Moving Beyond the Lexical Layer in Parsing Conversational Speech

Jeremy Gillmor Kahn

A thesis submitted in partial fulfillment
of the requirements for the degree of

Master of Arts

University of Washington

2005

Program Authorized to Offer Degree: Linguistics

University of Washington
Graduate School

This is to certify that I have examined this copy of a master's thesis by

Jeremy Gillmor Kahn

and have found that it is complete and satisfactory in all respects,
and that any and all revisions required by the final
examining committee have been made.

Committee Members:

_____
Richard A. Wright

_____
Emily M. Bender

_____
Mari Ostendorf

Date: _____

University of Washington

Abstract

Moving Beyond the Lexical Layer in Parsing Conversational Speech

Jeremy Gillmor Kahn

Chair of the Supervisory Committee:

Professor Mari Ostendorf
Electrical Engineering & Linguistics

Linguistics is often divided into S-linguistics (syntax and semantics) and P-linguistics (phonetics and phonology), interfacing at the lexical layer. This division carries the assumption that S and P linguistics are conditionally independent given the lexical sequence. However, speech contains paralexical information (such as prosody) that is not represented in the unadorned lexical sequence. This work takes a natural-language processing approach to relaxing this conditional independence assumption by examining ways in which statistical parsers can be improved by incorporating prosodic information. Prosody's impact on parser performance is explored in a series of automatic speech processing experiments on the SWITCHBOARD corpus of conversational telephone speech. These experiments explore three ways in which parser performance can be affected by paralexical information: at the sentence level (as defined by sentence boundaries), the sub-sentence level (as defined by intonational phrase boundaries), and the word level (in terms of recognized-word confidences). Within these levels, prosodically-marked disfluency information, such as self-corrections and hesitations, are also considered.

At the sentence level, this work demonstrates that state-of-the-art parser perfor-

mance is critically dependent on accurate sentence segmentation. At the sub-sentence level, it demonstrates that even when sentence boundaries are accurately determined, the posterior probabilities of symbolic prosodic boundary features can help statistical parsers to choose the correct parse. Finally, at the word level, this work introduces a new objective for speech-to-parse systems that jointly characterizes syntactic structure and word accuracy. Within this joint scoring framework, experiments show that considering alternate word hypotheses has a much greater impact on performance than alternate parse hypotheses.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

First and foremost, I must acknowledge the advice and guidance of my advisor Mari Ostendorf and readers Emily Bender and Richard Wright. Their counsel and encouragement in this work has been tremendously valuable; flaws remaining in this work are entirely my own.

I must also thank my research collaborators: Ciprian Chelba was very helpful in supporting the structured language model work that became Chapter 4. Matthew Lease and his advisors Eugene Charniak and Mark Johnson were absolutely essential collaborators in doing the research that became Chapter 5. Dustin Hillard's knowledge of speech recognition execution, architecture and hypothesis generation made Chapter 6 a reality, and Brian Roark and the rest of the Johns Hopkins Summer Workshop in Parsing and Spoken Structural Event Detection researchers made a usable SPARSEVAL tool, without which Chapter 6 could never have been completed.

My research benefited tremendously from being able to work in the Signal Speech and Language Interpretation laboratory at the University of Washington. In addition to an effective and powerful computing environment, the laboratory provides a context in which students, faculty and staff alike can share ideas, skills, and the sorts of secret handshakes that are difficult to learn from reading conference proceedings.

Finally, thanks to my parents, my brother, and to Dorothy and the Emerald Citizens for their love and support in this endeavor. I had a dream, and you were all in it.

DEDICATION

For the King of Bashan,

Irving Flint "Bud" Foote.

Your reign was too short.

Chapter 1

INTRODUCTION

The field of linguistics is often divided into phonetics and phonology on the one hand, and syntax and semantics on the other. But this division implies that there is a neat separation between these two — namely, at the lexical layer: a sequence of words or morphemes. Correspondingly, traditional natural-language processing tasks make a similar assumption, by treating (for example) the automatic speech recognition problem (getting words[1] from speech) separately from the parsing problem, which is usually defined as the extraction of syntactic structures from the lexical layer. However, this division of tasks and fields is based on an independence assumption: that these two sides have no interaction except through the lexical layer. Although it is convenient, this lexical-layer conditional independence assumption need not hold: prosody — tone, prominence and segmentation in speech — is a domain that naturally straddles the lexical layer, in that it is involved in both the phonetic form and articulation that make up the lexical items, and it is involved in the syntactic boundaries and structures of layers that the lexical items themselves make up, as well as higher-level processing.

This work takes a natural-language processing approach to relaxing the lexical-layer conditional independence assumption. Specifically, it examines ways in which parsers can be improved by incorporating information outside the simple lexical-layer:

[1]In the case of Chinese speech recognition, the definition of "word" is difficult to make precise, so the task is usually considered to be retrieving the correct characters. Nevertheless, the phrase "lexical layer" accomodates this concern.

from various levels of prosody, and from alternate speech recognition hypotheses. The rest of this chapter frames this approach in greater detail and lays out the structure for the rest of this work.

1.1 Linguistics, engineering, and the lexical layer

Accounts of linguistics (e.g. O'Grady et al. (1993)) often divide the field into six or so regions of research: phonetics and phonology "below" the lexical layer, in what is sometimes called **P-linguistics**; syntax, semantics, and "higher processing" (discourse, pragmatics, and planning) "above" the lexical level, in what is sometimes called **S-linguistics**. Morphology is occasionally treated as part of the syntax, e.g. in Haegeman (1991), and occasionally as part of the phonology, but it is almost never invoked in relationship to both. Morphology's border status (within the lexical layer) merely highlights the S/P divisions along the same border. This division of the linguistic field implies a division of labor; in a perceptual model, it is the role of P-linguistics to provide a representation of the lexical layer up to an S-linguistics component.

Natural language processing (NLP) approaches to dealing with real linguistic data, while they do not adopt exactly the same divisions, also divide up the process into more or less traditional tasks (e.g., the chapter headings of Jurafsky and Martin (2000)). These tasks work with many of the same basic units as the linguistics subfields: they may be divided into two groups (just as linguistics is divided): tasks that have the lexical layer as their output (e.g. automatic speech recognition, so-called "language models", and finite state morphologies), which one might call **P-NLP**, and tasks that begin from the lexical layer (e.g. parsing, topic detection, and summarization), which one might call **S-NLP**. A diagram of this arrangement is shown in figure 1.1, which includes the usual categories that each subfield assigns.

Both the engineering and the linguistics divisions of labor adopt an implicit as-

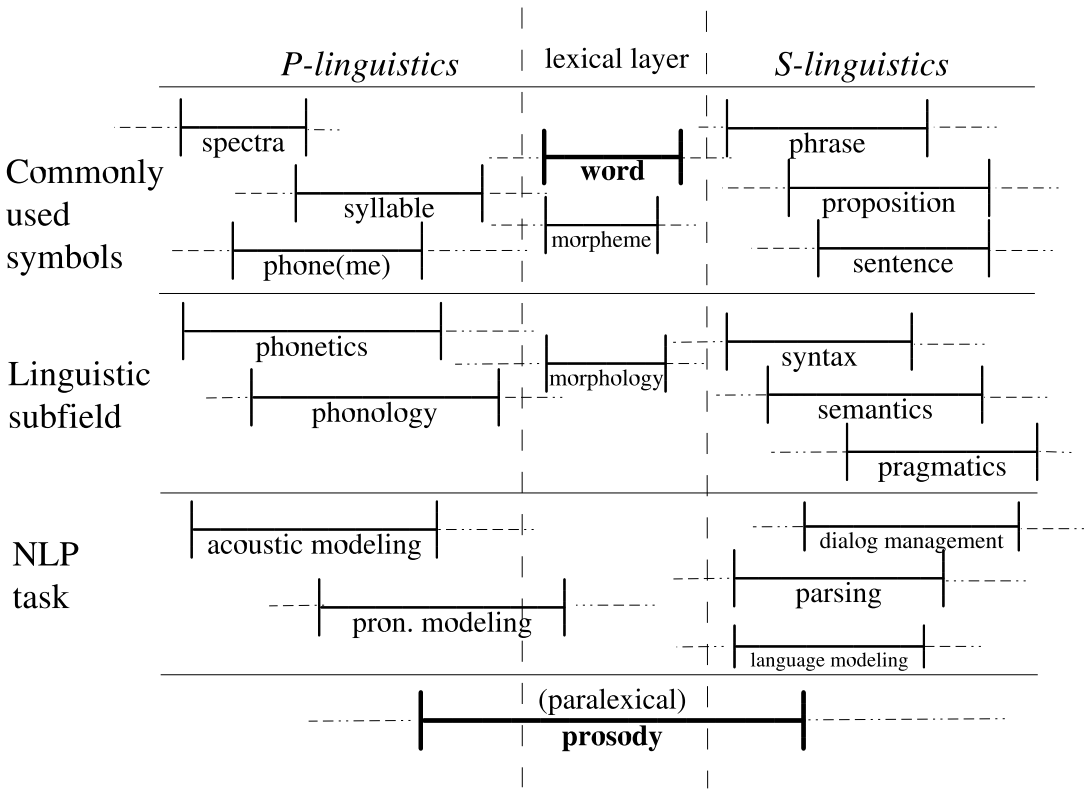


Figure 1.1: A diagram of the process of speech perception and understanding, as seen by P- and S-linguistics and P- and S-NLP. Solid lines represent core areas for each field; dotted lines represent peripheral interaction areas. Note that prosody (at the bottom of the figure), spans the entire lexical layer.

sumption of **lexical-layer conditional independence**: that the lexical sequence, usually as represented by the orthographic word, serves as an independence point between the sub-lexical components and the supra-lexical components. This assumption is a useful simplification of tasks for both fields: in linguistics, the lexical item seems to be cognitively important; in speech and natural language processing, it is relatively easy to gather corpora that are labeled with lexical items. In both engineering and linguistics, it is usually easy to get native speakers to agree on what lexical items make up a body of data.[2]

1.1.1 Expressing the lexical-layer conditional independence assumption in linguistics

In linguistics, this assumption is borne out in (for example) some syntactic theories, in which lexical (or morphological) items are found as whole roots at the beginning of syntax (Haegeman 1991). This assumption can also be seen in theories of lexical access in perception that take into account sub-word factors (Luce and Pisoni 1998) or word frequency (Bybee 1995) but not higher-level factors like syntactic phrases. In general, linguistics discussions often mention interactions between subareas in passing, but most linguistics research takes place within the S and P components. The boundaries of subareas are blurry *within* each of S and P linguistics – syntax and semantics are tightly interrelated in much research; the boundary between phonetics research and phonology research is blurry as well — but it is relatively easy to identify most linguistic research as either P-oriented or S-oriented, based on its position with respect to the lexical layer.

---

[2]Properly, one may only say that for those languages that (1) have orthographies, and (2) use a convention of word-separation marks (like whitespace), it is easy to get *literate* speakers, listeners, and readers to agree on what constitutes a word. Soliciting agreement on word-boundaries is more complex for languages like Chinese (with no whitespace separation) or languages with no standard written form. However, literate speakers of a given language can usually agree on some form of lexical item, whether it is the Chinese character, the Japanese *bunsetsu* phrase group, or some other language-specific, sub-phrasal division.

1.1.2 Statistical independence and conditional independence

In statistical models of complex data,when two random variables X and Y are statistically-dependent, then knowing the value of X gives us some information about the value of Y and vice versa. Expressing dependence or independence requires a description of the probability measure characterizing joint occurrences of the two (or more) variables a table expressing the relationship. Formally defining **statistical independence**: one says that X is independent of Y (which is sometimes written as $X \perp\!\!\!\perp Y$) when

$$p(x, y) = p(x)p(y) \ \forall x, y \ .$$

In other words, knowing whether X occurs makes it neither more nor less probable that Y occurs, and vice versa. One can also talk about **conditional independence**, which expresses that X and Y are conditionally independent given Z (written as $X \perp\!\!\!\perp Y|Z$):

$$p(x, y|z) = p(x|z)p(y|z) \ \forall x, y, z \ .$$

Once Z is known, knowing the value of X does not give any additional information about the value of Y, or vice versa.

1.1.3 Expressing the word-layer independence assumption in NLP

Thus, in an NLP approach to natural language, one can express this implied word-layer independence assumption as Assumption 1.1, which says that if the lexical sequence W is known, then the sub-lexical processes (P, the P-NLP component) provide no additional information to the supra-lexical processes (S, the S-NLP component):

$$P \perp\!\!\!\perp S \,|\, W \tag{1.1}$$

Assumption 1.1's expression is implicit in (for example) parsing work (Charniak 1993) that begins from lexical sequences, and work on speech recognition (Jelinek 1997) that optimizes system parameters to get the lexical sequence (and only the lexical

sequence) correct, even when it is used in a combined system that seeks to do (for example) spoken language understanding.

1.2 Relaxing the lexical-layer conditional independence assumption

The assumption that lexical sequences carry all the information needed from the sub-lexical P components to the supra-lexical S components is a useful one, and it serves as a useful simplification in both linguistics and natural language processing. However, built into this division of labor is the assumption that the lexical sequence is an adequate and complete representation of the sorts of information that are passed from these lower-level processes up to the higher-level processes.

These assumptions of modularity and independence, while not shared throughout linguistics, have focused research in areas away from these boundaries. However, it is quite productive and linguistically interesting to challenge these assumptions. Some work has recognized the limitation of this assumption: many spoken language systems (e.g. the JUPITER (Zue et al. 2000) and MOKUSEI (Nakano et al. 2001) spoken dialog weather information systems) use n-best lexical-sequence hypothesis lists as an interface between the speech recogition component (the P-NLP) and higher-level semantically-oriented processing (the S-NLP).

Relaxing the assumptions of modularity and independence requires one to examine how information from one side of the to-be-discarded lexical-information boundary could be used as a factor on the other side. For example, information that might previously have been used only in the P-component (e.g. pitch or duration) might be made useful in the S-component, e.g. in syntactic attachment decisions. Webster (2002), among others, suggests a model of psycholinguistic structure in which this might be possible. In the context of the existing fields and traditional NLP tasks, and especially in the context of improving a parser, this relaxation can be seen as a process of passing additional information "downstream" from the P-component to

the S-component.

The linguistics literature has long acknowledged that there are more complex interactions than this simple lexical layer, but has in general ignored those interactions for the sake of simplifying models and tractable research. The system of Lexical Phonology (e.g., in Kager and Zonneveld (1999)) explores some relaxation of this boundary, in allowing cross-word and intra-word effects to layer "on top" of each other. While some of this work is directly focused on the interface between phonology and the lexicon, work on the syntax-phonology interface (e.g., Selkirk (1984) and Gussenhoven (2004)) has begun to ask questions about the nature and expectations of this interaction.

For example, one could use (e.g.) phonetic or other acoustic information in making supra-lexical decisions, and supra-lexical information in making sub-lexical (phonetic and acoustic) decisions.. Of course, some information is clearly borne in information outside the lexical-sequence channel — English propositions may, for example, be understood as questions with certain pitch alterations.[3] Beyond that, Price et al. (1991) demonstrate that there are at least some syntactic decisions that can be disambiguated by non-lexical cues.

1.3 Prosody as a paralexical channel

If linguistics' division into subfields and natural language processing's traditional division into subtasks frame the lexical sequence as a channel for information from sub-lexical components, then one may challenge this framing (as in section 1.2) and ask whether there are other "out-of-band" channels that pass **paralexical** information from the sub-lexical components to the supra-lexical components. In other words, it may be useful to examine those properties of spoken language that remain useful in making supra-lexical decisions (e.g. syntactic and semantic decisions) but are not

[3]For some interesting contrasting data, see Safárová and Swerts (2004), which suggests that in conversation, this pattern of question intonation is not as robust as in careful speech.

completely captured by the lexical sequence. One property that spans this gap is **prosody**, linguistic information that (Couper-Kuhlen and Selting 1996:1) describes as "involving auditory parameters such as pitch, loudness and duration and the categories they jointly constitute." Thus, Figure 1.1 includes this prosodic "out-of-band", or **paralexical channel**.

Prosodic information (e.g., tones and phrasal groupings) is usually interpreted in linguistics as a separate symbolic layer of information (e.g., in Selkirk (1984)), even though it is — like all speech — carried in a signal that is continuous. Identifying these symbolic labels is a challenging task for speech processing, but a valuable first step in providing this additional symbolic layer to other components of a full system.

In addition, prosody may be involved in higher-level phenomena. The boundaries of spoken "sentence-like units" (SUs) as well as the various discourse-oriented phrases and lexical items, can interact with prosodic contours. As an NLP task, identifying these higher-level phenomena (SUs, speech repairs, and conversation markers like fillers) is known as structural metadata extraction, e.g. in Strassel (2003) and Liu et al. (2005).

As in section 1.2, this additional channel — in any of its forms — will serve as a vehicle with which to use engineering tools to explore relaxing the assumptions laid out in section 1.1 in order to improve performance on the parsing task.

1.4 Contributions and overview

In the experiments presented in this work, I demonstrate that relaxing the lexical-layer conditional independence assumption can improve parsing performance. I first demonstrate that sentence-level prosody — in the form of SU boundary identification in conversational speech — can impact parser performance, and then show that parser performance can be improved by incorporating posterior probabilities for sub-sentence symbolic prosody events. Finally, I demonstrate an improvement in performance at

a higher-level joint parsing task that incorporates word-level alternatives: combining a speech recognizer and a parser and selecting the best words and parse tree jointly. The remainder of this work is broken down as follows:

Chapter 2 includes a review of parsing technology and evaluation, speech recognition (insofar as it is related to the passing of additional information to the parser), prosody and some tactics for acquiring practically useful prosodic labels, and the various efforts to acquire structural metadata, like edit detection and SU detection.

In order to explore the possibilities of relaxing these assumptions, it is useful to use a corpus of real speech. In particular, conversational speech, with its rich negotiated turn-taking, is a good place to begin looking for useful opportunities to exploit the relaxation of the lexical-sequence independence assumption raised in Section 1.1. The research presented in this work is over SWITCHBOARD (Godfrey et al. 1992), a transcribed and annotated corpus of two-party telephone converstations. This corpus and its annotations are described in Chapter 3.

One common assumption hidden in the traditional division of NLP tasks is that the parsing task takes as an input the known word (or lexical) sequence, with known sentence boundaries. In conversational speech, (and many other speech sources) this assumption does not necessarily hold: even if the words are known, the edges of the sentence-like unit are not necessarily easy to detect in the audio input. In Chapter 4, I explore this assumption, and the repercussions in the parsing domain of applying automatic algorithms to try to detect these edge boundaries.

Another assumption implicit in the usual approach to the parsing task is that the lexical sequence is all the information needed to construct parses. In conversational speech, there is good reason to think that this assumption does not hold, and there is more useful information in the signal than just the lexical sequence alone. In Chapter 5, I explore methods to improve the selection of parse structures (syntactic trees) by using additional information from the paralexical prosody band.

Chapter 6 challenges the implicit reliance on the lexical sequence by combining

the traditional task of automatic speech recognition (which optimizes on the lexical sequence as output) and parsing (which assumes the lexical sequence as input) into a new single task: extracting good trees from the acoustic signal, bypassing the lexical measures and directly targeting instead a higher-level: words, and their interrelationships, as reflected in their syntactic dependencies.

Finally, Chapter 7 relates these contributions to each other, discusses the import of these results, and lays out a few directions for future research.

Chapter 2

BACKGROUND

This chapter includes brief introductions to two traditional human language technologies tasks (parsing and automatic speech recognition) and describes in some detail the kinds of prosodic characteristics and annotation that can help to improve parsing performance over conversational speech. Section 2.1 gives an overview of parsing technology and its application to conversational speech, and section 2.2 reviews the basics of speech recognition technology and its challenges in conversational speech. Section 2.3 discusses the ways that prosody can be described by linguistics, and also describes the higher-level cognitive and prosodic groupings that are used to separate and identify certain phenomena in conversational speech above the level of the word sequence. Section 2.4 discusses the relationship between these prosodic structures and syntactic structures and previous efforts to take advantage of these relationships. Finally, Section 2.5 discusses previous attempts to work on the problem of parsing automatically-recognized speech.

2.1 Parsing

Parsing is the extraction of syntactic structure from text (whether written or spoken). A **grammar** licenses some language: a possibly infinite set of word sequences that it allows. To describe natural languages, some level of recursively-nested **phrase structure** seems to be required. This **constituent structure**, the recursive grouping together of adjacent words and phrases into constituent phrases, is what parsers attempt to extract. Figure 2.1 shows an example bracketing of constituent structure

and the tree structure to which this bracketing corresponds.

A constituent structure (spans and labels) does not make up a complete representation of the structure of a sentence above the level of the word. For example, in figure 2.1, the labels do not reflect (in this example) the constraint that the subject noun phrase (*I*) restricts the form of the verb to *was* (that *were* is not acceptable in this position). These kinds of constraints require more sophisticated structure in addition to the constituent structure. Although there are regularities and language constraints that are not reflected by this analysis, for automatic approaches to grammar generation, labeled-constituent structure is nearly all that is available: much work is done based on the Penn Treebank (Marcus et al. 1993) labeled-constituent corpus. While there are other corpora that reflect more sophisticated relationships among the constituents (e.g. the hand- and automatically-generated HPSG-parsed treebank known as Redwoods (Oepen et al. 2002)), this thesis will consider only the Penn Treebank, specifically the conversational speech subset. Chapter 3 includes additional detail on the corpus used in this research.

2.1.1 Applications for parsers

From the point of view of linguistic study, designing a successful parser is interesting in its own right, but returning a useful parse structure is valuable for a number of other downstream applications. Knowledge-based machine translation systems (e.g. the LOGON project for Norwegian-English translation (Oepen et al. 2004)), information extraction systems (e.g. Schäfer (2003)) and other knowledge-driven systems use a parser to extract the structure and relationships among the words. Furthermore, the more data-driven approaches are rapidly starting to recognize the utility of identifying more than linear-sequence structure in other tasks. To name a few: current research in statistical question-answering and information extraction, e.g. Etzioni et al. (2005), requires at least the reliable knowledge of the structure of noun phrases; reading level assessment benefits from syntactic parse analysis (Schwarm and Ostendorf 2005);

```
(S
  (NP (PRP I) )
  (VP (VBD was)
    (ADJP (RB personally) (VBN acquainted)
      (PP (IN with)
        (NP (DT the)
            (NNS people) )))))
```

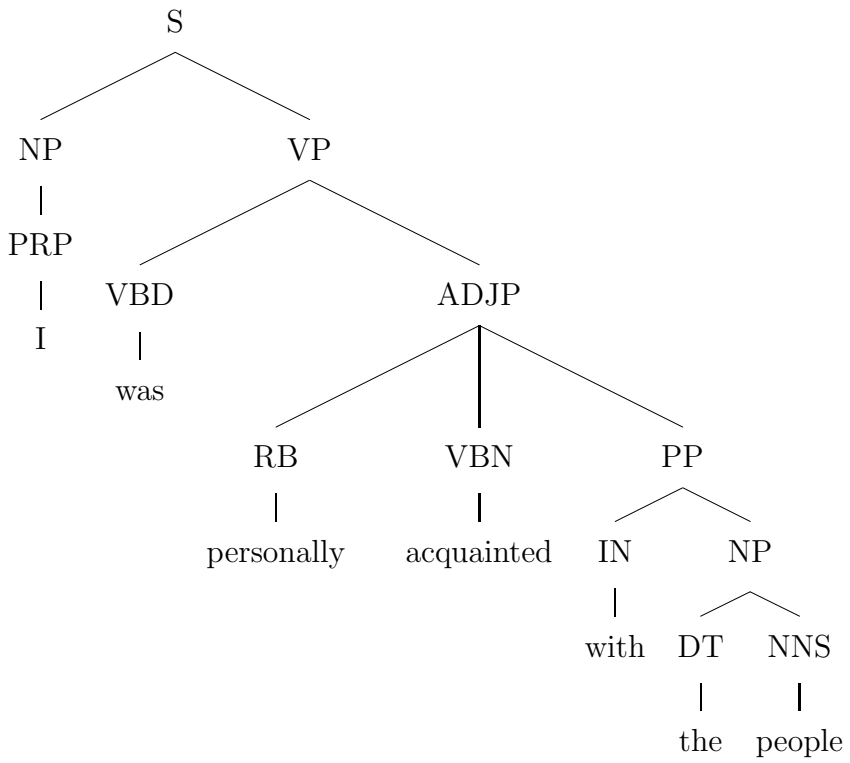


Figure 2.1: An example constituent structure for a short word sequence, as represented in Penn Treebank format or as a syntax tree. Note that each constituent span has a label. Every pair of matched parentheses (or subtree) can also be seen as a label over a specific substring of words. Also note that these labels do not reflect some grammatical categories like number.

language modeling (the prediction of likely word sequences) can (under some circumstances) be improved with good parse-tree knowledge (Chelba and Jelinek 2000, Chelba 2000, Roark 2001, Charniak 2001, Charniak et al. 2003); and even statistical machine translation (e.g. Yamada and Knight (2001) and Charniak et al. (2003)) has begun to explore the possibility of incorporating syntactic structure.

2.1.2 Context-free grammars (and some alternatives)

There are a variety of formalisms for specifying grammars for natural language. One very popular approach to modeling constituent structure (and the one adopted here) is to describe the constituents in a **context-free grammar** (or CFG) (Chomsky 1957), which consists of a set of **productions** and a lexicon.

Each production rule is made up of a left-hand symbol and a right-hand ordered list, as in figure 2.2.[1] The symbols that are used in the production rules fall into two groups: **terminals**, which are only allowed on the right side of productions, and **non-terminals**,[2] which may appear on the right or left side of productions. In a production, the left side represents a generalization over a sequence of the right side. Non-terminals reflect a generalization or grouping of constituents, abstracting away from the number or identity of the actual words. In a context-free grammar, then, the language L is the set of all word sequences w such that G contains rewrite rules that may be applied in an order such that the start symbol S may be rewritten as w.

A context-free grammar is not the only class of phrase-structure grammar. Alternatives include unification-based phrase-structure grammar frameworks like HPSG, outlined in Pollard and Sag (1994) and LFG, outlined in Bresnan (2001). In contrast

[1] In an extended variant of this format, alternations are allowed on the right-hand side of this list; for the purposes of this discussion, these can be understood as separate productions with a common left-hand side.

