A Robust Loose Coupling for Speech Recognition and Natural Language Understanding *

Eric K. Ringger
ringger@cs.rochester.edu
http://www.cs.rochester.edu/u/ringger/

The University of Rochester
Computer Science Department
Rochester, New York 14627

Technical Report 592

September 1995

---

*This technical report consists of the author's Ph.D. thesis proposal, appropriately modified from the version that was defended on May 23, 1995. Members of the thesis committee are James Allen, Richard Aslin, and Lenhart Schubert.

This work was supported by the University of Rochester Computer Science Department and ONR/ARPA research grant number N00014-92-J-1512.

Contents

1 Introduction **3**

1.1 Understanding Spoken Dialogue . . . 3

1.2 Speech Recognition . . . 5

1.2.1 State of the Art . . . 5

1.2.2 Problem: Recognition Errors on Spontaneous Speech . . . 6

1.2.3 Overview of Related Work . . . 7

1.2.4 Proposed Solution: Correcting Speech Recognition Errors . . . 8

1.3 Robust Parsing . . . 9

1.3.1 State of the Art . . . 9

1.3.2 Problem: Incremental Phrase Structure . . . 10

1.3.3 Overview of Related Work . . . 11

1.3.4 Proposed Solution: Identifying Chunks Using Prosody and Probabilities . 11

1.4 Summary of Contributions . . . 12

1.5 Proposal Organization . . . 13

2 Background and Related Work **14**

2.1 A Brief History of Speech Recognition to the Present . . . 14

2.2 Issues in Acoustic Modeling . . . 16

2.2.1 Discretizing the Speech Signal . . . 17

2.2.2 Hidden Markov Models . . . 17

2.3 Language Modeling . . . 21

2.3.1 Complexity of Language Models . . . 21

2.3.2 Collocation-based Models . . . 24

2.3.3 Phrase Structure Models . . . 25

2.4 Search and Speech Recognition/Parser Interfaces . . . 28

2.4.1 Tight Coupling . . . 29

2.4.2 Loose Coupling . . . 33

2.4.3 Semi-Tight Coupling . . . 35
2.5 Robust Parsing Strategies . . . 36
2.5.1 Robust Pattern Matching . . . 37
2.5.2 Robust Grammar-based Parsing . . . 38
2.6 Prosodic Analysis . . . 38
2.6.1 Prosodic Features . . . 39
2.6.2 Prosodic Features Aid Parsing . . . 39

3 Speech Recognition in the Trains Domain 41
3.1 HTK-based Speech Recognizer Trained from Trains Dialogue Corpus . . . 41
3.1.1 The TRAINS Dialogue Corpus . . . 41
3.1.2 SR Prototype . . . 42
3.1.3 Evaluation . . . 42
3.2 Language Models for Sphinx-II in TRAINS-95 . . . 43
3.2.1 The TRAINS-95 System . . . 43
3.2.2 Speech Recognition and Parser . . . 44
3.2.3 TRAINS-95 Dialogue Collection, Segmentation, and Transcription . . . 46
3.2.4 New Language Model . . . 46
3.3 Conclusions . . . 47

4 Improving the Loose Coupling 48
4.1 Correcting Speech Recognition Errors . . . 48
4.1.1 Motivation . . . 49
4.1.2 Statistical Machine Translation . . . 49
4.1.3 Correcting Recognition Errors Using a Post-Processor . . . 50
4.1.4 The Model . . . 50
4.1.5 Preliminary Results . . . 55
4.1.6 Other Issues . . . 55
4.2 Parsing Spontaneous Utterances . . . 56
4.2.1 Probabilistic Scores . . . 56
4.2.2 Prosodic Features . . . 56
4.3 Evaluation . . . 57
4.4 Data Collection . . . 57
4.5 Conclusions . . . 57

Abstract

The focus of this thesis proposal is to improve the ability of a computational system to understand spoken utterances in a dialogue with a human. Available computational methods for word recognition do not perform as well on spontaneous speech as we would hope. Even a state of the art recognizer achieves slightly worse than 70% word accuracy on (nearly) spontaneous speech in a conversation about a specific problem.

To address this problem, I will explore novel methods for post-processing the output of a speech recognizer in order to correct errors. I adopt statistical techniques for modeling the noisy channel from the speaker to the listener in order to correct some of the errors introduced there. The statistical model accounts for frequent errors such as simple word/word confusions and short phrasal problems (one-to-many word substitutions and many-to-one word concatenations). To use the model, a search algorithm is required to find the most likely correction of a given word sequence from the speech recognizer. The post-processor output should contain fewer errors, thus making interpretation by higher levels, such as parsing, more reliable.

Spontaneous speech is also challenging to process because it is more incremental than written language. Utterances frequently form brief phrases and fragments rather than full sentences; they tend to come in installments and refinements. Known methods for parsing do not perform as well as we would like in the face of these linguistic ambiguities and idiosyncrasies. Even state of the art algorithms for parsing spontaneous language sustain high error rates.

To address the incrementality of spontaneously spoken utterances, I will develop methods for segmenting a given utterance into "chunks" representing individual thoughts. Given an utterance of spontaneous speech, a tool for automatic prosodic feature extraction will analyze the output of the error-correcting post-processor and the acoustic waveform to generate prosodic cues. These cues will aid a robust parser using a prosody-wise grammar to identify the incremental phrases in the utterance and to provide a syntactic analysis.

These components will augment the TRAINS-95 conversational planning assistant.

Keywords: Dialogue, Prosody, Spoken Language Understanding, Spontaneous Speech, the TRAINS-95 **System.**

List of Figures

1.1 *Implementing Interpretation in Stages for Spoken Dialogue Understanding.* 4
1.2 *Post-Processing Speech Recognition to Correct Errors.* 8

2.1 *The Language Compiler for HARPY* [Lowerre and Reddy, 1986] 15
2.2 *The Architecture for Sphinx (adapted from* [Rudnicky *et al.*, 1994]). 16
2.3 *A Finite Markov Chain in Discrete Time with Stationary Transition Probabilities.* . 18
2.4 *A Discrete Hidden Markov Model with Finite, Vector-Valued Output Distributions.* 19
2.5 *Couplings Categorized by Tightness: (a) Tight; (b) Semi-Tight; (c) Loose.* 28
2.6 *Composite HMM Acoustic Model and Bigram Language Model.* 30
2.7 *A State/Time-Step Lattice Constructed from a Composite HMM/Bigram Model.* . . 31
2.8 *SR Output: (a) 1-best; (b)* n*-best (with scores); (c) Word Lattice (with scores).* . . 34

3.1 *Architecture of the* TRAINS-95 *System (adapted from* [Allen *et al.*, 1995]). 44

4.1 *Recovering Word-Sequences Corrupted in a Noisy Channel.* 51
4.2 *Alignment of a Hypothesis and the Reference Transcription.* 54

List of Tables

2.1 *Spectrum of Speech Recognition/Natural Language Understanding Interfaces Along the Tightness Dimension.* 29

3.1 *Word Recognition Performance.* 42

3.2 *Current Corpus of* TRAINS-95 *Dialogues.* 46

3.3 *Utterance Recognition Performance.* 47

3.4 *Word Recognition Performance.* 47

4.1 *Bigram Language Model for Post-Processor Search Example.* 52

4.2 *Simple Channel Language Model for Post-Processor Search Example.* 53

4.3 *Scored Hypotheses in Post-Processor Search Example.* 53

4.4 *Preliminary Results for First Draft Post-Processor (PP).* 55

Chapter 1

Introduction

1.1 Understanding Spoken Dialogue

For a human, interacting with a computer can be a tedious or impractical process. For most tasks, the human must come to the computer on the computer's terms and learn modes of communication that fit naturally with the computer's *modus operandi*. Given the computational abilities of the computer, it makes sense to offload the burden of communication onto the computer. This is where research in spoken language understanding (and other user-friendly interfaces) enters the picture. Spoken natural language is a promising communication mode for computer-human interaction because speaking comes very naturally to most people. For tasks in which the hands are otherwise engaged, natural speech can be ideal. Even for tasks that require other input modes, such as keyboard and mouse, spoken language can significantly enhance the ease and efficiency of communication for the human. Furthermore, speech has the potential to provide universal access to computers and related resources without specialized training. In a recent overview of speech technology, Rudnicky, Hauptmann, and Lee [1994] concluded, "Perhaps the greatest potential lies in the development of systems that combine recognition and synthesis to support conversational interaction between humans and computers in complex task domains."

Spoken dialogue is fundamentally the interchange of acoustic data between the computer and the human as they alternate in the roles of *speaker* and *listener*. In the role of speaker, a participant in the dialogue communicates information to the listener using an acoustic signal. By interpreting the spoken signal, the listener transforms the acoustic waveform into a meaning form, which she can manipulate and on which she can perform inference. We say that the listener "understands" the speaker if she is able to use the information in some meaningful way. For our purposes, a meaningful use of the information is manifested in a few possible ways: the conversation is maintained via a verbal response or another recognizable gesture (such as an update to a graphical interface), and/or a mutually agreeable plan for solving a particular problem evolves as a result of the conversation.

