Exploring Evidence for Shallow Parsing*

Xin Li **Dan Roth**

Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, IL 61801
xli1@cs.uiuc.edu *danr@cs.uiuc.edu*

Abstract

Significant amount of work has been devoted recently to develop learning techniques that can be used to generate partial (shallow) analysis of natural language sentences rather than a full parse. In this work we set out to evaluate whether this direction is worthwhile by comparing a learned shallow parser to one of the best learned full parsers on tasks both can perform — identifying phrases in sentences. We conclude that directly learning to perform these tasks as shallow parsers do is advantageous over full parsers both in terms of performance and robustness to new and lower quality texts.

1 Introduction

Shallow parsing is studied as an alternative to full-sentence parsing. Rather than producing a complete analysis of sentences, the alternative is to perform only partial analysis of the syntactic structures in a text (Harris, 1957; Abney, 1991; Greffenstette, 1993). A lot of recent work on shallow parsing has been influenced by Abney's work (Abney, 1991), who has suggested to "chunk" sentences to base level phrases. For example, the sentence *"He reckons the current account deficit will narrow to only $ 1.8 billion in September ."* would be chunked as follows (Tjong Kim Sang and Buchholz, 2000):

> [NP He] [VP reckons] [NP the current account deficit] [VP will narrow] [PP to] [NP only $ 1.8 billion] [PP in] [NP September] .

While earlier work in this direction concentrated on manual construction of rules, most of the recent work has been motivated by the observation that shallow syntactic information can be extracted using local information — by examining the pattern itself, its nearby context and the local part-of-speech information. Thus, over the past few years, along with advances in the use of learning and statistical methods for acquisition of full parsers (Collins, 1997; Charniak, 1997a; Charniak, 1997b; Ratnaparkhi, 1997), significant progress has been made on the use of statistical learning methods to recognize shallow parsing patterns — syntactic phrases or words that participate in a syntactic relationship (Church, 1988; Ramshaw and Marcus, 1995; Argamon et al., 1998; Cardie and Pierce, 1998; Munoz et al., 1999; Punyakanok and Roth, 2001; Buchholz et al., 1999; Tjong Kim Sang and Buchholz, 2000).

Research on shallow parsing was inspired by psycholinguistics arguments (Gee and Grosjean, 1983) that suggest that in many scenarios (e.g., conversational) full parsing is not a realistic strategy for sentence processing and analysis, and was further motivated by several arguments from a natural language engineering viewpoint.

First, it has been noted that in many natural language applications it is sufficient to use shallow parsing information; information such as noun phrases (NPs) and other syntactic sequences have been found useful in many large-scale language processing applications including information extraction and text summarization (Grishman, 1995; Appelt et al., 1993). Second, while training a full parser requires a collection of fully parsed sentences as training corpus, it is possible to train a

*This research is supported by NSF grants IIS-9801638, ITR-IIS-0085836 and an ONR MURI Award.

shallow parser incrementally. If all that is available is a collection of sentences annotated for NPs, it can be used to produce this level of analysis. This can be augmented later if more information is available. Finally, the hope behind this research direction was that this incremental and modular processing might result in more robust parsing decisions, especially in cases of spoken language or other cases in which the quality of the natural language inputs is low — sentences which may have repeated words, missing words, or any other lexical and syntactic mistakes.

Overall, the driving force behind the work on learning shallow parsers was the desire to get better performance and higher reliability. However, since work in this direction has started, a significant progress has also been made in the research on statistical learning of full parsers, both in terms of accuracy and processing time (Charniak, 1997b; Charniak, 1997a; Collins, 1997; Ratnaparkhi, 1997).

This paper investigates the question of whether work on shallow parsing is worthwhile. That is, we attempt to evaluate quantitatively the intuitions described above — that learning to perform shallow parsing could be more accurate and more robust than learning to generate full parses. We do that by concentrating on the task of identifying the phrase structure of sentences — a byproduct of full parsers that can also be produced by shallow parsers. We investigate two instantiations of this task, "chucking" and identifying atomic phrases. And, to study robustness, we run our experiments both on standard Penn Treebank data (part of which is used for training the parsers) and on lower quality data — the Switchboard data.

Our conclusions are quite clear. Indeed, shallow parsers that are specifically trained to perform the tasks of identifying the phrase structure of a sentence are more accurate and more robust than full parsers. We believe that this finding, not only justifies work in this direction, but may even suggest that it would be worthwhile to use this methodology incrementally, to learn a more complete parser, if needed.