[2] Some formalisms (e.g. Chomsky (1957)) divide the "non-terminals" category described here into into "pre-terminals" and "non-terminals"; this distinction is not necessary for the purposes of this discussion.

$$
\begin{aligned}
\text{S} &\rightarrow \text{NP VP} \\
\text{VP} &\rightarrow \text{VBD ADJP} \\
\text{ADJP} &\rightarrow \text{RB VBN} \\
\text{ADJP} &\rightarrow \text{RB VBN PP} \\
\text{PP} &\rightarrow \text{IN NP} \\
\text{NP} &\rightarrow \text{PRP} \\
\text{NP} &\rightarrow \text{DT NNS} \\
\text{NP} &\rightarrow \text{DT NN} \\
&\ldots \\
\text{PRP} &\rightarrow \text{I} \\
\text{PRP} &\rightarrow \text{me} \\
\text{VBD} &\rightarrow \text{was} \\
\text{VBD} &\rightarrow \text{swam} \\
\text{VBD} &\rightarrow \text{acquainted} \\
\text{VBN} &\rightarrow \text{acquainted} \\
\text{RB} &\rightarrow \text{personally} \\
\text{RB} &\rightarrow \text{often} \\
&\ldots
\end{aligned}
$$

Figure 2.2: Example productions for a context-free grammar.

with unification grammars, CFGs do not — in their usual set of labels — reflect details like number and gender agreement, although by subdividing the non-terminal set sufficiently, such a division is at least logically possible. However, the CFG formalism is substantially restricted (especially when compared with HPSG and LFG) in that it may only explicitly constrain one step in the tree: there may be no production rule that explicitly constrains the grandchildren of a node. (Of course, the effect can often be achieved by sufficiently constraining sets of interrelated rules, as in GPSG (Gazdar et al. 1985), but the complexity of development of such a system can be prohibitively difficult.)

2.1.3 Probabilistic CFGs

In the **probabilistic context-free grammar** (PCFG) approach, all word sequences are assigned some non-zero probability, but unusual productions are assigned a low likelihood. This assignment of likelihoods to various syntactic constructions has been suggested as a processing model (Abney 1995, Jurafsky 1996), and there is some psycholinguistic research that suggests that this model of (un)likelihood has some correlation with comprehension and reading difficulty (Levy 2005).

A PCFG assigns a likelihood to the entire sentence by building it up out of a product of the production likelihoods. Thus, PCFGs make a number of independence assumptions (described in (Charniak 1993)). In a PCFG, the likelihoods of the local decisions are explicitly limited in the same ways that the licensing of phrase structure was constrained in a CFG (in section 2.1.2). In other words, likelihoods are assigned based on a single production, with only one parent and its children involved. The overall likelihood of the tree is thus made of the product of tree-local single-element probabilities, where each element considers only its immediate children in computing its local likelihoods. No long-distance relationships are considered.[3]

[3]Long-distance relationships in the lexical sequence may be considered in a PCFG, but the PCFG assumptions are that only *tree-local* dependencies are considered.

While it is in principle possible to assign likelihood to productions by any method, most PCFGs are learned from a corpus of existing phrase-structures; the most common corpus is the Penn Treebank (Marcus et al. 1993). Unsmoothed likelihoods are the easiest to compute: the likelihood of a given production $p(s_i \rightarrow e_j)$ (expanding symbol s_i to the expansion sequence e_j) is specified to be:

$$p(s_i \rightarrow e_j) = \frac{c_{\text{corpus}}(s_i \rightarrow e_j)}{c_{\text{corpus}}(s_i)} \tag{2.1}$$

where $c_{\text{corpus}}(x)$ is the count of times that production or symbol x is used in the corpus. In practice, these likelihoods are frequently **smoothed**, or reduced slightly in order to reserve some probability mass for those productions that might appear in new data but did not appear in the corpus in order to avoid assigning zero probability to any string.

2.1.4 Lexicalized CFGs

As an extension to the description of constituent structure mentioned at the beginning of this section, natural language constituent structure is often understood to be **headed constituent structure**. In this representation, each constituent has one **head constituent**, which is the subconstituent that is the grammatical core of the parent. For example, the head constituent of a noun phrase is usually a noun. The **head word** of a phrase ϕ, then, may be defined recursively as the head word of ϕ's head constituent, or (if ϕ contains only one word) that word. A constituent structure is called **lexicalized** when each constituent is additionally annotated with its head word. Figure 2.3 shows an example lexicalized constituent structure.

Lexicalization allows the extension of CFGs and PCFGs. Both CFGs and PCFGs may now include production rules that specifically refer to these head words. (These productions must also specify which child phrase is the head phrase in each production.) Alternatively, a constituent structure may be converted to a headed constituent structure by algorithm, e.g. with the algorithm in Magerman (1995).

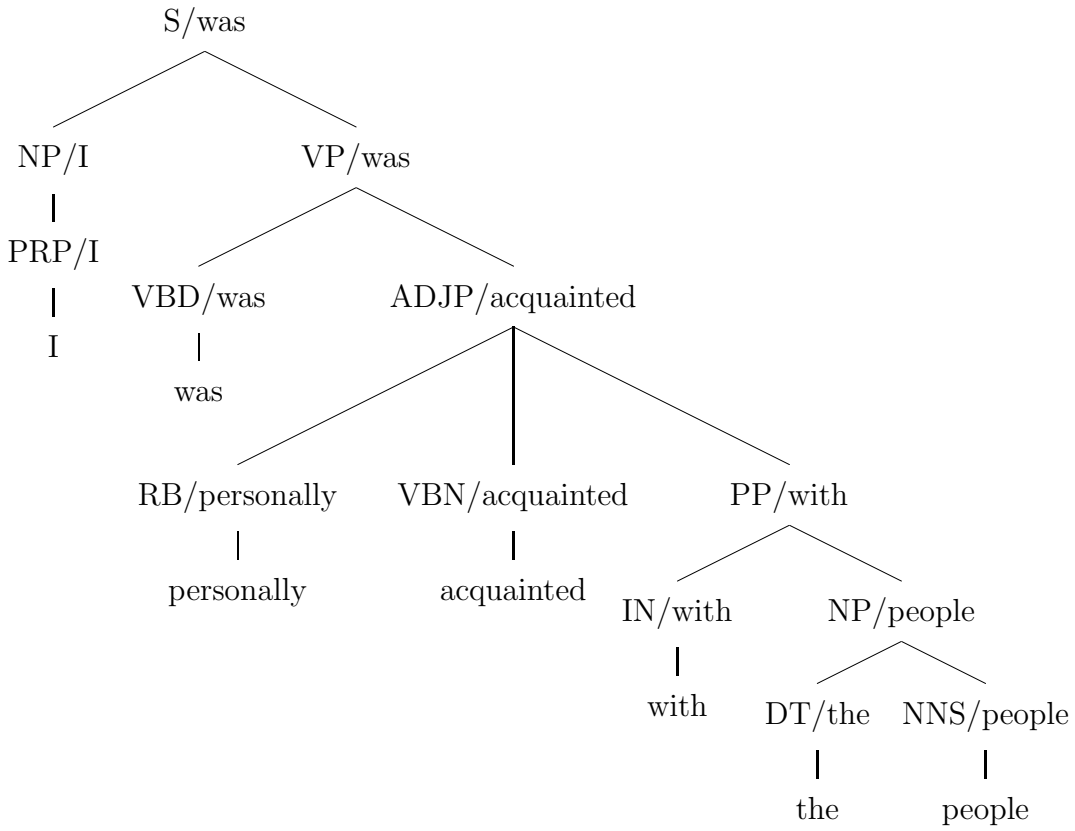


Figure 2.3: An example lexicalized constituent structure.

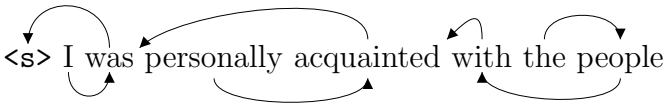


Figure 2.4: An example dependency structure, derived from the headed constituent structure in figure 2.3.

Lexicalizing a constituent structure allows the conversion of a constituent structure into a **dependency structure**: the head-words of the non-head phrases are marked as the dependents of the head word of each phrase. Each word, then, has exactly one dependency, except the top word (which we may consider dependent on an arbitrary start symbol, usually <s>). Figure 2.4 is the example dependency derived by this algorithm from the constituent structure described in figure 2.3.

Since Charniak (1993), the most popular approaches in working with PCFG-based parsers have incorporated lexicalization, basing conditional decisions on the head words of the component constituents, as well as the labels. PCFG-oriented systems (like Charniak (2000) and Charniak et al. (2003)) and head-dependency oriented systems (like Collins (2003) and Bikel (2004)) both use complex back-off schemes that assign likelihoods conditioned on the actual head-words, rather than merely the categories.[4]

2.1.5 Statistical parsing

The most common parsing task is to infer the correct constituent (relative to some human-annotated reference tree) structure over a word sequence. A PCFG — whether lexicalized or not — defines a method of selection of the best constituent structure, which is to choose the structure that is assigned the highest probability by the gram-

[4]Klein and Manning (2003a) have shown that grammar projections may be used to dramatically reduce the search space, and have also shown that with a deterministic adjustment to the label scheme ("grandparent annotation"), it may be possible to achieve comparable performance to lexicalized systems (Klein and Manning 2003b): lexicalization may not be entirely necessary! Despite these improvements, the state-of-the-art parsers over the standard task (the Penn Treebank) for the past several years have all been lexicalized (Charniak 2000, Collins 2003).

mar G:

$$\hat{s} = \operatorname*{argmax}_{s} p_G(s) \ . \tag{2.2}$$

A statistical parser thus must solve two problems: in training, it must infer the probabilities of the various productions from available training data, and at run-time, it must select the parse tree over the input word sequence that maximizes the total probability assigned to the sequence. The first task (training) may usually be accomplished by counting instances in a corpus, as suggested in section 2.1.3, while the second has usually been accomplished with heuristic search (e.g. Charniak (2000)), because the potential search space is exponentially large in the length of the sentence.

A parser, however, need not select only the one best parse. Instead of seeking the one best (with argmax), a PCFG-based parser can — at least in principle — return an ordered (or scored) list of the n best parse options, given the same input. Charniak (2000) and Bikel (2004) (a reimplementation of Collins (2003)) both offer options to return a pruned list of hypotheses and their scores as evaluated by their lexicalized PCFG.

2.1.6 Evaluation with PARSEVAL

Statistical parsers are usually evaluated against a held-out section of the corpora on which they are trained. To be experimentally valid (and scientifically interesting) the test corpus should consist of input word sequences that were not in the training corpus.

The usual evaluation measure for phrase-structure parsing (PARSEVAL, Black et al. (1991)) compares a hand-annotated phrase-structure to the structure proposed by the parser, and evaluates their agreement in terms of bracket precision, bracket recall, and bracket crossing. **Bracket crossing** describes the number of brackets in the hypothesized phrase structure that cross one of the brackets in the reference phrase structure — that is, the two brackets could not exist in the same phrase

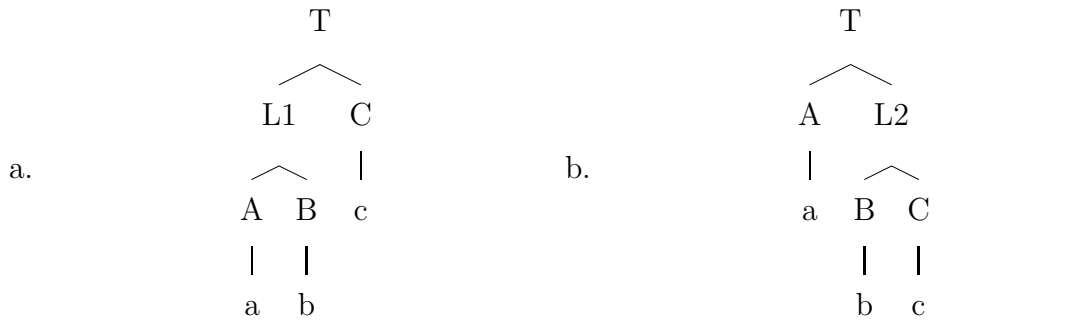


Figure 2.5: An example of bracket crossing. Bracketing (b) has exactly one bracket crossing with respect to bracketing (a): The L1 and L2 brackets from (a) and (b) respectively could not exist together in any well-formed bracketing, because they are not properly nestable.

structure, because the two spans overlap and neither contains the other completely. Figure 2.5 demonstrates an example bracket-crossing. As one might expect, a better parser will have a lower bracket-crossing count. In contrast to decreasing bracket-crossing, good parsers will try to *increase* **bracket precision** and **bracket recall**, which are measured in terms of the number of constituent labels that match between the hypothesized and reference structures. Precision and recall are measured using three component quantities: the number of hypothesized bracketings, the number of reference bracketings, and the number of correct bracketings, defined as follows:

hypothesized bracketings are those bracketings that are predicted automatically by the parser,

reference bracketings are those bracketings that are in the gold standard phrase structure (as defined by a hand-annotated reference file), and

correct bracketings are those bracketings that are in *both* the reference and the hypothesized bracketings.

Thus,

$$\text{bracket precision} = \frac{\#\text{ correct}}{\#\text{ hypothesized}} \tag{2.3}$$

$$\text{bracket recall} = \frac{\#\text{ correct}}{\#\text{ reference}} \tag{2.4}$$

The PARSEVAL measure provides for both **labeled** and **unlabeled** versions of these measures: in the former, the correct bracketings must match both label and span; in the latter, only the span is matched.

It is common to combine these precision and recall measures with the F**-measure**, the (weighted) harmonic mean of precision and recall:

$$F_\alpha = \frac{1}{\frac{\alpha}{\text{precision}} + \frac{1-\alpha}{\text{recall}}} \tag{2.5}$$

The term α in equation 2.5 reflects a differential weighting of precision vs. recall, but since there is no reason to prefer precision over recall, α is usually set to 0.5, yielding the commonly-used formula in equation 2.6:

$$F = \frac{2 \cdot \text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \tag{2.6}$$

2.1.7 Discriminative reranking of parse candidates

PCFGs are a maximum likelihood approach to selecting the best parse (or parses) from a candidate set. They effectively ask "Of all these parse candidates, which one is the most likely to have been generated by this PCFG?" Another approach is to ask instead: "Compared to the others here, which is the best parse?" "Best", of course, is a matter of definition, but since there are standard evaluation measures (e.g. PARSEVAL), we can measure "best" directly — and try to learn a model that separates good parses from bad ones based on the evaluation criterion.

By targeting the final evaluation measure as a direct optimization criterion, one can move away from the maximum-likelihood models that make up the PCFG. The PCFG (and lexicalized PCFG) independence assumptions require that the overall

probability assigned to a parse candidate must be built up over tree-local decisions. In this direct approach, as in Collins (2000) and Charniak and Johnson (2005), the n-best candidates selected by the PCFG may be treated as a set of candidates for discriminative reranking. These systems use the tree-local lexicalized-PCFG likelihoods as one of a wide variety of features, most of which are not reflected in the tree-local assumptions that make up the PCFG. The experiments in chapters 5 and 6 use just such a reranking framework.

2.1.8 Parsing spoken language

Parsing speech, poses a number of new challenges. First of all, the words themselves are not clearly delimited in the way that they are when presented as a written form. In a fully-automatic system, the parser would need to be downstream from a speech recognizer, with all the potential for error that is suggested by that arrangement. Even when the words are recognized entirely correctly (which is rare, especially for spontaneous speech), spoken language uses different grammatical patterns than written language does.

Spoken language is usually a different register than written language. For example, while a written English-language text may begin a new thought with *however*, a conversational speech text is far more likely to begin new thoughts with *anyway*, *but*, or *and*, as suggested by the examples below. In these examples, underlined regions highlight parts of the transcript that do not correspond well to written language.

A5: Uh-huh.

B6: uh, eight months ago I guess. And, uh, [we were + I was] fortunate in that I was personally acquainted with the, uh, people who, uh, ran the nursing home in our little hometown.

A6: Yeah.

B8: So I was very comfortable, you know, in doing it when it got to the point where we had to do it. But there's, well, I had an occasion for my mother-in-law who had fell and [needed to be +] you know could not take care of herself any more.

Note the disfluencies and fillers ("uh", "you know"), backchannels ("uh-huh", "yeah"), self-corrections ("we were – I was"), and planning failures ("fell" instead of *fallen*).

In addition to differences in register, written language is the product of a planning and editing process which is not usually applied to casual spoken language. Spoken language also introduces **speech repairs**, where the speaker breaks off in order to abandon or revise part of the current utterance, as in the example above. In the example above, + indicates the interruption point, and the bracketing indicates the region involved in the repair.

Finally, spoken language, unlike written language, does not have clear boundaries like sentence and paragraph breaks. In a conversation, speaker turns can be relatively easy to segment (e.g. in two-wire telephone speech, where the signal is often clearly from one wire or the other) or very difficult (single-microphone recordings of multi-party meetings). In either case, *within* a speaker turn, the edges of sentences (or sentence-like units) are not delimited by any straightforward markers: punctuation is not spoken.

In a full speech-understanding system, any parser would be dependent on word hypotheses generated by a speech recognition system; hence speech recognition is reviewed next. Section 2.5 discusses some existing work in that context and the difficulties of evaluating parses when even the input words are in question.

2.2 Speech Recognition

Automatic speech recognition (ASR) systems are generally made up of two components, an **acoustic model** and a **language model**. Speech recognizers try to select

the most likely word sequence $\hat{W}$ from the language L given an observation sequence O, according to the product of these two models:

$$\hat{W} = \underset{W \in L}{\operatorname{argmax}} \frac{p(O|W)p(W)}{p(O)} = \underset{W \in L}{\operatorname{argmax}} \overbrace{p(O|W)}^{\text{acoustic model}} \overbrace{p(W)}^{\text{language model}} \tag{2.7}$$

The acoustic model component assigns likelihoods to observations given words, and the language model component assigns likelihoods to word sequences.

2.2.1 Search complexity in ASR

The argmax term in equation 2.7 introduces a tremendous search space over all possible W: in natural languages L, the space of possible W is both potentially infinite (depending on the structure of the language) and effectively very large (there is a huge variety of sentences actually spoken in natural languages). Thus, practical approaches to this search problem incorporate strong conditional independence assumptions, which allow the use of dynamic-programming techniques like Viterbi decoding to drastically reduce the search space, in addition to n-best sentence hypothesis rescoring to incorporate more complex models that characterize longer-distance dependencies.

2.2.2 Evaluating ASR with word error rate

Speech recognition systems are conventionally evaluated with word error rate (WER), following equation 2.8:

$$\text{WER} = \frac{\text{deletions} + \text{insertions} + \text{substitutions}}{\#\text{ of correct words}} \tag{2.8}$$

Some variants (e.g. those implemented by the NIST `sclite` software utility (NIST 2005)) can be configured to ignore certain classes of substitution errors, e.g. contractions like *I'm* and *I am*, and to ignore certain classes of words entirely, like filled pauses (e.g. *uh*). Despite allowing some flexibility in this way, this error measure

still treats the word sequence as the primary objective, treating virtually all words as equal, and taking into account no measures above the level of the word sequence.

2.2.3 ASR and conversational speech

Many common applications for speech recognition are focused around communicating with or through machines (e.g. dialog systems like Dahl et al. (1994)), communicating with machine aid (e.g. speech-to-speech translation like Precoda et al. (2004)), or controlling machines (e.g. the medical transcription task, as in Mohr et al. (2003)). However, speech recognition has applications in human-human speech as well. Large corpora of human-human communication (e.g. in the MALACH project for Holocaust survivor narratives (Oard et al. 2002)) can contain thousands of hours of information, but be relatively inaccessible to researchers, due simply to their sheer size. A good indexing of these conversations — or other large corpora of human-human communication, such as meeting recordings (NIST 2004), call center data, voicemail, or congressional or other parliamentary hearings — would benefit from an automatic transcription of these conversations. Thus, ASR has an application in transcribing human-human conversations as well as in mediating human-computer interactions.

Automatic speech recognition performance varies widely depending on its context. Under ideal circumstances:

- the speech recordings are in noise-free environments, the recordings use high-quality microphones and high sampling rates, and the speech recognizer's training database of examples was recorded under the same circumstances,

- the training database includes a large body of spoken language from this same speaker, and

- the utterances to be recognized are well-separated by periods of silence, with careful and uniform articulation of each utterance.

However, this ideal situation is frequently compromised in practice. In most speech recognition tasks, systems are constrained to work in the real world: there may be noise in the environment, the recording devices available may be less than ideal, and there may be no training data for this speaker available, especially for automatic transcription of large multi-speaker corpora like MALACH. Robustness to these sources of variability is difficult and critically important.

Spontaneous conversational speech introduces further challenges beyond these technical ones: spontaneous conversational speech between humans rarely separates utterances with clean boundaries like periods of silence, and even when that speech is well-separated, spontaneous speech exhibits a wide variety of hypospeech and undershoot phenomena (Lindblom 1990), such that many words are not articulated in their citation form. For example, Greenberg's transcription studies (Greenberg et al. 2003) of the Switchboard corpus show dozens of pronunciations for short, high-frequency words like *and* and *the*.

Speech recognition of spontaneous conversational telephone speech (as in the Switchboard corpus, discussed in more detail in Chapter 3) combines all of these difficulties to present a challenging task for speech recognition research: less than ideal recording environments, spontaneous speech with all its attendant difficulties, and multiple speakers that differ between training and test environments.

2.3 Prosody and metadata

"Segments" of speech are usually considered to be the sequence of vowels and consonants that build up words. At the same time as articulating these segmental sequences, speakers of all human languages also execute changes in articulatory movements that exist at a larger timescale than the individual segment or syllable. In particular, one set of variations (those having to do with pitch, loudness, and tempo)

are known as prosody.[5] As discussed in Carmichael (2005), prosodic features serve a variety of purposes for a speaker in conversation: they can impart emotional state or affect; they can identify new, old, or contrasting information; they can identify overall discourse structure; they can identify register, dialect or social role; below the level of the word, prosodic structure can indicate sublexical structure (e.g. lexical stress), and there are strong dependencies between prosodic structure and segmental realization as well. Most interestingly for the purposes of this work, prosodic features of speech can indicate information about the grammatical structure and both internal and external boundaries of the utterance.

This work puts aside the other aspects of prosody, and focuses on the possibilities for prosody to indicate grammatical and boundary information for conversational utterances. In particular, this section focuses on the annotation standards and descriptions that may be useful in computational approaches to prosodic information.

2.3.1 Symbolic representations for prosody

In computational approaches like this work, it is useful to find a representation of the parameter in question (prosodic information) that can be made specific enough to be implemented on a computer. Of course, prosodic characteristics like amplitude and pitch can be recorded directly, but for the sake of generalization across (and within) speakers, it is useful to find and use abstractions of a higher level. This work attempts to identify high-level prosodic features that can help to inform parsing. What is sought, then, is a set of prosodic parameters that can be used in a more general way than the raw acoustic information.

One approach to identifying higher-level prosodic information is what Ladd calls

[5] In English, these characteristics almost never serve to distinguish among different words, but in tonal languages like (e.g.) Mandarin Chinese, different pitch and timing may distinguish among different lexical items. However, speakers of tonal languages still vary the pitch and timing information to carry paralexical information indicating phrase structure, provided that they match the tonal constraints imposed by the lexicon, and so all the information borne in prosody discussed here still applies to those languages.

```
and uh we were I was fortunate in that I was personally acquainted ...
  1 1   1   2p 1   1         1  1    4 1   1          1          1

 [ and uh we were ] [ I was fortunate in that ] [ I was personally acquainted ...]
```

Figure 2.6: An example word sequence, with break indices. The lower part of the figure represents the prosodic phrase grouping implied by the break indices.

the "Autosegmental/Metrical" (AM) model, which "adopts the phonological goal of being able to characterise contours adequately in terms of a string of categorically distinct elements, and the phonetic goal of providing a mapping from phonological elements to continuous acoustic parameters" (Ladd 1996:42). In this framework, the higher-level contour events (tones and boundaries) are associated with other language-features like words and syllables, in a more abstract way than strict alignment to particular timepoints. This idea draws its inspiration from the autosegmental approach to tonal phonology (Goldsmith 1979). The symbols and annotations described in this framework intend to be phonologically-relevant: that is, they reflect the prosodic variations that are meaningful in the language in question, much as the realization of lexical tone is a language-specific phenomenon (for those languages that have tone).

The most popular annotation standard in the AM tradition is the ToBI annotation system (Silverman et al. 1992). This system decomposes prosodic behavior into two main kinds of symbolic components: a linear sequence of **tones** and a hierarchical grouping of **intonational phrases** and lower-level prosodic phrases. The phrases are delimited by boundaries with **break index** values, indicating the degree of separation. Phrases may be nested, and the break index can be understood as an index of the size of the prosodic phrase just closed. Figure 2.6 has an example of a word sequence with break indices inserted, and the corresponding prosodic groupings reflected by these breaks.

A simplification of this variety of prosodic information (ToBI annotations of English) that is adopted in this work. In particular, it is worth noting that ToBI incorporates not just tones, but also boundary information. In ToBI, **break levels** indicate the connectedness of two words — every word is associated with a break level (that goes with its right side boundary). Break levels are non-recursive: they delimit a very small number of prosodic phrase types, which do not recursively nest, and fall in one of a small number of levels. This non-recursivity allows the phrases to be annotated only by their edges:

break level 0 describes a word that has been cliticized to its right neighbor;

break level 1 describes a continuous-speech boundary;

break level 2 describes a boundary that has either a phrasal lengthening (and no tone) or a phrasal tone (and no lengthening);

break level 3 describes a boundary that has a phrase tone and lengthening, and

break level 4 describes a boundary that has a boundary tone and phrase-final lengthening.

In addition, the annotation scheme includes a `p` diacritic, used for annotating disfluent boundaries. Since the `p` diacritic is never used with break levels 0, 3, or 4, this diacritic essentially introduces two other labels:

break "level" 1p describes a disfluent boundary with concurrent prosodic shortening (i.e. word fragments and abruptly shortened words), and

break "level" 2p describes a disfluent boundary with concurrent prosodic lengthening (i.e. hesitation)

2.3.2 Acoustic measures of prosody

Some alternative computational approaches to characterizing prosodic features are relatively close to the phonetic information. One example of this approach is Fujisaki's (2004) parameterization of f_0 for speech synthesis, which analyzes an f_0 curve as built from a "phrase command" (a long-timescale exponential decay) and smaller time-scale "accent commands" (local pitch-raising perturbations), each of which is parameterized with continuous time and height parameters. Another example of a phonetically-oriented computational model of prosody is Taylor's (2000) Tilt model, which incorporates information about amplitude, duration, and the f_0 curve shapes, assigning "events" with different f_0 shapes (tilts) to different sections of the speech signal. The Tilt model uses continuous parameters for tilt shape, but also removes the exponential phrase-command constraint from Fujisaki's model, allowing it to model a wider variety of long-term constraints. The Tilt model is a higher level of abstraction because it represents events as first-order objects in their own right, with secondary parameters representing their shape, while the Fujisaki parameters model the pitch perturbations of accent more directly.

