The Creation of a Large-Scale LFG-Based Gold Parsebank

Alexis Baird, Christopher R. Walker
Powerset/Microsoft
475 Brannan St., Suite 330
San Francisco, CA, USA
E-mail: {albaird, chriwalk}@microsoft.com

Abstract

Systems for syntactically parsing sentences have long been recognized as a priority in Natural Language Processing. Statistics-based systems require large amounts of high quality syntactically parsed data. Using the XLE toolkit developed at PARC and the LFG Parsebanker interface developed at Bergen, the Parsebank Project at Powerset has generated a rapidly increasing volume of syntactically parsed data. By using these tools, we are able to leverage the LFG framework to provide richer analyses via both constituent (c-) and functional (f-) structures. Additionally, the Parsebanking Project uses source data from Wikipedia rather than source data limited to a specific genre, such as the Wall Street Journal. This paper outlines the process we used in creating a large-scale LFG-Based Parsebank to address many of the shortcomings of previously-created parse banks such as the Penn Treebank. While the Parsebank corpus is still in progress, preliminary results using the data in a variety of contexts already show promise.

1. Introduction

Syntactic parsing systems have long been recognized as a priority in Natural Language Processing. In order to extract meaning from sentences, it is necessary to accurately identify the deep syntactic structure of the sentence. Indeed, the meaning of a linguistic expression is a function of the meaning of its parts *and* the manner in which they are assembled.

Rule-based grammatical systems can generate sufficient potential ambiguity that statistical approaches cannot be avoided—even if they are only applied to the problems of parse-selection and parse-ranking.

The creation of a well-behaved statistical parsing system requires large amounts of syntactically annotated data. Currently, most systems have been tuned to a single English parse corpus: the Penn Treebank (PTB) (Marcus et al., 1993). But PTB has some restrictive properties that make it difficult to apply to novel domains or more sophisticated applications. Most notably, PTB is restricted largely to data drawn from the Wall Street Journal (WSJ), which has a very narrow range of topics (and word senses). Also, PTB parses are typically shallow, sacrificing syntactic complexity in favor of annotator consistency. While PTB is certainly a suitable corpus for many needs, our goals for parser development required a less domain-restricted corpus with more detailed “deep-parse” annotations—and we believe that an active data creation program is the only route to adequate data, in terms of both volume and needs-tailoring.

In late 2008, the Parsebanking (PB) project grew out of this belief. The PB project was based on the theoretical foundations of Lexical-Functional Grammar (LFG) and was intended to provide high-quality syntactically annotated sentences with both constituent (c-) and functional (f-) structures (Kaplan et al., 1995). The project has yielded over 100,000 fully parsed sentences since active annotation began in March 2009. Our current annotation rate, which is still accelerating, is roughly 2000 sentences / week.

This paper documents the Best Practices of the project and reports on its progress to date. We will pay particularly close attention to tools, sampling practices, annotation practices, data quality, and data formatting.

2. Data

The source data is comprised of parsed sentences randomly selected from Wikipedia. Although Wikipedia presents some genre problems of its own, we believe it to be a more flexible source than The WSJ. However, since we were targeting parser improvements on Wikipedia, this belief was not a factor in our choice of sources.

Each sentence is run through our multi-stage NLP pipeline, resulting in a packed parse for the sentence (stored as a prolog file). These packed parses represent the entire choice space of possible parses via different combinations of lexical, morphological, and syntactic features. Any sentence that receives either a fragment parse or more than 200 possible parses is discarded.

The packed parse data is broken up into pairs of banks, each containing 500 sentences. Each bank has 200 sentences in common with its pair-mate, allowing for dual annotation and annotator agreement metrics. These dually annotated sentences are randomly dispersed throughout the banks, ensuring blindness.

The input format for the annotation tool is a prolog file generated by the LFG-Based XLE parser[1]—containing a packed representation of the choice space for all possible parses. The output format is a disambiguated prolog file that contains only the parse(s) chosen by the annotator.

