A Generation Framework for Grammar Development

Shashi Narayan
School of Informatics
University of Edinburgh
shashi.narayan@ed.ac.uk

Claire Gardent
CNRS, LORIA, UMR 7503
Vandoeuvre-lès-Nancy, F-54500, France
claire.gardent@loria.fr

Abstract

We developed a general framework for the development of a symbolic (hand-written) feature-based lexicalised tree-adjoining grammar (FB-LTAG). We choose natural language generation, surface realisation in particular, to question the capabilities of the grammar in terms of both accuracy and robustness. Our framework combines an optimised surface realiser with efficient error mining techniques. While generating from a large data set provided by the Generation Challenge Surface Realisation task, we improve both accuracy and robustness of our grammar significantly.

1 Introduction

We present a framework for the development of a hand-written feature-based lexicalised tree-adjoining grammar (FB-LTAG) for English. We use an XMG (Crabbé et al., 2013) based FB-LTAG (Alahverdzhieva, 2008) in our experiments. Unlike XTAG (The XTAG Research Group, 2001) where each rule is described manually, XMG based grammar uses a compact way of grammar writing using a meta grammar. Only rules in the meta grammar are described manually, these rules are later combined to generate the FB-LTAG grammar. Our grammar consists of roughly 1000 trees with a linguistic coverage similar to that of XTAG.

\begin{example}
\textbf{Input:} Shallow dependency structure
\begin{verbatim}
SROOT 1 0 donate cpos=vbd
SBJ 2 1 Sumitomo bank cpos=nn
OBJ 3 1 $500,000 cpos=cd
\end{verbatim}
\end{example}

\begin{verbatim}
  donate
/
  \_SBJ 2 1 Sumitomo bank cpos=nn
  \_OBJ 3 1 $500,000 cpos=cd
\end{verbatim}

\textbf{Output:} Sumitomo bank donated $500,000.

In the literature, parsing has been proposed for testing symbolic grammars in terms of both accuracy and coverage (Thurmair, 1990; van Noord, 2004; Sagot and de la Clergerie, 2006; de Kok et al., 2009). In this paper, we choose natural language generation (NLG), surface realisation in particular, to question the capabilities of the grammar while generating from a large set of shallow dependency trees (similar to the input shown in Example 1) provided by the Generation Challenge: Surface Realisation Shared task (SR Task, in short) (Belz et al., 2011). This SR benchmark consists of 26,725 sentences varying from minimum length 1 to maximum length 134 with an average length of 22. The purpose behind using this dataset was to check the robustness and the accuracy of the grammar, and also the efficiency of the surface realiser on a large benchmark.

2 Our Framework

Our framework consists of an optimised surface realisation algorithm together with error mining techniques. We describe them briefly in this section.

Structure-driven Lexicalist Generation. Symbolic surface realisation is very prone to the combinatorial problem (Kay, 1996) because of (i) strong lexical ambiguity, (ii) the lack of order information in the input, and (iii) intersective modifiers. We developed an optimised algorithm (TDBU-PAR, (Narayan and Gardent, 2012b)) which combines techniques and ideas from the head-driven (Shieber et al., 1990) and the lexicalist approaches (Espinosa et al., 2010; Carroll and Oepen, 2005; Gardent and Kow, 2005). On the one hand, rule selection is guided, as in the lexicalist approach, by the elementary units present in the input rather than by its structure. On the other hand, the structure of the input is used to provide top-down guidance for the search and thereby re-
strict the combinatorics. To further improve efficiency, the algorithm integrates three additional optimisation techniques: (i) polarity filtering from the lexicalist approach (Bonfante et al., 2004; Gardent and Kow, 2007); (ii) the use of a language model to prune competing intermediate substructures; and (iii) simultaneous rather than sequential parallelised top-down predictions.

Table 1 shows the advantages of the proposed algorithm. We compared our proposed system (TDBU-PAR) with a baseline system (BASELINE, (Narayan, 2011)). We saw that whereas BASELINE times out for longer sentences, the newly proposed system TDBU-PAR remains stable. Our system successfully terminates on all the SR Task input (26,725 sentences) with a coverage a of 38.73% and with a BLEU score of 0.675 for covered sentences.

Table 1: Comparison between generation times (seconds). To make these comparisons possible, average maximum arity of words present in dependency trees is 3.

<table>
<thead>
<tr>
<th>Length range (L)</th>
<th>Sentence</th>
<th>Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>TDBU-PAR</td>
<td>BASELINE</td>
</tr>
<tr>
<td>0−5</td>
<td>Total</td>
<td>1084</td>
</tr>
<tr>
<td></td>
<td>Succ</td>
<td>985</td>
</tr>
<tr>
<td>6−10</td>
<td>Total</td>
<td>2232</td>
</tr>
<tr>
<td></td>
<td>Succ</td>
<td>1477</td>
</tr>
<tr>
<td>11−20</td>
<td>Total</td>
<td>5705</td>
</tr>
<tr>
<td></td>
<td>Succ</td>
<td>520</td>
</tr>
<tr>
<td>(All)</td>
<td>Total</td>
<td>13661</td>
</tr>
<tr>
<td></td>
<td>Succ</td>
<td>2744</td>
</tr>
</tbody>
</table>

Further improvements

While efficient surface realisation and error mining helped improve coverage and precision, they also uncovered two additional sources of errors: (i) a formal error related to how multiple adjunction is represented in FB-TAG (Gardent and Narayan, 2015) and (ii) some errors with generating elliptical sentences (Gardent and Narayan, 2013). Addressing these issues lead to further improvements (Table 2) whereby, on the testset provided by the SR shared task, the final system has a BLEU score of 0.72 and a coverage of 0.81.

3 Conclusion

Efficient surface realisation and focused error mining allowed us to develop a large scale natural language generation system which is efficient, robust and accurate while integrating a symbolic grammar and lexicon. While this system was evaluated on newspaper text, because it relies on a symbolic grammar and lexicon, it should straightforwardly extend to other text genre. Similarly, while it was developed for unordered dependency tree inputs, it could also be of use in NLG applications which take as input tree-shaped data.
References