Problem-solving dialogue between two people exhibits remarkable properties. Such conversations rarely stall with one person waiting for the other. Speakers often recognize when they have misheard or misunderstood something, and they request information anew. Cognitive processes at many levels contribute to the robust flow of the conversation. In order for computer-human dialogue to be as robust, the computer system must be able to model many of the abilities of a human listener

and speaker.

Our ultimate objective is to construct a computational system capable of participating in such a dialogue, but this thesis will focus only on the role of the listener and the problems encountered there. To construct a successful listener, we must model and implement the interpretation process. We can conceive of an incremental transduction from incoming acoustic data to the meaning representation. Taking a reductionist attack on the problem, we will examine the interpretation process in stages. This brings us to the division of a Spoken Language Understanding system (a "listener") into components capable of performing each level of interpretation. The most common division of a system is into Speech Recognition (SR), Natural Language Understanding (NLU), and Dialogue Manager (DM) components as depicted in Figure 1.1.

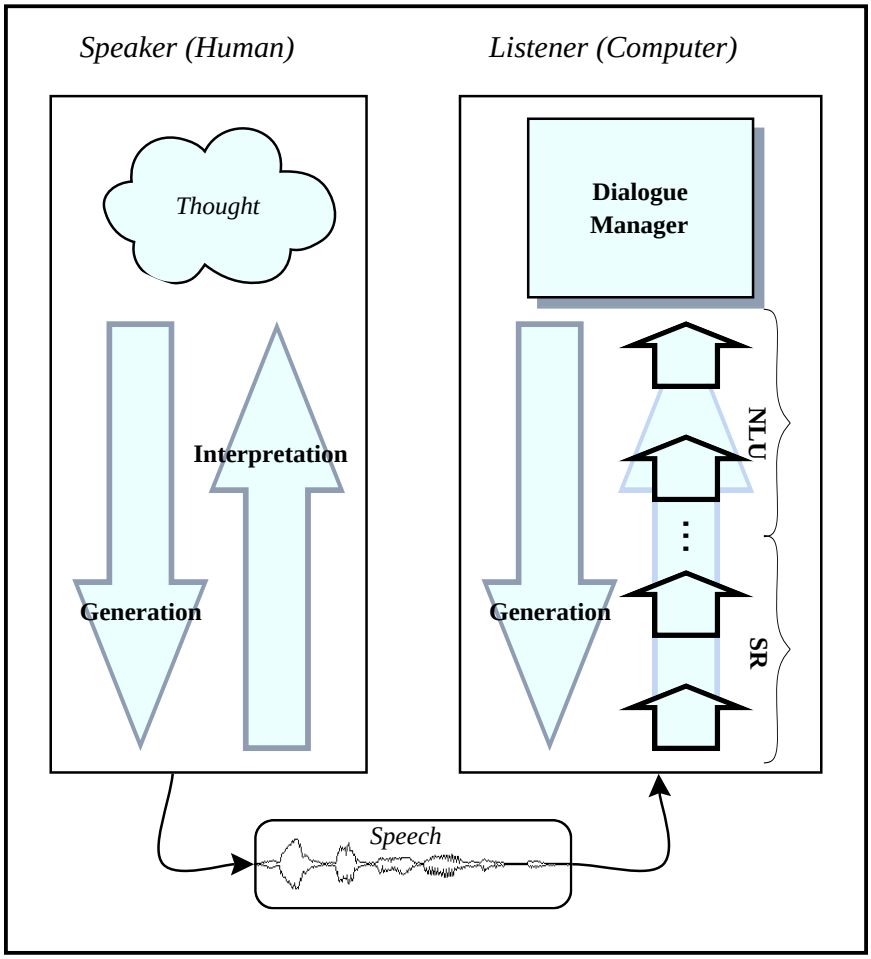


Figure 1.1: *Implementing Interpretation in Stages for Spoken Dialogue Understanding.*

This thesis proposal focuses on improving the ability of a computational system to understand spontaneous spoken utterances in a dialogue with a human by correcting misrecognitions and by using prosodic information to aid in the robust interpretation. For the remainder of this chapter, we first focus on the process of speech recognition. We review the state of the art in SR, present the pressing problem of recognition errors, and propose a technique for correcting some of them.

Second, we move across the interface with NLU to robust natural language parsing. There we summarize the state of the art, describe the problem of parsing incremental spontaneous speech, and propose a solution involving the use of prosodic features. The subsequent section provides a synopsis of the contributions to be made by this thesis.

1.2 Speech Recognition

1.2.1 State of the Art

Modern speech recognition technology is reliable and fast for certain kinds of speech. As a rule of thumb, highly constrained speech is easy to predict and, therefore, to recognize. For example, discrete speech, in which words are uttered in staccato fashion to clearly mark their boundaries, is easy to recognize. Furthermore, speech used for commanding and controlling a device can be constrained to contain only structured commands acceptable to the device; such speech can also be recognized with high accuracy. Moving up in complexity, recognizing a person's speech while the person reads continuously (fluently) from printed text (this kind of speech is referred to as "read speech" in much of the literature) that uses a large vocabulary (*ca*. 60,000 words) with high precision has come within reach in the past two years. Results from the 1995 ARPA Spoken Language Systems Technology Workshop [ARPA, 1995] indicate that performance on large vocabulary read speech is approaching 95% word accuracy. [1] If we remove the constraints provided by reading text, then we have *careful* spontaneous speech containing words from a medium-sized vocabulary (*ca*. 3000 words). Such speech can now be recognized with word accuracy above 97% in real-time. [2] It is important to note that this is not true spontaneous speech as most people use with one another; it is more paced and careful. Significantly, these feats have been achieved in a speaker-independent fashion.

The problem of recognizing human speech has been approached from several directions, including artificial neural networks (*e.g.*, [Waibel *et al.*, 1989]) and statistical modeling techniques [Waibel and Lee, 1990; Rabiner and Juang, 1993]. Spectral coding techniques, Hidden Markov Models, and simple models of language have been the mainstay of the most recent SR systems capable of recognizing large-vocabulary, real-time, speaker-independent, continuous speech. Also, recent increases in microprocessor speed and the inclusion of analog-to-digital converters in most systems have enabled speech recognition on workstation- and PC-class machines without additional special-purpose hardware (*c.f.* [Nguyen *et al.*, 1993]). It is therefore reasonable to expect that plug-and-play speech recognition software will be available for the mass-market in the near future. The 1994 formation of BBN Hark Systems Corporation, and the activities involving automatic dictation at Apple, Dragon Systems, IBM, Kurzweil, and Microsoft are indicative of this trend. Those who purchase such a speech recognizer will most likely want to regard it as a "black-box," having a clearly specified function and well-defined inputs and outputs but otherwise providing no hooks for altering or tuning its internal operations. For example, based on literature from BBN Hark Systems, their recognizer is already being used by customers for an automated directory assistance system, an air traffic control system, an air traffic control simulation and training system, and

[1] Note that such results are not reported as being computed in real-time or anywhere near real-time.

[2] Note that this figure considers only utterances that are "answerable" in the ATIS domain – only 75% of all utterances.

BBN's own voice-activated telephone dialing system [BBN HARK Systems, 1994]. Furthermore, the Hark system is being used at Sun Labs in their research on Speech User Interfaces [Yankelovich *et al.*, 1995]. As another example of using speech recognition as a black-box, several research labs have announced plans to make speech recognition available to the research community by running publicly accessible speech servers on the Internet.

1.2.2 Problem: Recognition Errors on Spontaneous Speech

For the computer, automatic understanding of true *spontaneous* speech is challenging because it is less constrained than the kinds of speech mentioned above. Recognizing spontaneous speech requires significant effort due to the numerous ambiguities that must be resolved. Spontaneous speech is replete with filled and silent pauses, human and environmental noise, and unknown and partial words [Young and Ward, 1993]. As a consequence of these acoustic ambiguities and idiosyncrasies, the available computational methods for analyzing speech do not perform as well as we would hope on the spontaneous speech that we want to handle. In fact, recognition errors are common: even a state of the art recognizer achieves only slightly worse than 70% word accuracy on (nearly) spontaneous speech in a conversation about a specific problem. (The examples below are drawn from problem-solving dialogues that we have collected from users interacting with the TRAINS system. TRAINS, the dialogues, and recognition results will be described in section 3.2.4.).