Each completed bank is a collection of 500 prolog files, accompanied by lists indicating which parses have received which quality rating (eg. “GOLD”, “NO GOOD”, etc.).

3. Annotation Process

3.1 Tool

The tool used for this project was developed at the University of Bergen.[2] It is a web interface that annotators can access remotely.

Rather than having annotators build c- and f-structures from scratch, the Bergen tool uses a series of decision points, referred to as *discriminants*. Any given discriminant can induce a binary partition on the choice space. The selection of a discriminant (or its complement) amounts to the selection of one of the two partition elements—reducing the choice space accordingly.

Annotators are presented with a sentence and all the parses identified by the parser. There may be anywhere from 2 to 200 parses. When there are fewer than 32 possible parses, the tool will display all of the possible c- and f-structures. When there are more than 32 parses, a more limited view of the choice space is provided along with all of the discriminants. See *Figure 1*.

When an annotator selects a discriminant, parses not consistent with that selection are removed from the choice space (and suppressed in the display). Discriminants offer annotators binary choices that are based on lexical, morphological, and syntactic features capable of efficiently partitioning the choice space. The syntactic features are broken into c-structure choices (e.g. “Does constituent X attach to constituent Y?”) and f-structure choices (e.g. “Is this constituent an adjunct of this verb?”, “Is this constituent an oblique of this verb?”). Discriminants are not completely independent. Some discriminants are redundant and others eliminate dependent discriminants when selected.

Some sentences will have more than one “correct parse” (eg. “She saw the man with the telescope.”); others will have none. Once a sentence has been disambiguated as much as possible, annotators rate the remaining parse(s) as *GOLD*, *NO GOOD*, or *OK*.

1. NO GOOD: the correct parse was not among the choices
2. GOLD: perfect parse(s)
3. OK: parse(s) with only minor mistakes.

For all “NO GOOD” sentences, annotators must write a comment indicating the type of error the parser made.

3.2 Annotation

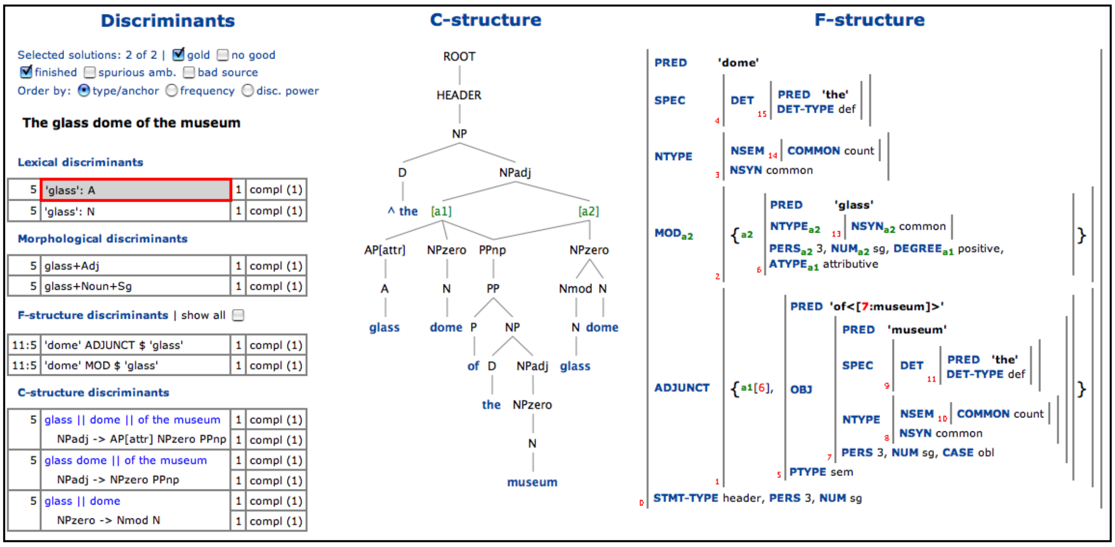
Our annotation team is made up of twelve annotators, all of whom have not just a linguistics background but also a strong syntax background, if not a specialization in LFG. All annotators work remotely, and the team relies heavily on email correspondence and a weekly teleconference as means of maintaining communication and by extension boosting inter-annotator agreement (IAA).

