Wide Coverage Symbolic Surface Realization

Charles B. Callaway
Istituto per la Ricerca Scientifica e Tecnologica
Istituto Trentino di Cultura, Italy (ITC-irst)
callaway@itc.it

Abstract

Recent evaluation techniques applied to corpus-based systems have been introduced that can predict quantitatively how well surface realizers will generate unseen sentences in isolation. We introduce a similar method for determining the coverage on the Fuf/Surge symbolic surface realizer, report that its coverage and accuracy on the Penn TreeBank is higher than that of a similar statistics-based generator, describe several benefits that can be used in other areas of computational linguistics, and present an updated version of Surge for use in the NLG community.

1 Introduction

Surface realization is the process of converting the semantic and syntactic representation of a sentence or series of sentences into the text, or surface form, of a particular language (Elhadad, 1991; Bateman, 1995). Most surface realizers have been symbolic, grammar-based systems using syntactic linguistic theories like HPSG. These systems were often developed as either proof-of-concept implementations or to support larger end-to-end NLG systems which have produced limited amounts of domain-specific texts.

As such, determining the generic coverage of a language has been substituted by the goal of producing the necessary syntactic coverage for a particular project. As described in (Langkilde-Geary, 2002), the result has been the use of regression testing with hand-picked examples rather than broad evaluations of linguistic competence. Instead, large syntactically annotated corpora such as the Penn TreeBank (Marcus et al., 1993) have allowed statistically based systems to produce large quantities of sentences and then more objectively determine generation coverage with automatic evaluation measures.

We conducted a similar corpus-based experiment (Callaway, 2003) with the FUF/SURGE symbolic surface realizer (Elhadad, 1991). We describe a direct comparison with HALOGEN (Langkilde-Geary, 2002) using Section 23 of the TreeBank, showing that the symbolic approach improves upon the statistical system in both coverage and accuracy. We also present a longitudinal comparison of two versions of FUF/SURGE showing a significant improvement in its coverage and accuracy after new grammar and morphology rules were added. This improved version of SURGE is available for use in the NLG community.

2 Related Work in Wide Coverage Generation

Verifying wide coverage generation depends on (1) a large, well structured corpus, (2) a transformation algorithm that converts annotated sentences into the surface realizer's expected input form, (3) the surface realizer itself, and (4) an automatic metric for determining the accuracy of the generated sentences. Large, well structured, syntactically marked corpora such as the Penn TreeBank (Marcus et al., 1993) can provide a source of example sentences, while automatic metrics like simple string accuracy are capable of giving a fast, rough estimate of quality for individual sentences.

Realization of text from corpora has been approached in several ways. In the case of Ratnaparkhi's generator for flight information in the air travel domain (Ratnaparkhi, 2000), the transformation algorithm is trivial as the generator uses the corpus itself (annotated with semantic information such as destination or flight number) as input to a surface realizer with an n-gram model of the domain, along with a maximum entropy probability model for selecting when to use which phrase.

FERGUS (Bangalore and Rambow, 2000) used the Penn TreeBank as a corpus, requiring a more substantial transformation algorithm since it requires a lexical predicate-argument structure instead of the TreeBank's representation. The system uses an underlying tree-

```
(S (NP-SBJ                    ((cat clause)
       (NP (JJ overall)        (process ((type ascriptive) (tense past)))
           (NNS sales)))       (participants
   (VP (VBD were)                   ((carrier ((cat common) (lex "sale") (number plural)
       (ADJP-PRD                                (describer ((cat adj) (lex "overall")))))
           (RB roughly)                (attribute ((cat ap) (lex "flat")
           (JJ flat))))                            (modifier ((cat adv) (lex "roughly")))))))
```

Figure 1: A Penn TreeBank Sentence and Corresponding SURGE Input Representation

based syntactic model to generate a set of possible candidate realizations, and then chooses the best candidate with a trigram model of the Treebank text. An evaluation of three versions of FERGUS on randomly chosen Wall Street Journal sentences of the TreeBank showed simple string accuracy up to 58.9%.

Finally, Langkilde's work on HALOGEN (Langkilde-Geary, 2002) uses a rewriting algorithm to convert the syntactically annotated sentences from the TreeBank into a semantic input notation via rewrite rules. The system uses the transformed semantic input to create millions of possible realizations (most of which are grammatical but unwieldy) in a lattice structure and then also uses n-grams to select the most probable as its output sentence. Langkilde evaluated the system using the standard train-and-test methodology with Section 23 of the TreeBank as the unseen set.