In these approaches (as in the symbolic labels approaches above), the acoustic measures of prosody must be normalized for the speaker's possible pitch range, loudness and baseline. To be used in an engineering or scientific perspective, the parameters would ideally be generalizable to any speaker of that language. To do this generalization, the parameters learned here (or the tonal and break levels described in ToBI) must be matched to the speaker and utterance. From an algorithmic point of view, without this normalization, the raw values are far too distributed and scattered to be useful in extracting prosodic measures of any kind (a large man's "high" pitch accent target may be substantially lower than a small woman's "low").

2.3.3 Metadata in speech

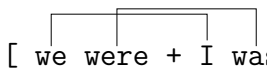
As shown in the previous sections, one may consider prosody information on a continuum, from very raw acoustic information on one end, towards more categorical and descriptive "phonological" information on the other. One may extend in the latter direction and examine other classes of symbolic representation. Some (e.g. Wightman (2002)) criticize ToBI for its decision to represent prosodic characteristics at a high phonological level, but not quite so high that native speakers find it easy to assign labels. Wightman's recommendation ("label what you hear") asks instead that labelers target higher-level representations that are easy for annotators to assign. One annotation and extraction effort that might be considered a candidate for this kind of approach is the metadata extraction task (Strassel 2003), which seeks to automatically identify **metadata** on speech. There are a number of kinds of metadata that are interesting to algorithmically extract, including the location of conversation-structuring words like fillers, backchannels, and discourse markers. In addition, metadata is usually understood to include the boundaries of **sentence-like units (SUs)** and **edit regions**. This work focuses in particular on the last two categories.

Sentence-like units (SUs) are intended to reflect a "whole thought" within a speaker's discourse. The SU may or may not correspond to a grammatical sentence by written standards — it may, for example, correspond only to a single backchannel like *yeah* — but the metadata annotation target is to model the human segmentation behavior, so an interaction with a syntactic model is to be expected. A transcript of a speech sample, segmented into SUs, is presented below:

```
uh eight months ago I guess.  / And, uh, we were + I was fortunate
in that / I was personally acquainted with the, uh, people who,
uh, ran the nursing home...
```

Edit regions, on the other hand, are closely tied to the online nature of speech: whether due to planning difficulties or to other concerns, speakers will sometimes

revise or repeat what they are saying "midstream" in a **disfluency**. A disfluent **edit region** is characterized with several points and regions: the **interruption point** (which is the point at which the speaker broke off), the **reparandum** (the part to be repaired), and the **repair** (the speech provided after the interruption point, intended to replace the reparandum). The reparandum and editing phrase together may be referred to as an *edit* or *edit region*. Edit regions are difficult to model with HMM or PCFG models, because these models can induce only linear or tree-structured dependencies between words. However, the relationship between reparandum and repair seems to be quite different: the repair is often a "rough copy" of the reparandum, using the same or very similar words in roughly the same order. A language model characterizing this dependency with hidden stack operations is proposed in Heeman and Allen (1999). An example edit region is below:



While there are other ways of describing disfluency (e.g., the ToBI guidelines list a `disfl` tag for secondary annotation), the characterization of the region above matches that of the Meteer et al. (1995) annotation effort, and there is therefore a reasonably-sized corpus of conversational speech (see Chapter 3) that has been annotated for these SUs and edit regions.

2.3.4 Automatic detection of prosodic and metadata events

In a working system that operates over real input data, the detection of these metadata events (and other prosody events as well) can be quite challenging, due to the complex relationship between the high-level events (like SU boundaries) and low-level events (like pitch or amplitude changes), not to mention the interaction among (e.g.) prosodic events like tone, prominence and boundaries.

Shriberg and Stolcke (2004), among others, describe a variety of applications for prosody models that operate from the raw acoustic features, with no intermediate

symbolic prosody representation. Their system uses lower-level acoustic cues (e.g. f_0 and normalized f_0) to discover metadata (and even higher level events like dialog acts). Indeed, they argue that modeling intermediate structures (like tones and prosodic phrases) is costly in computational terms, expensive to annotate, and may even lead to a degradation of performance, because of an optimization on the intermediate target rather than the higher-level task. Nevertheless, a variety of systems model these intermediate prosodic features.

Among those systems that try to predict prosody parameters and events, we may break the research into two groups. Some research has tried to predict the parameters of the acoustic models (those described in section 2.3.2), and other research tries to predict the symbolic labels assigned by human annotators (section 2.3.1). Rossi et al. (2002) is one of the former, and proposes a mechanism for the automatic extraction of Fujisaki parameters. Cutugno et al. (2002) and Martin (2004) provide automatic tools for prosodic analysis at the segment level and below, and Mixdorff (2002) proposes (inter alia) that there may be automatic ways to relate Fujisaki parameters to ToBI labels.

Boundary prediction is one of the areas of focus in this work, because prosodic phrase boundaries seem to be correlated with syntactic phrase boundaries. Nevertheless, prosodic phrase boundary identification is not an easy task: prosodic phrase boundaries occur with a wide variety of low-level features, and some of those raw features that might seem reasonable as boundary cues (e.g., pause length after a word) may occur in situations that nevertheless do not correspond to a deep boundary (especially when disfluencies are taken into consideration). As Ladd (1996:235) writes:

> It is universally assumed that one of the functions of prosody is to divide up the stream of speech into chunks or phrases of one sort or another.... Despite the apparent universality of the chunking function, however, [these

> phrases and their boundaries] are remarkably difficult to define and to identify consistently. [The] boundaries seem to take on a bewildering variety of manifestations, from a clear pause accompanied by a local f_0 fall or rise, to a subtle local slowing or pitch change that defies unambiguous definition.

Correspondingly, research efforts to automatically extract boundary labels use learning techniques that can take advantage of multiple input features. Wightman and Ostendorf (1994) use decision trees and a Markov sequence to model ToBI-style break indices. Vereecken et al. (1998) use cascaded perceptrons to do automatic prosody labeling on a small corpus of careful speech in six languages. Batliner et al. (1999) use linear discriminant analysis to identify a handful of powerful features that identify symbolic prosody boundaries from a very large feature vector. Chen et al. (2004)a uses an artificial neural-network; later work by Chen et al. (2004)b uses maximum-likelihood Gaussian mixture models to do prosody prediction; and Ananthakrishnan and Narayanan (2005) uses coupled hidden Markov models. These last three systems all also pull in syntactic features (part-of-speech tags) to assist with prosodic labeling.

Speech segmentation into sentence-like chunks (the identification of SUs) has become a problem of some interest over the last few years, especially with the creation of the EARS task (Liu et al. 2005). Shriberg et al. (2000) use lower-level prosody information to extract these higher-level tasks (sentences), while a variety of systems (Stolcke et al. 1998, Warnke et al. 1999, Kim and Woodland 2001, Ferrer et al. 2002, Srivastava and Kubala 2003) try to predict other high-level prosodic boundaries as well (e.g., in the case of Kim and Woodland (2001), the location of punctuation). More recent work uses a variety of classifiers (Liu et al. 2005) to approach the SU classification problem in the EARS framework.

Efforts to identify interruption points (e.g. Liu et al. (2003), Baron et al. (2002)) have largely been using the same classification frameworks as SU, because the two

tasks are interrelated: interruption points and sentence boundaries are boundary events. Other tools are used to identify edit regions. One promising direction for this latter task takes advantage of the syntactic similarity between the repair and the reparandum (something not well-represented by a context-free grammar), and combines a conditional random field with a model of syntactic quality of the remainder (Johnson and Charniak 2004).

Chapter 4 explores how data of this form (especially SU boundary decisions) impact the efficacy of a statistical parser.

2.4 Prosody and syntax

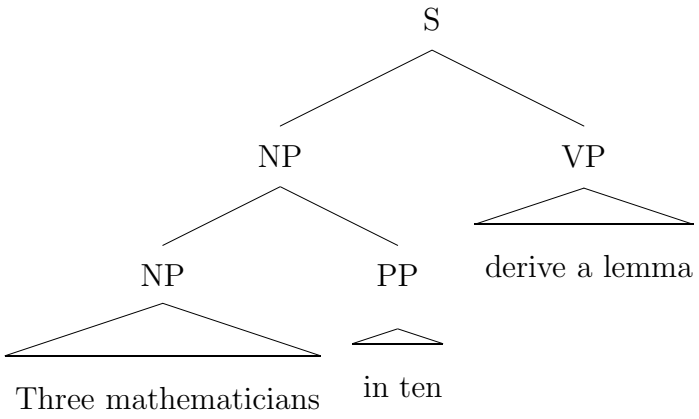
While Charniak and Johnson (2001) demonstrate that syntax can be used to help identify edits, it is worth investigating the relationship between syntax and prosody in more detail. While most linguistic theory acknowledges that there is some relationship between syntax and prosodic structure, this acknowledgement is noticeably vague about what the nature of this relationship is. In this section, I explore what proposals have been made to relate these areas.

2.4.1 Theories relating prosody to syntax

Syntactic boundaries and prosodic boundaries seem to correspond — at least in some circumstances, and there is a variety of linguistic literature that tries to make sense of this. Most of these models adopt the approach that the two structures are compatible but separate,[6] and that neither syntactic nor prosodic structure is entirely determined by the other. Their interaction is complex, as reflected by the variety of proposals for how they fit together.

Thus, one major question in relating prosody to syntax is when a prosodic phrase

[6]However, there are some exceptions that treat syntax and prosody together, e.g. Steedman (2000) and Blache and Hirst (2001). The Construction Grammarians also have made proposals in this direction (e.g. Brenier and Michaelis (2005)).



(a) [Three mathematicians in ten] [derive a lemma]

(b) *[Three mathematicians] [in ten derive a lemma]

Figure 2.7: Acceptable (a) and unacceptable (b) prosodic phrasings. Note that both prosodic phrasings end at constituent boundaries. Selkirk attributes the unacceptability of (b) to a violation of her "sense unit" constraint.

may (or may not) correspond to a syntactic phrase. Figure 2.7 (derived from Selkirk (1984:292)) demonstrates an example sentence that allows some phrasings but not others. Selkirk (1984:285) accounts for this difference in acceptability by proposing that prosodic phrase structure may be broken into groups at any position, but that these prosodic phrases must also meet some semantico-syntactic constraints. In particular, she proposes that each prosodic phrase (IP in her use) must form a "sense unit":

> Two constituents C_i, C_j form a sense unit if any of the following hold:
>
> a C_i modifies C_j (a head)
>
> b C_i is an argument of C_j (a head)
>
> c C_i and C_j together form a syntactic constituent.

(adapted from Selkirk (1984:291))

In addition, Selkirk proposes that phrasal breaks are stronger (have an increased number of "silent demibeats") depending on the number of constituents, phrases, and top-level "daughter phrases of S" that end at that point.

Auer (1996) points out that turn changes in conversational speech tend to happen at closed syntactic boundaries, but argues that these boundaries interact such that

> A context-free, purely syntactic definition of a closed syntactic gestalt is difficult, even impossible to give. There is a certain temptation to define a minimal syntactic gestalt as consiting of a finite verb plus its obligatory arguments (in full or anaphorically abridged form). However, it is not clear that what is obligatory can be stated in ways which do not recur to sematics or pragmatics.... (Auer 1996:61)

In more recent phonological theory, Optimality Theory (OT) has become quite popular. In OT approaches, decisions are not made by rule but by multiple ranked constraints. In this framework, multiple candidates compete and are eliminated by successively lower-ranked constraints. Thus the winning candidate is the one that violates the fewest high-ranked constraints. This theory can account for large-scale tendencies that are not uniformly followed, in that these constraints may be violated if a higher-ranked constraint eliminates competitors. Gussenhoven (2004), and the earlier work by Truckenbrodt (1999), among others, propose high-ranking constraints on the interaction between prosodic and syntactic phrasing, e.g. Truckenbrodt's Wrap-XP constraint:

> Wrap-XP: Each XP is contained in a phonological phrase.

If this constraint is ranked fairly highly, they propose, it may account for the general tendency of phonological phrases to coincide with *some* syntactic phrase. In this

theoretical approach, other constraints, including one encouraging an optimal phrase length, account for the non-uniformity of application of this constraint.

2.4.2 Psycholinguistic evidence relating prosody and syntax

With these several theories as a framework, one might expect to find quantitative work that explores how speakers and listeners use this correlation between syntactic and prosodic structure. As summarized in Cutler et al. (1997), listeners' boundary decisions may be based on durational information or other pre-boundary lengthening, or pitch contour variation (e.g., in Wightman et al. (1992), which explores durational cues for syntax). In more recent work, prosody has been demonstrated to be interestingly correlated phrase and clause boundaries in Swedish spontaneous speech (Strangert 2004, Heldner and Megyesi 2003) and (planned) Korean speech (Kang and Speer 2002).

Furthermore, phrasal attachment decisions (especially PP attachment decisions) have been demonstrated to be influenced by prosodic structure, for example in Price et al. (1991), Marslen-Wilson et al. (1992), Pynte and Prieur (1996), and more recently in Kang and Speer (2004) and Mani (2004).

Schafer et al. (2000), Snedeker and Trueswell (2003), and Kraljic and Brennan (2005) found that speakers produce prosodic cues for syntactic disambiguation — even when contextual information available allowed for only one interpretation. Furthermore, Kraljic and Brennan (2005) also demonstrate that, in those situations where the context allowed ambiguity, listeners often do not even consider the unintended interpretation (as determined by eye-tracking). As they write,

> This evidence that prosody plays a strong and reliable role in disambiguation is good news for researchers working on machine processing of spoken dialog, because it means that prosodic cues are likely to reflect syntactic structure in a consistent way, and so prosody is well worth pursuing as a

cue for machine parsers of human speech. (pp. 213–214)

Accordingly, we turn to the history of work done in using prosody in parsing speech.

2.4.3 Prosody and parsing

There have been a variety of efforts to adapt prosodic information into parsing models. Some early attempts use prosody to prune bad parse choices (Bear and Price 1990) or to rank candidate parses (Veilleux and Ostendorf 1993, Ostendorf et al. 1993). Bakenecker et al. (1994) and Kompe et al. (1997) do this pruning as well, and use the prosody to help drive a rule-oriented translation in the VERBMOBIL project (Nöth et al. 2000). Some work also uses prosody to identify empty categories (the results of transformational movement) in the VERBMOBIL project (Batliner et al. 1996, Batliner et al. 1998, Batliner et al. 2001). Work in this lineage includes spoken-language understanding work that targets not syntactic structure but higher-level tasks (Nöth et al. 2000, Nöth et al. 2002) like travel-information translation, or sentiment analysis and task-dependent translation, as in Bhagat et al. (2003). In another approach, Core and Schubert (1999) uses parsing and prosody information together to identify repair regions.

Although PCFG-driven parsers are now the state of the art in parsing (as discussed in section 2.1.3), most of the approaches described here are based on earlier work that did not use PCFG parsers. Other formalisms used have included Seneff's (1992) TINA speech understanding system (Veilleux and Ostendorf 1993), a "concept-based" approach (Mayfield et al. 1995), or a "chunky semantic parser" (Haas et al. 1998, Boros et al. 1998), rather than a context-free grammar. Some recent work in trying to integrate PCFG parsing and prosody (Gregory et al. 2004) tries (and reports failure) to integrate prosody into parsing by inserting new word tokens with raw prosodic features after some words. However, by introducing prosody information as lexical tokens, this approach reduces the available history for the probabilistic

learner, and the raw-feature token descriptions do not reflect any relationship to the perceptual prosodic phrase. In this work, I explore the introduction of perceptually-labeled prosody in Collins' reranking model (Collins 2000), which does not introduce this problem of reduced history. Chapter 5 explores a way to incorporate prosodic information into a statistical parser.

2.5 ASR and parsing

In addition to information from prosody, it is also interesting to incorporate uncertainty in the word sequence itself into the parsing problem. Speech understanding systems incorporate word-recognition uncertainty, and incorporate various kinds of parsing based on the word-level uncertainty.

The JUPITER (Zue et al. 2000) English weather dialog-system, and the derived Chinese system MUXING (Wang et al. 2000) and Japanese system MOKUSEI (Nakano et al. 2001) use the TINA natural language parser (Seneff 1992), which is an adaptable but manually-constructed parser over limited domains. The parse targets in this context are usually at a higher-level of predicate logic, with types like `APlace` and `RelTime`, rather than "noun phrase" or "adverb". In these systems, an n-best recognizer hypothesis list is generated and then searched by the parser to find the optimal parse structure. These parse choices are integrated based on a combination of recognizer score, dialog context, and syntactic and semantic well-formedness. Likewise, the limited-domain travel dialog systems like EVAR (Gallwitz et al. 1998, Nöth et al. 2001), SQEL (Aretoulaki et al. 1998), and VERBMOBIL, as well as the Berkeley Restaurant Project (Jurafsky et al. 1995) also do not target parses directly, but target task-specific relations of the sort that could directly build database queries (rather than language-general syntactic structure). Haas et al. (1998) and Batliner et al. (2001) use somewhat more general parsing approaches, but still target task-specific "concepts" rather than syntactic structure directly.

There has not been very much work on using task-neutral syntactic parsers on speech, especially in relatively unrestricted domains such as the Switchboard corpus of conversational speech. One of the best-known recent efforts is Charniak and Johnson (2001). Though not focused on parsing speech, parsers have been used to reduce word-error rate automatic speech recognition by using the syntactic structure as a model of long-distance interrelationships between words, as discussed in section 2.1.1 (Stolcke et al. 1996, Chelba and Jelinek 2000, Chelba 2000, Roark 2001, Charniak 2001, Charniak et al. 2003).

Furthermore, there has also not been much work studying the sorts of degradation to parse accuracy that comes along with parsing in the downstream from an ASR engine. Some of the task-specific systems mentioned above (e.g. Zue et al. (2000)) evaluate success based on the discovery of correct understanding in the dialog context (as a part of the concept-driven speech understanding task). This work has been developed mostly with the PARSEVAL metric (Black et al. 1991), described in section 2.1.6. This lack of research may be due to restrictions imposed by the PARSEVAL evaluation measure: when the words are different between the reference transcript and the hypothesis, it is difficult to say whether a particular span is in both.

To cope with this problem, the 2005 Johns Hopkins CLSP Workshop in Parsing and Spoken Structural Event Detection developed SPARSEVAL, a revised parse evaluation measure that takes into account dependency relationships among words instead of spans.[7] The SPARSEVAL metric converts CFG trees into dependency trees (using Magerman's head-finding algorithm and head percolation of the words at the leaves) and treats each dependency tree as a bag of triples $\langle w_d, r, w_h \rangle$ where w_d is the dependent word, r is a symbol describing the relation, and w_h is the dominating (head) word. SPARSEVAL describes the overlap (in terms of precision and recall) between the "gold" bag-of-triples and the hypothesized bag-of-triples. In order to get a single

[7] I was a collaborator and was involved in the planning of the workshop and the design of this metric.

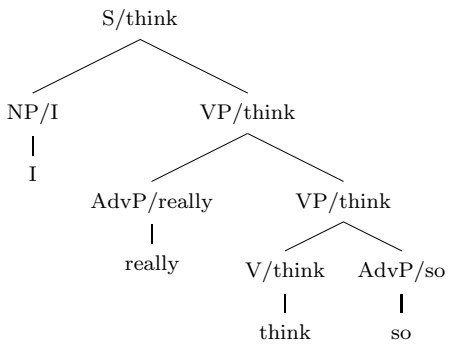
measure for distance from the correct tree, the systems used in this work use the F-measure combination of precision and recall. Although the reference and hypothesized dependencies are both coming from CFG trees and may therefore suffer some distortion in the conversion to dependencies, the SPARSEVAL strategy assumes that this distortion should be similar for reference and ASR transcripts and for different CFG-based parsers cases. Figure 2.8 includes reference and two example hypothesized trees.

Overall, SPARSEVAL allows a principled incorporation of both word accuracy and accuracy of parse relationships. Since every triple involves two words, this measure depends heavily on word accuracy, but in a more complex way than word error rate, the standard speech recognition evaluation metric. Figure 2.8 demonstrates a number of properties of the SPARSEVAL measure. Although both (b) and (c) have the same word error (one substitution each), they have very different precision and recall behavior. As the figure suggests, the SPARSEVAL measure overweights "key" words. All words appear exactly once in the left (dependent) side of the triple, but only the heads of phrases appear on the right. Thus, those words that are the lexical heads of many other words (such as *think* in the figure) are multiply-weighted by this measure. Head words are multiply weighted because getting head words wrong impacts not only the triples where that head word is dependent on some other token, but also the triples where some other word depends on that head word. Non-head words are not involved in so many triples.

2.6 Summary

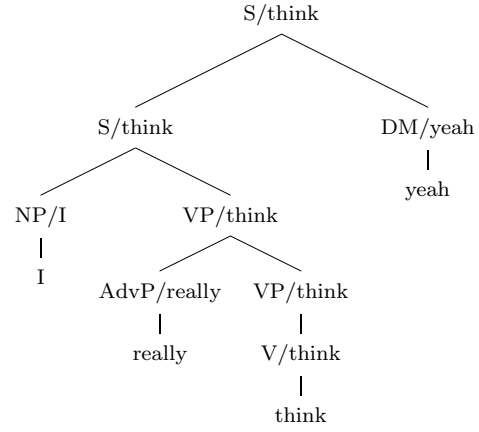
This chapter has given a review of current robust (statistical) parsing technology and a basic review of speech recognition technology. In addition, it reviews the theory and practice of prosody and prosodic labeling of recorded speech, as well as some other higher-level groupings (such as the SU). This chapter also presents a summary of

(a)



(I, S/NP, think)
(really, VP/AdvP, think)
(think, <s>/S, <s>)
(so, VP/AdvP, think)

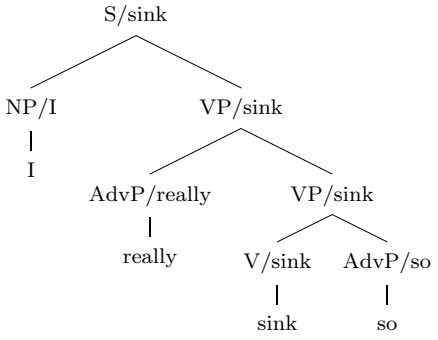
(b)



Precision = $\frac{3}{4}$, Recall = $\frac{3}{4}$
Word Error Rate = $\frac{1}{4}$

(I, S/NP, think)
(really, VP/AdvP, think)
(think, <s>/S, <s>)
(yeah, S/DM, think)

(c)



Precision = $\frac{0}{4}$, Recall = $\frac{0}{4}$
Word Error Rate = $\frac{1}{4}$

(I, S/NP, sink)
(really, VP/AdvP, sink)
(sink, <s>/S, <s>)
(so, VP/AdvP, sink)

Figure 2.8: A SPARSEVAL example that includes a reference tree (a) and two hypothesized trees (b,c) with alternative word sequences. Each tree lists the dependency triples that it contains; bold triples in the hypothesized trees indicate triples that overlap with the reference tree. Although all have the same parse structure, tree (c) is penalized more heavily (no triples right) because it gets the head word *think* wrong.

previous attempts to parse conversational speech, and examined the previous work in relating prosody to syntax. Although previous work has demonstrated some successes in parsing conversational speech transcripts, there is little work in syntactic parsing of conversational speech working in the context of speech recognition error as well.

Evaluation of parsing in the context of speech raises further problems, since neither the traditional speech recognition measure (WER) nor the common PARSEVAL parsing measure are adequate for the task of evaluating parse hypotheses based on hypothesized words. This chapter has also reviewed a recently-developed alternative evaluation measure for this situation, SPARSEVAL, which was developed with my collaboration.

The remainder of this thesis explores ways to connect parsing, prosody and speech recognition systems to incorporate existing solutions in each domain and to improve beyond the previous work in combining these systems.

Chapter 3

CORPUS

Experiments with a parser in the context of a speech-processing system require a speech corpus with (at a minimum) a test set that has been hand-labeled for syntactic structure, against which the parser can be tested. Ideally, such a corpus would include a sizable amount of data, in order to allow a large portion of the data to be used for training while still reserving some part unseen as a test corpus. Additionally, if symbolic prosody symbols (those symbols that reflect human judgement) are to be made useful, then some way of getting at those judgements must also be available. The Switchboard corpus (Godfrey et al. 1992) is just such a corpus.

3.1 The Switchboard corpus

The data used in this work comes from the Switchboard corpus, which is a collection of English-language conversational telephone speech. Paid volunteers participated in a five-minute telephone conversation with a stranger on one of about 40 pre-selected topics. About 2500 conversations were collected, with speakers from around the United States. Not all speakers were represented equally; approximately 50 speakers had more than one hour of conversation recorded (to be used as a test environment for speaker identification), but many others (about 450) appear in one to twenty conversations. These conversations, although artificial in that there is no purpose to the conversation (other than data collection), nevertheless exhibit many of the challenging properties of natural conversation: hesitations, restarts, repairs — in short, all manner of **disfluent speech** — as well as sentence structure and organization

that does not necessarily reflect the same sorts of patterns that one might expect in written or otherwise edited English.

The original transcription effort was followed by other transcription efforts, e.g. the work of Deshmukh et al. (1998), sometimes called the "Mississippi State transcripts", which are now considered a more accurate transcription. This latter work includes a number of improvements to the transcript. This effort to re-transcribe the corpus points out an important difficulty: that of human error in the reference transcription. Transcribing speech is difficult, and transcribing conversational speech is extremely difficult. The ISIP (1997) transcripts use multiple passes and additional quality control checks to get to a better reference texts.

For the purposes of this research, there are several relevant additional annotation efforts on the Switchboard corpus (beyond improvements to the transcriptions). Section 3.2 outlines these efforts and gives some examples of this annotation; section 3.3 discusses the reconciliation of these different annotations, and section 3.4 outlines the partitioning of the corpus into groupings for experimental use.