2 Experimental Design

In order to run a fair comparison between full parsers and shallow parsers — which could produce quite different outputs — we have chosen the task of identifying the phrase structure of a sentence. This structure can be easily extracted from the outcome of a full parser and a shallow parser can be trained specifically on this task.

There is no agreement on how to define phrases in sentences. The definition could depend on downstream applications and could range from simple syntactic patterns to message units people use in conversations. For the purpose of this study, we chose to use two different definitions. Both can be formally defined and they reflect different levels of shallow parsing patterns.

The first is the one used in the chunking competition in CoNLL-2000 (Tjong Kim Sang and Buchholz, 2000). In this case, a full parse tree is represented in a flat form, producing a representation as in the example above. The goal in this case is therefore to accurately predict a collection of 11 different types of phrases. The chunk types are based on the syntactic category part of the bracket label in the Treebank. Roughly, a chunk contains everything to the left of and including the syntactic head of the constituent of the same name. The phrases are: adjective phrase (ADJP), adverb phrase (ADVP), conjunction phrase (CONJP), interjection phrase (INTJ), list marker (LST), noun phrase (NP), preposition phrase (PP), particle (PRT), subordinated clause (SBAR), unlike coordinated phrase (UCP), verb phrase (VP). (See details in (Tjong Kim Sang and Buchholz, 2000).)

The second definition used is that of *atomic phrases*. An atomic phrase represents the most basic phrase with no nested sub-phrases. For example, in the parse tree,

> ((S (NP (NP Pierre Vinken) , (ADJP (NP 61 years) old) ,) (VP will (VP join (NP the board) (PP as (NP a nonexecutive director)) (NP Nov. 29))) .))

`Pierre Vinken`, `61 years`, `the board`, `a nonexecutive director` and `Nov. 29` are atomic phrases while other higher-level phrases are not. That is, an atomic phrase denotes a tightly coupled message unit which is just above the level of single words.

2.1 Parsers

We perform our comparison using two state-of-the-art parsers. For the full parser, we use the one developed by Michael Collins (Collins, 1996; Collins, 1997) — one of the most accurate full parsers around. It represents a full parse tree as a set of basic phrases and a set of dependency relationships between them. Statistical learning techniques are used to compute the probabilities of these phrases and of candidate dependency relations occurring in that sentence. After that, it will choose the candidate parse tree with the highest probability as output. The experiments use the version that was trained (by Collins) on sections 02-21 of the Penn Treebank. The reported results for the full parse tree (on section 23) are recall/precision of 88.1/87.5 (Collins, 1997).

The shallow parser used is the SNoW-based CSCL parser (Punyakanok and Roth, 2001; Munoz et al., 1999). SNoW (Carleson et al., 1999; Roth, 1998) is a multi-class classifier that is specifically tailored for learning in domains in which the potential number of information sources (features) taking part in decisions is very large, of which NLP is a principal example. It works by learning a sparse network of linear functions over a pre-defined or incrementally learned feature space. Typically, SNoW is used as a classifier, and predicts using a winner-take-all mechanism over the activation value of the target classes. However, in addition to the prediction, it provides a reliable confidence level in the prediction, which enables its use in an inference algorithm that combines predictors to produce a coherent inference. Indeed, in CSCL (constraint satisfaction with classifiers), SNoW is used to learn several different classifiers – each detects the beginning or end of a phrase of some type (noun phrase, verb phrase, etc.). The outcomes of these classifiers are then combined in a way that satisfies some constraints – non-overlapping constraints in this case – using an efficient constraint satisfaction mechanism that makes use of the confidence in the classifier's outcomes.

Since earlier versions of the SNoW based CSCL were used only to identify single phrases (Punyakanok and Roth, 2001; Munoz et al., 1999) and never to identify a collection of several phrases at the same time, as we do here, we also trained and tested it under the exact conditions of CoNLL-2000 (Tjong Kim Sang and Buchholz, 2000) to compare it to other shallow parsers. Table 1 shows that it ranks among the top shallow parsers evaluated there [1].

Table 1: **Rankings of Shallow Parsers in CoNLL-2000**. See (Tjong Kim Sang and Buchholz, 2000) for details.