These systems represent a statistical approach to wide coverage realization, turning to automatic methods to evaluate coverage and quality based on corpus statistics. However, a symbolic realizer can use the same evaluation technique if a method exists to transform the corpus annotation into the realizer's input representation. Thus symbolic realizers can also use the same types of evaluations employed by the parsing and MT communities, allowing for meaningful comparisons of their performance on metrics such as coverage and accuracy.

3 The Penn TreeBank

The Penn TreeBank (Marcus et al., 1993) is a large set of sentences bracketed for syntactic dependency and part of speech, covering almost 5 million words of text. The corpus is divided into 24 sections, with each section having on average 2000 sentences. The representation of an example sentence is shown at the left of Figure 1.

In general, many sentences contained in the TreeBank are not typical of those produced by current NLG systems. For instance, newspaper text requires extensive quoting for conveying dialogue, special formatting for stock reports, and methods for dealing with contractions. These types of constructions are not available in current general purpose, rule-based generators:

- Direct and indirect quotations from reporters' interviews (Callaway, 2001):
  "It's turning out to be a real blockbuster," Mr. Sweig said.
- Incomplete quotations:
  Then retailers "will probably push them out altogether," he says.
- Simple lists of facts from stock reports:
  8 13/16% high, 8 1/2% low, 8 5/8% near closing bid, 8 3/4% offered.
- Both formal and informal language:
  You've either got a chair or you don't.
- A variety of punctuation mixed with text:
  $55,730,000 of school financing bonds, 1989 Series B (1987 resolution).
- Combinations of infrequent syntactic rules:
  Then how should we think about service?
- Irregular and rare words:
  "I was upset with Roger, I fumpered and schmumpered," says Mr. Peters.

By adding rules for these phenomena, NLG realizers can significantly increase their coverage. For instance, approximately 15% of Penn TreeBank sentences contain either direct, indirect or incomplete written dialogue. Thus for a newspaper domain, excluding dialogue from the grammar greatly limits potential coverage. Furthermore, using a corpus for testing a surface realizer is akin to having a very large regression test set, with the added benefit of being able to robustly generate real-world sentences.

In order to compare a symbolic surface realizer with its statistical counterparts, we tested an enhanced version of an off-the-shelf symbolic generation system, the FUF/SURGE (Elhadad, 1991) surface realizer. To obtain a meaningful comparison, we utilized the same approach as

Realizer	Sentences	Coverage	Matches	Covered Matches	Total Matches	Accuracy
SURGE 2.2	2416	48.1%	102	8.8%	4.2%	0.8542
SURGE+	2416	98.9%	1474	61.7%	61.0%	0.9483
HALOGEN	2416	83.3%	1157	57.5%	47.9%	0.9450

Table 1: Comparing two SURGE versions with HALOGEN [Langkilde 2002].

HALOGEN, treating Section 23 of the Treebank as an unseen test set. We created an analogous transformation algorithm (Callaway, 2003) to convert TreeBank sentences into the SURGE representation (Figure 1), which are then given to the symbolic surface realizer, allowing us to measure both coverage and accuracy.

4 Coverage and Accuracy Evaluation

Of the three statistical systems presented above, only (Langkilde-Geary, 2002) used a standard, recoverable method for replicating the generation experiment. Because of the sheer number of sentences (2416), and to enable a direct comparison with HALOGEN, we similarly used the simple string accuracy (Doddington, 2002), where the smallest number of Adds, Deletions, and Insertions were used to calculate accuracy: 1 - (A + D + I) / #Characters.

Unlike typical statistical and machine learning experiments, the grammar was "trained" by hand, though the evaluation of the resulting sentences was performed automatically. This resulted in numerous generalized syntactic and morphology rules being added to the SURGE grammar, as well as specialized rules pertaining to specific domain elements from the texts.

Table 1 shows a comparative coverage and accuracy analysis of three surface realizers on Section 23 of the Penn TreeBank: the original SURGE 2.2 distribution, our modified version of SURGE, and the HALOGEN system described in (Langkilde-Geary, 2002). The surface realizers are measured in terms of:

- *Coverage*: The number of sentences for which the realizer returned a recognizable string rather than failure or an error.
- *Matches*: The number of identical sentences (including punctuation/capitals).
- *Percent of covered matches*: How often the realizer returned a sentence match given that a sentence is produced.
- *Percent of matches for all sentences*: A measure of matches from all inputs, which penalizes systems that improve accuracy at the expense of coverage (Matches / 2416, or Coverage * Covered Matches).
- *Accuracy*: The aggregate simple string accuracy score for all covered sentences (as opposed to the entire sentence set).

The first thing to note is the drastic improvement between the two versions of SURGE. As the analysis in Section 3 showed, studying the elements of a particular domain are very important in determining what parts of a grammar should be improved. For instance, the TreeBank contains many constructions which are not handled by SURGE 2.2, such as quotations, which account for 15% of the sentences. When SURGE 2.2 encounters a quotation, it fails to produce a text string, accounting for a large chunk of the sentences not covered (51.9% compared to 1.1% for our enhanced version of SURGE).