Recognition errors frequently occur at function words as demonstrated by the following examples. In the examples, the HYP tag indicates the SR system's *hypothesis*, and the REF tag indicates the *reference* transcription.

```
REF:  SEND THE  TRAIN    FROM MILWAUKEE TO CHARLESTON
HYP:  SEND THAT TRAIN AT FROM MILWAUKEE TO CHARLESTON

REF:  GREAT NOW I    WANT TO GO FROM WASHINGTON    TO BOSTON
HYP:  GREAT NOW I'LL WANT A  GO FROM WASHINGTON IN TO BOSTON

REF:  RIGHT SEND THE  TRAIN FROM MONTREAL TO CHARLESTON
HYP:  RATE  SEND THAT TRAIN FROM MONTREAL TO CHARLESTON
```

Errors also plague content words. As in the above examples, misrecognitions may be as simple as substituting a single word for another.

```
REF:  GO TO BALTIMORE
HYP:  GO TO OUTSIDE
```

Here is an utterance with a replacement of a single word by multiple smaller words:

```
REF:  GO FROM CHICAGO TO TOLEDO
HYP:  GO FROM CHICAGO TO TO LEAVE AT
```

The following utterance is an example of the opposite error in which multiple words are erroneously concatenated to make a longer word:

```
REF: GO DIRECTLY FROM BUFFALO TO PITTSBURGH
HYP: GO DIRECTLY FROM BUFFALO TENTH
```

The severity of a misrecognition does not necessarily stop with replacements, however. Here are more complex examples in which adjacent words are misrecognized and in which the hypothesized words overlap the boundary between the reference words:

```
REF:      GREAT OKAY NOW WE COULD GO FROM SAY  MONTREAL TO WASHINGTON
HYP:  I'M GREAT OKAY NOW WEEK IT  GO FROM CITY MONTREAL TO WASHINGTON

REF: SEND THE TRAIN FROM CHARLESTON   TO BOSTON
HYP: SATURDAY AND   FROM CHARLOTTE IN TO BOSTON
```

Introducing a few spontaneous artifacts complicates the problem for the recognizer:

```
REF: UH     NO   NO UM  BRING THE TRAIN FROM CHARLESTON TO LEXINGTON
     THROUGH CINCINNATI
HYP: BUTTON KNOW NO OWN BRING THE TRAIN FROM CHARLESTON TO LEXINGTON AT
     THREE   WASN'T CITY
```

Finally, even for a recognizer that is trained independently of the speaker, some people who do not roughly match the profile of those in the training set simply cannot be "understood" by it. Here is an example that demonstrates the severity of the problem.

```
REF: SEND  THE TRAIN FROM    CHARLOTTE TO    CHICAGO
HYP: SALAD X J MEAN  FRIENDS DIRECTLY  SHOOT HOW ABOUT
```

(Interestingly enough, the recognizer does seem to have properly segmented the utterance even though no single segment is accurately labeled.)

Every one of these errors is a stumbling block for understanding. Successful understanding requires significant effort on the computer's part, including effort to repair or recover from recognition errors.

1.2.3 Overview of Related Work

We concur with others that some of these recognition errors can be repaired by enhancing the SR/NLU interface. Some work has attempted to overcome recognition errors during higher level NLU processing. The simplest approach involves a loose coupling in which the SR module provides a single sentence hypothesis to subsequent modules. Several researchers have termed this approach "throwing a sentence over the wall." The positive side to this approach is its simplicity. The downfall is that the SR module makes no attempt either to consult higher level knowledge sources or to forward ambiguities to later modules for possible resolution.

A second and more sophisticated approach to overcoming recognition errors shows more promise. In this approach, the SR component generates a list of the best transcription hypotheses, feedback from the parser to the recognizer is minimal to non-existent, and the parser evaluates the list using higher-level knowledge in order to re-rank the alternatives and select the best one. Hence, here the SR system relies on the ability of subsequent components to resolve ambiguities. Unfortunately, when the SR output does not contain every correct word of the spoken utterance, no amount of rescoring will repair the omission. However, another advantage to this approach is that

independent groups can develop the individual SR and NLU modules without intimate collaboration; *i.e.*, the NLU components can be designed to regard the SR system as a black-box. For this reason, many research groups have focused on such loose couplings.

Other work has attempted to *prevent* (rather than simply overcome) recognition errors by coupling more tightly with higher level NLU processes. One approach involves the use of high-level grammars to constrain the operation of the SR. One problem with this approach is its computational complexity; no real-time recognizer for a vocabulary of significant size has tightly incorporated such higher-level models. A second problem is the inflexibility of most grammar formalisms in handling deviant input. When a grammar expects a well-formed "sentence," it will most likely fail to successfully guide the recognition process when applied to an agrammatical utterance.

1.2.4 Proposed Solution: Correcting Speech Recognition Errors

I will explore novel methods for using a speech recognizer in a loosely coupled arrangement; in other words, the setting of my work includes the assumption that the SR module is a black-box from the point of view of subsequent modules. I will design and implement techniques for post-processing the output of the SR in order to reduce recognition errors. To explain the idea, I will refer to Figure 1.2. Current statistical speech recognition models are based on the assumption that when

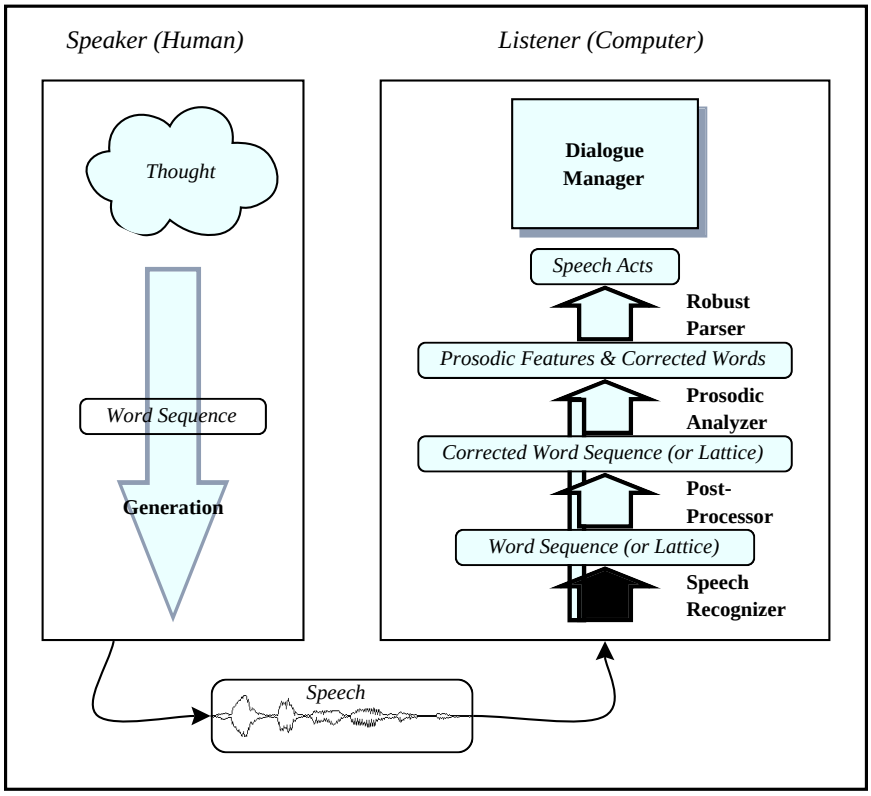


Figure 1.2: *Post-Processing Speech Recognition to Correct Errors.*

someone wishes to speak, they (incrementally) generate a word sequence from which the acoustic

signal is constructed (see Figure 1.2). From the point where the words are generated and uttered, the spoken acoustic signal is transmitted and heard by the listener. This pathway is assumed to be imperfect and to introduce noise; hence, it is called a "noisy channel." I extend this pathway by including the SR "black-box" at the end. Thus, my noisy channel begins with the speaker's word sequence and culminates at the output of the SR module (see Figure 1.2). I then adopt statistical techniques for modeling that channel in order to correct some of the errors introduced there.

The statistical model I will adopt describes some of the effects on utterances in a specific task domain when they pass through the entire noisy channel. Specifically, it should account for frequent errors such as simple word/word confusions and short phrasal problems (one-to-many word substitutions and many-to-one word concatenations). In addition to the model, a suitable search algorithm is required. This algorithm will use the model to find the most likely correction for a given word sequence from the SR module. Consequently, the post-processor output should contain fewer errors, thus making the task of interpretation by higher levels, such as parsing, more reliable.

The information in the model cannot be expressed in existing low-level language models. However, the model's content could be utilized by the speech recognizer itself if the recognizer were suitably modified. However, the statistical model will be trained on utterances in a particular domain; therefore, the post-processor truly does belong outside of the recognizer, which we view as a general-purpose component.

The primary advantage to this approach over existing approaches for correcting SR errors lies in its ability to introduce options that are not available in the SR module's output. As mentioned above, existing rescoring tactics cannot do so. One disadvantage lies in the need to actually train an additional error correction model.

One open issue is the form of the post-processor's input and output. I will perform experiments involving two possible representations. In the simple configuration, the input and output of this post-processor will be a single word-sequence. In the more complex representation, the input and output will be a word-lattice with probabilistic tags. This will also require that the parser be modified to parse word-lattices.