Parsers	Precision(%)	Recall(%)	F_β(%)
[KM00]	93.45	93.51	93.48
[Hal00]	93.13	93.51	93.32
[CSCL]*	93.41	92.64	93.02
[TKS00]	94.04	91.00	92.50
[ZST00]	91.99	92.25	92.12
[Dej00]	91.87	91.31	92.09
[Koe00]	92.08	91.86	91.97
[Osb00]	91.65	92.23	91.94
[VB00]	91.05	92.03	91.54
[PMP00]	90.63	89.65	90.14
[Joh00]	86.24	88.25	87.23
[VD00]	88.82	82.91	85.76
Baseline	72.58	82.14	77.07

2.2 Data

Training was done on the Penn Treebank (Marcus et al., 1993) Wall Street Journal data, sections 02-21. To train the CSCL shallow parser we had first to convert the WSJ data to a flat format that directly provides the phrase annotations. This is done using the "Chunklink" program provided for CoNLL-2000 (Tjong Kim Sang and Buchholz, 2000).

Testing was done on two types of data. For the first experiment, we used the WSJ section 00 (which contains about 45,000 tokens and 23,500 phrases). The goal here was simply to evaluate the full parser and the shallow parser on text that is similar to the one they were trained on.

[1] We note that some of the variations in the results are due to variations in experimental methodology rather than parser's quality. For example, in [KM00], rather than learning a classifier for each of the 11 different phrases, a discriminator is learned for each of the $\binom{12}{2}$ phrase pairs which, statistically, yields better results. [Hal00] also uses 5 different parsers and reports the results of some voting mechanism on top of these.

Our robustness test (section 3.2) makes use of section 4 in the Switchboard (SWB) data (which contains about 57,000 tokens and 17,000 phrases), taken from Treebank 3. The Switchboard data contains conversation records transcribed from phone calls. The goal here was two fold. First, to evaluate the parsers on a data source that is different from the training source. More importantly, the goal was to evaluate the parsers on low quality data and observe the absolute performance as well as relative degradation in performance.

The following sentence is a typical example of the SWB data.

> *Huh/UH ,/, well/UH ,/, um/UH ,/, you/PRP know/VBP ,/, I/PRP guess/VBP it/PRP 's/BES pretty/RB deep/JJ feelings/NNS ,/, uh/UH ,/, I/PRP just/RB ,/, uh/UH ,/, went/VBD back/RB and/CC rented/VBD ,/, uh/UH ,/, the/DT movie/NN ,/, what/WP is/VBZ it/PRP ,/, GOOD/JJ MORNING/NN VIET/NNP NAM/NNP ./.*

The fact that it has some missing words, repeated words and frequent interruptions makes it a suitable data to test robustness of parsers.

2.3 Representation

We had to do some work in order to unify the input and output representations for both parsers. Both parsers take sentences annotated with POS tags as their input. We used the POS tags in the WSJ and converted both the WSJ and the SWB data into the parsers' slightly different input formats. We also had to convert the outcomes of the parsers in order to evaluate them in a fair way. We choose CoNLL-2000's chunking format as our standard output format and converted Collins' parser outcome into this format.

2.4 Performance Measure

The results are reported in terms of precision, recall, and $F_\beta(\beta = 1)$ as defined below:

$$\text{Precision} = \frac{\text{Number of correct proposed patterns}}{\text{Number of proposed patterns}}$$

$$\text{Recall} = \frac{\text{Number of correct proposed patterns}}{\text{Number of correct patterns}}$$

$$F_\beta = \frac{(\beta^2 + 1) \cdot \text{Recall} \cdot \text{Precision}}{\beta^2 \cdot \text{Precision} + \text{Recall}}$$

We have used the evaluation procedure of CoNLL-2000 to produce the results below. Although we do not report significance results here, note that all experiments were done on tens of thousands of instances and clearly all differences and ratios measured are statistically significant.

3 Experimental Results

3.1 Performance

We start by reporting the results in which we compare the full parser and the shallow parser on the "clean" WSJ data. Table 2 shows the results on identifying all phrases — chunking in CoNLL-2000 (Tjong Kim Sang and Buchholz, 2000) terminology. The results show that for the tasks of identifying phrases, learning directly, as done by the shallow parser outperforms the outcome from the full parser.

Table 2: **Precision & Recall for phrase identification (chunking)** for the full and the shallow parser on the WSJ data. Results are shown for an (weighted) average of 11 types of phrases as well as for two of the most common phrases, NP and VP.

	Full Parser			Shallow Parser		
	P	R	F_β	P	R	F_β
Avr	91.71	92.21	91.96	93.85	95.45	94.64
NP	93.10	92.05	92.57	93.83	95.92	94.87
VP	86.00	90.42	88.15	95.50	95.05	95.28

Next, we compared the performance of the parsers on the task of identifying atomic phrases[2]. Here, again, the shallow parser exhibits significantly better performance. Table 3 shows the results of extracting atomic phrases.