Our scoring mechanism matches every node/edge pair in the f-structure for two independently produced parses of a given sentence. Depending on the importance of the f-structure relation to the downstream semantics, we then weight the feature accordingly. For example, differences in OBJECT labels receive much higher weight than differences in a ±HUMAN label. The feature weights are based on the needs of our semantics team who use the parses as input to their system. While this scoring method has the result of scoring even f-structures in which there was no choice for the annotators, we believe this still gives us a substantial signal from which to measure IAA. Throughout the course of the project, we have seen our IAA numbers increase from 97.5 (unweighted)/96.2 (weighted) before plateauing within the past two months at around 98.7 (unweighted)/97.5 (weighted).

Currently about 60% of all sentences are marked “GOLD”, while 25-30% are marked “NO GOOD”. We have seen each of these numbers increase by about 5-10% over time—a phenomenon that we believe to be a product of increased annotator confidence (in the past annotators often used “OK” when they were unsure about a parse).

[1] See XLE documentation from PARC: http://www2.parc.com/isl/groups/nltt/xle/

[2] LFG Parsebanker Interface Wiki can be found at: http://maximos.aksis.uib.no/Aksis-wiki/LFG_Parsebanker_Interface



Discriminants

Selected solutions: 2 of 2 | ☑ gold □ no good
☑ finished □ spurious amb. □ bad source
Order by: ● type/anchor ○ frequency ○ disc. power

The glass dome of the museum

Lexical discriminants

5	'glass': A	1	compl (1)
5	'glass': N	1	compl (1)

Morphological discriminants

5	glass+Adj	1	compl (1)
5	glass+Noun+Sg	1	compl (1)

F-structure discriminants | show all □

11:5	'dome' ADJUNCT $ 'glass'	1	compl (1)
11:5	'dome' MOD $ 'glass'	1	compl (1)

C-structure discriminants

5	glass || dome || of the museum	1	compl (1)
	NPadj -> AP[attr] NPzero PPnp	1	compl (1)
5	glass dome || of the museum	1	compl (1)
	NPadj -> NPzero PPnp	1	compl (1)
5	glass || dome	1	compl (1)
	NPzero -> Nmod N	1	compl (1)

Figure 1: Screenshot of the LFG Parsebanker Interface

4. “Silver Standard”

While the data we have created is accurate, the design of the annotation task results in a gap in the data coverage—hence the use of “silver standard” rather than “gold standard”. Since annotators are only presented with the output of the parser, any sentences that do not have the correct parse in their original choice space will not be represented in the corpus. While this method still gives us good precision numbers for a subset of all possible sentences, it potentially impacts system training and development.

One alternative would be to restructure the annotation completely such that annotators build c- and f-structures from scratch. However, this would not only create an overwhelming task for annotators but also negatively impact inter-annotator agreement—introducing inconsistencies into the final corpus and severely reducing throughput rates. Since the parses for even simple sentences are hugely complicated, breaking the task into a series of binary choices creates a much more streamlined annotation process. We believe the enormous acceleration of data creation as well as an increase in inter-annotator agreement metrics is worth the potential gaps in corpus coverage. Other work done in named-entity annotation (Ganchev et al., 2007) and automatic-content extraction (Medero et al., 2006) have shown the effectiveness of using decision points over more free-form annotation.

Iterative failure analysis should close the gaps over time. Allowing annotators to edit trees directly (i.e. without grammatical constraints) might offer a mechanism to close coverage gaps more quickly, but it’s not clear that this is an appropriate role for annotators. We feel it is better to leave grammar development to grammar engineers—annotators operate better within well-defined constraints. In order to jump-start the feedback loop, we’ve asked annotators to write standardized comments on all *NO GOOD* sentences, with a description of the changes necessary to fix the parse.