3.2 Annotation of the Switchboard corpus

Beyond the word transcription, the Switchboard corpus has been annotated for several additional kinds of paralexical and supra-lexical information. This section outlines three of those annotation efforts.

3.2.1 Treebank annotation

The Penn Treebank project (Marcus et al. 1993) annotates a number of different texts with syntactic structure. In the third phase of this project (TB3), a portion of the Switchboard corpus (about 1300 conversation sides) was annotated for syntactic structure, assigning constituent-label bracketing over each sentence span. Although this TB3 syntactic labeling was done by linguists, the labels were assigned based on

the transcribed text, not by listening to the actual speech. Since the the original transcription did not proceed according to clear punctuation guidelines, the text-based segmentation by the transcribed periods is inconsistent. This inconsistency motivates some of the data-manipulation work described in section 3.3.3.

The syntactic annotation here corresponds to no particular contemporary theory of syntax, although some decisions (to include null elements, for example) make it incompatible with some. The part-of-speech tags and syntax labels are the same as those used to tag the rest of the Penn Treebank project, with a few additional labels that annotate characteristics of spoken language like edit regions and partial words.

3.2.2 Disfluency annotation (SU boundaries)

Two interesting types of prosodic information available for conversational speech are the edges of "sentence-like" units (SUs) and the position and structure of edit regions. To target this question, the disfluency annotation effort by Meteer et al. (1995) included a markup standard that delimited (among other things) a "slash unit" (SU) and edit structure, as described in section 2.3.3. These were annotated by listeners who were provided with the actual acoustics. These boundaries (unlike the syntactic boundaries described in section 3.2.1) are thus fairly faithful to the conversational turns of the spoken language; furthermore, the annotators distinguished between a complete SU (`/`) and an incomplete SU `/-`. Several years after the Meteer disfluency annotation was performed, LDC developed new "V5" guidelines for annotating SUs (Strassel 2003, LDC 2004), and significant progress (e.g. Liu et al. (2005)) has been made in automatically recovering SU boundaries annotated according to these later standards (Strassel 2003).

Since the ultimate goal here is to incorporate boundary detection into a fully automated speech parsing system, and since there is more Meteer-style annotated data than V5 data, these Meteer-annotated SUs have been automatically adjusted to be more like the V5 LDC standard, which is the reference format for future work.

3.2.3 Annotation with prosodic structure

In addition to the SU annotation of Switchboard described above (section 3.2.2), Ostendorf et al. (2001) labeled a subset of the database (portions of 124 conversation sides) for a subset of the ToBI annotation. This corpus makes up slightly less than 4.7 hours of conversation, less than 10% of the corpus annotated for syntactic structure, and these labels were carefully time-aligned to the Mississippi State transcripts mentioned above.

This labeling provides a valuable additional source of information for the structure of conversational speech. In this corpus, each utterance is hand-labeled with boundary tones (but not accent tones), accents, break indices, and additional diacritics indicating a disfluent boundary.

For the purposes of this research, the break index annotations are the most interesting. These are identified with the right edge of every word, and indicate the degree of the prosodic phrase that ends at that point (as discussed in Chapter 2, section 2.3.1). If the boundary between words was judged disfluent by the annotator, the break index would be marked with a diacritic `p`. This diacritic was only attached to break indices `1` and `2` by design. Additionally, when the annotator was uncertain of a particular level, this uncertainty was sometimes indicated with a trailing minus, e.g. a `4-` indicates an uncertainty between `3` and `4`.

The set of possible break indices is quite large (at least ten), so these indices are mapped to a smaller set because of the small size of the dataset. Specifically, this work uses only 3 classes of prosodic breaks: intonational phrase (annotated with `4` or `4-`), disfluent break (annotated with `1p` or `2p`), and a class that subsumes all other labels. Phrasal prominence (pitch accents) and tones are not used in this work. Chapter 5 describes how this information is used to bootstrap first to a larger corpus and, having done that, how this larger (noisily annotated) corpus of prosodic information is used to improve parse selection. This simple 3-class system is relatively theory-neutral

and language-independent in that essentially all languages have a notion of fluent and disfluent segmentation.

3.3 Reconciling annotation differences

Since these annotations are based on different word transcriptions and conventions for defining a "sentence", to combine them raises a number of concerns. This section outlines the steps taken to combine these annotations into a consistent multiply-annotated corpus that matches the sort of output that might appear as the product of a speech-to-parse system.

3.3.1 Transcript and treebank normalization

In order to make the reference treebanks as much like correctly recognized speech as possible, punctuation and null elements are pruned from the reference treebank, as are "semantic role" diacritic labels. Case is normalized to lowercase throughout. Also, for the sake of retaining good association between the transcript and syntax, the original transcripts were retained (despite the existence of the improved Mississippi State transcripts). Contracted auxiliaries and negations are left as separated tokens (*do n't* and *I 'll*, as they are in the treebank, and acronyms are left as single tokens. Changing these transcriptions would require hand-correcting the parses, which was beyond the scope of this thesis.

3.3.2 Reconciling SUs to V5 mapping

While the later (V5) specifications of SU and IP annotation do not correspond exactly to the specification done by the original (Meteer) annotation, the correspondences are close enough to use a mapping of the original SU notation (SRI 2003). The primary differences between the two are in the annotation of incomplete sentences and restarts. The newer (LDC 2004) mappings are the target format for the state-

of-the-art systems, so it is useful to work with data that is normalized to carry the structure that the best systems generate.

3.3.3 Treebank resegmentation by SU

Since this work's goal is to be able to parse speech in a completely automated system, sentence boundaries will need to be automatically detected in speech recognition for any downstream parsing. Unfortunately, annotation guidelines used for Switchboard treebanking did not specify how sentence boundaries should be identified, and annotation was performed based on the transcript without reference to the underlying audio.

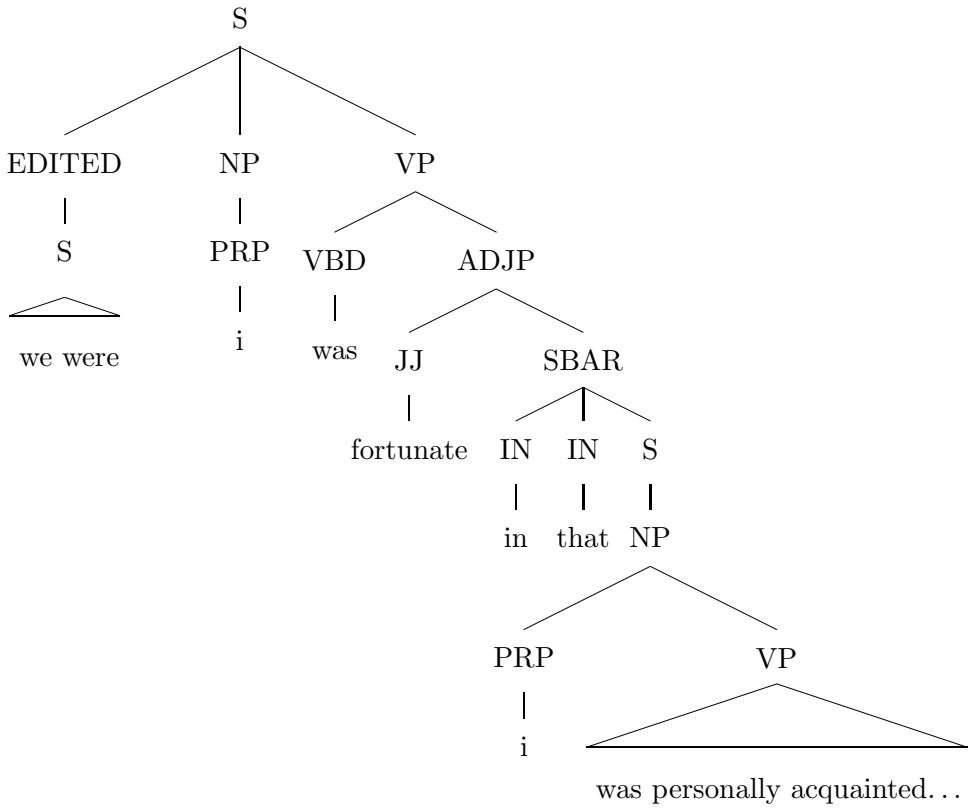
In some cases, the text-based syntactic annotation infers structure where none belongs, treating (for example) *and* as a conjunction rather than a discourse marker that begins a new, separate sentence, or treating a separated discourse marker (like *uh-huh*) as an attached component of a nearby sentence. Thus, sentence boundaries used by the treebank annotators are largely irreproducible, and may not be faithful to the intents of the speakers.

An example error due to the annotation based on transcribed words and punctuation is given below. In the SU-annotated sequence below, it is quite clear that the word *that* is a deictic pronoun, because the slash `/` indicates the end of an SU (a "whole thought" according to the annotation specification):

```
B.6: [ we were, + I was ] fortunate in that /
     I was personally acquainted with the, {F uh, }
     people who, {F uh, } ran the nursing home
     in our little hometown. /
```

However, the transcript includes only punctuation, and no SU `/` marks, and the syntactic annotators incorrectly treat *that* as a subordinating conjunction. The treebank

(incorrectly) annotates the following structure over these two SUs (after case normalization and punctuation):



The correct annotation would probably be the following (note the bold nodes, which are different).

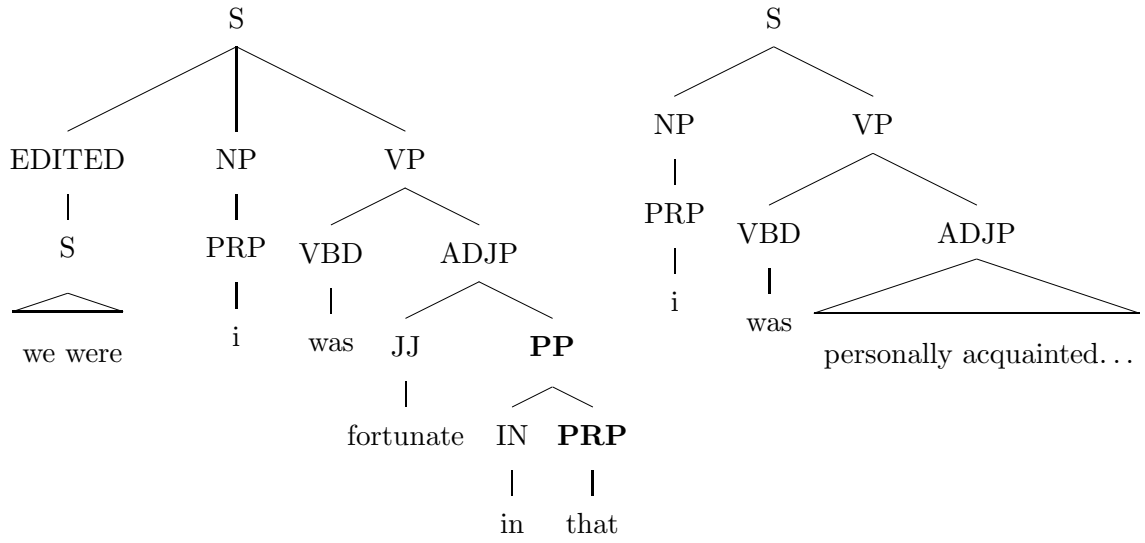


Table 3.1: Statistics on the Switchboard division used. This table represents the "large" corpus.

Section	Sides	SUs	Words
Train	1,031	87,599	659,437
Tune	126	13,147	103,500
Test	128	8,726	61,313
Total	1,285	109,472	824,250

Overall, SU and treebank segments disagree at a rate of about 5% in each direction, due mostly to the analysis of some discourse markers as conjunctions (treebank "sentences" of >1 SU) and the separation of backchannels into separate treebank sentences (SUs of >1 treebank "sentence", e.g. "yeah, yeah." vs. "yeah. yeah.").

Accordingly, in this work I segmented the Switchboard treebank to correspond to the SU-annotated sentence boundaries. In cases where the original syntactic constituents span multiple SUs, I discard any constituents violating the SU boundary, and in the event that an SU spans a treebank sentence boundary, I insert a new top-level constituent SUGROUP to produce a proper tree (which is evaluated like any other constituent in the reference tree).

While this SU resegmentation makes it difficult to compare experimental results to past work on the Switchboard treebank, this is a necessary step towards developing a more realistic baseline for fully automated parsing of speech. Furthermore, this work (Chapter 4) is the first work on parsing in the context of automatic SU-boundary detection, and this allows a baseline of reference SU performance.

3.4 Experimental partitioning

Of the 1300 total conversation sides annotated with parses, data mismatch (e.g. invalid treebanks or mis-assignment of turns to speakers) caused a small number of

Table 3.2: Statistics on the Switchboard division used. This table represents the "small" corpus.

Section	Sides	Words
Train & Tune	816	566,083
Test	128	61,313
Total	944	627,396

conversation sides to be discarded. Statistics on the **full corpus** division are given in Table 3.1. During development, experiments were carried out on the *tune* section. The *test* section was reserved for a single, final test run.

Some early experiments in Chapter 4 were carried out on a smaller **subset corpus** (Table 3.2, which shares its test segment with the full corpus described above). In this smaller corpus, there are 816 conversation sides in the train and tune combined (566,083 words).

Chapter 4

PARSING AND SENTENCE-LEVEL PROSODIC INFORMATION

In this chapter,[1] I explore the effects of automatic sentence boundary and interruption-point detection on the performance of a variety of statistical parsers.

Several efforts have looked at detecting sentence boundaries in speech, e.g. (Kim and Woodland 2001, Huang and Zweig 2002). Metadata extraction efforts, described further below, extend this task to include identifying self-interruption points (IPs) that indicate a disfluency or restart. This chapter explores the usefulness of identifying boundaries of sentence-like units (referred to as SUs) and IPs in parsing conversational speech.

As discussed in Chapter 2, early work in parsing conversational speech was rule-based and limited in domain, (e.g., Mayfield et al. (1995)). Results from another rule-based system (Core and Schubert 1999) suggest that standard parsers can be used to identify speech repairs in conversational speech. Heeman and Allen (1999) intertwine speech recognition, discourse-marker identification, turn segmentation and repair modeling into a single model, using a language model that is sensitive to repair structure, but does not examine syntactic structure in the task it explores. More recent work in statistically parsing conversational speech (Charniak and Johnson 2001) has examined the performance of a parser that removes edit regions in an earlier step. Edit (or self-correction) detection research (e.g. Liu et al. (2003) and

[1]Ciprian Chelba supported the work in this chapter by providing the Structured Language Model parser and valuable discussion. Joungbum Kim provided the SU-IP detection results. Preliminary results from this work have been published in Kahn et al. (2004).

Johnson and Charniak (2004)) has begun to look at the detection of disfluent regions in speech. Some systems (e.g. Kim (2004)) detect disfluent regions in the context of detecting sentence (SU) boundaries, and others (e.g. Liu et al. (2005)) detect SUs and IPs independently but use post-processing heuristics with both detected boundaries to reconcile the two annotations. Still others (e.g. Johnson and Charniak (2004)) use syntactic structure to help identify speech repairs, but assume that sentence boundaries are known. In this work, parsers are trained on the complete (human-specified) segmentation, with edit-regions included. The experiments in this chapter use all of the words within edit regions, anticipating that making the parallel syntactic structures of the edit region available to the parser can improve its performance in identifying that structure. Experiments in the next chapter consider an alternative framework for handling edits.

In order to ensure that these results are general to most statistical parsers (that is, that they do not exploit some peculiarity of a particular statistical parsing scheme), these experiments use three different parsers described in section 4.1. For SU and IP detection, this work uses a detection system (Kim 2004) that integrates prosodic and lexical cues and a naïve (silence-based) system. Both of these systems are described in section 4.2. Section 4.3 describes the experimental framework, and section 4.4 explores the results in the context of the questions this work seeks to answer. Full results tables are available in Appendix A.

4.1 Parsers

All three of the parsers used in this system are fundamentally oriented around probabilistic, lexicalized context-free grammars. However, each has a different approach to the use of the lexicalized PCFG information. These experiments use all three in order to ensure that any differences due to prosodic information — the focus of these experiments — cannot be attributed to a quirk of the parser.

4.1.1 Structured Language Model

The structured language model (SLM), proposed by Chelba and Jelinek (2000), is a language model that operates by keeping a stack of hypothesized lexicalized-CFG partial-parse structures, and evaluating the probability of new words conditioned on the class and headword of the previous two recently closed phrases. Thus, the probability tables are constrained in much the same way as a canonical lexicalized PCFG, with a few modifications for backoff order and local context. The SLM's "parser" module gives probabilities for building new structure that are conditioned on exactly the same tree-local fields on which a lexicalized PCFG is conditioned: the lexical headwords and categories of the parent and child nodes. Like a lexicalized PCFG, probabilities are assigned to a tree as a whole, based on probability tables that take into account only tree-local constraints. This probability table can be explored as a language model, or as a parser. These experiments use the trained probabilities as a parser.

4.1.2 Charniak parser

Unlike the SLM, the Charniak parser (Charniak 2000) is more directly intended as a parser. This parser uses a "maximum-entropy inspired" approach to do smoothing of possibly-redundant features in conditioning the generative prediction of the tree structure. In addition, the Charniak parser uses a wider variety of conditioning factors, not all of which are strictly tree-local in the PCFG paradigm — for example, the left sibling of the parent node is a factor considered in the Charniak parser. This system (also unlike the SLM) uses a heuristic candidate-pruning strategy internally, and then calculates the probabilities of the various candidates using an expanded conditioning space. The Charniak parser also includes some grammar-specific tunings: for example, those tags that tend to represent coordination structures are explicitly coded into the learner. The experiments in this chapter use the July 11, 2005 version

of the Charniak parser.[2]

4.1.3 Bikel parser

The Bikel parser (Bikel 2004) is a Java re-implementation and generalization of the Collins (2003) parser. In this system, the ultimate output is (once again) a CFG-oriented tree, in the form that the Penn treebanks generate. In addition, this Collins/Bikel model focuses its conditioning on phrases' lexical heads and builds outwards from those, rather than using the complex smoothing of the Charniak parser. Its internal conditioning is oriented around the head words, not the head category, but this is fundamentally akin to other lexicalized PCFG constraints — the word-dependency focus taken in this system is merely more closely oriented to the lexicalization than to the phrase categories. Internally, the Collins/Bikel model, like the Charniak parser, specifically hand-codes for a notion of conjunction, and furthermore adds special case probabilistic models for noun and verb phrases. The experiments in this chapter use the 0.9.9 release of the Bikel parser.

4.2 SU and IP detection

This section describes the various schemes for automatically detecting SUs and IPs that are used in these experiments.

The automatic system used here for SU and IP detection is that of Kim et al. (2004) and Kim (2004), modulo differences in the organization of the training and test data. It combines decision tree models of prosody with a hidden event language model in a hidden Markov model (HMM) framework for detecting events at each word boundary, similar to Liu et al. (2003). One difference is that the implementation used here includes lexical pattern matching features (sequential matching words or POS tags) in the decision tree as well as prosody cues. In addition, this work uses a joint

[2]The Charniak parser is released with dates but no version numbers.

Table 4.1: SU performance statistics for the detectors used in these experiments, over the RT03 test set.

	R	P	F	SER
Naïve	43.2	79.0	55.9	68.8
Auto	75.7	87.7	81.3	35.0

representation of SU and IP boundary events rather than having separate detectors, and separately models IP events associated with fillers vs. edits, since only the edit IPs are of interest for this work. Using this model on the DARPA RT-03F metadata test set (NIST 2003) gives 35.0% slot error rate[3] (SER) for SUs (75.7% recall, 87.7% precision), and 68.8% SER for edit IPs (41.8% recall, 79.8% precision) on reference transcripts, using the `rt_eval` scoring tool. While these error rates are relatively high, it is a difficult task and the SU performance was at the state of the art at the time of the original experiments.

Since a reasonable guess at "sentence" segmentation in a conversation might simply look for long pauses, this work also includes a decision tree classifier to predict SU events based only on the pause duration after a word boundary. This model serves as a baseline condition, referred to here as the "naïve" predictor since it makes no use of other prosodic or lexical cues that are important for preventing IPs or hesitations from triggering false SU detection. The naïve predictor has SU SER of 68.8% (43.2% recall, 79.0% precision), roughly twice that of the HMM, with a large loss in recall. Note that because this naïve system is based on a simple pause model, it does not predict any interruption points.

Table 4.1 shows the performance statistics for the various SU prediction systems,

[3]Slot error rate is calculated as

$$\text{SER} = \frac{\#\text{ errors}}{\#\text{ events}}$$

and the number of SU and IP events is small compared to the number of words.

Table 4.2: IP performance statistics for the **auto** system used in these experiments, over the RT03 test set.

	R	P	*F*	SER
Auto	41.8	79.8	54.9	68.8

as tested on the DARPA RT-03F metadata test set (NIST 2003). Table 4.2 shows the performance statistics for IP prediction, although the IP performance figures are not comparable to the DARPA evaluation, since the focus here is restricted to IPs associated with edit disfluencies. The results for both for an oracle system would be 100% precision and recall, and 0% SER.

4.3 Experimental framework

With the flexibility in choice of parser, and the available SU and IP information, there are a number of experimental variables. This section describes the common framework for these experiments and their evaluation, and the experimental variables that were explored.

4.3.1 Experimental variables

These experiments vary a number of parameters in order to get a good picture of the problem space presented here. In addition to varying the parsers described in section 4.1 (SLM, Charniak, and Bikel), these experiments also use three different SU/IP systems: **oracle**, which uses the mapped (but originally human-annotated) V5 annotations described in Chapter 3, section 3.2.2; **auto**, which uses the Kim et al. (2004) automatic system, and **naïve**, which uses the simple decision tree classifier; these two latter systems are described in section 4.2. In addition, for the oracle and auto SU/IP conditions, these experiments also vary whether or not the predicted IPs are included as word tokens. (Where the IPs are included, they are included

in both training and test corpora.) Finally, because the text-oriented parsing literature reports great success in using punctuation as additional word tokens, and the original treebank annotation of the SWITCHBOARD corpus includes punctuation (at least before normalization), one final experimental condition includes hand-annotated punctuation along with oracle SUs as an alternative "extended oracle" condition.

4.3.2 Experimental setup

In all experiments, the parser to be used is trained on a corpus with the reference SU segmentation, which included the matching IP and/or punctuation tokens (that is, with the IP or punctuation condition matching the test corpus). The test corpus conversation sides are segmented according to the experimental SU/IP condition; if IPs are included, the IPs come from the same condition. These hypothesized segments (with or without internal IP tokens) are presented as whole sentences to the parser, and the resulting parse trees for that conversation side are presented to evaluation (described below).

4.3.3 Evaluation

These experiments evaluate parser performance by using bracket precision and recall scores, as well as bracket-crossing, using an implementation (Sekine and Collins 1997) of the PARSEVAL metric (Black et al. 1991). This bracket-counting metric for parsers requires that the input words (and, by implication, sentences) be held constant across test conditions. Since our experiments deliberately vary the segmentation, these experiments evaluate each conversation side as a single "sentence" in order to obtain meaningful results across different segmentations. This top-level sentence is constructed by attaching the parser's proposed constituents for each SU to a new top-level constituent (labeled TIPTOP, on analogy with the standard sentence-level root tag TOP). Thus, two different segmentations can be compared over the same word sequence, because the segmentations will agree at least at the beginning and

end of the conversation. Segmentation errors (of course) cause some mismatches, but that possibility is part of the impact of segmentation — which is one of the factors being evaluated here.

For evaluation, the TIPTOP bracket (which always contains the entire conversation side) is ignored (as is TOP, in the standard PARSEVAL evaluation), so the conversation-side scoring technique does not interfere with accurate bracket counting, but allows segmentation errors to be evaluated at the level of bracket-counting.

The SLM parser (unlike the Bikel and Charniak systems) generates binary trees, but the syntactic structures given as truth (from SWITCHBOARD) often include subtrees with more than two branches. The parse trees used for training the SLM insert bar-level nodes to transform N-ary trees into binary ones; the reverse mapping of SLM-produced binary trees back to N-ary trees is done by simply removing the bar-level constituents (or ignoring the inserted bar-level constituents in PARSEVAL). Finally, to compare the IP-present conditions with the non-IP conditions, IP tokens are ignored when matching labeled spans.

All of these experiments report labeled bracket precision (P), labeled bracket recall (R), and the F-measure (harmonic mean of precision and recall), all of which increase as the hypothesized structures match the reference structures more closely. They also report bracket-crossing (BX), which *decreases* as the hypothesized structures match the reference more closely. Note that these bracket-crossings are much higher than results usually reported because the bracket-crossings are counted per conversation side (due to the TIPTOP evaluation described above).

Results in this chapter differ from other results reported in Charniak and Johnson (2001) because the results presented here use PARSEVAL scores directly, rather than the relaxed-edit scoring metric that Charniak and Johnson (and Chapter 5) use. The results are slightly different from the results in Kahn et al. (2004) because that work used only the SLM, the subset corpus, and some differences in the corpus re-segmentation process.

Table 4.3: The performance of three parsers at recovering SWITCHBOARD tree structure, given words segmented along the reference SU boundaries, when trained on the subset corpus described in Chapter 3.

	R	P	*F*	BX
SLM	77.45	68.40	72.64	46.80
Charniak	80.85	72.24	76.30	40.18
Bikel	82.42	83.36	82.89	36.77

4.4 Results

The complete experimental results tables are available in Appendix A, but these data are fairly complex due to the number of variables. This section pulls out some comparisons from these experiments to examine the effects of the experimental variables on performance. Significance testing is done with the stratified shuffle as suggested in Yeh (2000).

4.4.1 Comparison among parsers

These three parsers do not perform equivalently on conversational speech. As Table 4.3 demonstrates, the Bikel parser performs much better than the Charniak and SLM systems. The evaluation measure used here is discussed in section 4.3.3, which includes an evaluation of the structure within edit regions, and the results presented in Table 4.3 are results over the SU-resegmented word structures, not the original punctuation-annotated sentences. IP information is not provided to the parser.

It is worth noting that the SLM is at a disadvantage here: its training is optimized to predict words, not structure, and so its parse structure performance is (perhaps not surprisingly) weaker. The Charniak system used here does not incorporate explicit speech repair modeling. In their latest work, Johnson and Charniak (2004) also support a model for speech repairs, and use a scoring measure that discounts errors

within speech repair regions, and hence their reported performance is better than that presented here. That the Bikel system performs better on this (raw PARSEVAL) measure suggests that it may be doing a better job of modeling structure within edit regions than the Charniak system.

