A Look At Parsing and Its Applications [*]

Matthew Lease, Eugene Charniak, Mark Johnson, and David McClosky
Brown Laboratory for Linguistic Information Processing (BLLIP)
Brown University
{mlease,ec,mj,dmcc}@cs.brown.edu

Abstract

This paper provides a brief introduction to recent work in statistical parsing and its applications. We highlight successes to date, remaining challenges, and promising future work.

Introduction

The study of syntax in linguistics seeks to uncover the underlying recursive structure governing human language usage – the infinite ways words combine to form phrases, and how those phrases in turn combine to eventually form sentences. In addition to explaining what strings do and do not constitute valid uses of language, syntax also plays a more significant role: most syntactic theories define a systematic relationship between syntactic structure and meaning, and at minimum it seems we must know which words modify which other words to get the correct interpretation. For example, in *Jane saw the man with the binoculars*, alternative readings vary whether the prepositional phrase *with the binoculars* modifies the verb (i.e. the binoculars are being used to see) or the noun after it (i.e. the man has the binoculars). Consequently, accurate recovery of syntax (i.e. parsing) is widely viewed as a necessary precursor to building systems capable of understanding natural language.

Most work in statistical parsing has centered on the Penn Treebank (PTB), a collection of about two million words from newspaper text and telephone conversations manually annotated for syntax (Marcus *et al.* 1993). This resource provides a valuable testbed for developing and evaluating statistical methods for parsing, and the best performing system to date achieves about 92% f-measure[1] in matching manually annotated syntax (McClosky, Charniak, & Johnson 2006). While the holy grail of fully understanding language remains largely elusive, parsing has in the meantime been usefully applied to a variety of tasks in natural language processing (NLP).

This paper briefly describes our statistical parsing model before going on to highlight a few successes to date in applying parsing to various NLP tasks. The goal of our presentation is to improve community familiarity with parsing and how it can be usefully applied in NLP, and we welcome readers to download our parser[2] for use in their own work.

The Parsing Model

This section describes our parsing model. The first stage consists of a lexicalized probabilistic context-free grammar (PCFG) (Collins 1997; Charniak 2000). Parsing accuracy has been subsequently improved by adding a second-stage, maximum-entropy model to rerank candidates produced by the PCFG (Charniak & Johnson 2005). Most recently, use of semi-supervised learning has further improved parsing accuracy (McClosky, Charniak, & Johnson 2006).

The PCFG

The lexicalized PCFG (Charniak 2000) uses maximum likelihood to estimate probabilities for the various syntactic rules seen in the training data. Lexicalized conditioning enables the parser to learn, for example, that whereas *give* commonly takes two arguments (e.g. *give a dog a bone*), the verb *donate* rarely if ever occurs in such a context. Use of lexicalization has led to significantly improved performance over early non-lexical models (Charniak 1997).

A parse π consists of a hierarchy of syntactic *constituents* in which we assume each constituent has a single *head* word most representative of the entire constituent's syntactic function. Our model can be understood as a top-down process of generating for each constituent c first the head word's part-of-speech (POS) tag t, then the head word h itself, and finally c's expansion e into sub-constituents given its label l (e.g. is it a noun or verb phrase) and relevant history H (information outside c deemed to be important). Thus we have

$$p(\pi) = \prod_{c \in \pi} p(t \mid l, H) \cdot p(h \mid t, l, H) \cdot p(e \mid l, t, h, H)$$

where c is an implicit subscript of all terms. All distributions here are heavily backed off and smoothed using Chen's method (1996) and lightly pruned to remove unhelpful statistics. To find the most likely parse for a given sentence s, one just selects the π that maximizes the conditional probability of π given s. On the Wall Street Journal (WSJ) in PTB, the PCFG achieves about 90% f-measure in matching (hand-annotated) labelled constituent structure.

[*] We thank the support of NSF grants 0121285, LIS9720368, and IIS0095940, and DARPA GALE contract HR0011-06-2-0001.

[1] The harmonic mean of precision and recall: $\frac{2PR}{P+R}$.