5. Advantages over PTB

Unlike the Penn Treebank, our Parsebank project offers enhanced functional syntactic information that can be passed to a semantic system. For example, while PTB can only distinguish whether a given prepositional phrase attaches to a an adjacent verb, our corpus goes on to distinguish what function the prepositional phrase serves (adjunct, oblique, oblique agent, etc.), as well as more nuanced features such as tense, number, aspect, etc.

Furthermore, the size of our corpus makes it an extremely valuable resource for training, evaluating and testing parsing systems. Already, we have produced over 100,000 sentences, which rivals or exceeds the size of the WSJ part of the PTB.

An active annotation program means that we can build corpora that are narrowly tailored to our specific needs. We have already applied the data-creation process to a number of “specialty” samples, such as search-engine queries or sentences leading-off a document. Once the infrastructure is in place for improvements in parse-ranking, we will also be able to target datasets based upon the active learning requirements of the models.

Finally, our source data relies primarily on a random sample of sentences from Wikipedia and therefore is much less domain-restricted. Without such a domain-restriction in the training and evaluation corpus, systems can be developed to be more flexible.

6. Applications

Discriminant-driven parsebanking allows for the extraction of derivative data supporting, among other things:

1. Parse Ranking
2. Parser/Parse-Ranker Evaluation
3. POS Tagging
4. Shallow (“Chunk”) Parsing

The infrastructure for the majority of intended applications is still under development, so it is difficult to make any concrete assessment of the utility of the data we have created. Efforts are already underway to use the Parsebanking Project in parse-ranking training and evaluation. Early experiments are strongly indicative of significant improvement.

In the meantime, we have begun to seek out shallower applications that can be supported with data extracted from our gold parse. The percentage “GOLD” and percentage “NO GOOD” stats provided above illustrate how the annotation process itself can be leveraged to support a small set of terse evaluation metrics.

We have had early successes with the extraction of POS-tagged data (in PTB format) for use in training and testing independent POS taggers.

It is important to note that the success of the Parsebank project was driven less by some inherent advantage of LFG than by the availability of a reasonably good toolkit supporting automatic pre-parsing of the dataset. That said, the richness of the LFG representation makes LFG → X conversions viable. While LFG parsebanking supports the extraction of the shallower PTB-style trees, the reverse is not true.

Already the efforts of the Parsebanking project have yielded promising results and while the PTB offered ground-breaking and valuable data to the field at the time of its creation, the Parsebanking Project has the potential to build further on that contribution.

7. References

Ganchev, Kuzman, Fernando Pereira, Mark Mandel, Steven Carroll, and Peter White. (2007). Semi-Automated Named Entity Annotation. In *Proceedings of the Linguistic Annotation Workshop*, pp.53-56.

Kaplan, Ronald M. and Joan Bresnan. (1995). Lexical- Functional Grammar: A Formal System for Grammatical Representation. *Formal Issues in Lexical-Functional Grammar*, ed. Mary Dalrymple, Ronald M. Kaplan, John T. Maxwell III, Annie Zaenen, pp. 29-130. CSLI, Stanford.

Linguistic Data Consortium. (1999). Catalog #: LDC99T42; http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC99T42.

LFG Parsebanker Interface Wiki. http://maximos.aksis.uib.no/Aksis-wiki/LFG_Parsebanker_Interface

Marcus, M. P., Santorini, B. and Marcinkiewicz, M. A. (1993). Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics*, Vol. 19, No. 2, pp. 313-330.

Medero, Julie, Kazuaki Maeda, Stephanie Strassel and Christopher Walker. (2006). An Efficient Approach for Gold-Standard Annotation: Decision Points for Complex Tasks, *LREC: Fifth International Conference on Language Resources and Evaluation.*

Riezler, Stefan, Tracy H. King, Ronald M. Kaplan, Richard Crouch, John T. Maxwell, and Mark Johnson. (2002). Parsing the Wall Street Journal using a Lexical-Functional Grammar and Discriminative Estimation Techniques.. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL'02*), Philadephia, PA.

XLE Documentation from PARC: http://www2.parc.com/isl/groups/nltt/xle/