4.4.2 Comparison across SU/IP conditions

To investigate the impact of high-level sentence-level segmentation (SU boundaries) and IP prediction on the performance of state-of-the-art parsers, results for the different IP and SU conditions are given in Table 4.4. Note that for the +IP cases, reference (hand-marked) IPs are used in the "reference SU" condition, and automatically detected IPs are used in the "auto IP" condition. The same results are presented in Figure 4.1. For clarity, only the F scores are presented here; bracket-crossing, precision and recall all follow these trends and are available in Appendix A. Results are reported for systems trained on the subset corpus, since the SLM was only trained on that data set. The same findings hold for the Bikel and Charniak parsers when trained on the full corpus.

As Table 4.4 demonstrates, getting the SUs right is critically important to accurate parsing. Ignoring the +IP rows in Table 4.4 for the moment, one can see that naïve SU segmentation is quite spectacularly damaging to parse quality, as measured by the PARSEVAL F-score. (SU differences are all significant at $p < 0.0001$.) However, the automatic systems recover some of this performance loss: in comparison to the naïve SU detector (using only pauses), using the state-of-the-art automatic SU detector gives a 26% reduction in error $(1 - F)$, despite a relatively high SU error rate (as calculated from the Bikel system). Using reference SUs gives a 50% reduction in error. While the Bikel system is the best performer, it has the greatest sensitivity to segmentation: the Charniak parser, by contrast, gets a 21% reduction in error in moving from the naïve SUs to the automatic SUs, and a 42% reduction from moving from naïve to reference SUs.

Table 4.4: SU conditions and IP conditions, using the SLM, Charniak and Bikel parsers. Only F score is reported for clarity. All results here are with systems trained on the subset corpus.

SLM			
	SU condition		
	Reference	Auto	Naïve
+IP	72.89	69.01	—
−IP	72.64	67.46	63.01

Charniak			
	SU condition		
	Reference	Auto	Naïve
+IP	78.38	68.78	—
−IP	76.30	67.89	59.24

Bikel			
	SU condition		
	Reference	Auto	Naïve
+IP	84.22	74.99	—
−IP	82.89	74.56	65.72

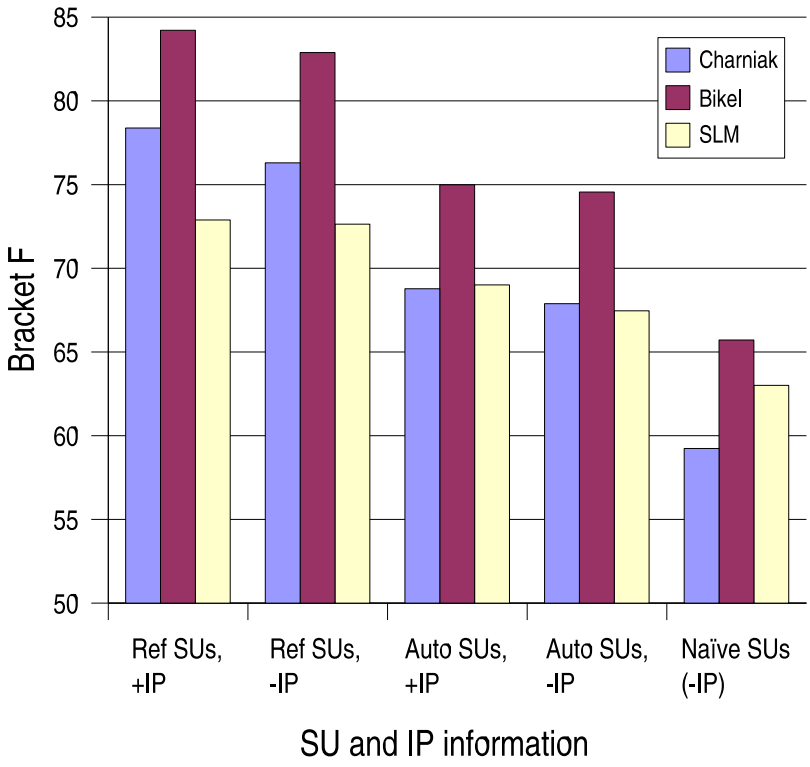


Figure 4.1: F measure for SU and IP conditions, using all three parsers.

Table 4.5: Running the SLM (trained over the small subset corpus) over reference SUs: with and without IPs, with and without punctuation.

	R	P	F	BX
−IP −punct	77.45	68.40	72.64	46.80
+IP −punct	77.98	68.42	72.89	44.56
−IP +punct	68.89	**77.98**	73.15	41.66
+IP +punct	**78.65**	73.97	**76.23**	**35.84**

IP information is useful to all three systems (±IP differences in F are all significant, $p < 0.002$). For the best systems (Bikel, Charniak), gains are bigger when using oracle segmentation and oracle IPs than when using automatically identified IPs. Using reference IPs gives an 8% reduction in error $(1 - F)$ with reference SUs in the Bikel parser, and a 9% reduction with the Bikel parser. This result is different from (but complementary to) Johnson et al.'s (2004) result that automatically-predicted IPs help edit detection, which is known to help parsing as well.

4.4.3 Comparing IPs to punctuation

It is interesting to compare IPs (which reflect a single kind of suprasegmental information — the position of an interruption point) with punctuation, which is used to indicate a variety of syntactic and discourse effects in written language. When oracle SUs are known, then IPs and punctuation reflect two different types of sentence-internal events. In principle, punctuation should be a superset of IP information, but there was no explicit specification for annotating self-interruption given to the transcribers. Table 4.5 shows the results of running the SLM parser with reference segmentation, and compares the effects of including or excluding prosody and syntax markup. Adding punctuation with no IPs included gives an 11% reduction in bracket crossings, and a 14% improvement in precision, but at the cost of an 11% decrease

in recall, which suggests that in the context of correct SUs, providing punctuation is a mixed blessing. When punctuation is provided together with IPs, however, all measurements improve, suggesting that a richer representation of structural metadata (beyond SUs and IPs) would be useful.

4.5 Discussion

The experiments in this chapter demonstrate that sentence-level prosodic events — specifically, SU identification — drastically impact parser performance. It is possible that low SU recall in the naïve segmenter created a much larger parse search space, leading to more opportunities for errors. The improvement to parse performance provided by the automatic system is promising, and suggests that current lines of research to improve metadata extraction can have a direct impact on the performance of other higher-level natural language applications that deal with conversational speech, such as parsing.

Inaccuracies in automatic segmentation can contribute to poor PARSEVAL performance in two ways: on the one hand, the segmentation itself can prevent some spans from being correctly identified (the segmentation crosses some syntactic spans); on the other hand, segmentation can hide useful context information from the parser and derail the parser's probability estimates. Results from Kahn et al. (2004) on a related test set strongly suggest that the latter is the problem: that study reported that more than 90% of correct bracketings are still possible under the "naïve" segmentations.

In addition, including IP information is consistently helpful to both the Bikel and the Charniak parsers (especially in the context of reference SU and IP prosodic events). This improvement is somewhat in contrast with results reported in Gregory et al. (2004), which reported no improvement from including low-level prosody tokens (reflecting changes in f_0 and other low-level phenomena). In that experiment, it appeared that introducing new word terms and additional tokens into the word

string introduced data sparsity and occupied spaces in the parser's local history that would otherwise be occupied by information-bearing terms. However, by using the perceptually-based IP tokens (instead of the raw prosody tokens used by Gregory et al. (2004)), these experiments showed an improvement, especially in the context of automatic SU detection. The finding that inserting these IP terms as words improves the parse score (despite their cost to the parse history) demonstrates that IPs are useful, information-bearing units. These experiments exemplify one way to exploit these units to improve parser performance; other ways are explored in Johnson et al. (2004).

As a related experiment, the results presented in section 4.4.3 suggest that punctuation and IP information are both useful, and bear *complementary* kinds of information, possibly because punctuation is a fairly reliable indicator of some kinds of prosodic phrase boundaries. The next chapter explores the effect of including prosodic phrase information in information provided to a parser.

Chapter 5

PARSE SELECTION USING PROSODIC PHRASE STRUCTURE

In the previous chapter, experiments with punctuation suggested that it would be interesting to explore symbolic prosody features below the level of the sentence. Similarly, it has been shown that automatically detected IPs based on prosodic cues (Liu et al. 2005) are useful in the reranking stage of a tree-adjoining grammar (TAG) based speech repair detection system (Johnson et al. 2004). Intonational phrase boundaries (and the boundaries of disfluencies) may also be useful in assisting a parser in discovering the appropriate phrase structure.

This chapter[1] identifies a set of useful prosodic cues for parsing conversational speech, and shows how such features can be effectively incorporated into a reranking statistical parsing model (Charniak and Johnson 2005). In addition, it demonstrates the impact of automatic edit detection on parse performance.

5.1 Architecture

This work generates a **parse cohort** (set of candidate parses) from each input sentence using an off-the-shelf, k-best parser, and then uses symbolic prosody features (and the more standard syntactic features) to rescore the candidate parses within

[1]Matthew Lease, Eugene Charniak, and Mark Johnson supported the work in this chapter by providing support for their parser and edit detector (Johnson et al. 2004, Charniak and Johnson 2005). This work grew out of a collaboration with Matthew Lease in the design and execution of these experiments. Darby Wong assisted (Wong et al. 2005) in bootstrapping prosodic labels from the small hand-labeled set over the entire treebanked section of the SWITCHBOARD corpus. This work was published as Kahn et al. (2005).

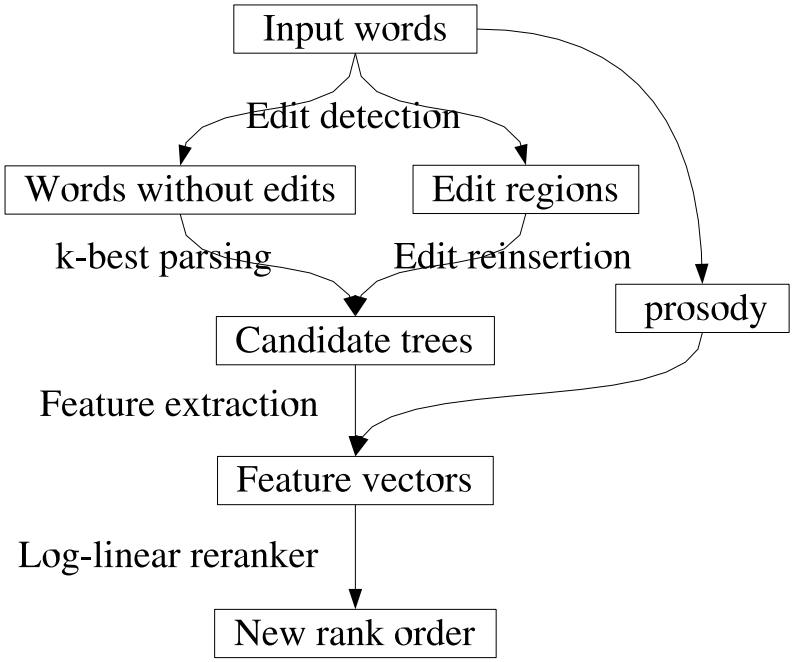


Figure 5.1: Parse-reranking architecture that incorporates edit detection and incorporates prosodic features.

the cohort. Architecturally, this system (shown in Figure 5.1) combines earlier models proposed for parse reranking (Collins 2000) and filtering out edit regions before parsing (Charniak and Johnson 2001), motivated by Charniak and Johnson's (2001) claim that probabilistic context-free grammars (PCFGs) perform poorly at detecting edit regions. This claim is empirically validated in section 5.4: two state-of-the-art PCFGs (Bikel 2004, Charniak and Johnson 2005) are both shown to perform significantly below a state-of-the-art edit detection system (Johnson et al. 2004).

The system operates as follows:

1. Words in an edit region are identified and removed.

2. Each resulting string is parsed to produce a cohort of k candidate parses.

3. Edit-region words are reinserted into the candidates with a new part-of-speech

tag EW. Consecutive sequences of edit words are inserted as single, flat EDITED constituents.

4. Features (syntactic and/or prosodic) are extracted for each candidate, i.e. candidates are converted to feature vector representation.

5. The candidates are rescored by the reranker to identify the best parse in the cohort.

The use of Collins' parse reranking model has several advantages for this work. In addition to incorporating prosody without distorting the lexical history available to the parser, the discriminative model makes it relatively easy to experiment with a variety of prosodic features, something that is considerably more difficult to do directly with a typical PCFG parser.

The use of the Charniak-Johnson approach of separately detecting disfluencies is motivated by their result that edit detection error degrades parser accuracy, but this work also includes experiments that omit this step (forcing the PCFG to model the edits) and confirms the practical benefit (asserted by Charniak and Johnson (2001)) of separating responsibilities between the edit detection and parsing tasks. Two PCFGs (Bikel 2004, Charniak and Johnson 2005) are evaluated on edit detection and compared their performance to a state-of-the-art TAG-based edit detection system (Johnson et al. 2004).

5.2 Baseline system

This work adopts an existing parser-reranker as a baseline (Charniak and Johnson 2005). The parser component supports k-best parse generation, and the reranker component is used to rescore candidate parses proposed by the parser.

The reranker attempts to select from the cohort of k candidates $T = \{t_1, \ldots t_k\}$ the parse $t^\star \in T$ with the highest bracket F-score (in comparison with a hand-annotated

reference), as in (Collins 2003). To accomplish this, a feature-extractor converts each candidate parse $t \in T$ into a vector of real-valued features $f(t) = (f_1(t), \ldots, f_m(t))$. For example, 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 $\lambda' f(t)$ (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 bracket F-score parse $t^\star$ conditioned on T (together with a Gaussian regularizer to prevent overtraining). Informally, λ is chosen to make high F-score parses $t^\star$ as likely as possible under the (conditional) distribution defined by f and λ. As in Collins (2000), training data for the reranker is generated by reparsing the training section of the corpus, using $n-1$ folds as training data to train the parser for generating training candidates from the n-th fold. For these experiments, $n = 10$.

Conditioning on the cohort T (those parses generated from the same input) is accomplished in part by **de-moding**, since those features that have the same values across all parses of the same input are not useful in discrimination. In order to identify such features more readily and to make the matrix of feature values more sparse, the count or weight associated with each candidate feature f_j in every candidate $t \in T$ is de-moded:

$$f_j(t) = f_j(t) - \text{mode}_j(T)$$

where $\text{mode}_j(T)$ represents the most common value of $f_j(t)$ where $t \in T$. Thus, when all (or nearly all) of the values of $f_j(t)$ are identical across all $t \in T$, the values associated with $f_j(t)$ for each t will all (or nearly all) be zero, allowing a more compact representation and speeding the machine-learning process.

Before training the λ values, some coarse pruning is done on the features across the entire set of training cohorts: let c_j be the count of training cohorts that include at least one candidate t with a non-zero value of $f_j(t)$ (after demoding). Feature j is discarded from the feature vector of all candidates in all cohorts if $c_j < \theta$. For the

experiments in this chapter, $\theta = 5$.

The system from Charniak and Johnson (2005) also includes a syntactic feature extractor that identifies interesting syntactic relationships not included in the PCFG parsing model (but used in the reranker). These features are primarily related to non-local dependencies, including parallelism of conjunctions, the number of terminals dominated by coordinated structures, right-branching root-to-leaf length, lexical/functional head pairs, and n-gram style sibling relationships. Since the features are lexicalized — many of them count the occurrences of particular words — it is difficult to count the total number of possible features, but these experiments use on the order of 50,000 features after pruning the extracted set.

The architecture separates responsibilities between the edit detection and parsing tasks based on earlier evidence that improved edit detection leads to better parsing accuracy (Charniak and Johnson 2001). To provide a competitive baseline for our parsing experiments, an off-the-shelf, state-of-the-art TAG-based model is the primary edit detector (Johnson et al. 2004). However, by varying this edit detector component, this system also provides an environment for exploring edit detection task accuracy (Section 5.5.1) and its impact on parse performance.

5.3 Symbolic prosody features

As discussed in Chapter 2, section 2.3.1, many theories of prosody have a symbolic representation for prosodic phrasing, which adopts a representation of phrase boundaries. These phrase boundaries are instantiated in the speech signal with various combinations of acoustic cues (f_0, energy, timing), but are modeled in these theories as categorical boundary events. Symbolic break labels are used here as a mechanism to consolidate information about these acoustic cues (and the perceptual, categorical events that they represent). These break labels (described in Chapter 3, section 3.2.3) represent linguistic notions of intonational phrases and hesitation phenomena.

This work uses symbolic events directly because the intermediate representation of the break label (between raw acoustic values and the higher-level syntactic structure) simplifies training on sparse acoustic structures. Just as acoustic models for speech recognition are trained on an intermediate represntation (the phone), a small number of break indices can serve to simplify modeling this large and interdependent set of continuously-valued acoustic cues. Instead of representing these break labels as categorical symbols (present or absent), though, this system passes a posterior probability of a categorical label to the parsing component rather than making a hard decision.

5.3.1 Extending hand-annotated labels to the whole treebank

It might seem impractical to use prosodic labels as an intermediate representation, because of the skills required to annotate prosodic corpora for supervised learning. The system used here overcomes this problem by taking advantage of a weakly-supervised learning strategy (Wong et al. 2005) to bootstrap its way to a weakly-annotated prosodic corpus.

The strategy used here is similar to that proposed by Nöth et al. (2000), which uses categorical labels defined in terms of syntactic structure and pause duration. However, their system's category definitions are without reference to human perception, while the category definitions used here take advantage of automatically-learned relations between hand-labeled perceptual events, hand-labeled syntax, and other acoustic cues, without predetermining the relation or requiring a direct coupling to syntax.

Initial break prediction models were trained on a small set of labeled data from the train section of the treebanked portion of the Switchboard corpus (Ostendorf et al. 2001), using both parse and acoustic cues to predict a reasonable first labeling of the entire treebanked training corpus. This first labeling is used to label more data with an EM algorithm that iterates between:

1. predicting probabilities of prosodic breaks in the unlabeled data, based on the

current model, again using parse and acoustic features, and

2. re-estimating the model using the predicted probabilities as weighted counts.

This iteration is stopped after 3 iterations, and the resulting "noisy labels" are used to train a new break prediction model that is derived only from the acoustic sequence. The labels assigned by the acoustic-only break-prediction system are then used in the parsing experiments.

Decision trees are the model in each stage, in part because of an interest in analyzing the prosody-syntax relationships learned. The baseline prosody detection system trained on hand-labeled data has error rates of 9.6% when all available cues are used (both syntax and prosody) and 16.7% when just acoustic and part-of-speech cues are provided (our target environment). Using weakly supervised (EM) training to incorporate unannotated data (and bootstrap from some syntactic knowledge about the training corpus) led to a 15% reduction in error rate to 14.2% for the target word sequences. The final decision tree was used to generate posteriors for each of the three classes of prosodic breaks, one for each word in a sentence, which is an annotated corpus to be used as input prosodic features for the syntactic-prosody feature extraction described in the next section. For more details, see Wong et al. (2005).

5.3.2 Prosodic features for parse reranking

Prosodic break posterior probabilities (as calculated in the previous section) by themselves do not help discriminate among parse candidates, since the prosodic break information is predicted on a per-word basis and these experiments hold the word sequence constant across all parse candidates t in a cohort T. In order to use the prosodic features to help discriminate among parse candidates, prosody features must be combined with syntactic features to create combined features whose values vary across parse candidates.

One strategy for extracting syntactic features that are not easily detected by a PCFG is to look at the counts of syntactic subconstituents with various signatures. The strategy presented here is closely related to some of the features used in the baseline system (Charniak and Johnson 2005). A **syn-signature** here is an equivalence class of subconstituents that will be added to the same count: for example, one signature might be noun phrases with a height of two or three nodes from the NP root to the lowest leaf, exactly two words, and a right-side depth of exactly two. The equivalence class is characterized by a 4-tuple:

$$(l, w, h, d) = \textsc{synsig}(c)$$

where c is a tree subconstituent, l is the label of c, w is a 3-valued indicator of the length in words of c ($w \in \{1, 2\text{–}5, 6+\}$), h is a 3-valued indicator of the maxiumum depth from c's root to any leaf word ($w \in \{1, 2\text{–}3, 4+\}$), and d is a 3-valued indicator of the depth from c's root to the rightmost leaf word ($w \in \{1, 2\text{–}3, 4+\}$).

Note that two different syntactic structures will have a different list of subconstituents, and these subconstituents will usually be associated with different syn-signatures. Thus, counts of these syn-signature equivalence classes may be used to discriminate among different parses of the same word sequences.

To incorporate prosody in this model, I extend the syn-signature to include the prosodic break class (`1`, `4`, or `p`) at the right edge of the constituent, extending the 4-tuple syn-signature into a 5-tuple **synpros-signature**. However, since the prosodic information available to this system is available in terms of posterior probabilities, in this system I adopt not the count of synpros-signatures, but the *weighted count* of synpros-signatures, with the weights set to a coarsened posterior probability of the prosodic break at the right edge of the constituent. Coarsening makes more values effectively zero (through the de-moding described in section 5.2) and reduces issues of data sparsity, despite the real-valued posterior probabilities introduced by the break

Require: T is a cohort of parse candidates and their feature vectors, made up of candidate representations $\{t_1, \ldots t_K\}$. Each of these t_i has a feature vector with a value $t_{i,s}$ for each possible signature s.

for all parse candidates $t_i \in T$ **do**

for all subconstituents $c \in t_i$ **do**

$s \Leftarrow \textsc{synsig}(c)$

for all break level $B \in \{\mathtt{p}, \mathtt{1}, \mathtt{4}\}$ **do**

$s_B \Leftarrow$ the synpros-signature of syn-signature s extended by break B

$p_B \Leftarrow$ posterior probability of break B after c

$t_{i,s_B} \Leftarrow t_{i,s_B} + \textsc{coarsen}(p_B)$

end for

end for

end for

$\textsc{demode}(T)$

Figure 5.2: Algorithm to extract syntax/prosody features from a cohort T of parse candidates hypothesized over the same input lexical sequence.

labels. The coarsening is characterized as follows:

$$\textsc{coarsen}(p) = \text{round}(3 \cdot p)$$

where p is a posterior probability of a break value.

The algorithm in Figure 5.2, finally, describes the process of combining syntactic signature features with prosodic features. Thus, when using the prosody features for reranking, each candidate tree t is characterized by prosodically-weighted counts of various synpros-signature subconstituents.

This algorithm relates prosodic features to syntactic ones in a data-driven way: the learner will (ideally) learn to associate high counts of some synpros-signatures with good parses, and high weighted counts of other synpros-signatures with bad

parses, and likewise for low (or negative) weighted counts.

The baseline set of syntactic features (from Charniak and Johnson (2005)) includes a class of features that is very similar to the syn-signatures described here. However, the baseline set of features also includes lexically-specific information via association with specific words. When the baseline features are used in experiments below, they always include these lexical features as well. The prosodic features assigned by the algorithm in this section, on the other hand, do not use lexicalized features at all: they use only constituent label, length and shape. By lexicalizing, the baseline feature set is still much larger than even the extended syn-pros signature weighted counts: the lexicalized baseline features use about 50,000 features after pruning (as described in section 5.2) while the synpros-signature weighted-count features have approximately 1300 after pruning.

5.4 Edit detection experiments

Edit detection accuracy is evaluated on a per-word basis, where any tree terminal is considered an edit word if and only if it is dominated by an EDITED constituent in the gold tree. The PCFGs are trained on the training section of the treebank data with the flattened, edit regions included[2] and then used to parse the test data.[3] The TAG-based detector (as in Johnson et al. (2004)) uses a source-channel model, with a source model trained on trees with EDITED nodes removed, and a channel model trained on the Penn Treebank disfluency-annotated files. As the results in Table 5.1 attest, both PCFGs performed significantly below the TAG-based detector.

---

[2]Training on flattened EDITED nodes improved detection accuracy for both PCFGs: as much as 15% for Bikel-Collins.

[3]For the Charniak parser, edits were detected using only its PCFG component in 1-best mode, not its 2nd stage reranker.

Table 5.1: Edit word detection performance for two word-based PCFGs and the TAG-based edit detector. Error is calculated as the number of mistakes divided by the number of true events. F here is a harmonic mean of word-based precision and recall in identifying edit-region words.

Edit Detector	Edit F-score	Edit Error
Bikel-Collins PCFG	65.3	62.1
Charniak PCFG	65.8	59.9
TAG-based	77.1	43.6

5.5 Parsing experiments

5.5.1 Experimental variables

The primary focus of this work is to evaluate the extent to which prosodic cues could complement or replace lexical and syntactic features in parse reranking. Accordingly, three flavors of **feature extraction** are used: syntactic and lexical features, prosodic features, and both sets of features combined. For all feature extraction conditions, the first-stage score for each parse (generated by the k-best parser) was always included as a feature. The syntactic and lexical features are exactly those used in Charniak and Johnson (2005), lexicalized and all. The prosodic features, as described in section 5.3.2, do not incorporate any lexicalization.

A second parameter varied in our experiments was the method of upstream **edit detection** employed prior to parsing: PCFG, TAG-based, and oracle knowledge of treebank edit annotations (Section 5.4).

5.5.2 Evaluation

The experiments in this chapter use the **relaxed edited** revision (Charniak and Johnson 2001) to the standard PARSEVAL evaluation metric, which penalizes systems that get EDITED spans wrong (edit detection error), but does not penalize disagreements

in the attachment or internal structure of edit regions. This metric fits the task assuming that, while it is necessary to detect repairs so that they can be filtered out, there is usually little reason to try to recover syntactic structure in regions of speech that have been repaired or restarted by the speaker. The revised metric thus reflects this relative lack of interest in the EDITED regions by penalizing systems that get the EDITED span wrong (repair detection error), but not penalizing disagreements in the attachment or internal structure of edit regions:

- EDITED constituents in the treebank are flattened and adjacent EDITED constituents are merged, and

- there is no penalty for including or excluding EDITED constituents from any adjacent edge.

If a parse candidate proposes an edge e that abuts an EDITED edge, but the gold edge e' with the same label subsumes all the same leaf positions plus exactly that abutting EDITED edge, then e is considered a correct edge for this scoring. Likewise, if the gold edge e' does *not* include that EDITED edge, but the proposed parse e includes it, k is still considered correct.

