \frac{\#\text{semantic triples covered by the grammar}}{\#\text{all semantic triples}} \tag{1}$$

The precision of our approach is 95.96%, and the recall is 79.10%. We can see from the result that the grammar extracted has very high quality. By mannually checking the grammar, we find out that our grammar also has clear linguisitic interpretation which is an advantage over [3] and [4]. However, the recall is relatively low. It's mainly due to that some outputs of the stanford parser are not consistent and contain a lot of errors especially for long sentence. Our extraction algorithm is sensitive to this type of errors and we decide to give up all of the grammar segments for a sentence if we encouter any error in the extraction procedure. In other words, we only keep the grammar segments if the extraction is successful for the whole sentence.

Conclusions and Future Work

In this paper, we proposed a new semi-automatic approach to extract semantic-aligned Lexicalized Tree Adjoining Grammar from a parallel corpus. The experimental result shows that our method achieves high precision but relatively low recall mainly due to the upstream errors from the syntacti- cal parser. Meanwhile, the grammar extracted by us is more similar to that crafted by human experts which has better linguistic interpretation than previous work.

There are still several possibilities that our work can be improved. First of all, the extracted gram- mar can be integrated into a NLG system and evaluated by the output of the NLG system. Secondly, the semantics alignment step in our approach still requires human annotation. This step may be made fully automatic based on the word similarity between the surface form and the semantic triples. Thirdly, our grammar extraction algorithm can be made more robust to upstream parser errors.

References

[1] Eva Banik, Claire Gardent, Donia Scott, Nikhil Dinesh, and Fennie Liang. KBGen-Text Generation from Knowledge Bases as a New Shared Task. *International conference on natural language generation 2012*, 2012.

[2] John Chen and VK Shanker. Automated extraction of TAGs from the Penn Treebank. *New developments in parsing technology*, 2005.

[3] D DeVault, David Traum, and Ron Artstein. Making grammar-based generation easier to deploy in dialogue systems. *Ninth sigdial workshop on discourse and dialogue (sigdial)*, (June):198– 207, 2008.

[4] D DeVault, David Traum, and Ron Artstein. Practical grammar-based NLG from examples.

Proceedings of the fifth international natural language generation conference, 2008.

[5] Arianna D'Ulizia, Fernando Ferri, and Patrizia Grifoni. A survey of grammatical inference methods for natural language learning. *Artificial intelligence review*, 36(1):1–27, 2011.

[6] Aravind K Joshi and Yves Schabes. Tree-adjoining grammars. In, *Handbook of formal languages*, pages 69–123. Springer, 1997.

[7] Dan Klein and Christopher D Manning. Accurate unlexicalized parsing. In *Proceedings of the 41st annual meeting on association for computational linguistics-volume 1*. Association for Computational Linguistics, 2003, pages 423–430.

[8] F Xia. Automatic grammar generation from two different perspectives. PhD thesis. 2001.

[9] Fei Xia. Extracting tree adjoining grammars from bracketed corpora. *Proceedings of the 5th natural language processing pacific rim symposium*, 1999.

[10] Fei Xia and M Palmer. From Treebanks to Tree-Adjoining Grammars. *Supertagging: using complex lexical descriptions in natural language processing*:1–39, 2006.