Unsupervised Psycholinguistics: Using Text-to-Text Realization to Model Language Production

Jeremy Cole
College of IST
Pennsylvania State University
University Park, PA 16802
jrcole@psu.edu

David Reitter
College of IST
Pennsylvania State University
University Park, PA 16802
reitter@psu.edu

Abstract

This paper covers the beginning of a system implemented to pursue psycholinguistic experiments on large unannotated datasets. The system's goal is to attempt to analyze language production with a cognitive modeling approach. This paper will discuss progress with the system so far, current setbacks, and directions forward.

1 Introduction

Text-to-Text Realization (TTR) has impressively demonstrated that the linguistic toolchain and a set of representations have reached the point where they can analyze the broad range of sentences found in corpora, and that they can recover sentences in generated output. While these tools are interesting to certain applications, they do not presently allow us to systematically validate the underlying components in terms of their psychological and linguistic plausibility. While systems that rely more explicitly on cognitive architectures and deep linguistic encoding have made great progress past simple example sentences, they have yet to reach the coverage of TTR.

The goal of the system is to unify these two paradigms, allowing for statements of cognitive plausibility to be tested on large datasets. The current version of the system built on White's realization algorithm (White, 2006) using the principles of ACT-R activation (Anderson, 1983). Below, we will discuss the current toolchain, what we've learned, and possible directions forward.

2 Parsing

The first step of TTR is to take sentences in natural language and transform them into a semantic parse. The goal of this step is to produce something that captures the meaning of intended statement, without capturing any syntactic information. These parses form the input structures for the realizer. Since the realizer is not given the syntax, it can make all of the syntactic choices itself. In this way, we can examine how different versions of the realizer more or less accurately capture the same syntactic choices that a human would make in that situation.

However, semantic representation is an open question: presumably, a good method would resemble the way ideas are combined to form sentences in thought, which is difficult to observe. White chose to use Hybrid-Logic Dependency Semantics (HLDS), which Baldridge et al make the case for (Kruijff and Kruijff-Korbayova, 2001). The case is also made that HLDS can easily be coupled with a grammar formalism (Baldridge and Kruijff, 2002). For the purpose of TTR, this might be completely fine; however, this introduces the syntax that we were trying to avoid. The exact parser White uses, which is bundled with the OpenCCG toolkit, is not purely semantic. Sentences which mean the same thing can still have very different parses. See Table 1 for examples.

This is a serious problem from a principled approach, as it makes it difficult to test even classical example sentences that were handled by earlier cognitive models (Reitter et al., 2011). However, there are several groups working on semantic parsing who have made parsers more recently than the one packaged with White's realization algorithm.

3 Realization

The next step of TTR is to take the parses and transform it back into sentences. For obvious reasons, the input to realizer cannot be too specified to the initial sentence, or nothing was accomplished.

The realizer, as it is, works in a fairly simple way. It has a scoring mechanism for partial realizations and a system for generating them. The scoring mechanism is the portion we modified:

Sentence	Parse
I threw the red ball to my black dog.	@w1((throw.01 ^ <mood>dcl ^ <tense> past ^ <Arg0>(w0:ORGANIZATION ^ I) ^ <Arg1>(w4 ^ ball ^ <num>sg ^ <Det>(w2 ^ the) ^ <Mod>(w3 ^ red) ^ <Mod>(w5 ^ to ^ <Arg1>(w8 ^ dog ^ <num>sg ^ <Det>(w6 ^ my) ^ <Mod>(w7 ^ black))))))
I threw my black dog the red ball.	@w1(throw.01 ^ <mood>dcl ^ <tense> past ^ <Arg0>(w0:ORGANIZATION ^ I) ^ <Arg1>(w4 ^ dog ^ <num> sg ^ <Det>(w2 ^ my) ^ <Mod>(w3 ^ black) ^ <Mod>(w5 ^ the ^ <Arg1>(w7 ^ ball ^ <num> sg ^ <Mod>(w6 ^ red)))))
I mailed the examples to my adviser.	@w1(mail.01 ^ <mood>dcl ^ <tense>past ^ <Arg0>(w0:ORGANIZATION ^ I) ^ <Arg1>(w3 ^ example ^ <num> pl ^ <Det>(w2 ^ the) ^ <Mod>(w4 ^ to ^ <Arg1>(w6 ^ adviser ^ <num>sg ^ <Det>(w5 ^ my)))))
The examples were mailed to my adviser by me.	@w2(PASS ^ <mood>dcl ^ <tense>past ^ <Arg0>(w1 ^ example ^ <num>pl ^ <Det>(w0 ^ the)) ^ <Arg1>(w3 ^ mail ^ <tense>past ^ <Arg0>(w4 ^ to ^ <Arg0>(w6 ^ adviser ^ <num>sg ^ <Det>(w5 ^ my) ^ <Mod>(w7 ^ by ^ <Arg1>(w8 ^ me)))) ^ <Arg1>w1))