1.3 Robust Parsing

1.3.1 State of the Art

On the natural language understanding front, recent research has yielded robust parsers for arbitrary text, such as the broad-coverage parser under development at Microsoft Research [Richardson, 1994]. This parser is able to generate useful analyses despite agrammaticalities in the input.

Other parsers have been designed specifically for robustly analyzing transcriptions of spoken language. As we have seen robustness is required both to recover from recognition errors by the SR module and to handle performance errors by the human speaker. Examples include the Gemini parser at SRI [Dowding *et al.*, 1993], the TINA parser at MIT [Seneff, 1992], the Phoenix [Ward, 1990] and GLR* [Lavie, 1994] parsers at CMU, and the Delphi parser at BBN [Stallard and Bobrow, 1992]. One technique commonly applied by these robust parsers is the use of domain knowledge to limit ambiguity and to thereby enable faster parsing.

For spoken language systems, the marriage of speech recognition and NLU has been reasonably successful. One particularly successful direction has been domain-specific spoken query answering systems. The (D)ARPA Air Travel Information System (ATIS) initiative resulted in several systems capable of answering spoken queries in the air travel domain [ARPA, 1995]. The Berkeley Restaurant Project (BeRP) [Jurafsky *et al.*, 1994] system is similar to ATIS systems in so far as each utterance is independent of the previous conversation; no dialogue state is maintained. Other systems simplify the nature of dialogue maintain exclusive initiative by rigidly controlling the conversation and by prompting the human for information. Work at the Oregon Graduate Institute [Oviatt, 1994] relaxes the system-initiative constraint but seeks to guide user responses into brief sentences to potentially eliminate spoken disfluencies. Thus, some existing work takes advantage of stateless utterances, as in ATIS, or of constrained input expectation, as in the OGI work. However, most natural human-human dialogue does not proceed in either fashion. Each party in a dialogue is free to bring new information to the table at arbitrary points in time.

1.3.2 Problem: Incremental Phrase Structure

Problems arise in parsing spontaneous speech because speaker performance errors are frequent. People often speak agrammatically, and they repair their utterances on the fly. Furthermore, spontaneous dialogue is replete with false-starts, interruptions, and sub-dialogues [Heeman and Allen, 1994b]. Spontaneous speech is also challenging to process because it is more incremental than written language. Utterances frequently form brief phrases and fragments rather than full sentences. They tend to come in installments and refinements and to make use of topic-comment phrase structure [Brown and Yule, 1983]. For example, the following utterance from the TRAINS Dialogue Corpus (TDC) illustrates topic-comment structure. (The dialogue identifier and utterance number of each TDC utterance are provided as (dialogue:utterance). The TDC will be described in more detail later.)

> so from Corning to Bath – how far is that (d92-1:utt5)

Known methods for parsing do not perform as well as we would like in the face of linguistic ambiguities and idiosyncrasies such as incremental phrase structure. Even state of the art algorithms for parsing spontaneous language sustain high error rates (*c.f.* [ARPA, 1995]). To address the incrementality of spontaneously spoken utterances, we require methods for segmenting a given utterance into "chunks" representing individual thoughts.

In written language, capitalization and punctuation aid in segmenting an utterance into individual thoughts. For spoken language, pitch, intonation, loudness, rhythm, and stress serve this purpose [Waibel, 1987]. These prosodic features are known to play a critical role for humans in their perception of speech. Many levels of interpretation can use prosodic cues from the speech signal to enhance the speed and accuracy of the analysis [Pierrehumbert and Hirschberg, 1990; Rowles and Huang, 1992]. For example, in the TDC utterance quoted above, the "- -" indicates a pause. This simple prosodic annotation helps to separate the topic from the question.

The following single turn from the TDC further illustrates how prosodic cues can aid in robustly analyzing spontaneous speech:

> not at the same time okay we're going to hook up engine E2 to the boxcar at Elmira and send that off to Corning now while we're loading the boxcar with oranges at Corning we're gonna take engine E3 and sent it over to Corning hook it up to the tanker car and send it back to Elmira

The listener needs to segment this into individual utterances in order to successfully interpret them. Without prosodic information, the structure of this string is highly ambiguous. For instance, consider the middle part of the turn where the word "now" occurs. It may be a temporal adverbial, as in the single statement "Send that off to Corning now, while we're loading the boxcar." On the other hand, "now" could be the first word in a new section of the discourse, such as "Send that off to Corning. Now, while we're loading the boxcar, . . . " The different phrasings yield radically different interpretations. Listening to the dialogue, the prosody clearly indicates the latter interpretation.

This leads to the need to define a unit of comprehension for spoken language that does not correspond to a sentence or a complete turn. This idea is not new. Halliday [1967] proposed that intonation serves to break up speech into informational units, while Gee and Grosjean [1983] proposed that the prosodic structure is the basic structure in language processing. It is likely that a model combining prosody with higher-levels of interpretation will be necessary to form these units.

1.3.3 Overview of Related Work

Although prosodic cues are used by humans in segmenting and understanding spontaneous speech, current robust parsing strategies typically rely only on phrase structure models and heuristics, neglecting prosodic features. Simply put, available information is not being adequately used, which again motivates the need to improve the interface between SR and NLU. [3]

Techniques for extracting useful prosodic cues do exist. Wightman and Ostendorf have developed algorithms for the automatic labeling of prosodic phrasing and for the automatic detection of prominences, syllables emphasized by the speaker [Wightman and Ostendorf, 1994]. These two prosodic features, phrasing and prominence, mark syntactic information needed for disambiguation. However, these labeling algorithms are limited because they were developed for use on speech from professional FM radio news announcers *reading* scripts. Consequently, their current algorithm performs poorly on spontaneous speech.

Using the Wightman and Ostendorf algorithm, Ostendorf, Wightman, and Veilleux [1993] have reported using automatically detected prosodic phrasing to evaluate possible parses. They achieved performance approaching that of human listeners. Unfortunately, this approach is external to the parsing framework where significant influence can be had. Work by Harper *et al.* [1994] actually integrates prosodic constraints into a constraint-based parsing framework.

1.3.4 Proposed Solution: Identifying Chunks Using Prosody and Probabilities

Faced with the use of a modern SR system that neglects all but the most basic prosodic cues (silence and filled pauses), I will use Wightman and Ostendorf's tool for prosodic analysis to analyze the output of the error-correcting post-processor proposed above as shown in Figure 1.2. Given an utterance of spontaneous speech, this tool will take the acoustic waveform and the post-processor output to generate prosodic phrasing and prominences. Necessary modification for handling spontaneous speech will be investigated, possibly in conjunction with Colin Wightman (currently at the New Mexico Institute of Mining and Technology).

[3] This problem is related to the fact that most modern SR systems typically use only phonetic features characterized by the spectral properties of the speech. They effectively discard prosodic features [Waibel, 1987; Waibel, 1988] except for some non-verbal sounds and silence [Ward, 1989].

I will first augment the error-correcting post-processor proposed above to incorporate the simple prosodic cues (silences and filled pauses) provided by the SR module itself. Then, once the prominence-and-phrasing tool is operational, it will be wedged between the post-processor and the parser. I will perform experiments to determine how useful the following criteria are in segmenting the input while parsing: prosodic phrasing and prominence, phrase-structure rules, probabilistic scores, and key-word spotting. I will augment the grammar with statistical information in addition to prosodic features and constituents to guide the parser in preferring more probable and prosodically plausible sequences of phrases.

1.4 Summary of Contributions

This thesis will contribute models and methods for overcoming speech recognition errors and for dealing with the incremental nature of spontaneous speech. These ends will be reached by the following means:

- a model and working implementation of a post-processor to correct recognition errors for a third-party speech recognition system;
- modifications to the Wightman and Ostendorf prosodic analysis for robust behavior on spontaneous speech; and
- a prosody-wise parser and grammar for handling the incremental nature of spontaneous speech in general settings.

Conceptual contributions of these models include the following:

- modern speech recognition components can be used in a loosely-coupled fashion (as a black-box) for robustly interpreting utterances in a dialogue with a human;
- accurate spontaneous utterance segmentation can be facilitated by statistics, prosodic cues, and robust parsing techniques.

I will measure the ability of the overall spoken language understanding system to perform on three levels:

1. Word recognition accuracy;
2. Parser accuracy;
3. End-to-end performance of the TRAINS-95 system.

For a particular utterance, word recognition accuracy is measured by comparing a human transcription with the output of the recognizer (or the post-processor). This metric will indicate how well the post-processor aids the SR component's performance. Parser accuracy is measured by comparing a human interpretation of an utterance with the output of the robust parser. This second metric will measure how well the robust-parser aids in successful interpretation. Third, end-to-end performance is measured in terms of how many turns (on average) a user requires to accomplish a fixed task using the entire TRAINS-95 system. This benchmark will be useful for measuring how well the new components contribute to the overall problem of spoken language understanding.