3.2 Robustness

Next we present the results of evaluating the robustness of the parsers on lower quality data. Table 4 describes the results of evaluating the same parsers as above, (both trained as before on the

[2]As a side note — the fact that the same program could be trained to recognize patterns of different level in such an easy way, only by changing the annotations of the training data, could also be viewed as an advantage of the shallow parsing paradigm.

Table 3: **Precision & Recall for atomic phrase identification** on the WSJ data. Results are shown for an (weighted) average of 11 types of phrases as well as for the most common phrase, NP. VP occurs very infrequently as an atomic phrase.

	Full Parser			Shallow Parser		
	P	R	F_β	P	R	F_β
Avr	88.68	90.45	89.56	92.02	93.61	92.81
NP	91.86	92.16	92.01	93.54	95.88	94.70

same WSJ sections) on the SWB data. It is evident that on this data the difference between the performance of the two parsers is even more significant.

Table 4: **Switchboard data:** Precision & Recall for phrase identification (chunking) on the Switchboard data. Results are shown for an (weighted) average of 11 types of phrases as well as for two of the most common phrases, NP, VP.

	Full Parser			Shallow Parser		
	P	R	F_β	P	R	F_β
Avr	81.54	83.79	82.65	86.50	90.54	88.47
NP	88.29	88.96	88.62	90.50	92.59	91.54
VP	70.61	83.53	76.52	85.30	89.76	87.47

This is shown more clearly in Table 5 which compares the relative degradation in performance each of the parsers suffers when moving from the WSJ to the SWB data (Table 2 vs. Table 4). While the performances of both parsers goes down when they are tested on the SWB, relative to the WSJ performance, it is clear that the shallow parser's performance degrades more gracefully. These results clearly indicate the higher-level robustness of the shallow parser.

3.3 Discussion

Analyzing the results shown above is outside the scope of this short abstract. We will only provide one example that might shed some light on the reasons for the more significant degradation in the results of the full parser. Table 6 exhibits the results of chunking as given by Collins' parser. The four columns are the original words, POS tags, and the phrases — encoded using the BIO scheme (B- beginning of phrase; I- inside the phrase; O- outside the phrase) — with the true annotation and Collins' annotation.

Table 5: **Robustness:** Relative degradation in F_β results for Chunking. For each parser the result shown is the ratio between the result on the "noisy" SWB data and the "clean" WSJ corpus data.

	Full Parser	Shallow Parser
	F_β	F_β
Avr	.89	.93
NP	.95	.96
VP	.86	.92

The mistakes in the phrase identification (e.g., in "word processing applications") seem to be a result of assuming, perhaps due to the "um" and additional punctuation marks, that this is a separate sentence, rather than a phrase. Under this assumption, the full parser tries to make it a complete sentence and decides that "processing" is a "verb" in the parsing result. This seems to be a typical example for mistakes made by the full parser.

Table 6: **An example: a parsing mistake**

WORD	POS	TRUE	Collins
Um	UH	B-INTJ	B-INTJ
COMMA	COMMA	O	I-INTJ
Mostly	RB	O	I-INTJ
COMMA	COMMA	O	O
um	UH	B-INTJ	B-INTJ
COMMA	COMMA	O	O
word	NN	B-NP	**B – NP**
processing	NN	I-NP	**B – VP**
applications	NNS	I-NP	**B – NP**
and	CC	O	O
COMMA	COMMA	O	O
uh	UH	B-INTJ	B-INTJ
COMMA	COMMA	O	O
just	RB	B-ADVP	B-PP
as	IN	B-PP	I-PP
a	DT	B-NP	B-NP
dumb	JJ	I-NP	I-NP
terminal	NN	I-NP	I-NP
..	.	O	O

4 Conclusion

Full parsing and shallow parsing are two different strategies for parsing natural languages. While full parsing provides more complete information about a sentence, shallow parsing is more flexible, easier to train and could be targeted for specific, limited subtasks. Several arguments have been used in the past to argue, on an intuitive basis, that (1) shallow parsing is sufficient for a wide range of applications and that (2) shallow parsing could be more reliable than full parsing in handling ill-formed real-world sentences, such as sentences that occur in conversational situations.

While the former is an experimental issue that is still open, this paper has tried to evaluate experimentally the latter argument. Although the experiments reported here only compare the performance of one full parser and one shallow parser, we believe that these state-of-the-art parsers represent their class quite well. Our results show that on the specific tasks for which we have trained the shallow parser – identifying several kinds of phrases – the shallow parser performs more accurately and more robustly than the full parser. In some sense, these results validate the research in this direction.