5.5.3 Improvements in best-parse measure

Table 5.2 shows the F-scores for the top-ranked parses after reranking, where the first-stage PCFG (Charniak) parser was run with a beam-size (k of k-best) of 50. The first row shows the lower bound for reranked parsing accuracy on each edit condition — using no reranking other than the PCFG confidence. For comparison, the table also includes the oracle-rate F (the last row), calculated by selecting the parse with the best F-score available in the cohort. As the oracle-rate shows, there is considerable room for improvement. Statistical significance was computed using a non-parametric shuffle test similar to that described in Bikel (2004). For all three edit detection con-

Table 5.2: Parsing F-score of various feature and edit-detector combinations, using the framework outlined in this chapter. Parse candidates are generated with the Charniak parser.

	PCFG	TAG	Reference edits
Edit F (Table 5.1)	65.8	77.1	(100.0)
Parser 1-best	84.4	84.8	87.0
Prosodic features	85.0	85.5	87.6
Syntactic features	85.3	85.7	87.8
Combined features	85.5	86.0	88.1
Oracle-rate	92.6	93.2	95.2

ditions, the improvement from using the combined features over either prosodic or syntactic features in isolation was significant ($p < 0.005$). Similarly, for all three feature extraction conditions, the improvement from using the TAG-based edit detector instead of the PCFG edit detector is also significant ($p < 0.001$). Interestingly, the TAG's 33% reduction in edit detection error relative to the PCFG yields only about 17% of the parse accuracy differential between the PCFG and reference edit conditions. There remains a difference in parse F-score between the TAG and reference edit detection conditions to be realized by further improvements in edit detection. Feature training for the syntactic feature condition resulted in a learned weight λ vector with approximately 50K features, while the prosodic features used only about 1300 features. Despite this difference in the length of the λ vectors, the prosodic feature condition achieved roughly 75% of the improvement of the syntactic features.

5.5.4 Improvements to the parse order

In some situations, e.g. for language modeling, improving the ordering and weights of the entire parse set is important. While the results in section 5.5.3 show that the top

Table 5.3: Reranked-oracle rate parse F-score for the top s parses, as ranked by each feature set.

s	1	2	3	5	10	25
PCFG	87.0	89.8	91.0	92.2	93.4	94.6
Pros.	87.6	90.3	91.5	92.7	93.9	94.8
Syn.	87.8	90.6	91.8	93.0	94.0	94.9
Comb.	88.1	90.9	92.0	93.1	94.1	95.0

parse after reranking improves with a combination of prosodic and syntactic features, there is no guarantee of a corresponding improvement in the overall ordering. One way to look at overall ordering is to consider the **reranked oracle rate**: the best selection over the top s reranked parses in the cohort.

Table 5.3 reports reranked-oracle rate over the reranked set. The error for each feature condition, relative to using the PCFG parser in isolation, is shown in Figure 5.3. Both the table and figure show that the reranked beam achieves a consistent trend in parse accuracy improvement relative to the PCFG beam, similar to what is demonstrated by the 1-best scores (Table 5.2).

5.6 Discussion

Experiments in this chapter show a consistent improvement in parsing accuracy over conversational speech by incorporating prosodic information into the parse selection process along with non-local syntactic information. These improvements are shown to be robust to the difficulties introduced by automatic edit detection and, in addition to improving the one-best performance, to improve the overall ordering of the parse candidates.

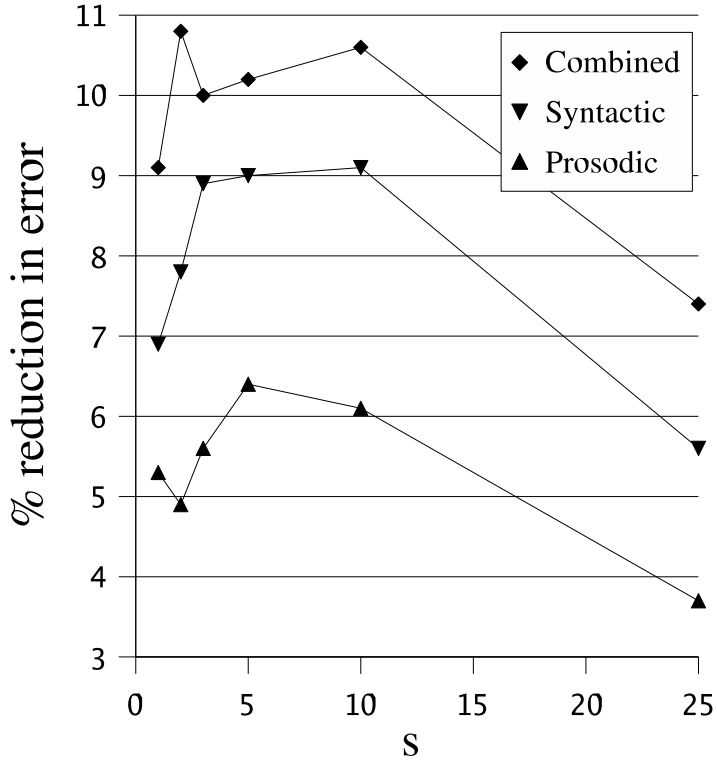


Figure 5.3: Reduction in error (Error $= 1 - F$) for the s-best reranked-oracle relative to the parser-only oracle, for different feature rerankings. Note that combined features achieve a reduction in error relative to either feature set alone.

Prosodic and syntactic features are both useful in this reranking context, and there is a gain to using them together. While the gains are not additive, since the prosodic feature selection incorporates some constituent-structure information, the combined gains are statistically significant.

Chapter 6

PARSING AUTOMATIC SPEECH RECOGNIZER TRANSCRIPTS

This chapter[1] addresses a different perspective on relaxing the lexical-layer independence assumption than the previous chapters. It relaxes the lexical-layer independence assumption, not by incorporating paralexical (prosody) information into parsing, but rather by allowing the reranking system flexibility in choosing alternative word sequences and providing acoustic confidence information along with these new word-sequence hypotheses.

Syntactic parsers, as discussed in Chapter 2, section 2.1, are usually considered to operate over a *given* set of words. A good parser models the probability of a tree given a word sequence $p(t|\vec{w})$, where t represents the parse tree, and $\vec{w}$ represents the words presented as input. Under some circumstances, a PCFG parser can be used as a language model to predict the likelihood of different word hypotheses. However, under these circumstances, the parse structure t itself is marginalized, because this approach attempts to select $\hat{w}$ to maximize the total possible probability associated with that word sequence $\sum_t p(t, w)$. Evaluation of parser models usually has *not* considered the possibility that the input words may be wrong, emphasizing instead correct span labels (e.g. the PARSEVAL measure) or other measures of good tree structure.

As discussed in Chapter 2, section 2.2, automatic speech recognition (ASR) attempts to maximize $p(\vec{w}|x)$, where x is the input acoustic signal (and sentence seg-

[1]Dustin L. Hillard helped with this project by supporting the SRI recognizer for the generation of these toolkits, through script support for the rerank learner component, and through valuable planning conversations.

mentation). Additional structure over these words $\vec{w}$ is not typically considered beyond its use in n-gram language modeling, and performance is usually measured in terms of word error rate (WER). This evaluation measure can be unhelpful for some applications because it does not directly reflect any of the relationships among the words: for example, function and content words are weighted equally, and drastically meaning-changing substitutions are no worse than the deletion of a determiner.

In a speech-to-parse system, the only input component is the input acoustic stream x. One way to combine the previous two tasks (ASR and parsing) into a speech-to-parse system might be to select an optimal tree structure $\hat{t}$ and set of words $\hat{w}$ to maximize $p(t, w|x)$. This task reflects not only the validity of the word sequence w, but also attempts to capture the relations among the words as reflected by the tree structure t.

To explore these questions, this chapter lays out an architecture (section 6.1), presents a set of experiments (section 6.2) and concludes with some discussion of these results (section 6.3).

6.1 Architecture

Like the previous chapter, this chapter uses a reranking system adapted from Charniak and Johnson (2005). Unlike the previous chapter, this chapter implements a system for generating ASR word hypotheses w and parse tree hypotheses over those words t^w and then selecting (or ranking) the best $\langle w, t^w \rangle$ hypothesis from the speech-to-parse hypothesis **cohort**. The best hypothesis is selected to minimize error rate with respect to the reference parse-tree and transcripts. This system uses the N-best output sequences from a speech-recognizer $w_i \in W$ (where $|W| \leq N$) and cascades each w_i into a M-best parser, generating up to M parses $t_j^{w_i} \in T^{w_i}$ for each proposed w_i (where $|T^{w_i}| \leq M$).

The system attempts to select the best pair $\left\langle \hat{w}, \hat{t^w} \right\rangle$ from the $M \times N$ resulting pairs

of parses ($t_j^{w_i}$) and word hypotheses (w_i). The selection is automatically learned, based on a variety of features, including

- ASR confidence $p(w_i|x)$ of a hypothesized word sequence w_i,
- Parser confidence $p(t_j^{w_i}|w_i)$ of a hypothesized parse $t_j^{w_i}$, given a hypothesized word sequence w_i, and
- Parse features: counts of various structure types.

For this work, the large Switchboard corpus described in Chapter 3 is divided into the training, development and testing sets. This corpus provides reference words and parse trees that allow the system to train and evaluate on similar corpora.

The system generates N-best ASR hypotheses over each SU segment, then feeds each of these hypotheses to a statistical parser, generating M-best parses for each ASR hypothesis, creating an $M \times N$ *cohort* for each SU. Higher-level parse features are extracted from each element of the cohort, and the cohort is re-ranked using these extracted features as well as the ASR and parser confidence scores. In this respect, the system described here is an extension of the framework proposed by Charniak and Johnson (2005); indeed, it uses components of their framework. However, it adapts their architecture in two important respects: first, instead of assuming a single word sequence as input, this architecture passes a collection of speech recognition hypotheses to the parser, and reranks over a cohort of word-and-parse hypotheses. Second, this system optimizes its reranking not on the PARSEVAL metric but the SPARSEVAL metric, described below in section 6.1.1. Figure 6.1 outlines this system, which is related to the system outlined in the previous chapter (figure 5.1).

6.1.1 Evaluation

The task, as discussed above, is to select the best words-and-tree, given only the acoustic stream. To describe *best*, the system targets the closest match to the reference

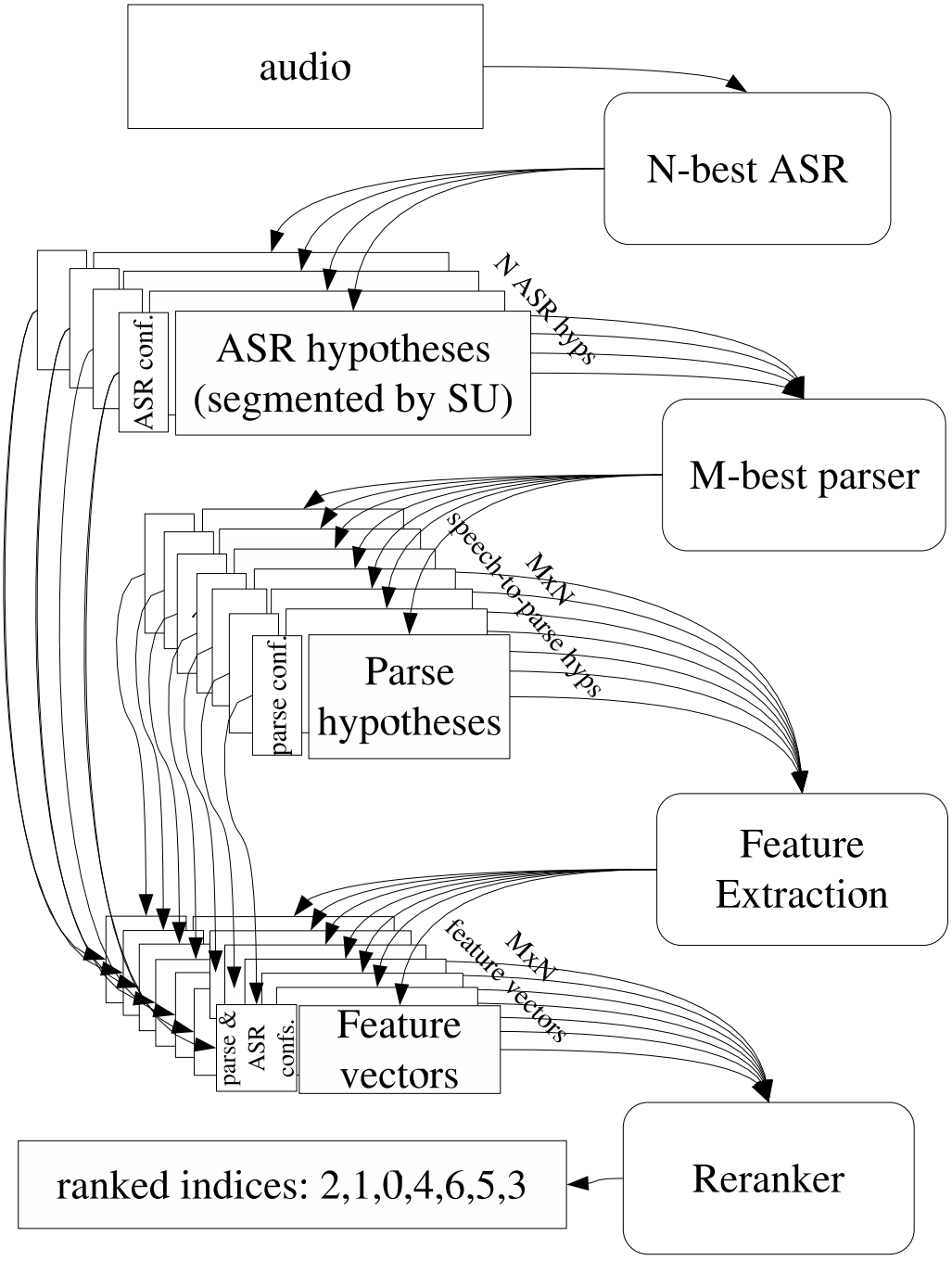


Figure 6.1: The architecture of this speech-to-parse system. Compare to the transcript-to-parse architecture in Chapter 5.

corpus. However, the distance measure involved here is potentially troublesome.

The PARSEVAL (Black et al. 1991) evaluation method for scoring parses reports bracket precision, bracket recall, and bracket matching over leaf-identical trees. This metric poses problems in the current task, because ASR errors (especially deletion and insertion) make it difficult to evaluate the bracket-matching performance when comparing a reference transcript (and tree) with ASR-hypothesized words (and a parser-hypothesized tree). It is difficult to consistently compare word spans over a sequence when the endpoint words may be deleted or inserted. Should a deleted word be included in the span or not? Conversely, the standard ASR metric is word error rate, which is evaluated *only* on the leaves of any parse tree, and does not consider tree structure at all when doing such a comparison. Thus, word error rate is not sufficient as an evaluation strategy for $p(t, w|x)$.

Because the usual evaluation metrics do not work well with this problem, this work invokes the SPARSEVAL evaluation metric, which measures error in the prediction of both tree structure and (leaf) words, as discussed in section 2.5.

6.1.2 ASR hypothesis generation

As an ASR system, this system uses an updated version of the 5× real-time system used by SRI in the Fall 2004 RT evaluations (Stolcke 2004). On the test set (128 conversation sides), this system's baseline WER is 23.8%. The system performs a single recognition and adaptation pass, and produces 1000 or more ranked hypotheses per waveform segment, which are then rescored with a number of knowledge sources, such as higher-order n-gram language models, pronunciation scores, and duration models. The final stage extracts the hypothesis with the lowest expected WER from a confusion network. A confusion network (Mangu et al. 2000) is a compacted representation of a word lattice or N-best list, where the complexity of the lattice or list representation is reduced to a simpler form that maintains all possible paths (and more) by transforming the space to a series of slots that each have word hypotheses (and null

arcs) and associated posterior probabilities.

Since the segmentation of the ASR output is based on silence, it does not agree with the SU segmentation described in Chapter 3, and therefore also does not agree with the syntactic segmentation used in the reference data. This system requires N-best lists per SU, not per ASR segment. To acquire this resegmentation, 100-best hypothesis lists are generated and transformed into confusion networks. All the confusion networks for a conversation side are concatenated, the confusion network slots are aligned with the reference transcript words, and then the confusion network is cut at the reference SU boundaries given in the reference transcript. Finally, an N-best list is generated from each of these SU-segmented confusion networks to obtain the top N word hypotheses for each SU segment. The confidences that the ASR system generates for each of these ASR hypotheses are retained for the reranking step (section 6.1.5).

Although the Switchboard corpus is included in the training data for the SRI system, this should only be a concern if the performance is unrealistically good — an overly optimistic estimate of the difficulties of speech recognition might throw the results of these experiments into doubt with respect to their generalizability to a real-world system. However, the WER reported for this system is not optimistic, in part because the system is only a 5× real-time system: 20× real-time systems have WER of less than 20%.

6.1.3 Parse hypothesis generation

To generate the M-best parse candidates from each ASR hypothesis, this system uses the first (hypothesis-generation) stage of an updated version of the parser from Charniak and Johnson (2005), trained on the training section of the corpus (using the reference words). This system is a state-of-the art lexicalized statistical parser, and it produces a confidence for each of the M-best parses it hypothesizes. Its overall confidence, like any CFG-based parser, is based on a series of tree-local predictions.

Laughter and other non-word tokens are stripped from the input word hypotheses, and ASR hypotheses that have no words are discarded; because reference SUs are given, there is guaranteed to be at least one word per SU. This strategy gives the overall speech-to-parse system a small piece of information that would not be available without reference SUs. Thus, this strategy is critically dependent on the Meteer et al. (1995) annotation of SUs, because without this SU annotation it would be impossible to discard empty sequences. However, empty sentences could be handled by introducing an empty sentence penalty term that is learned in the reranking (section 6.1.5).

6.1.4 Feature extraction

To extract non-local features of the candidate sentences, we use the syntactic feature-extractor from the reranking stage of Charniak and Johnson (2005) (the same feature extraction component that is used for the syntactic features in Chapter 5). This feature-extractor extracts binary features and counts-of-features for each member of the cohort and normalizes those features within each cohort (by demoding the cohort, as described in Chapter 5). For space and speed considerations, this work keeps only those features that appear with non-zero values in the demoded cohorts at least 2000 times; that is, the pruning threshold θ described in section 5.2 is set to 2000. This pruning leaves approximately 140,000 of these long-distance syntactic features. The feature set here is much larger because of the diversity of lexical features within cohorts (in chapter 5, the word sequences within each cohort were held constant) and because the candidate sets (cohorts) are much larger here (as many as 400 candidates in a cohort). Larger cohorts means more opportunities for some features to appear with non-zero values after de-moding, and thus more may escape the pruning. Prosody features (as described in chapter 5) are not used here.

6.1.5 Reranking with average perceptrons

After feature extraction, each cohort now has a wide variety of normalized features for each candidate, along with ASR and parser confidences for each candidate in the cohort as well. The cohort is reranked using the average-perceptron learner from Charniak and Johnson (2005), trained to maximize the F-score of the candidate chosen from each cohort. Average perceptrons (Freund and Schapire 1999) are noted for their speed in training and their relative robustness to the sorts of extremely high dimensional spaces that this problem presents.

As in Collins (2000) (and the previous chapter), training data for the reranker is generated by reparsing the training section of the corpus, using $n - 1$ folds as the training data to train the parser for generating training candidates from the n-th fold. For these experiments, $n = 10$.

6.1.6 Consolidating multiple learners

The high dimensionality of the data involved in this task incurs another problem: system memory constraints limit reranker training to roughly one-tenth of the available data set. Each training cohort is assigned at random to one of ten different learners. Test cohorts are reranked by each of the ten learners, and their rankings are are consolidated according to the average rank assigned over the ten learners. To give an example, the new first-rank candidate will thus be the cohort member with the smallest average rank assigned by the ten learners.

6.2 Experiments & Results

This section shows the results from varying some components of the system outlined in section 6.1. The feature set used for reranking includes log probability confidence scores for both the ASR system and the parse system. The ASR system's confidences are included as raw scores, as well as normalized so that the probability over each

Table 6.1: Upper (oracle) and lower (1×1) bounds for the reranking system, as measured by SPARSEVAL F score.

	Recall	Precision	F-score
20×20 oracle (upper bound)	62.5	70.9	66.5
1×1 baseline (lower bound)	48.3	50.9	49.6

cohort sums to 1. The parse scores are provided as raw and normalized per cohort as well, and (as an additional measure) provided such that the parse scores over each ASR hypothesis also sum to one (normalized per ASR hypothesis).

For some of these experiments, the test cohorts have their beam reduced: N is the number of ASR hypotheses that were considered, and M is the number of parse hypotheses that were considered, ranked by the confidence assigned by that component. In all cases, the parser and ASR components are run with $M = N = 20$, but in some experiments these lists were pruned for size at test time. Thus an $N = 5, M = 15$ experiment represents an experiment where only the top 5 recognizer hypotheses (out of 20) were considered, and from those top 5, only the top 15 parse candidates (out of 20) on each ASR hypothesis were considered. Test cohort size is not always equal to $N \times M$ in every cohort because in some circumstances (usually short sentences), one or both of the components may not emit their full complement of hypotheses.

6.2.1 Baselines

As an upper bound, the best that the system can do is to pick the best possible candidate from the cohort every time. As a lower bound, and a reasonably good baseline, one might reduce the reranking strategy to the simple (1×1) algorithm of selecting the top ASR hypothesis (as measured by ASR confidence) and using the top parse for that word sequence (as measured by the parser confidence). The results for

Table 6.2: Oracle SPARSEVAL performance (F) over various numbers of parse candidates, given reference words. Best from recognized-word 20×20 cohort (leftmost column) included for comparison.

best from recognized cohort	1-best	Top-5	Top-10	Top-15	Top-20
66.5	69.6	74.7	76.0	76.6	77.0

these two bounding systems are presented in Table 6.1.

To evaluate the dependence of the SPARSEVAL measure on speech-recognition accuracy, and as a higher upper-bound, one may consider the same SPARSEVAL measure but when parsing the reference words. Table 6.2 and Figure 6.2 report the best possible performance; for contrast, the oracle performance from the 20×20 real recognized cohort is included.

6.2.2 20×20 *experiments*

One approach to improving on the 1×1 lower bound reranking would be a simple interpolation, ignoring the higher-order parse features (those generated by the feature extraction), and including only the raw parser and ASR confidences. The task then is to find the new optimal parse $\hat{c}$ as a linear interpolation of the log scores emitted by the ASR system and the parser.

$$\hat{c} = \underset{c}{\operatorname{argmax}}\, \lambda \log p_{\text{ASR}}(c) + (1 - \lambda) \log p_{\text{parser}}(c)$$

In this ($M = 20, N = 20$) experiment, λ is optimized on 448 conversation sides from the training set, and $\lambda = 0.88$ was found to be a near-optimal weight. As shown in Table 6.3, this method achieved a small but significant ($p < 0.005$) improvement in precision and no significant improvement in recall. The F improvements were thus marginally significant ($p < 0.02$). However, this improvement is only 3% of the possible improvement in the candidate pool.

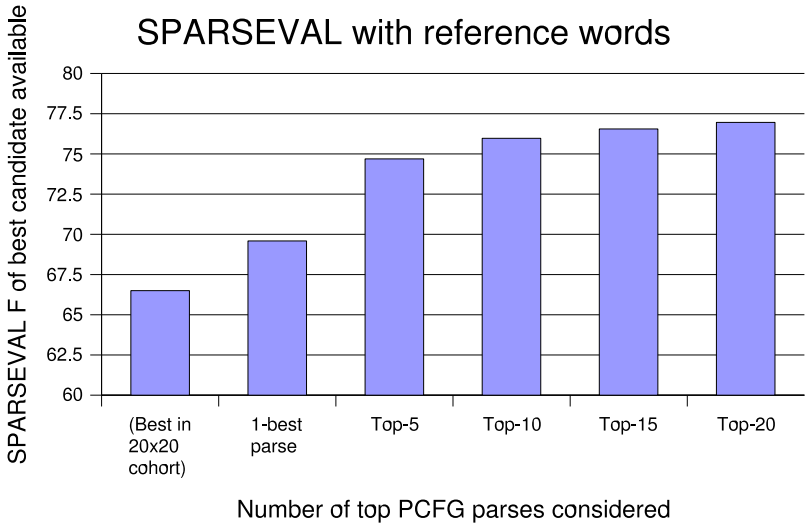


Figure 6.2: Oracle SPARSEVAL performance (F) over various numbers of parse candidates, given reference words. Best from recognized-word 20×20 cohort (leftmost column) included for comparison.

Table 6.3: SPARSEVAL F-score improvements using average perceptron over syntactic feature set and confidences vs. using linear interpolation of confidences.

	Recall	Precision	F-score
oracle (upper bound)	62.5	70.9	66.5
average perceptron	51.8	54.8	53.2
interpolation	48.8	51.5	50.1
1-best (lower bound)	48.3	50.9	49.6

Table 6.4: Word error rates for the selected candidate. Note that the WER for the candidates chosen by the average perceptron (column 2) is lower than the WER for the candidates proposed by the ASR one-best words (column 1), but higher than the WER for the candidates in the optimal SPARSEVAL score (column 3).

	ASR 1-best	Av. perceptron reranked	Oracle SPARSEVAL
WER %	23.5	22.8	25.1

More features are available to the reranker than merely the confidence scores for the parser. The average-perceptron reranker can take advantage of the much larger feature set extracted from the candidates, and adding these features to the perceptron's training and test data gives a substantial improvement in reranking, as shown in Table 6.3. As the upper bound demonstrates, the average perceptron reranker achieves 27% of the possible improvement (3.6% absolute improvement in F) in the $M = 20, N = 20$ reranking scenario.

The perceptron reranking does not optimize on the word sequence itself. Instead, it optimizes on the SPARSEVAL measure of dependency performance. Nevertheless, it is interesting to examine the word error rate of the parse candidates chosen under each of these conditions in Table 6.4. The WER for the average perceptron case is actually lower than the WER for the 1-best hypothesized ASR words, but the WER for the oracle SPARSEVAL score is *higher* than the 1-best system. The increase in error may be real, for example because of the lower weight placed on non-head function words (like determiners) by SPARSEVAL, but it may also be an artifact of noisy transcripts. Since the scoring was based on the original reference word transcripts (and not the improved Mississipi State transcripts) for consistency with the parses, the added "errors" in the oracle case may in fact be corrections.