1.5 Proposal Organization

The second chapter of this thesis proposal discusses background issues and related work regarding the correction of speech recognition errors and approaches to dealing with the incremental structure of spontaneous speech. The third chapter presents work that I have completed toward the proposed goals. The fourth and final chapter lays out the work that must be completed in order to establish the claims posed here.

Chapter 2

Background and Related Work

In order to address the two specific thesis problems with the state of the art in spoken language understanding, we need to examine existing approaches for reducing speech recognition errors and for handling incremental phrase structure.

Since we concur with others in the field that interpretation can be done in stages, we have a need for intermediate representations of language. On every level, from the acoustic signal to the final meaning, the needed representations motivate the understanding system's architecture and analytical ability. (These models may also be useful for their predictive ability and for generation.) Research in computational linguistics and signal processing has produced useful models of human language on many levels. The points in the continuum from acoustic signal to meaning representation where researchers have chosen to delineate a stage of interpretation are surprisingly well agreed upon. At the lowest levels, models based on information theory and stochastic processes prevail, and at the higher levels, models based on linguistic theory predominate. However, there is some overlap.

This chapter examines the stages of interpretation but only in so far as they relate to our two problems: dealing with recognition errors and incremental phrase structure. In the first section, we summarize the ideas that historically made lasting impressions on SR technology and that lead to the main issues of this thesis. At the ground floor of interpretation, modern speech recognition consists of three components: acoustic models, language models, and search algorithms to employ those models. We explore the most successful acoustic models in the second section. The third section examines the issue of language modeling. What kind(s) of language model(s) should be used? This depends on two competing issues: language model explanatory power and permissiveness versus search algorithm complexity. With a fixed amount of CPU cycles, the tradeoff must be made. The fourth section investigates the issue of search as it pertains to the SR/NLU interface. In the fifth section, we discuss strategies for robust parsing. Finally, in the sixth section, we examine the use of prosody features for combating syntactic ambiguity in the parsing process.

2.1 A Brief History of Speech Recognition to the Present

In the past three decades, speech recognition (SR) has evolved from a vision to real implementations with clearly understood theories. The first significant milestones were reached as part of the ARPA Speech Understanding Research Project of 1970-1976. The outcome demonstrated

the validity of claims that simple language models were sufficient to make accurate recognition possible [Klatt, 1977] for well-constrained tasks.

The precursor to most modern SR systems, HARPY was built at CMU in the mid 1970s as part of that first ARPA SR initiative [Lowerre and Reddy, 1986]. It bested the other systems in the competition by meeting or exceeding all of the ARPA goals. Klatt [1977] observes that the primary reason for its success was its simple but appropriately constraining language model. HARPY used a very small grammar for representing requests, and it used finite-state specifications for each word in terms of its phonemes. A language compiler first expanded the grammar into a network of words and then converted that network to a network of possible phoneme sequences. The language compiler for HARPY is depicted in Figure 2.1. The idea of a network of legal phonemes carries

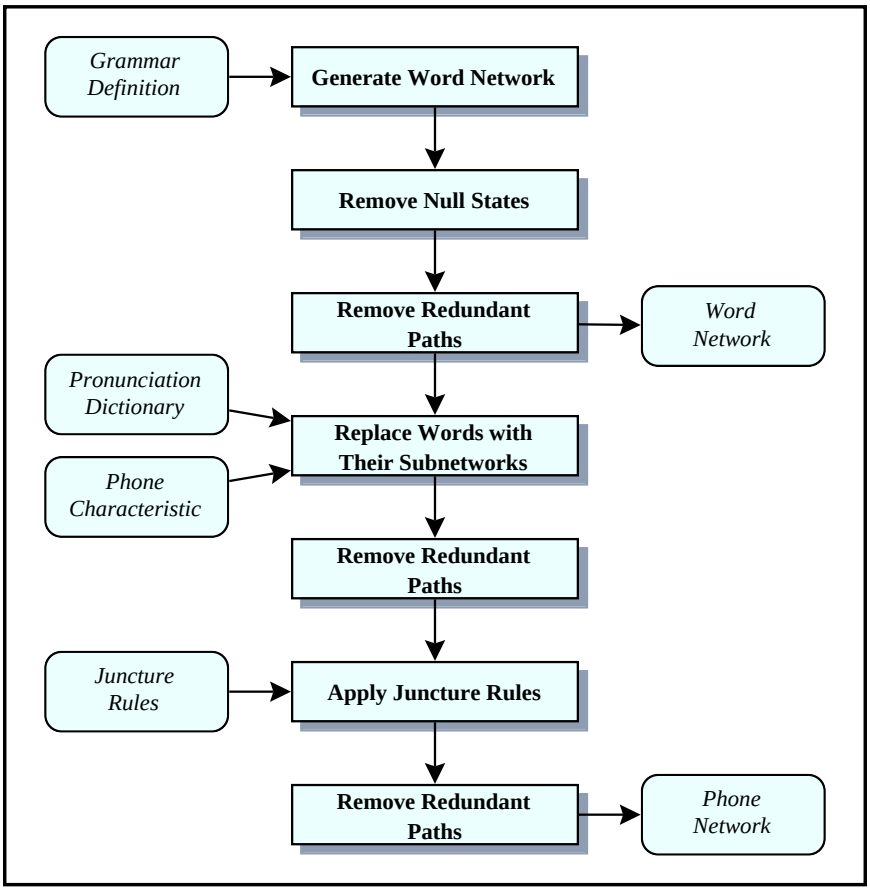


Figure 2.1: *The Language Compiler for HARPY* [Lowerre and Reddy, 1986].

over into subsequent work. Given an input, HARPY performed a search through the network by making probabilistic judgments about acoustic matches between phoneme nodes in the network and segments of the acoustic signal and by pruning low probability paths. This idea is also a clear precursor for subsequent search algorithms

Speech Recognition research at IBM during the 1970s and 80s made landmark contributions in acoustic and language modeling for the purpose of increasing word recognition accuracy even in difficult tasks [Jelinek, 1985]. Statistical techniques were motivated by informa-

tion and communication theory and implemented in the Raleigh and Tangora systems. Hidden Markov Models (HMMs) were used for the first time for acoustic models, and n-gram language models were used to constrain the recognition search [Bahl *et al.*, 1983; Katz, 1987; Jelinek, 1990]. HMMs have since been adopted by many modern SR systems, and n-gram models have been adopted in most if not all modern systems. Since that time, whether to specify a language using a hard-and-fast deterministic model or a permissive probabilistic model has been an important design issue. For tasks requiring a specific command language, deterministic models are best (*e.g.*, see [Huang *et al.*, 1995]). Given a word sequence, a deterministic model will either accept or reject the sequence by definition; there is no room for ill-formed word-sequences. For general-purpose tasks such as dictation and dialogue, the probabilistic approach has served best. A properly trained probabilistic model assigns every sequence of words a non-zero (possibly very small) probability. Also, probabilistic models can be used to easily rank several analyses.

The Sphinx system was built during the late 1980s at CMU [Lee, 1989; Lee *et al.*, 1990; Lee, 1990]; it perfected the art of using HMMs to recognize sentences drawn from a medium-sized vocabulary of English words. It innovatively synthesized and improved on HMM techniques for acoustic modeling in order to reach high performance recognition of speaker-independent, continuous speech. Its architecture set a common standard for others to follow. Figure 2.2 depicts the system components (square boxes), knowledge sources (rounded boxes), and their relationships (on-line relationships are depicted by solid arrows, and off-line relationships are depicted by dotted lines). The Sphinx-II system was built upon this foundation [Huang *et al.*, 1992; Huang *et al.*, 1993b;

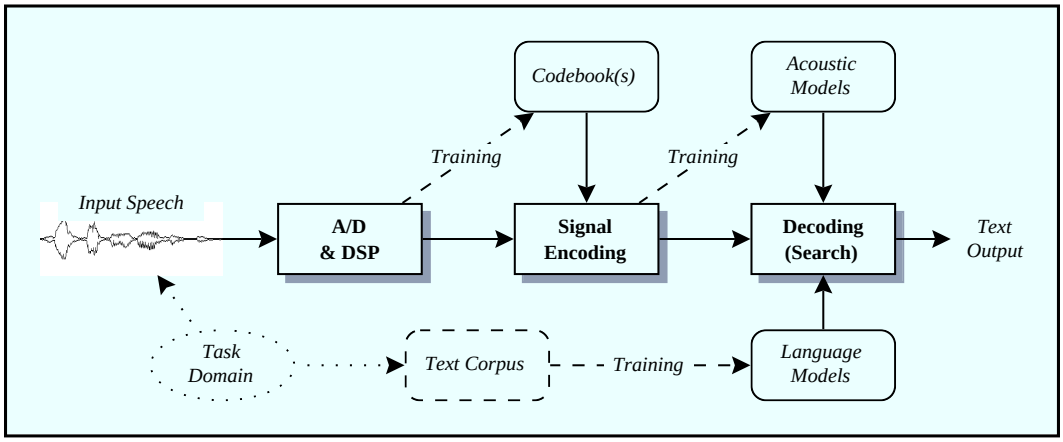


Figure 2.2: *The Architecture for Sphinx (adapted from* [Rudnicky *et al.*, 1994]).