[2] ftp://ftp.cs.brown.edu/pub/nlparser

The Reranker

Using the efficient k-best parsing algorithm described in (Jimenez & Marzal 2000; Huang & Chang 2005), the PCFG is modified to propose a set of a k parse candidates instead of the single mostly likely analysis. A *reranker* (Collins 2000; Charniak & Johnson 2005) then selects from this set $T = \{t_1, \dots t_k\}$ the parse $t^\star \in T$ with the highest f-measure (in comparison to a hand-annotated reference). This reranking paradigm is useful in allowing us to incorporate other interesting syntactic relationships that are more difficult to express in the PCFG's generative model.

A feature-extractor converts each candidate parse $t \in T$ into a vector of real-valued features $f(t) = (f_1(t), \dots, f_m(t))$ (e.g., the value $f_j(t)$ of the feature f_j might be the number of times a certain syntactic structure appears in t). The reranker training procedure associates each feature f_j with a real-valued weight λ_j, and $\lambda' \cdot f(t)$ (the dot product of the feature vector and the weight vector λ) is a single scalar weight for each parse candidate. The reranker employs a maximum-entropy estimator that selects the λ that minimizes the log loss of the highest f-measure parse $t^\star$ conditioned on T (together with a Gaussian regularizer to prevent over-fitting). Informally, λ is chosen to make high f-measure parses as likely as possible under the (conditional) distribution defined by f and λ.

In terms of performance, the reranker achieves about 91% f-measure in matching hand-annotated syntax on WSJ.

Semi-Supervised Learning

In recent work (McClosky, Charniak, & Johnson 2006), the reranking-parser is used to automatically parse the North American News Corpus[3] which consists of about 24 million sentences drawn from various news sources (sentence boundaries were induced by a simple discriminative model). For each sentence, the most likely parse produced by the reranking-parser was then combined with the original training data (with weighting) and used to retrain. This improved parsing accuracy on WSJ to just over 92%.

Genre Portability

As might be expected, a parser trained on one genre of text (e.g. WSJ) tends to perform less accurately when evaluated on a different genre. Since we cannot expect millions of words to be manually annotated as training data in every genre of interest, there has been significant work in studying how parsers can be made more portable across genres (Sekine 1997; Gildea 2001; Steedman *et al.* 2003). Somewhat surprisingly, the semi-supervised model just discussed is *not* over-trained on the newspaper genre, but in fact achieves significantly improved parsing accuracy across genres[4], making parsing both more accurate and more portable than ever.

Language Modeling

This section introduces the language model (LM), which assigns to a sequence of words a probability estimating the likelihood of the sequence's occurrence in practice (relative to all other possible sequences). We describe how a parsing language model operates and compare performance to n-gram methods. Next, we introduce the noisy-channel paradigm and discuss its use in NLP. Finally, we cite several noisy-channel type tasks in which improved performance has been achieved by using a parser LM instead of n-grams.

	Perplexity		
Model	**Alone**	**+Trigram**	**WER**
Trigram	≈ 167	–	13.7
Xu, Chelba & Jelinek	151.2	144.2	12.3
Roark	152.3	137.3	12.7
Charniak	130.2	126.1	11.9

Table 1: Perplexity (the $base_2$ antilog of per-word cross-entropy) results of syntax-based language models on a "speech-like" version of WSJ (smaller is better). Word Error Rate (WER) is for n–best list rescoring on HUB–1 lattices.

N-gram vs. Parsing Language Models

N-gram LMs, in various smoothed and backed-off incarnations, have dominated LM usage to date (Goodman 2001). We focus attention here on the widely used Knesser-Ney (KN) smoothed trigram. KN is one of the most successful smoothing techniques known, though one whose efficacy only recently has begun to be really understood (Goldwater, Griffiths, & Johnson 2006; Teh 2006).

An alternative (or complementary) approach to n-gram modeling is to employ a generative parser LM (Charniak 2001), which estimates the probability of a string s by summing over a large sample of its likely parses

$$p(s) = \sum_{\pi} p(\pi, s)$$

Table 1 compares language modeling accuracy on a speech-like version of WSJ of three syntactic language models (Charniak 2001; Roark 2001; Xu, Chelba, & Jelinek 2002) compared to a KN-smoothed trigram. Word error rate (WER) of corresponding automatic speech recognition (ASR) on HUB–1 (read WSJ) lattices is also shown. In all cases, syntactic models outperform the n-gram, and interpolating the two yields further improvement.