6.2.3 Beam size experiments

The reranking component of this system can only improve the overall performance by re-ordering the cohort so that better candidate hypotheses may be selected. Thus, an upper bound on the performance of this system is the oracle rate: the best performance available in the candidate pool. In this architecture, the candidate pool is limited by the beam size of the ASR and parsing systems. In these experiments, reranker parameters are trained on ($M = 20, N = 20$) cohorts over 90% of the training set, and the remaining 10% was used as a development set, which was used to explore these values in order to preserve the test set for further experiments. These experiments explore the effect (on both oracle rate and reranked performance) of pruning the testing cohorts to smaller values of M and N. In effect, they explore the effects of pre-pruning the cohort by initial ASR confidences (reducing N) and by parse confidences (reducing M).

This work first examines how the **pruned oracle rate** changes by varying N and M. Figure 6.3 shows the F-measures over this development corpus if the best candidate is chosen from each pruned cohort. As the figure shows, the oracle rate falls off quite steeply as N and M get small (especially as N — the ASR beam — shrinks). However, most (64%) of the oracle improvement in the top 20×20 is in the top 5×5, which indicates that the overall ordering performed by the ASR and parser confidences is — on the large scale — useful.

In addition to oracle rates, it is also interesting to consider how this pruning affects the performance of the reranker — shrinking the hypothesis space reduces the best *possible* score, but are these differences perceptible to the reranker? Figure 6.4 shows the F-measure achieved over the test corpus by the reranker, when the input candidates are constrained by a smaller M and N. Unlike the oracle rate, the reranker seems to plateau at about $M = 5, N = 10$, suggesting that although the reranker can identify good parses that are near the top of the $M \times N$ lists, it is not able to take

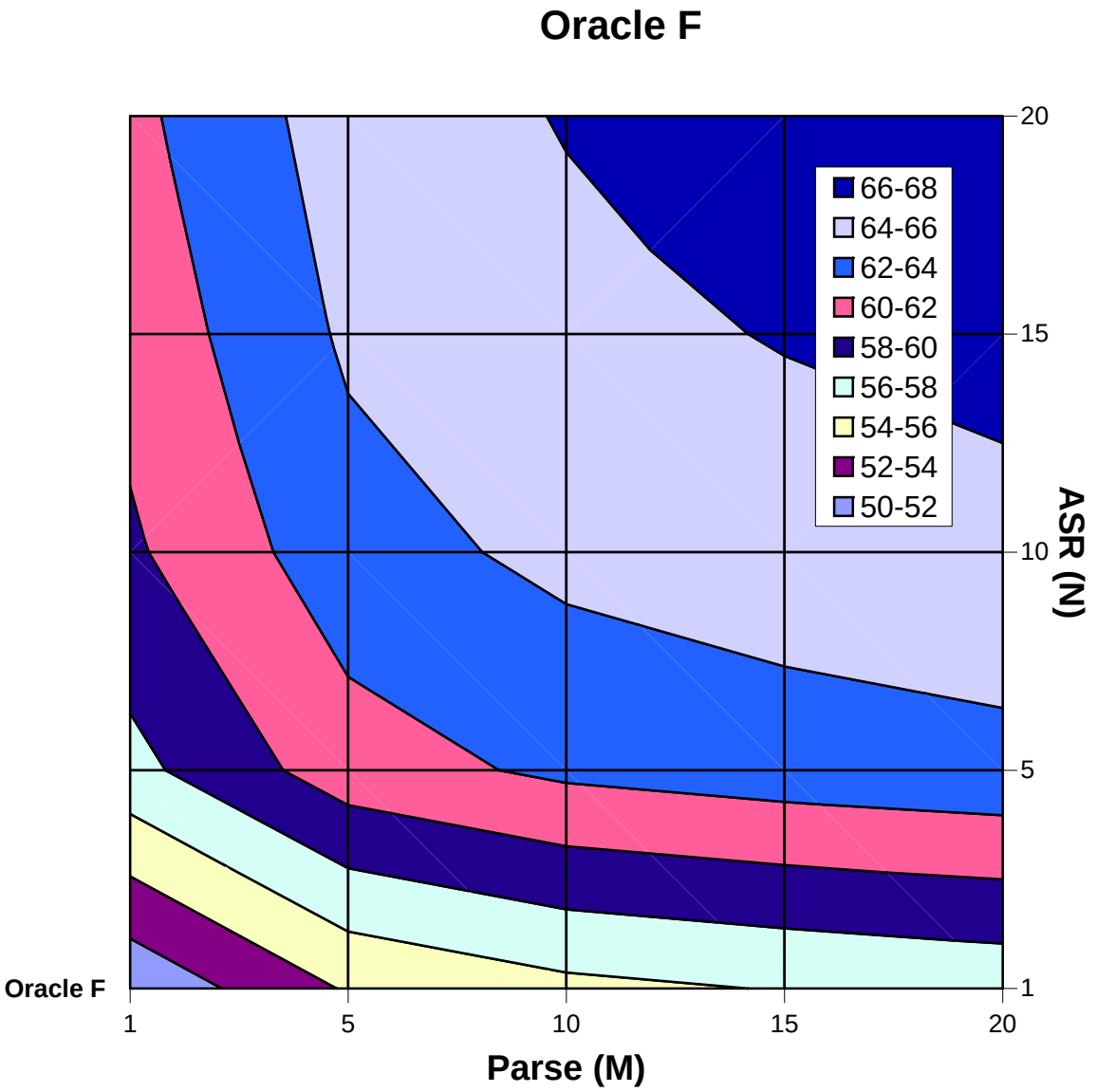


Figure 6.3: The effects of varying M and N on oracle performance. Pruned-oracle rate of F score, over varying numbers of ASR (N) and parse (M) hypotheses, on a development set.

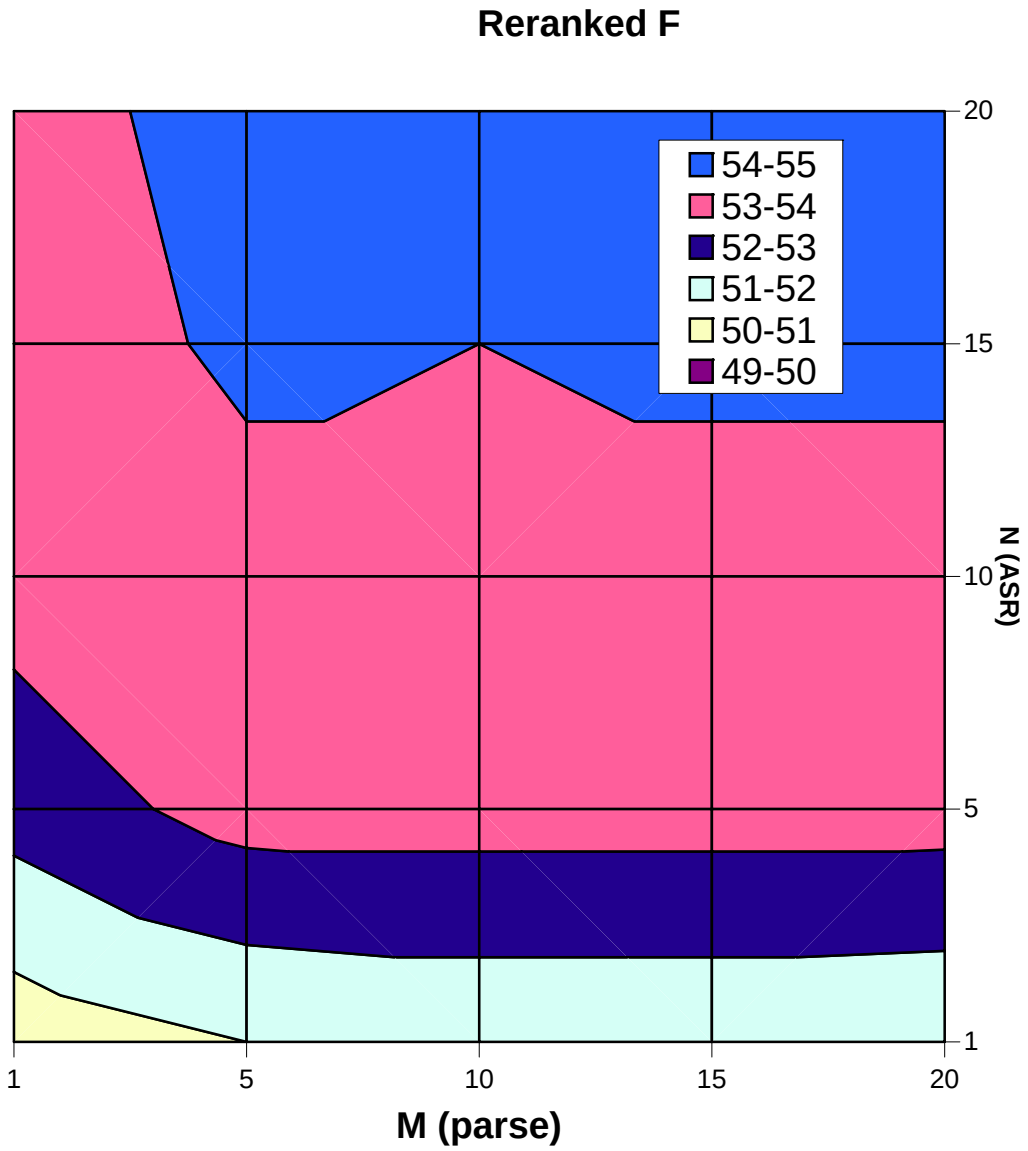


Figure 6.4: The effects of varying M and N on reranked performance (F). The F-score achieved by the reranker, over varying numbers of ASR (N) and parse (M) hypotheses, on a development set.

advantage of the continuing potential improvements available in the larger candidate set.

6.3 Discussion

The experiments presented in this chapter demonstrate that, on a dependency-oriented measure of speech-to-parse quality (SPARSEVAL), it is critically important to consider alternate hypotheses in the word sequence. Compared to a simple pipelining of ASR and parsing, the speech-to-parse system presented here achieves 27% of the possible improvement over a simple 1×1 pipelining of a 20×20 speech-to-parse candidate pool by using syntax features and subsystem confidences in an average perceptron reranking framework.

Nevertheless, it is worth noting that both the parser and the ASR system make valuable contributions to the reduction in error rate, even though they interface only through hypothesis lists. The higher-level syntactic features that are used as features for the reranking were initially designed (by Charniak and Johnson (2005)) for reranking parse candidates when the words are given, but they still serve well to improve the selection from the parse candidate pool.

Overall, the SPARSEVAL measure is highly sensitive to speech recognition error. As Figure 6.2 demonstrates, if the system is given the reference words, the 1-best parse hypothesis outperforms even the oracle (best-available) candidate from the recognized 20×20 cohort. Even beyond this difference, the poor performance of the whole system on the SPARSEVAL performance measure suggests that there is still much work to be done in parsing speech. As argued in section 2.5, the SPARSEVAL measure reflects the overall difficulty of the problem of getting word dependency relationships correct in the context of recognized speech and parsing.

Several observations about the component systems may be made from the results presented here. For all these experiments with speech-to-parse cohorts (including the

oracle and 1×1 baselines), the system precision is consistently higher than recall. In addition, the system is more robust to pruning down the parse hypotheses than it is to pruning down the ASR hypotheses: in the $M \times N$ pruning experiments, reranking only the ASR hypotheses by taking only the top parse still yields most (89%) of the F score improvements that the system can perform on the full 20×20 cohort. On the other hand, keeping only the top ASR hypothesis and reranking 20 parses over that word sequence only provides 18% of the possible improvement.

ASR variability is valuable to the reranking component, and not necessarily because of the difficulty of the ASR problem. The SPARSEVAL oracle candidates from each cohort yield a net *loss* in WER that, if real (despite the noise associated with transcript errors) suggests that an intermediate optimization on the lexical layer (as represented by word-error rate) may hurt a speech-to-parse system overall.

In summary, the results in this chapter demonstrate that speech-to-parse systems should not make the assumption that the top hypothesis of a speech recognizer is the correct word sequence. Instead, a good speech-to-parse system must target the final parse evaluation measure — in this case, SPARSEVAL. To relate this observation to the central thesis of this work: the 1-best lexical sequence is not an optimal interface layer between a speech recognizer and a parser in a speech-to-parse system. Introducing uncertainty (in the form of N-best recognizer hypothesis lists) into the interface between the speech recognizer and the parser permits a significant improvement in the quality of a speech-to-parse system.

Chapter 7

CONCLUSION

This work advances the thesis that the lexical-layer conditional independence assumption, while convenient, does not hold. It has demonstrated that paralexical and non-lexical information can improve the performance of parsers and speech-to-parse systems. This chapter summarizes the contributions made in support of this thesis and describes some directions for future work to extend this area of research.

7.1 Summary of contributions

Chapters 1 and 2 introduce the problem of lexical layer conditional independence and present background material to frame the problem in both the linguistics and natural-language processing domains. Chapter 3 describes the corpus on which these experiments were performed, and the various forms of annotation (lexical, syntactic, and prosodic) which support supervised training and allow these experiments to be evaluated.

Chapter 4 presents a set of experiments that demonstrate that two kinds of high-level prosodic information can impact statistical parsers: sentence-level segment (SU) boundaries and interruption points (IPs). This work is the first work to show parsing results that include automatic sentence segmentation; previous work on parsing speech has been based on the unrealistic assumption that segmentation is given. The performance of three different PCFG-based parsers is demonstrated to be assisted by the presence of accurate SU boundaries and IPs (or rather, hindered by their absence). In addition, this chapter demonstrates that different parsers are affected differently.

Thus, it is important for the designer of a speech-to-parse system to consider the whole problem, not just the parser performance.

In Chapter 5, it is demonstrated that sub-sentence prosodic information (in the form of ToBI-style break indices) can be used in a parse-reranking framework to improve the selection of a good parse from a pool of parse candidates generated by a PCFG-based parser. Although previous work has shown that prosody can be used in parse pruning (and to reduce parsing computational costs), this is the first work to show an improvement in performance due to the incorporation of prosodic information.

Finally, Chapter 6 also demonstrates one of the first uses of the SPARSEVAL evaluation metric to evaluate parses of automatic speech recognition hypotheses. It demonstrates that uncertainty about the word sequence is a critically important factor in selecting the best word sequence and parse from a combined speech-to-parse system.

7.2 Future work

This work provides a demonstration that relaxing the lexical-layer independence assumption can actually improve parsing (and speech-to-parse performance). This section explores possible work that might extend this observation further in a variety of ways.

7.2.1 Extension to other languages

All the work presented here was done on the English-language Switchboard corpus of telephone speech. Several components are required to extend the systems used here to other languages beyond English. First and foremost, a lexically-transcribed and syntactically-treebanked corpus of speech for training and evaluation of the component systems would be needed. Corpora to do this research on other languages exist in part: Arabic and Chinese CALLFRIEND corpora of telephone speech, for

example, are available through the Linguistic Data Consortium (LDC) catalog, as are treebanks of newswire texts in those languages, but as yet the LDC does not support a single transcribed, treebanked corpus of conversational speech in any language but English.

The main system components used here — a statistical parser, a speech segmenter, and a speech recognition system — are in principle language-independent. Furthermore, the combination strategies used here have no dependence on any language properties specific to English. In the systems used here, both the Charniak parser and the SRI speech recognizer have a number of optimizations on the English language (and in the case of the Charniak parser, the Penn Treebank tag set). However, work is being done to extend these systems to other languages. For example, the Charniak parser includes a Chinese language adaptation, Bikel and Chiang (2000) apply statistical parsing models to the Chinese treebank, and the SRI speech recognizer has recently been extended to work on Mandarin Chinese as well (Hwang et al. 2004).

The secondary components (systems for prosodic label extraction and segmentation) are even less dependent on language-specific adjustments. The Kim (2004) SU and IP detector, for example, uses only raw acoustic features (duration, pitch, and energy) and lexical cues, which are learned from a forced alighment of the orthographic transcription to the speech waveform. Although this alignment is not particularly language-specific (questions of orthography and word boundaries arise as always), the recognizer (which must be language-specific) is required to do the alignment. The prosody extraction component from Wong et al. (2005) requires only a comparable small labeled component. Since the prosody annotations used here have very little theoretical baggage (they distinguish only no-break, fluent-break, and disfluent-break), it should be easy to annotate a corpus with these distinctive categories in most if not all languages. However, the question of intra-word disfluent breaks may well raise problems in highly-agglutinating languages like Turkish or Finnish (to name

two), and it may be necessary to annotate break indices at the level of the morpheme instead. Of course, in a highly-agglutinating language, a statistical parser may need to incorporate sub-word information as well.

Thus, the biggest questions for extending this work to other languages revolve around the recording, transcription and annotation of an appropriate corpus in the new target language.

7.2.2 New metrics for measuring speech-to-parse performance

The SPARSEVAL measure used in Chapter 6 is a dependency graph projection: it attempts to match the dependency tree represented by the hypothesized and reference trees. In doing so, it only looks at the local dependencies between words: a projection of the entire graph onto local dependencies only. In a more sophisticated projection, one might want to look at longer-distance graph matching: not only is it important to connect the right words to their immediate heads, it may also be important to get the overall graph topology right. As spoken-language understanding systems approach this problem from a task-specific direction, extended syntactic measurements like SPARSEVAL approach the problem of higher-level evaluation "from the bottom up" — i.e. building up from linguistic analysis towards higher-level tasks.

7.2.3 Combining segmentation work into speech-to-parse systems

Chapter 4 demonstrates that SU segmentation is critically important to parsers, and Chapter 6 demonstrates that the speech-to-parse problem depends on quality hypothesis generation from both the ASR and the parse component. However, integrating these two tasks may be quite a challenge. One direction to explore (though a computationally-expensive one) might be to incorporate an additional hypothesis-ranking step that ranks segmentation hypotheses based on the ASR and parse hypotheses generated over those segments.

7.2.4 Variant learners in reranking environments

Due to memory constraints, the experiments in Chapter 6 used a consolidation of ten independent learners, whose results were aggregated by summing ranks. However, this rank-summing approach is not necessarily the ideal ranking solution. Another strategy might be to look at summing the *scores* (rather than the ranks), in a bagging approach, so that the learners' "opinions" about relative ranks can be continuous rather than ordered.

The experiments in Chapters 5 and 6 use maximum-entropy and average-perceptron reranking schemes respectively. In fact, the Charniak and Johnson (2005) parser from which the rerankers are taken provides a variety of reranking tools, which could be investigated to improve the speech-to-parse results.

7.2.5 Training different rerankers for different kinds of cohorts

In addition to looking for paralexical information like SU boundaries and IPs, it may be useful to look into adding discourse or dialog act information to parse selection. In a speech-to-parse environment, that information may not be easy to extract, but at a minimum, sentence length alone could be a useful criterion by which to divide the reranking training data, following the intuition that the syntactic structures of short sentences (backchannels, simple requests, affirmation and some kinds of negation) are very different than the syntactic structures of longer sentences (e.g. explanations). It is possible that the criteria that characterize a good parse of a short lexical sequence are not the same as those that characterize a good parse of a long lexical sequence. Thus, speech-to-parse cohorts could be handled by different rerankers, based on the distribution of the lengths of their hypothesized word sequences.

7.2.6 Additional prosody information

Although Chapters 4 and 5 demonstrate that prosodic features are useful to improving parser performance, the field of possible prosodic and other non-lexical features has by no means been exhausted. For example (and staying close to the system of symbols used already) using ToBI prominence and tone marks (in addition to prosodic break indices) merits further investigation in improving parse reranking.

In addition, prosodic information bears a complex relationship to transcribed punctuation. Chapter 4 explores this question, but there remain many possible parallels between the two. For example, it might be interesting to look at whether punctuation information is still useful to a system that is also presented with prosodic labeling — that is, is punctuation useful *per se* or is it serving as a marker for prosodic boundaries?

7.2.7 Incorporating segmentation and sub-sentential prosody

Chapter 4 demonstrates that segmentation impacts parser performance, and that punctuation and IP detection can improve the parsing of speech transcripts. Likewise, Chapter 5 demonstrates a way that prosody can be used to improve parser performance. Several questions may be answered in the intersection of these two insights: Does adding prosodic information help as much (or more) when the SU segmentation available is not the reference SUs? Does it help as much (or more) when IP information is present?

7.2.8 Incorporating prosody and word-uncertainty into speech-to-parse systems

The baseline reranking features (from Charniak and Johnson (2005)) do not incorporate any notion of word uncertainty or other characteristics from the acoustic sequence except for characteristics derived from the lexical sequence itself. To extend a speech-to-parse system like the one presented in Chapter 6, one approach might incorporate

the prosody components in Chapter 5. To use this strategy, the prosody labels generated over the reference data would need to be re-generated for each ASR hypothesis, since the generation process involves word identity. However, this architectural work is not intractable.

In addition, speech recognition uncertainty at the word level (rather than the sentence level) might be included as a feature in its own right. The reranking features might be extended to incorporate higher-level speech information (like the IP information that proved useful in Chapter 4). Including measures of ASR uncertainty at the word level into the features included in the speech-to-parse candidates (especially the certainty of the headword) may also improve the overall performance in the speech-to-parse task, above and beyond the improvements seen in Chapter 6.

BIBLIOGRAPHY

Abney, Steven. 1995. Chunks and dependencies: Bringing processing evidence to bear on syntax. In J. Cole, G. Green, and J. Morgan (Eds.), *Computational Linguistics and the Foundations of Linguistic Theory*, 145–164. CSLI.

Ananthakrishnan, Sankaranarayanan, and Shrikanth Narayanan. 2005. An automatic prosody recognizer using a coupled multi-stream acoustic model and a syntactic-prosodic language model. In *Proc. ICASSP*.

Aretoulaki, Maria, Stefan Harbeck, Florian Gallwitz, Elmar Nöth, Heinrich Niemann, Jozef Ivanecky, Ivo Ipsic, Nikola Pavesic, and Vaclav Matousek. 1998. SQEL: A multilingual and multifunctional dialogue system. In *Proc. Intl. Conf. on Spoken Language Processing*.

Auer, Peter. 1996. On the prosody and syntax of turn-continuations. In *Prosody in Conversation* (Couper-Kuhlen and Selting 1996), 57–100.

Bakenecker, G., U. Block, A. Batliner, R. Kompe, E. Nöth, and P. Regel-Brietzmann. 1994. Improving Parsing by Incorporating 'Prosodic Clause Boundaries' into a Grammar. In *Proc. Intl. Conf. on Spoken Language Processing*, Vol. 3, 1115–1118, Yokohama.

Baron, Don, Elizabeth Shriberg, and Andreas Stolcke. 2002. Automatic punctuation and disfluency detection in multi-party meetings using prosodic and lexical cues. In *Proc. Intl. Conf. on Spoken Language Processing*, Vol. 2, 949–952, Denver.

Batliner, A., A. Feldhaus, S. Geissler, T. Kiss, R. Kompe, and E. Nöth. 1996. Prosody, empty categories and parsing - A success story. In *Proc. Intl. Conf. on Spoken Language Processing*, Vol. 2, 1169–1172, Philadelphia, PA. Found via Ostendorf CCAI paper to-read prosody parsing.

Batliner, A., R. Kompe, A. Kießling, M. Mast, H. Niemann, and E. Nöth. 1998. M = Syntax + Prosody: A syntactic–prosodic labelling scheme for large spontaneous speech databases. *Speech Communication* 25:193–222.

Batliner, A., E. Nöth, J. Buckow, R. Huber, V. Warnke, and H. Niemann. 2001. Whence and Whither Prosody in Automatic Speech Understanding: A Case Study. In M. Bacchiani, J. Hirschberg, D. Litman, and M. Ostendorf (Eds.), *Proc. of the Workshop on Prosody and Speech Recognition 2001*, 3–12, Red Bank, NJ.

Batliner, Anton, Jan Buckow, Richard Huber, Volker Warnke, Elmar Nöth, and Heinrich Niemann. 1999. Prosodic feature evaluation: Brute force or well designed? In *Proc. International Congress of Phonetic Sciences*, 2315–2318, San Francisco, August.

Bear, John, and Patti J. Price. 1990. Prosody, syntax and parsing. In *Meeting of the Association for Computational Linguistics*, 17–22.

Bel, Bernard, and Isabelle Marlien (Eds.). 2004. *Speech Prosody 2004*, Nara, Japan. ISCA.

Bhagat, S., H. Carvey, and E. Shriberg. 2003. Automatically generated prosodic cues to lexically ambiguous dialog acts in multi-party meetings. In *Proc. International Congress of Phonetic Sciences*.

Bikel, Dan. 2004. *On the Parameter Space of Lexicalized Statistical Parsing Models*. PhD thesis, University of Pennsylvania.

Bikel, Daniel M., and David Chiang. 2000. Two statistical parsing models applied to the Chinese treebank. In *Second Chinese Language Processing Workshop*.

Blache, Philippe, and Daniel Hirst. 2001. Aligning prosody and syntax in property grammars. In *Proceedings of EuroSpeech 2001*.

Black, E., S. Abney, D. Flickenger, C. Gdaniec, R. Grishman, P. Harrison, D. Hindle, R. Ingria, F. Jelinek, J. Klavans, M. Liberman, M. Marcus, S. Roukos, B. Santorini, and T. Strzalkowski. 1991. A procedure for quantitatively comparing syntactic coverage of English grammars. In *Proc. 4th DARPA Speech & Natural Lang. Workshop*, 306–311.

Boros, M., J. Haas, V. Warnke, E. Nöth, and H. Niemann. 1998. How Statistics and Prosody can guide a Chunky Parser. In *Proc. of the AIII Workshop on Artificial Intelligence in Industry*, 388–398.

Brenier, Jason M., and Laura A. Michaelis. 2005. Optimization via syntactic amalgam: Syntax-prosody mismatch and copula doubling. *Corpus Linguistics and Linguistic Theory* 1(1):45–88.

Bresnan, Joan. 2001. *Lexical-functional syntax*. Malden, Mass.: Blackwell.

Bybee, Joan. 1995. Regular morphology and the lexicon. *Language and Cognitive Processes* 10(5):425–455.

Carmichael, Lesley Marie. 2005. *Situation-based intonation pattern distribution in a corpus of American English*. Linguistics, University of Washington, Seattle.