Huang *et al.*, 1993a]. Its innovations include better acoustic models and the ability to share data between complex models (called "tying" in the literature) so that training data is better utilized. Systems using Sphinx-II consistently outperformed competitors in the ATIS evaluations of the early 1990s [ARPA, 1995].

2.2 Issues in Acoustic Modeling

Each facet of the speech recognition problem has been approached from several directions. Beginning with the HARPY phone network framework and phone matching algorithm, a successful

extension involved generalizing to flexible word and phone templates. The word and phone templates were compared against the incoming speech via a dynamic programming [Bellman, 1957] algorithm called by some "Dynamic Time Warping" (DTW) (see for example [Sakoe and Chiba, 1978]). Subsequently, innovations in stochastic modeling provided a unified framework for performing probabilistic phoneme and word matches via a single well-defined search. These innovations involve spectral coding techniques and Hidden Markov Models [Rabiner and Juang, 1993]. Other acoustic models have been constructed using artificial neural networks [Waibel *et al.*, 1989]. In most SR systems, the relationships of the acoustic model to the search algorithm and language models are essentially identical to those of an HMM. This section will briefly examine the relevant signal processing ideas and the use of HMMs for modeling the acoustic and durational properties of speech; however, many of the ideas can be abstracted for other acoustic models.

2.2.1 Discretizing the Speech Signal

At the lowest level, we are interested in accurately modeling the acoustic speech signal [Rabiner, 1989; Huang *et al.*, 1993b]. The goal is to discretize the speech signal in order to view speech as a time series (sequence of symbols). This stage involves sampling the signal via an analog to digital converter, dividing it into overlapping windows [1], and characterizing each window of sound in terms of a vector of spectral parameters. The most common method for determining these parameters is Linear Predictive Coding (LPC) [Rabiner, 1989]. However, many sets of parameters are possible. For example, parameter types include linear prediction filter coefficients, linear prediction reflection coefficients, LPC Cepstral coefficients, mel-frequency Cepstral coefficients, and mel-filter bank channel outputs [Young and Woodland, 1993].

Describing the speech signal in terms of a sequence of spectral vectors allows for further discretization using vector quantization [Gray, 1984]. This process takes a vector of spectral parameters and finds the nearest neighbor in a table (code-book) of N prototypical vectors, statically pre-computed in a training phase. Thus, the final result characterizes the given speech signal as a sequence of vectors from a small set of size N. It is also possible to characterize each window of the speech signal using several vectors, each consisting of a different class of parameters. In this case, each vector for the given window is vector-quantized using a corresponding, distinct code-book [Lee, 1989; Lee *et al.*, 1990].

2.2.2 Hidden Markov Models

Recent speech recognition research has been dominated by the successful application of stochastic modeling techniques. These techniques consider the speech signal as a time-series that can be modeled by a stochastic process. The most commonly used stochastic process model for speech is a Hidden Markov Model (*c.f.* [Rabiner and Juang, 1986; Poritz, 1988; Rabiner, 1989; Lee, 1989; Lee *et al.*, 1990; Huang *et al.*, 1993b]). We will examine the underlying assumptions and, later, the relevant search algorithms for employing them.

[1] Window width is typically on the order of ten milliseconds.

Definition

First of all, let $(\Omega, \mathcal{F}, \mathcal{P})$ be a *probability space*, where

- Ω is the sample space
- $\mathcal{F}$ is the family of events
- $\mathcal{P}$ is the associated probability measure [Rozanov, 1977].

For a fixed t, let $X(t)$ be a random variable (a function from Ω to some range set $\mathcal{Q}$). Then, when t is drawn from an index set T, the sequence of variables $\{X(t)\}$ is a *stochastic process* on the given probability space [Taylor and Karlin, 1994]. A stochastic process $\{X(t)\}$ is said to be a *Markov process* if and only if it possesses the *Markov Property*, namely that the probability of $X(t_1)$ is conditionally independent of $X(t)$ given $X(t_0)$, for any $t < t_0$. In other words,

$$(\forall t < t_0)\ \mathrm{P}[X(t_1) \mid X(t_0), X(t)] = \mathrm{P}[X(t_1) \mid X(t_0)] \quad . \tag{2.1}$$

For a Markov process, the common range set $\mathcal{Q}$ for each random variable in the process is called the *state space*.

Now consider a subclass of the Markov processes with the following characteristics:

- The number, N, of states in $\mathcal{Q}$ is finite.
- The index set T is discrete. *e.g.*, $T = \mathbf{Z}^+$, the set of all positive integers.
- The transition probabilities $a_{m,n}^{k,k+1} = \mathrm{P}[X(k+1) = q_n \mid X(k) = q_m] = a_{m,n}$ are stationary ($q_m, q_n \in \mathcal{Q}$). *i.e.*, they do not depend on the value of $k \in T$. These transition probabilities form a stochastic matrix: $\underline{\underline{\mathbf{A}}} = \{a_{i,j}\}$.
- The distribution for $X(1)$ (the "initial distribution") is specified by the probability vector $\underline{\pi}$.

Such a Markov process is called a *finite Markov chain (FMC) in discrete time with stationary transition probabilities* [Luenberger, 1979; Taylor and Karlin, 1994]. A small example of such a FMC is shown in Figure 2.3.

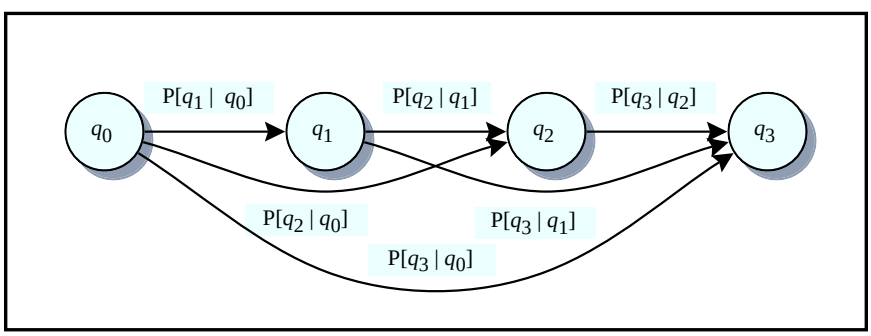


Figure 2.3: *A Finite Markov Chain in Discrete Time with Stationary Transition Probabilities.*

As a final refinement, consider the following conditions:

- For every state $q_i \in \mathcal{Q}$, there is a random variable Y_i with distribution b_i.
- Each Y_i takes on one of M values from the *output alphabet* $\mathcal{W}$, which contains either scalars or vectors. This alphabet can be discrete or continuous. We will assume the discrete case, where b_i is a probability vector $\underline{\mathbf{b}}_i = (\mathrm{P}[Y_i = v_{i,1} \mid q_i], \mathrm{P}[Y_i = v_{i,2} \mid q_i], \ldots, \mathrm{P}[Y_i = v_{i,M} \mid q_i])$. Also, for conciseness, we will often denote the set $\{\underline{\mathbf{b}}_i \mid q_i \in Q\}$ as the $N \times M$ matrix $\underline{\underline{\mathbf{B}}}$, where the i-th row corresponds to $\underline{\mathbf{b}}_i$.

A FMC in discrete time with stationary transition probabilities and such per-state random variables is called a discrete *Hidden Markov Model*. The variables Y_i model the output probability associated with each state in the model. Because each state of an HMM generates an output according to the output probability distribution, HMMs are also called probabilistic functions of Markov processes in the earlier literature (*c.f.* [Baum, 1972; Levinson *et al.*, 1983]). For a HMM, only the output is visible; the actual state sequence is "hidden." An example of a small HMM (based on the Markov chain from Figure 2.3) is shown in Figure 2.4.