While these results show parsing language models to be complementary if not superior given an equivalent amount of training data (a million words here), web corpora have now enabled n-grams to be trained on half a billion words, for example, and no syntactic model has yet been trained on anything close to this much data (nor will there ever likely be half a billion words manually annotated for syntax).

However, we have shown that unannotated data *can* improve both the perplexity and word-error rates of our first-stage PCFG parsing language model (Hall & Johnson 2003). Moreover, the semi-supervised model discussed earlier (McClosky, Charniak, & Johnson 2006) will directly improve our ability to automatically annotate new data for training our parsing language model. Future experiments will study precisely how the model scales to very large corpora.

[3] Linguistic Data Consortium (LDC) Catalog ID LDC95T21

[4] A full description of these findings is currently in submission.

As a closing remark, KN smoothing was recently surpassed (and rather dramatically) by a novel use of random forest (RF) modeling (Xu & Jelinek 2004). As with syntactic models, however, scalability is again an issue. In this case, computational complexity is demanding, and if one has to choose between RFs with less data and KN smoothing with more data, the latter wins. If RFs can be made more efficient, however, it may benefit syntactic models as well by better smoothing their internal probability distributions.

Noisy-Channel Modeling

Given some observed data O, it is often useful to imagine it has an underlying source form S about which we have some prior knowledge (easily expressed via Bayes Rule):

$$\widehat{s} \quad = \quad \operatorname*{argmax}_S \mathrm{P}(S|O) = \operatorname*{argmax}_S \mathrm{P}(O|S)\mathrm{P}(S)$$

In NLP, this *noisy-channel* setup has been widely used to model an underlying string of words being transformed into some other observable form: acoustics for speech recognition (Hall & Johnson 2003), words in a foreign language for machine translation (Charniak, Knight, & Yamada 2003), a disfluent transcript to be cleaned (Johnson & Charniak 2004), a long paraphrase to be shortened (Turner & Charniak 2005), etc. While the *channel model* $\mathrm{P}(O|S)$ differs for each task, what remains constant is $\mathrm{P}(S)$, the LM. In this framework, a better prior directly implies an improved posterior, and all of the tasks cited here have shown improved accuracy from using a parser LM instead of n-grams.

Tasks

Speech Recognition: the acoustic channel model identifies words based on sound, but similar sounding words and phrases are often confused (e.g. *a*, *uh*, and *the*, or as a more colorful example, *recognize speech* and *wreck a nice beach*). However, given such possibilities to choose between (via a lattice or n-best lists from the acoustic model), the LM can help us discriminate between alternatives by providing prior knowledge on the probability of each path though the lattice, penalizing with low probability word sequences that are unlikely to occur in practice. Table 1 compares WER achieved using syntactic and n-gram language modeling on this task (Hall & Johnson 2003).

Machine Translation (MT): the channel model translates a foreign language string into English, and given multiple possibilities for the translation, the LM again helps choose between alternatives. One study found that by moving from an n-gram to a parsing language model, human judges deemed 50% more of translations to be perfect, 200% more to be grammatically correct, and an equivalent number to semantically correct (Charniak, Knight, & Yamada 2003). Despite these findings, output produced using the n-gram LM received higher scores from BLEU, a popular n-gram based metric for automatic evaluation of MT output. The difference observed here between human judgements and BLEU scores suggest closer agreement may be possible by incorporating syntactic information into evaluation.

Disfluency Modeling: we observe a possibly disfluent utterance in conversational speech and want to find a "cleaned" fluent version indicating what the speaker meant to say. For example, in an observed utterance *I want a flight to Boston, uh, I mean Denver*, we assume the speaker meant to say simply *I want a flight to Denver* but inadvertently inserted *uh I mean* due to cognitive and practical constraints. While an engaged listener can usually filter out disfluencies subconsciously, disfluencies have been shown to negatively impact both the readability of transcribed speech (Jones *et al.* 2003) and the accuracy of automated analysis performed on it (Charniak & Johnson 2001; Harper *et al.* 2005). To model disfluencies in the noisy-channel paradigm, disfluency insertion is handled by the channel and the LM estimates the string probabilities of proposed cleaned versions. For this task, the parsing LM yielded a 3% f-measure improvement (12.5% error reduction) over an n-gram model (Johnson & Charniak 2004).