Table 1: The first two sentences illustrate the difference between Prepositional Object and Double Object, while the second two illustrate the difference between Passive and Active voice.

we replaced the ngram scorer with an estimate for how activated the phrase would be under ACT-R principles.

However, after careful study, we began to believe that perhaps the input is too specified for our purposes. While we wanted to do away with syntax in the input completely to allow the realizer to make the decision, it currently receives and uses the parts of speech of each token in the semantics. Furthermore, it generally considers the best realization to be one where it has replicated the parts of speech from the original sentence.

Unfortunately, this methodology leads the scoring to be insensitive to change. Even choosing the scores randomly does pretty well, and optimizing a model is almost impossible because the models will not be substantially different, as the realizer still wants to create the same syntactic structure.

Furthermore, because the parts of speech are known, the model will not substantially vary its syntax. The ideal model would sometimes choose semantically-equivalent but syntactically-different variants of the same sentence based on the context and probability. However, the current model will generally only vary morphology.

For a way forward, the model will have to learn how to make syntactic choices on its own. The first step is a parse that's divorced from syntax; the second step is to recreate the syntax from scratch. To our knowledge, making syntactic choices for any arbitrary semantics is an unsolved problem. We have some ideas for methods to solve it, including using ACT-R to associate syntactic rules with lexical combinations.

4 Evaluation

The last piece of the system is a method to evaluate the actual models. For instance, if two models generate different sentences, how do we claim one sentence is better than the other, or at least more human?

Originally, we relied on ROUGE (Lin, 2004). ROUGE is a metric that has been used to judge semantic equivalence in a variety of applications, such as summarization. The goal of ROUGE is to estimate *semantic* equivalence: does the automatically generated summary have the same meaning as the one created manually? However, we are not interested in semantic equivalence: presumably, a valid realizer would always choose sentences that have the same semantics. What we truly wish to investigate is exact equivalence, so that we can look at the similarity in syntax.

A simple method is to use edit distance, a family of metrics for comparing two vectors. Levenshtein distance, a common variant of edit distance, has often been applied to spell-checking, though in this case, we look at vectors of words in sentences, instead of vectors of characters in words. Since edit distance is normally applied to compare two things exactly, it seems a good choice to go beyond semantic equivalence.

5 Conclusion

In our time studying linguistic realizations, we have made progress toward our goal, despite limitations in available tools. We have developed a path forward to overcome obstacles in parsing, choosing syntactic rules, internal scoring, and model evaluation. The experiments ran so far show some promise in our ability to investigate psycholinguistic phenomenon: several variables relevant to the production process have been identified and isolated. However, until the modifications discussed in this report are made, a valid study of these variables is difficult, if not impossible. We have identified possible solutions to these problems that we will pursue in the coming months.

References

John Anderson. 1983. A Spreading Activation Theory of Memory. *Journal of Verbal Learning and Verbal Behavior*.

Jason Baldridge and Geert-Jan M. Kruijff. 2002. Coupling CCG and hybrid logic dependency semantics. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 319–326. Association for Computational Linguistics.

Geert-JanM. Kruijff and Ivana Kruijff-Korbayova. 2001. A hybrid logic formalization of information structure sensitive discourse interpretation. In Vclav Matousek, Pavel Mautner, Roman Moucek, and Karel Tauser, editors, *Text, Speech and Dialogue*, volume 2166 of *Lecture Notes in Computer Science*, pages 31–38. Springer Berlin Heidelberg.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, pages 74–81.

David Reitter, Frank Keller, and Johanna D. Moore. 2011. A computational cognitive model of syntactic priming. *Cognitive Science*, 35(4):587–637.

Michael White. 2006. Efficient Realization of Coordinate Structures in Combinatory Categorial Grammar. *Research on Language and Computation*, 4(1):39–75, May.