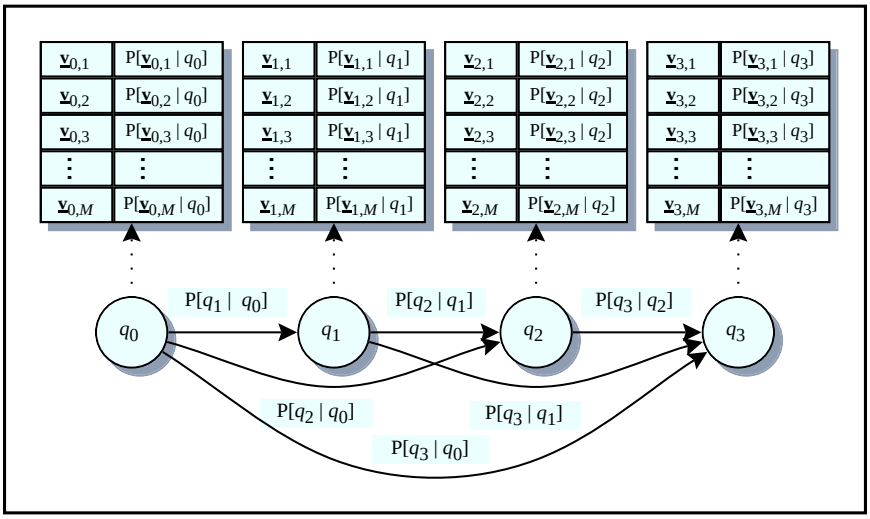


Figure 2.4: *A Discrete Hidden Markov Model with Finite, Vector-Valued Output Distributions.*

Given a state space $\mathcal{Q}$ and an output alphabet $\mathcal{W}$, a HMM is fully-specified by a three-part model $\lambda = (\underline{\underline{\mathbf{A}}}, \underline{\underline{\mathbf{B}}}, \underline{\pi})$ that describes the initial configuration of the model, how transitions in the model occur, and how symbols are generated. Having presented this definition, it is worth mentioning that other definitions exist. For example [Lee, 1990] uses an alternative in which the output variables are associated with the transition arcs between states rather than with the states themselves. [2] Furthermore, in the above definition, we focused on a discrete output probability distribution; however, continuous densities are also possible. For continuous HMMs, densities are usually constructed as Gaussian mixtures for purposes of straightforward computation(*e.g.*, [Young and Woodland, 1993]). A compromise between the two approaches, called semi-continuous HMMs (SCHMMs), has been defined by Huang [Huang and Jack, 1989].

[2] These two models are the probabilistic counterparts to Moore and Mealy machines, respectively (*c.f.* [Hopcroft and Ullman, 1979]).

It should be noted that the standard HMM models make some false assumptions for expediency's sake. For example, as noted above, the Markov principle assumes that speech features occurring at one time are uncorrelated with and independent of other recently occuring features that occurred even ten milliseconds earlier in the previous frame. Several research labs have developed hybrid artificial neural network/hidden Markov models for acoustic models that improve upon traditional HMMs by modeling correlations among simultaneously occuring speech features and between current and recent features.

A HMM can be viewed from both a generation and an analysis perspective. Because we are interested in using HMMs for analyzing sequences of symbols, the generation of the speech signal is modeled as a sequence of symbols generated by a HMM [Rabiner, 1989]. An observation sequence $\underline{\mathbf{O}} = (O_1, O_2, \ldots, O_T)$ is generated as follows:

1. Set $t = 1$, and choose an initial state $q_t (= q_1)$, according to the initial state distribution, π.
2. While $t < T$, do the following:
   (a) Choose O_t according to the symbol probability density $\underline{\mathbf{b}}_{q_t}$ for state q_t.
   (b) Choose q_{t+1} according to the state transition probability density $\underline{\mathbf{a}}_{q_t}$ for state q_t.
   (c) Increment t.

For recognition, there are three key problems to solve in order for HMMs to be useful [Rabiner and Juang, 1986]. The first problem is at the heart of the training issue: given an observation sequence $\underline{\mathbf{O}}$, how do we adjust the model parameters λ to maximize $\mathrm{P}[\underline{\mathbf{O}} \mid \lambda]$? A solution would allow us to take observations of known state sequences and tune a model to explain those observations. The solution typically involves an instance of the Expectation Maximization (EM) algorithm called Baum-Welch reestimation [Poritz, 1988]. This algorithm iteratively estimates each parameter of an HMM.

Second, given the model $\lambda = (\underline{\underline{\mathbf{A}}}, \underline{\underline{\mathbf{B}}}, \underline{\pi})$, for an arbitrary observation sequence $\underline{\mathbf{O}} = (O_1, O_2, \ldots, O_T)$, how do we compute the probability of the observation sequence $\mathrm{P}[\underline{\mathbf{O}} \mid \lambda]$? The solution can be used in selecting the best model that matches an arbitrary input sequence. Thus, assuming we have a library of $\|\mathcal{V}\|$ distinct HMMs, [3] the recognition problem consists of finding λ_m such that

$$m = \arg \max_{1 \leq m \leq \|\mathcal{V}\|} \mathrm{P}[\underline{\mathbf{O}} \mid \lambda_m] \quad . \tag{2.2}$$

The solution to this problem is computed by the Forward procedure [Baum, 1972]. [4]

The third problem is the *decoding* problem: given an observation sequence $\underline{\mathbf{O}}$, how do we identify a state sequence $\underline{\mathbf{q}} = (q_1, q_2, \ldots, q_T)$ that is optimal in some meaningful sense? In other words, how do we discover the "hidden" state sequence that accounts for a given observation sequence. This involves identifying $\underline{\mathbf{q}} = (q_1, q_2, \ldots, q_T)$ such that

$$\underline{\mathbf{q}} = \arg \max_{\underline{\mathbf{q}} \in \mathcal{Q}^T} \mathrm{P}[\underline{\mathbf{O}} \mid \underline{\mathbf{q}}] \quad . \tag{2.3}$$

The solution to this problem is computed by the Viterbi algorithm, which we will examine in greater detail in the section on search.

[3] For a vocabulary $\mathcal{V}$, we will use $\|\mathcal{V}\|$ to denote its size.

[4] Baum-Welch reestimation employs the Forward procedure and its analogue, the Backward procedure. In fact, Baum-Welch reestimation is referred to by some as the Forward-Backward algorithm.

2.3 Language Modeling

Acoustic and durational models alone do not provide all of the constraints necessary for successful recognition for large vocabulary tasks. Experiments with large vocabulary tasks indicate that word recognition without the aid of a language model yields no better than roughly 70% word accuracy rates

Humans use many non-acoustic, higher-level sources of information to aid in the recognition process. Similarly, we will argue in this section that accurate language models are required for more accurate (*i.e.*, fewer errors) automatic speech recognition. However, there is an important trade-off here between language model complexity and the efficiency of recognition (or "decoding") algorithms that use the models. State of the art systems trade the benefits of a complex language model for the efficiency of simpler algorithms. The best compromise currently consists of HMMs and n-gram back-off language models (defined below) using a beam search for decoding. Systems using these models alone are capable of recognizing large vocabulary, speaker-independent, continuous speech with reasonable accuracy and in real-time. Needless to say, such models are insufficient for the full understanding task where some kind of syntactic and semantic analyses are required. We address this issue in detail in the next section.

This section will proceed as follows: first, we discuss measures of language model complexity; second, we discuss collocation-based models of language; and third, we examine phrase structure models.

2.3.1 Complexity of Language Models

We would like to have an objective measure of language model quality for evaluating those models that can be used in a spoken language understanding system. The most common route is to define this measure in terms of information theory [Shannon, 1948], which is concerned with sources of information [Jelinek, 1990]. For this purpose, we'll adopt Jelinek's orthographic definition of a word as a *word form* defined by its spelling. Two differently spelled inflections or derivations of the same stem are considered different words. Homographs with different parts of speech or meanings nonetheless constitute the same word.

In the spirit of our discussion regarding dialogue between a speaker and a listener, a speaker generates words chosen from some finite vocabulary $\mathcal{V}$ that is also known to the listener. For the sake of discussion, we need to assume that there is a statistical law that governs the speaker's choice of output words. When generating a word, the speaker provides information by removing the listener's uncertainty about the identity of the word. Thus, from an information theoretic point of view, the speaker provides more information if its outputs are more uncertain. [5]

Given a vocabulary $\mathcal{V}$, uncertainty (and therefore information) is maximal if each of the possible words is spoken with equal probability and independently of previously chosen words. Shannon's approach measures the amount of information provided by a speaker as a number I:

$$I = \log_2 ||\mathcal{V}|| \tag{2.4}$$

[5] This theory is usually couched in terms of "sources" and "symbols," but it is adapted here in terms of "speakers" and "words."

The logarithmic measure in equation 2.4 makes sense if we consider that speaking two words from $\mathcal{V}$ is equivalent to speaking one word from a corresponding uniform source $\mathcal{V}' = \mathcal{V} \times \mathcal{V}$ with size $\|\mathcal{V}'\| = \|\mathcal{V}\|^2$. Intuitively, this act should provide twice as much information. In fact, this is so:

$$I' = \log_2 \|\mathcal{V}'\| = \log_2 \|\mathcal{V}\|^2 = 2 \log_2 \|\mathcal{V}\| = 2I \quad . \tag{2.5}$$

Because the logarithm is base two, the information is measured in bits.

Entropy

Let w denote a word that is spoken with *a priori* probability $\mathrm{P}[w]$. For speakers that choose their words from $\mathcal{V}$ independently of one another, we can sum the weighted information per word over all words in $\mathcal{V}$ to derive an expression for the total information provided by a speaker. This quantity is the *entropy* H of the speaker:

$$H = -\sum_{w \in \mathcal{V}} \mathrm{P}[w] \cdot \log_2 \mathrm{P}[w] \quad . \tag{2.6}$$

Note that if $\mathrm{P}[w] = 1/\|\mathcal{V}\|$ for all words $w \in \mathcal{V}$, the source is uniform and equation 2.6 simply reduces to equation 2.4. Furthermore, the *Fundamental Coding Theorem of information theory* asserts that on average, representing a word generated by a speaker of entropy H requires at least H bits. Also, from equations 2.4 and 2.6, a speaker of entropy H provides as much information as one that chooses words equiprobably from a vocabulary of size 2^H.