Sentence Compression: this task involves finding a "equivalent" shorter paraphrase or summary of a sentence (at a finer level of granularity, proposing various shortened alternatives according to a length vs. information loss tradeoff). Under the noisy-channel rubric, the channel model proposes portions of the sentence to remove, and the LM estimates the acceptability of the remaining words. Using a parsing LM rather than an n-gram, 10% more compression (i.e. shorter paraphrases) is achieved with no loss in grammaticality or information (Turner & Charniak 2005).

Direct Use of Syntax

This section describes tasks in which syntactic features derived from parser output have been directly applied.

Information Extraction (IE): Parsing has been used in a variety of information extraction (IE) systems. In Miller et al. (2000), simple entity and relation annotations were added on top of syntax, and the parser was trained to recover both in parallel. In MUC-7, the system finished second in two tasks (the winning system was hand-crafted). In Surdeanu et al. (2003), predicate-argument structures were induced from parse trees with trivial rules mapping predicate arguments to domain-specific template slots. Again the system performed quite well though was bested by one hand-crafted for domain-specific pattern matching. Recently there has been fast growing interest in performing IE on biomedical research papers for automatic database curation, and parsing is being increasingly applied here as well (Park 2001).

NL Database Queries: Natural language interfaces have the potential to simplify our interactions with devices and software. A recent study annotated logical form on surface-level queries (e.g. *what states border Texas?*) and automatically learned a mapping between surface and logical forms via induced syntax (Zettlemoyer & Collins 2005). For two domains, 87% f-measure was achieved in mapping NL queries to correct logical form.

Comma Detection: Syntactic features derived from parser output have been used to automatically detect where commas should be inserted into text (Shieber & Tao 2003). Results achieved about 73% f-measure and showed detection accuracy increased smoothly and incrementally as parsing accuracy improved. In addition to being useful in tasks like generation and MT, this is particularly compelling for annotating transcribed speech. As discussed earlier, ASR output alone does not produce readable transcriptions (Jones *et al.* 2003), and parsing studies have shown the presence of commas, which commonly indicate constituent boundaries, significantly improves parsing accuracy on transcribed speech (Gregory, Johnson, & Charniak 2004).

Sentence Boundary Detection: In a similar vein, sentence boundary detection is also important for automatic speech transcription to improve both its readability (Jones *et al.* 2003) and the accuracy of automatic analysis performed on it (Harper *et al.* 2005). Recently, syntactic features extracted from parser output have been used to rerank output of a state-of-the-art sentence boundary detection system, and accuracy was improved for both the best-case of manually transcribed (reference) words and the fully-automatic case of ASR output (Harper *et al.* 2005).

Machine Translation: Early work in MT sought an interlingua which could serve as a nexus in translating between all language pairs. At the opposite extreme, statistical MT systems have primarily translated at the word-level. Recent work has begun investigating syntactic-transfer as a compromise between these two poles (Knight & Marcu 2005); unlike interlingua, we have concrete syntactic representations that have been broadly applied, and syntax provides a deeper understanding of linguistic structure and commonalities shared across languages than we get from words alone.

Conclusion

We have provided a brief introduction to recent work in statistical parsing and its applications. While we have already seen parsing usefully applied to a variety of tasks, really we are only beginning to scratch the surface in developing systems capable of deep understanding of natural language.

References

Charniak, E., and Johnson, M. 2001. Edit detection and parsing for transcribed speech. In *Proc. NAACL*, 118–126.

Charniak, E., and Johnson, M. 2005. Coarse-to-fine *n*-best parsing and MaxEnt discriminative reranking. In *Proc. Assoc. for Computational Linguistics (ACL)*, 173–180.

Charniak, E.; Knight, K.; and Yamada, K. 2003. Syntax-based language models for statistical machine translation. In *MT Summit IX*. Intl. Assoc. for Machine Translation.

Charniak, E. 1997. Statistical parsing with a context-free grammar and word statistics. In *Proc. AAAI*, 598–603.

Charniak, E. 2000. A maximum-entropy-inspired parser. In *Proc. N. American ACL (NAACL)*, 132–139.

Charniak, E. 2001. Immediate-head parsing for language models. In *Proc. Assoc. for Computational Linguistics (ACL)*, 116–123.

Chen, S. F., and Goodman, J. 1996. An empirical study of smoothing techniques for language modeling. In *Proc. Assoc. for Computational Linguistics (ACL)*, 310–318.