Charniak, Eugene. 1993. *Statistical Language Learning*. Cambridge, Massachusetts: The MIT Press.

Charniak, Eugene. 2000. A maximum-entropy-inspired parser. In *Proc. NAACL*, 132–139.

Charniak, Eugene. 2001. Immediate-head parsing for language models. In *Proc. ACL*.

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

Charniak, Eugene, and Mark Johnson. 2005. Coarse-to-fine n-best parsing and maxent discriminative reranking. In *Proc. ACL*, 173–180, Ann Arbor, Michigan, June. Association for Computational Linguistics.

Charniak, Eugene, Kevin Knight, and Kenji Yamada. 2003. Syntax-based language models for statistical machine translation. In *Machine Translation Summit*. Intl. Association for Machine Translation.

Chelba, Ciprian. 2000. *Exploiting syntactic structure for natural language modeling*. PhD thesis, Johns Hopkins University, Maryland.

Chelba, Ciprian, and Frederick Jelinek. 2000. Structured language modeling. *Computer Speech and Language* 14(4):283–332.

Chen, Ken, Mark Hasegawa-Johnson, and Aaron Cohen. 2004a. An automatic prosody labeling system using ANN-based syntactic-prosodic model and GMM-based acoustic-prosodic model. In *Proc. ICASSP*.

Chen, Ken, Mark Hasegawa-Johnson, Aaron Cohen, and Jennifer Cole. 2004b. A maximum likelihood prosody recognizer. In *Speech Prosody 2004* (Bel and Marlien 2004).

Chomsky, Noam. 1957. *Syntactic Structures*. The Hague: Mouton.

Collins, Michael. 2000. Discriminative reranking for natural language parsing. In *ICML '00: Proceedings of the Seventeenth International Conference on Machine Learning*, 175–182, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Collins, Michael. 2003. Head-driven statistical models for natural language parsing. *Computational Linguistics* 29(4):589–638.

Core, M., and K. Schubert. 1999. Speech repairs: A parsing perspective. In *Satellite Meeting ICPHS 99*.

Couper-Kuhlen, Elizabeth, and Margaret Selting (Eds.). 1996. *Prosody in Conversation: Interactional Studies*. Cambridge University Press.

Cutler, Anne, Delphine Dahan, and Wilma van Donselaar. 1997. Prosody in the comprehension of spoken language: a literature review. *Language and Speech* 40(2):141–201.

Cutugno, Francesco, L. D'Anna, M. Petrillo, and E. Zovato. 2002. APA: Towards an automatic tool for prosodic analysis. In *Speech Prosody 2002* (SP 2002).

Dahl, Deborah A., Madeleine Bates, Michael Brown, William Fisher, Kate Hunicke-Smith, David Pallett, Christine Pao, Alexander Rudnicky, and Elizabeth Shriberg. 1994. Expanding the scope of the ATIS task: the ATIS-3 corpus. In *Proc. HLT*.

Deshmukh, Neeraj, Aravind Ganapathiraju, Andi Gleeson, Jonathan Hamaker, and Joseph Picone. 1998. Resegmentation of SWITCHBOARD. In *Proc. ICASSP*.

Etzioni, Oren, Michael Cafarella, Doug Downey, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S. Weld, and Alexander Yates. 2005. Unsupervised named-entity extraction from the web - an experimental study. *Artificial Intelligence*.

Ferrer, Luciana, Elizabeth Shriberg, and Andreas Stolcke. 2002. Is the speaker done yet? faster and more accurate end-of-utterance detection using prosody in human-computer dialog. In *Proc. Intl. Conf. on Spoken Language Processing*, Vol. 3, 2061–2064, Denver.

Freund, Yoav, and Robert E. Schapire. 1999. Large margin classification using the perceptron algorithm. *Machine Learning* 37(3):277–296.

Fujisaki, Hiroya. 2004. Information, prosody, and modeling — with emphasis on tonal features of speech. In *Speech Prosody 2004* (Bel and Marlien 2004).

Gallwitz, Florian, M. Aretoulaki, M. Boros, J. Haas, S. Harbeck, R. Huber, H. Niemann, and E. Nöth. 1998. The Erlangen Spoken Dialogue System EVAR: A State–of–the–Art Information Retrieval System. In *Proceedings of 1998 International Symposium on Spoken Dialogue (ISSD 98)*, 19–26, Sydney, Australia.

Gazdar, Gerald, Ewan Klein, Geoffrey K. Pullum, and Ivan A. Sag. 1985. *Generalized Phrase Structure Grammar*. Blackwell Publishing.

Godfrey, J. J., E. C. Holliman, and J. McDaniel. 1992. Switchboard: Telephone speech corpus for research and development. In *Proc. ICASSP*, Vol. I, 517–520, San Francisco.

Goldsmith, John A. 1979. *Autosegmental Phonology*. Garland.

Greenberg, Steven, Hannah Carvey, Leah Hitchcock, and Shuangyu Chang. 2003. Temporal properties of spontaneous speech — a syllable-centric perspective. *Journal of Phonetics* 31:465–485.

Gregory, Michelle, Mark Johnson, and Eugene Charniak. 2004. Sentence-internal prosody does not help parsing the way punctuation does. In *Proc. NAACL*.

Gussenhoven, Carlos. 2004. *The phonology of tone and intonation*. Cambridge University Press.

Haas, Jürgen, Manuela Boros, Elmar Nöth, Volker Warnke, and Heinrich Niemann. 1998. A Concept for a Prosodically and Statistically Driven Chunky Semantic Parser. In *Text, Speech, Dialog*, 357–362.

Haegeman, Liliane. 1991. *Introduction to Government and Binding Theory*. Oxford: Basil Blackwell.

Heeman, Peter A., and James F. Allen. 1999. Speech repairs, intonational phrases, and discourse markers: modeling speakers' utterances in spoken dialogue. *Computational Linguistics* 25(4):527–571.

Heldner, Mattias, and Beáta Megyesi. 2003. Exploring the prosody-syntax interface in conversations. In *Proc. International Congress of Phonetic Sciences*.

Huang, J., and G. Zweig. 2002. Maximum entropy model for punctuation annotation from speech. In *Proc. Eurospeech*.

Hwang, Mei-Yuh, Xin Lei, Tim Ng, Ivan Bulyko, Mari Ostendorf, Andreas Stolcke, Wen Wang, Jing Zheng, Venkata Ramana Rao Gadde, Martin Graciarena, Man-Hung Siu, and Yan Huang. 2004. Progress on Mandarin conversational telephone

speech recognition. In *Intl. Symposium on Chinese Spoken Language Processing*. IEEE.

ISCA. 2002. *Speech Prosody 2002*, Aix-en-Provence, France, April.

ISIP. 1997. Mississippi State transcriptions of SWITCHBOARD. http://www.isip.msstate.edu/projects/switchboard/.

Jelinek, Frederick. 1997. *Statistical methods for speech recognition*. MIT Press.

Johnson, Mark, and Eugene Charniak. 2004. A TAG-based noisy channel model of speech repairs. In *Proc. ACL*, 33–39.

Johnson, Mark, Eugene Charniak, and Matthew Lease. 2004. An improved model for recognizing disfluencies in conversational speech. In *Proc. of the Rich Text 2004 Fall Workshop*.

Jurafsky, Daniel. 1996. A probabilistic model of lexical and syntactic access and disambiguation. *Cognitive Science* 20(2):137–194.

Jurafsky, Daniel, and James H. Martin. 2000. *Speech and Language Processing: An introduction to natural language processing, computational linguistics, and speech recognition*. Prentice Hall.

Jurafsky, Daniel, Chuck Wooters, Jonathan Segal, Andreas Stolcke, Eric Fosler, Gary Tajchman, and Nelson Morgan. 1995. Using a stochastic context free grammar as a language model for speech recognition. In *Proc. ICASSP*.

Kager, Ren, and Wim Zonneveld (Eds.). 1999. *Phrasal Phonology*. Nijmegen University Press.

Kahn, Jeremy G., Matthew Lease, Eugene Charniak, Mark Johnson, and Mari Ostendorf. 2005. Effective use of prosody in parsing conversational speech. In *Proc. HLT/EMNLP*. To appear.

Kahn, Jeremy G., Mari Ostendorf, and Ciprian Chelba. 2004. Parsing conversational speech using enhanced segmentation. In *Proc. HLT-NAACL*, 125–128.

Kang, Soyoung, and Shari Speer. 2002. Prosody and clause boundaries in Korean. In *Speech Prosody 2002* (SP 2002).

Kang, Soyoung, and Shari R. Speer. 2004. Prosodic disambiguation of participle constructions in English. In *Speech Prosody 2004* (Bel and Marlien 2004).

Kim, J. 2004. Automatic detection of sentence boundaries, disfluencies and conversational fillers in spontaneous speech. Master's thesis, University of Washington.

Kim, J., S. E. Schwarm, and M. Ostendorf. 2004. Detecting structural metadata with decision trees and transformation-based learning. In *Proc. HLT*.

Kim, J.-H., and P. Woodland. 2001. The use of prosody in a combined system for punctuation generation and speech recognition. In *Proc. Eurospeech*, 2757–2760.

Klein, Dan, and Chris Manning. 2003a. A* parsing: Fast exact viterbi parse selection. In *Proc. HLT-NAACL*.

Klein, Dan, and Chris Manning. 2003b. Accurate unlexicalized parsing. In *Proc. ACL*, 423–430.

Kompe, R., A. Kiessling, H. Niemann, E. Nöth, A. Batliner, S. Schachtl, T. Ruland, and H. U. Block. 1997. Improving parsing of spontaneous speech with the help of prosodic boundaries. In *Proc. ICASSP*, 811–814, Munich, Germany.

Kraljic, Tanya, and Susan E. Brennan. 2005. Prosodic disambiguation of syntactic structure: for the speaker or for the addressee? *Cognitive Psychology* 50:194–231.

Ladd, D. Robert. 1996. *Intonational Phonology*. Cambridge University Press.

LDC. 2004. Simple metadata annotation specification. Technical report, Linguistic Data Consortium. Available at http://www.ldc.upenn.edu/Projects/MDE.

Levy, Roger. 2005. Processing difficulty in verb-final clauses matches syntactic expectations. Presented at the 2005 annual meeting of the Linguistic Society of America.

Lindblom, Björn. 1990. *Explaining Phonetic Variation: A sketch of the H&H theory*. Kluwer Academic Publishers. 403–439.

Liu, Y., E. Shriberg, and A. Stolcke. 2003. Automatic disfluency identification in conversational speech using multiple knowledge sources. In *Proc. Eurospeech*, Vol. 1, 957–960.

Liu, Yang, Elizabeth Shriberg, Andreas Stolcke, Barbara Peskin, Jeremy Ang, Dustin Hillard, Mari Ostendorf, Marcus Tomalin, Phil Woodland, and Mary Harper. 2005. Structural metadata research in the EARS program. In *Proc. ICASSP*.

Luce, Paul A., and David B. Pisoni. 1998. Recognizing spoken words: The neighborhood activation model. *Ear and Hearing* 19(1):1–36.

Magerman, David M. 1995. Statistical decision-tree models for parsing. In *The Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*, 276–283, San Francisco. The Association for Computational Linguistics, Morgan Kaufman.

Mangu, Lidia, Eric Brill, and Andreas Stolcke. 2000. Finding consensus in speech recognition: word error minimization and other applications of confusion networks. *Computer Speech and Language* 373–400.

Mani, Nivedita. 2004. The role of prosody in parsing ambiguous sentences. In *Speech Prosody 2004* (Bel and Marlien 2004).

Marcus, Mitchell P., Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: the Penn treebank. *Computational Linguistics* 19(1).

Marslen-Wilson, William D., L. K. Tyler, P. Warren, P. Grenier, and C. S. Lee. 1992. Prosodic effects in minimal attachment. *The Quarterly Journal of Experimental Psychology* 45A(1):73–87.

Martin, Philippe. 2004. WinPitchPro — a tool for text to speech alignment and prosodic analysis. In *Speech Prosody 2004* (Bel and Marlien 2004).

Mayfield, L., M. Gavalda, Y. Seo, B. Suhm, W. Ward, and A. Waibel. 1995. Parsing real input in JANUS: a concept-based approach. In *Proc. TMI 95*.

Meteer, Marie, Ann Taylor, Robert MacIntyre, and Rukmini Iyer. 1995. Dysfluency annotation stylebook for the switchboard corpus. Technical report, Linguistic Data Consortium (LDC).

Mixdorff, Hansjörg. 2002. Speech technology, ToBI, and making sense of prosody. In *Speech Prosody 2002* (SP 2002).

Mohr, David N., David W. Turner, Gregory R. Pond, Joseph S. Kamath, Cathy B. De Vos, and Paul C. Carpenter. 2003. Speech recognition as a transcription aid: A

randomized comparison with standard transcription. *Journal of the American Medical Informatics Association* 10(1).

Nakano, Mikio, Yasuhiro Minami, Stephanie Seneff, Timothy J. Hazen, D. Scott Cyphers, James Glass, Joseph Polifroni, and Victor Zue. 2001. MOKUSEI: A telephone-based Japanese conversational system in the weather domain. In *Proc. Eurospeech*.

NIST. 2003. Rich Transcription Fall 2003 Evaluation (RT-03F). Technical report, NIST, http://www.nist.gov/speech/tests/rt/rt2003/fall/.

NIST. 2004. Meeting room project at NIST. Documentation available on the web at http://www.nist.gov/speech/test_beds/mr_proj/.

NIST. 2005. NIST speech recognition scoring toolkit (SCTK). Technical report, NIST, http://www.nist.gov/speech/tools/.

Nöth, E., A. Batliner, A. Kießling, R. Kompe, and H. Niemann. 2000. VERBMOBIL: The use of prosody in the linguistic components of a speech understanding system. In *IEEE Trans. on Speech and Audio Processing*, Vol. 8,5, September.

Nöth, E., A. Batliner, V. Warnke, J. Haas, M. Boros, J. Buckow, R. Huber, Florian Gallwitz, M. Nutt, and H. Niemann. 2002. The use of prosody in automatic dialogue understanding. *Speech Communication* 36(1).

Nöth, E., M. Boros, J. Fischer, Florian Gallwitz, J. Haas, R. Huber, H. Niemann, G. Stemmer, and V. Warnke. 2001. Research issues for the next generation spoken dialogue systems revisited. *Lecture Notes in Computer Science* 2166:341+.

Oard, Douglas, Dina Demner-Fushman, Jan Hajic, Bhuvana Ramabhadran, Samuel Gustman, William Byrne, Dagobert Soergel, Bonnie Dorr, Philip Resnik, and

Michael Picheny. 2002. Cross-language access to recorded speech in the MALACH project. In *Proceedings of the Text, Speech, and Dialog Workshop.*

Oepen, Stephan, Helge Dyvik, Jan Tore Lønning, Erik Velldal, Dorothee Beermann, John Carroll, Dan Flickinger, Lars Hellan, Janne Bondi Johannessen, Paul Meurer, Torbjørn Nordgård, and Victoria Rosén. 2004. Som å kapp-ete med trollet? Towards MRS-based Norwegian - English Machine Translation. In *Proceedings of the 10th International Conference on Theoretical and Methodological Issues in Machine Translation*, Baltimore, MD.

Oepen, Stephan, Dan Flickinger, Kristina Toutanova, and Christopher D. Manning. 2002. LinGO Redwoods: A rich and dynamic treebank for HPSG. In *Proceedings of The First Workshop on Treebanks and Linguistic Theories (TLT2002)*, Sozopol, Bulgaria.

O'Grady, William, Michael Dobrovolsky, and Mark Aronoff. 1993. *Contemporary Linguistics: An Introduction.* St. Martin's Press.

Ostendorf, Mari, Izhak Shafran, Stefanie Shattuck-Hufnagel, Lesley Carmichael, and William Byrne. 2001. A prosodically labeled database of spontaneous speech. In *Proc. of the ISCA Workshop on Prosody in Speech Recognition and Understanding*, 119–121, 10.

Ostendorf, Mari, Colin W. Wightman, and Nanette M. Veilleux. 1993. Parse scoring with prosodic information: an analysis/synthesis approach. *Computer Speech and Language* 7:193–210.

Pollard, Carl J., and Ivan A. Sag. 1994. *Head-driven phrase structure grammar.* Stanford: CSLI.

Precoda, Kristin, Horacio Franco, Ascander Dost, Michael Frandsen, John Fry, Andreas Kathol, Colleen Richey, Susanne Riehemann, Dimitra Vergyri, Jing Zheng, and Christopher Culy. 2004. Limited-domain speech-to-speech translation between English and Pashto. In *Proc. HLT*. In demo sessions.

Price, Patti J., Mari Ostendorf, Stefanie Shattuck-Hufnagel, and C. Fong. 1991. The use of prosody in syntactic disambiguation. *Journal of the Acoustical Society of America* 90(6):2956–2970.

Pynte, Joël, and Bénédicte Prieur. 1996. Prosodic breaks and attachment decisions in sentence parsing. *Language and Cognitive Processes* 11(1/2):165–191.

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

Rossi, Pierluigi Salvo, Francesco Palmieri, and Francesco Cutugno. 2002. A method for automatic extraction of Fujisaki-model parameters. In *Speech Prosody 2002* (SP 2002).

Safárová, Marie, and Marc Swerts. 2004. On recognition of declarative questions in english. In *Speech Prosody 2004* (Bel and Marlien 2004).

Schafer, Amy J., Shari R. Speer, Paul Warren, and S. David White. 2000. Intonational disambiguation in sentences production and comprehension. *Journal of Psycholinguistic Research* 29(2):169–182.

Schäfer, Ulrich. 2003. WHAT: An XSLT-based infrastructure for the integration of natural language processing components. In *Proceedings of the HLT-NAACL 2003 Workshop on Software Engineering and Architecture of Language Technology Systems*.

Schwarm, Sarah, and Mari Ostendorf. 2005. Reading level assessment using support vector machines and statistical language models. In *Proc. ACL*, 523–530, Ann Arbor, Michigan, June. Association for Computational Linguistics.

Sekine, S., and M. Collins. 1997. EVALB. As in Collins ACL 1997; http://nlp.cs.nyu.edu/evalb/.

Selkirk, Elisabeth O. 1984. *Phonology and Syntax: The relation between sound and structure*. MIT Press.

Seneff, Stephanie. 1992. TINA: a natural language system for spoken language applications. *Computational Linguistics* 18(1):61–86.

Shriberg, Elizabeth, et al. 2000. Prosody-based automatic segmentation of speech into sentences and topics. *Speech Communication* 32(1–2):127–154.

Shriberg, Elizabeth, and Andreas Stolcke. 2004. Direct modeling of prosody: An overview of applications in automatic speech processing. In *Speech Prosody 2004* (Bel and Marlien 2004).

Silverman, Kim, Mary Beckman, John Pitrelli, Mari Ostendorf, Colin Wightman, Patti Price, Janet Pierrehumbert, and Julia Hirschberg. 1992. ToBI: A standard for labeling English prosody. In *Proc. Intl. Conf. on Spoken Language Processing*, 867–870.

Snedeker, Jesse, and John Trueswell. 2003. Using prosody to avoid ambiguity: Effects of speaker awareness and referential context. *Journal of Memory and Language* 48:102–130.

SRI. 2003. Description of Meteer/TB3 to LDC/V5 mapping. Technical report, SRI.

Srivastava, A., and F. Kubala. 2003. Sentence boundary detection in Arabic speech. In *Proceedings of the Eighth European Conference on Speech Communication and Technology*, 949–952.

Steedman, Mark. 2000. Information structure and the syntax-phonology interface. *Linguistic Inquiry* 31(4):649–689.

Stolcke, Andreas. 2004. Speech-to-text research at SRI-ICSI-UW. In *Proceedings of the NIST Rich Transcription Workshop*.

Stolcke, Andreas, Ciprian Chelba, David Engle, Victor Jimenez, Lidia Mangu, Harry Printz, Eric Ristad, Roni Rosenfeld, Dekai Wu, Fred Jelinek, and Sanjeev Khudanpur. 1996. Dependency language modeling. Technical report, CLSP, Johns Hopkins. WS96 Project Report.

Stolcke, Andreas, Elizabeth Shriberg, Rebecca Bates, Mari Ostendorf, Dilek Hakkani, Madelaine Plauché, Gökhan Tür, and Yu Lu. 1998. Automatic detection of sentence boundaries and disfluencies based on recognized words. In R. H. Mannell and J. Robert-Ribes (Eds.), *Proc. Intl. Conf. on Spoken Language Processing*, Vol. 5, 2247–2250, Sydney. Australian Speech Science and Technology Association.

Strangert, Eva. 2004. Speech chunks in conversation: Syntactic and prosodic aspects. In *Speech Prosody 2004* (Bel and Marlien 2004).

Strassel, S. 2003. *Simple Metadata Annotation Specification V5.0*. Linguistic Data Consortium.

Taylor, Paul. 2000. Analysis and synthesis of intonation using the Tilt model. *Journal of the Acoustical Society of America* 107(3):1697–1714.

Truckenbrodt, Hubert. 1999. On the relation between syntactic phrases and phonological phrases. *Linguistic Inquiry* 30(2):219–255.

Veilleux, Nanette M., and Mari Ostendorf. 1993. Prosody/parse scoring and its application in ATIS. In *Proc. ARPA Human Language Technology Workshop '93*, 335–340, Princeton, NJ.

Vereecken, Halewijn, Jean-Pierre Martens, Cynthia Grover, Justin Fackrell, and Bert Van Coile. 1998. Automatic prosodic labeling of 6 languages. In *Proc. ICASSP*.

Wang, C., S. Cyphers, X. Mou, J. Polifroni, Stephanie Seneff, J. Yi, and Victor Zue. 2000. MUXING: A telephone-access Mandarin conversational system. In *Proc. Intl. Conf. on Spoken Language Processing*.

Warnke, Volker, Elmar Noth, Heinrich Niemann, and Georg Stemmer. 1999. A segment based approach for prosodic boundary detection. In *TSD*, 199–202.

Webster, Gabriel. 2002. *Toward a psychologically and computationally adequate model of speech perception*. PhD thesis, University of Washington, Linguistics.

Wightman, Colin W. 2002. ToBI or not ToBI? In *Speech Prosody 2002* (SP 2002).

Wightman, Colin W., and Mari Ostendorf. 1994. Automatic labeling of prosodic patterns. *IEEE Transactions on Speech and Audio Processing* 2(4):469–481.

Wightman, Colin W., Stefanie Shattuck-Hufnagel, Mari Ostendorf, and Patti J. Price. 1992. Segmental durations in the vicinity of prosodic phrase boundaries. *Journal of the Acoustical Society of America* 91(3):1707–1717.

Wong, Darby, Mari Ostendorf, and Jeremy G. Kahn. 2005. Using weakly supervised learning to improve prosody labeling. Technical Report UWEETR-2005-0003, University of Washington Electrical Engineering Dept.

Yamada, Kenji, and Kevin Knight. 2001. A syntax-based statistical translation model. In *Meeting of the Association for Computational Linguistics*, 523–530.

Yeh, Alexander. 2000. More accurate tests for the statistical significance of result differences. In *COLING 18*, Vol. 2, 947–953.

Zue, Victor, Stephanie Seneff, James R. Glass, Joseph Polifroni, C. Pao, Timothy J. Hazen, and L. Hetheringon. 2000. JUPITER: a telephone-based conversational interface for weather information. *IEEE Transactions on Speech and Audio Processing* 8(1):85–96.

Appendix A

PERFORMANCE OF VARIOUS PARSERS WITH VARIOUS SU AND IP CONDITIONS

Chapter 4 discusses the performance of various parsers with a variety of SU and IP conditions. This appendix includes the complete performance results for the experiments described in that chapter.

Each of these experiments reports bracketing precision (P) and recall (R) (and the F harmonic mean of these), in which high numbers indicate better matches between hypothesized and reference parse structures. Also reported is the average bracket crossing (BX), in which low numbers indicate better matches between reference and hypothesis parse structures. Note that these average bracket-crossings are much higher than results usually reported because the bracket-crossings are counted per conversation side (due to the TIPTOP evaluation described in Chapter 4, section 4.3.3).

A.1 Small corpus experiments

The experimental results presented in this section were performed on the smaller corpus described in chapter 3. For discussion, please see Chapter 4.

Table A.1: Performance of the Structured Language Model as a parser, trained on the subset corpus.

Segmentation	R	P	*F*	BX
Naïve	68.05	58.67	63.01	80.40
Auto −IP	72.47	63.09	67.46	66.21
Auto +IP	74.25	64.46	69.01	58.09
Reference −IP	77.45	68.40	72.64	46.80
Reference +IP	77.98	68.42	72.89	44.56
Reference +Punct −IP	68.89	77.98	73.15	41.66
Reference +Punct +IP	78.65	73.97	76.23	35.84

Table A.2: Performance of the Charniak parser, trained on the subset corpus.

Segmentation	R	P	*F*	BX
Naïve	62.64	56.19	59.24	144.20
Auto −IP	71.79	64.39	67.89	89.36
Auto +IP	72.87	65.13	68.78	88.43
Reference −IP	80.85	72.24	76.30	40.18
Reference +IP	83.38	73.95	78.38	36.37

Table A.3: Performance of the Bikel parser, trained on the subset corpus.

Segmentation	R	P	*F*	BX
Naïve	66.53	64.93	65.72	111.27
Auto −IP	74.47	74.66	74.56	72.29
Auto +IP	75.05	74.93	74.99	72.18
Reference −IP	82.42	83.36	82.89	36.77
Reference +IP	84.14	84.31	84.22	34.23

A.2 Full corpus experiments

The tables presented in this section were performed after training on the full corpus described in table 3.1.

Table A.4: Performance of the Charniak parser, trained on the full corpus.

Segmentation	R	P	*F*	BX
Naïve	62.78	56.28	59.35	144.06
Auto −IP	71.81	64.45	67.93	89.51
Auto +IP	72.87	65.13	68.78	88.43
Reference −IP	80.85	72.24	76.30	40.18
Reference +IP	83.38	73.95	78.38	36.37

Table A.5: Performance of the Bikel parser trained over the full corpus.

Segmentation	R	P	*F*	BX
Naïve	66.92	65.21	66.05	111.25
Auto −IP	75.07	75.17	75.12	71.02
Auto +IP	75.48	75.36	75.42	70.94
Reference −IP	82.91	83.79	83.35	36.37
Reference +IP	84.45	84.59	84.52	34.52