Clearly, there are several directions for future work that this preliminary work suggests. First, in our experiments, the Collins' parser is trained on the Treebank data and tested on the lower quality data. It would be interesting to see what are the results if lower quality data is also used for training. Second, our decision to run the experiments on two different ways of decomposing a sentence into phrases was somewhat arbitrary (although we believe that selecting phrases in a different way would not affect the results). It does reflect, however, the fact that it is not completely clear what kinds of shallow parsing information should one try to extract in real applications. Making progress in the direction of a formal definition of phrases and experimenting with these along the lines of the current study would also be useful. Finally, an experimental comparison on several other shallow parsing tasks such as various attachments and relations detection is also an important direction that will enhance this work.

5 Acknowledgments

We are grateful to Vasin Punyakanok for his advice in this project and for his help in using the CSCL parser. We also thank Michael Collins for making his parser available to us.

References

S. P. Abney. 1991. Parsing by chunks. In S. P. Abney R. C. Berwick and C. Tenny, editors, *Principle-based parsing: Computation and Psycholinguistics*, pages 257–278. Kluwer, Dordrecht.

D. Appelt, J. Hobbs, J. Bear, D. Israel, and M. Tyson. 1993. FASTUS: A finite-state processor for information extraction from real-world text. In *Proc. International Joint Conference on Artificial Intelligence*.

S. Argamon, I. Dagan, and Y. Krymolowski. 1998. A memory-based approach to learning shallow natural language patterns. In *COLING-ACL 98, The 17th International Conference on Computational Linguistics*.

S. Buchholz, J. Veenstra, and W. Daelemans. 1999. Cascaded grammatical relation assignment. In *EMNLP-VLC'99, the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, June.

C. Cardie and D. Pierce. 1998. Error-driven pruning of Treebanks grammars for base noun phrase identification. In *Proceedings of ACL-98*, pages 218–224.

A. Carleson, C. Cumby, J. Rosen, and D. Roth. 1999. The SNoW learning architecture. Technical Report UIUCDCS-R-99-2101, UIUC Computer Science Department, May.

E. Charniak. 1997a. Statistical parsing with a context-free grammar and word statistics. In *Proc. National Conference on Artificial Intelligence*.

E. Charniak. 1997b. Statistical techniques for natural language parsing. *The AI Magazine*.

Kenneth W. Church. 1988. A stochastic parts program and noun phrase parser for unrestricted text. In *Proc. of ACL Conference on Applied Natural Language Processing*.

M. Collins. 1996. A new statistical parser based on bigram lexical dependencies. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics*, pages 184–191.

M. Collins. 1997. Three generative, lexicalised models for statistical parsing. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics*.

J. P. Gee and F. Grosjean. 1983. Performance structures:a psycholinguistic and linguistic appraisal. *Cognitive Psychology*, 15:411–458.

G. Greffenstette. 1993. Evaluation techniques for automatic semantic extraction: comparing semantic and window based approaches. In *ACL'93 workshop on the Acquisition of Lexical Knowledge from Text*.

R. Grishman. 1995. The NYU system for MUC-6 or where's syntax? In B. Sundheim, editor, *Proceedings of the Sixth Message Understanding Conference*. Morgan Kaufmann Publishers.

Z. S. Harris. 1957. Co-occurrence and transformation in linguistic structure. *Language*, 33(3):283–340.

M. P. Marcus, B. Santorini, and M. Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330, June.

M. Munoz, V. Punyakanok, D. Roth, and D. Zimak. 1999. A learning approach to shallow parsing. In *EMNLP-VLC'99, the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, June.

V. Punyakanok and D. Roth. 2001. The use of classifiers in sequential inference. In *NIPS-13; The 2000 Conference on Advances in Neural Information Processing Systems*.

L. A. Ramshaw and M. P. Marcus. 1995. Text chunking using transformation-based learning. In *Proceedings of the Third Annual Workshop on Very Large Corpora*.

A. Ratnaparkhi. 1997. A linear observed time statistical parser based on maximum entropy models. In *EMNLP-97, The Second Conference on Empirical Methods in Natural Language Processing*, pages 1–10.

D. Roth. 1998. Learning to resolve natural language ambiguities: A unified approach. In *Proc. National Conference on Artificial Intelligence*, pages 806–813.

E. F. Tjong Kim Sang and S. Buchholz. 2000. Introduction to the CoNLL-2000 shared task: Chunking. In *Proceedings of CoNLL-2000 and LLL-2000*, pages 127–132.