Collins, M. 1997. Three generative, lexicalised models for statistical parsing. In *Proc. Assoc. for Comp. Linguistics*, 16–23.

Collins, M. 2000. Discriminative reranking for natural language parsing. In *Machine Learning: Proc. of the 17th Intl. Conference (ICML 2000)*, 175–182.

Gildea, D. 2001. Corpus variation and parser performance. In *Proc. Empirical Methods in NLP (EMNLP)*, 167–202.

Goldwater, S.; Griffiths, T.; and Johnson, M. 2006. Interpolating between types and tokens by estimating power-law generators. In *Advances in Neural Information Processing Systems (NIPS) 18*.

Goodman, J. T. 2001. A bit of progress in language modeling extended version. Technical Report 2001-72, Microsoft Research.

Gregory, M.; Johnson, M.; and Charniak, E. 2004. Sentence-internal prosody does not help parsing the way punctuation does. In *Proc. N. American ACL (NAACL)*, 81–88.

Hall, K., and Johnson, M. 2003. Language modelling using efficient best-first bottom-up parsing. In *Automatic Speech Recognition and Understanding Workshop*. IEEE ASRU 2003.

Harper, M., et al. 2005. *Johns Hopkins Summer Workshop Final Report on Parsing and Spoken Structural Event Detection*.

Huang, L., and Chang, D. 2005. Better k-best parsing. Technical Report MS-CIS-05-08, Dept. of Comp. Sci., U. of Pennsylvania.

Jimenez, V. M., and Marzal, A. 2000. Computation of the n best parse trees for weighted and stochastic context-free grammars. In *Joint Intl. Workshops on Advances in Pattern Recognition (IAPR)*.

Johnson, M., and Charniak, E. 2004. A TAG-based noisy channel model of speech repairs. In *Proc. Assoc. for Computational Linguistics*, 33–39.

Jones, D., et al. 2003. Measuring the readability of automatic speech-to-text transcripts. In *Proc. Eurospeech*, 1585–1588.

Knight, K., and Marcu, D. 2005. Machine translation in the year 2004. In *Intl. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, 965–968.

Marcus, M., et al. 1993. Building a large annotated corpus of English: The Penn Treebank. *Comp. Linguistics* 19(2):313–330.

McClosky, D.; Charniak, E.; and Johnson, M. 2006. Effective self-training for parsing. In *Proc. N. American ACL (NAACL)*.

Miller, S.; Fox, H.; Ramshaw, L.; and Weischedel, R. 2000. A novel use of statistical parsing to extract information from text. In *Proc. N. American ACL (NAACL)*, 226–233.

Park, J. C. 2001. Using combinatory categorical grammar to extract biomedical information. *IEEE Intelligent Systems* 16(6).

Roark, B. 2001. Probabilistic top-down parsing and language modeling. *Computational Linguistics* 27(2):249–276.

Sekine, S. 1997. The domain dependence of parsing. In *Proc. Applied Natural Language Processing (ANLP)*, 96–102.

Shieber, S. M., and Tao, X. 2003. Comma restoration using constituency information. In *Proc. NAACL*, 221–227.

Steedman, M., et al. 2003. Bootstrapping statistical parsers from small datasets. In *Proc. European ACL (EACL)*, 331–338.

Surdeanu, M.; Harabagiu, S.; Williams, J.; and Aarseth, P. 2003. Using predicate-argument structures for information extraction. In *Proc. Assoc. for Computational Linguistics (ACL)*, 8–15.

Teh, Y. 2006. A bayesian interpretation of interpolated kneserney. Technical report, National University of Singapore.

Turner, J., and Charniak, E. 2005. Supervised and unsupervised learning for sentence compression. In *Proc. Assoc. for Computational Linguistics (ACL)*, 290–297.

Xu, P., and Jelinek, F. 2004. Random forests in language modeling. In *Proc. Empirical Methods in NLP (EMNLP)*, 325–332.

Xu, P.; Chelba, C.; and Jelinek, F. 2002. A study on richer syntactic dependencies for structured language modeling. In *Proc. Assoc. for Computational Linguistics (ACL)*, 191–198.

Zettlemoyer, L. S., and Collins, M. 2005. Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. In *Proc. UAI*.