Additionally, a number of morphology enhancements, such as contractions and punctuation placement contributed to the much higher percentage of exact matches. While some of these are domain-specific, many are broader generalizations which although useful, were not included in the original grammar because they were not encountered in previous domains or arose only in complex sentences.

On all four measures the enhanced version of SURGE performed much better than the statistical approach to surface realization embodied in HALOGEN. The accuracy measure is especially surprising given that statistical and machine learning approaches employ maximization algorithms to ensure that grammar rules are chosen to get the highest possible accuracy. However, given that the difference in accuracy from Surge 2.2 is relatively small while its quality is obviously poor, using such accuracy measures alone is a bad way to compare surface realizers.

Finally, the coverage difference between the enhanced version of SURGE and that of HALOGEN is especially striking. Some explanations may be that statistical systems are not yet capable of handling certain linguistic phenomena like long-distance dependencies (due to n-gram ap-

proaches), or given that statistical systems are typically robust and very unlikely to produce no output, that there were problems in the transformation algorithm that converted individual sentence representations from the corpus.

5 Additional Benefits

The evaluation approach presented here has other advantages besides calculating the coverage and accuracy of a grammar. For instance, in realizers where linguists must add new lexical resources by hand, such a system allows them to generate text by first creating sample sentences in the more familiar TreeBank notation. Sentences could also be directly generated by feeding an example text to a parser capable of producing TreeBank structures. This would be especially useful in new domains to quickly see what new specialized syntax they might need.

Additionally, the transformation program can be used as an error-checker to assist in annotating sentences in a new corpus. Rules could be (and have been) added alongside the normal transformation rules that detect when errors are encountered, categorize them, and make them available to the corpus creator for correction. This can extend beyond the syntax level, detecting even morphology errors such as incorrect verbs, typos, or dialect differences.

Finally, such an approach can help test parsing systems without the need for the time-consuming process of annotating corpora in the first place. If a parser creates a TreeBank representation for a sentence, the generation system can then attempt to regenerate that same sentence automatically. Exact matches are highly likely to have been correctly parsed, and more time can be spent locating and resolving parses that returned very low accuracy scores.

6 Conclusions and Future Work

Recent statistical systems for generation have focused on surface realizers, offering robustness, wide coverage, and domain- and language-independence given certain resources. This paper represents the analogous effort for a symbolic generation system using an enhanced version of the FUF/SURGE systemic realizer. We presented a grammatical coverage and accuracy experiment showing the symbolic system had a much higher level of coverage of English and better accuracy as represented by the Penn TreeBank. The improved SURGE grammar, version 2.4, will be made freely available to the NLG community.

While we feel that both coverage and accuracy could be improved even more, additional gains would not imply a substantial improvement in the quality of the grammar itself. The reason is that most problems affecting accuracy lie in transforming the TreeBank representation as opposed to the grammar, which has remained relatively stable.

References

S. Bangalore and O. Rambow. 2000. Exploiting a probabilistic hierarchical model for generation. In *COLING–2000: Proceedings of the 18th International Conference on Computational Linguistics*, Saarbruecken, Germany.

John A. Bateman. 1995. KPML: The KOMET-penman (multilingual) development environment. Technical Report Release 0.8, Institut für Integrierte Publikations- und Informationssysteme (IPSI), GMD, Darmstadt.

Charles Callaway. 2001. A computational feature analysis for multilingual character-to-character dialogue. In *Proceedings of the Second International Conference on Intelligent Text Processing and Computational Linguistics*, pages 251–264, Mexico City, Mexico.

Charles B. Callaway. 2003. Evaluating coverage for large symbolic NLG grammars. In *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence*, pages 811–817, Acapulco, Mexico, August.

George Doddington. 2002. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings of the 2002 Conference on Human Language Technology*, San Diego, CA, March.

Michael Elhadad. 1991. FUF: The universal unifier user manual version 5.0. Technical Report CUCS-038-91, Dept. of Computer Science, Columbia University.

Irene Langkilde-Geary. 2002. An empirical verification of coverage and correctness for a general-purpose sentence generator. In *Second International Natural Language Generation Conference*, Harriman, NY, July.

M. Marcus, B. Santorini, and M. Marcinkiewicz. 1993. Building a large annotated corpus of English: The PennTreeBank. *Computational Linguistics*, 26(2).

Adwait Ratnaparkhi. 2000. Trainable methods for surface natural language generation. In *Proceedings of the First North American Conference of the ACL*, Seattle, WA, May.