We extend equation 2.6 for sequences $\underline{\mathbf{w}}_{1,n}$ of words $(w_1, w_2, \ldots, w_n)$, each from $\mathcal{V}$, so that, in general, we have the *per-word entropy*

$$H = -\lim_{n \to +\infty} \left[\frac{1}{n} \cdot \sum_{\underline{\mathbf{w}}_{1,n}} \mathrm{P}[\underline{\mathbf{w}}_{1,n}] \cdot \log_2 \mathrm{P}[\underline{\mathbf{w}}_{1,n}] \right] \quad . \tag{2.7}$$

One special case of this is when the speaker's choice of words is independent, *i.e.*, when

$$\mathrm{P}[\underline{\mathbf{w}}_{1,n}] = \mathrm{P}[w_1] \cdot \mathrm{P}[w_2] \cdot \cdots \cdot \mathrm{P}[w_n] \quad . \tag{2.8}$$

Then equation 2.7 reduces to equation 2.6. Another special case occurs when the statistical law underlying the speaker's choice of words can be described by a stochastic process (defined above in 2.2.2) that is *ergodic*. Simply put, when the source is "well behaved" so that sufficiently long sequences of words are typical of it and can be used to characterize it, then equation 2.7 reduces to

$$H = -\lim_{n \to +\infty} \left[\frac{1}{n} \cdot \log_2 \mathrm{P}[\underline{\mathbf{w}}_{1,n}] \right] \quad . \tag{2.9}$$

Thus, assuming that a speaker is ergodic (to use the terminology somewhat loosely), entropy can be directly estimated from long sequences of spoken words using equation 2.9 with a sufficiently large value of n.

We can now apply these concepts to language models. Given a body (or *corpus*) of text, we can measure the amount of information per word of the system that generated the corpus:

$$H = -\frac{1}{n} \cdot \log_2 \mathrm{P}[\underline{\mathbf{w}}_{1,n}] \quad , \tag{2.10}$$

where n is the size of the corpus. Here, H is the average number of bits required to specify a word. For our purposes, the most significant consequence of this is that equation 2.10 is also an estimate of the difficulty of recognizing speech from a speaker using the same "mechanism" that gave rise to the corpus. The reason is that a speech understanding system must extract exactly H bits of information from the acoustic data in order to accurately determine a spoken word. Expressed another way: a larger entropy implies a smaller ability to predict the next word and, hence, weaker guidance for the SR system.

Clearly, in order to determine H in equation 2.10, we require the actual probabilities $\mathrm{P}[\underline{\mathbf{w}}_{1,n}]$ of word sequences in the language. Jelinek observes that these probabilities are unknowable in practice. At best, we can only construct a language model from which we can compute estimates $\hat{\mathrm{P}}[\underline{\mathbf{w}}_{1,n}]$. Thus, from the point of view of the hearer, an estimate of the difficulty of recognizing speech generated by a speaker using the same mechanism that generated the given corpus is measured by the *log-probability* of the corpus:

$$LP = -\frac{1}{n} \cdot \log_2 \hat{\mathrm{P}}[\underline{\mathbf{w}}_{1,n}] \quad . \tag{2.11}$$

Assuming ergodic behavior of the speaker as before, it is possible to show that $LP \geq H$ [Jelinek, 1990]. Thus, in practice it is possible to compute an upper-bound on the entropy of the corpus.

Perplexity

Working in terms of entropy or log-probability measured in information bits may be somewhat unintuitive. As an alternative, we measure the difficulty of recognizing a spoken utterance (using a particular language model) by the average size of the word set (a subset of $\mathcal{V}$) from which the recognizer must choose at any particular decision point during the analysis [Brown *et al.*, 1992a]. For a sequence $\underline{\mathbf{w}}_{1,n} = (w_1, w_2, \ldots, w_n)$ of words, $\hat{\mathrm{P}}[\underline{\mathbf{w}}_{1,n}]$ is an estimate provided by the given language model of the probability that the speaker will generate them. Then the *perplexity* PP is defined as:

$$PP = 2^{LP} = \hat{\mathrm{P}}[\underline{\mathbf{w}}_{1,n}]^{-\frac{1}{n}} \quad . \tag{2.12}$$

Thus, to a first order approximation (disregarding the acoustic distances between particular words), the recognition task can be considered as difficult as the recognition of a language with PP equally likely words. Hence, perplexity is a measure of the average "branching" of a text or utterance relative to a language model. For speech recognition, it is the number of options an SR system must consider at any point in its search for the best word transcription of its input.

Thus, when confronted with recognizing an arbitrary utterance $\underline{\mathbf{w}}_{1,n}$, we can simplify the task by decreasing LP and hence PP. We must find the smallest possible valid estimate of $\hat{\mathrm{P}}[\underline{\mathbf{w}}_{1,n}]$ by choosing a precise language model that yields a tight upper-bound. This motivates the need for good language models and provides us with an easily computable metric for evaluating them.

Cross-Entropy

Assume for the sake of argument that there is a single "correct" model for a particular speaker of English. When attempting to recognize real (spoken) English from that speaker, we do not possess

the correct model. However, we would like a measure of how useful our available models are. One such measure is the *cross entropy* [Brown *et al.*, 1992a]. With respect to a particular model $\mathcal{M}$ that provides probability estimates $\hat{\mathrm{P}}_{\mathcal{M}}[\underline{\mathbf{w}}_{1,n}]$, the per-word cross entropy $H_{\mathcal{M}}$ of a speaker is defined in a manner analogous to per-word entropy, where the second factor in equation 2.7 is replaced by the estimate $\hat{\mathrm{P}}_{\mathcal{M}}[\underline{\mathbf{w}}_{1,n}]$ provided by the available model. As before, if the speaker is ergodic, then the expression simplifies as in equation 2.9. The cross entropy with respect to the correct language model for the speaker is a lower bound on the cross entropy for any other language model. Hence, one can compute the cross entropies of competing models and infer that the model with the smallest cross entropy is the best of the set.

2.3.2 Collocation-based Models

With the information theoretic notion of entropy in mind, it is possible to construct simple language models having reasonably small perplexities. For a well-behaved speaker, words that are frequently spoken together in one setting will very likely be spoken together in another setting with that same frequency. This observation motivated the creation of language models based on word collocation.

The goal of the statistical approach to speech recognition is to determine the most likely word sequence (*i.e.*, transcription) $\underline{\hat{\mathbf{w}}}_{1,n} = (w_1, w_2, \ldots, w_n)$ given an acoustic signal $\underline{\mathbf{s}}$ containing speech.

Now, given that $\underline{\mathbf{s}}$ was observed, a recognizer using this approach would yield the desired result, namely the word sequence $\underline{\hat{\mathbf{w}}}_{1,n}$ such that

$$\underline{\hat{\mathbf{w}}}_{1,n} = \underset{\underline{\mathbf{w}}_{1,n}}{\operatorname{argmax}}\, \mathrm{P}[\underline{\mathbf{w}}_{1,n} \mid \underline{\mathbf{s}}] \quad . \tag{2.13}$$

This requires the construction of a model of the prior probability density $\mathrm{P}[\underline{\mathbf{w}}_{1,n}]$ of any given word sequence [Jelinek, 1990]. This probability is interpreted as the probability that a speaker will say the string in question. Using Bayes' rule with an appropriate acoustic conditional probability density $\mathrm{P}[\underline{\mathbf{s}} \mid \underline{\mathbf{w}}_{1,n}]$ (approximated using labeled speech corpora), we can determine the posterior probabilities $\mathrm{P}[\underline{\mathbf{w}}_{1,n} \mid \underline{\mathbf{s}}]$ needed for equation 2.13:

$$\mathrm{P}[\underline{\mathbf{w}}_{1,n} \mid \underline{\mathbf{s}}] = \frac{\mathrm{P}[\underline{\mathbf{w}}_{1,n}] \cdot \mathrm{P}[\underline{\mathbf{s}} \mid \underline{\mathbf{w}}_{1,n}]}{\mathrm{P}[\underline{\mathbf{s}}]} \quad . \tag{2.14}$$

Since equation 2.13 is maximizing with respect to a fixed (given) $\underline{\mathbf{s}}$, by equations 2.13 and 2.14 the recognizer actually needs only to find the word sequence $\underline{\hat{\mathbf{w}}}_{1,n}$ such that:

$$\underline{\hat{\mathbf{w}}}_{1,n} = \underset{\underline{\mathbf{w}}_{1,n}}{\operatorname{argmax}}\, \mathrm{P}[\underline{\mathbf{w}}_{1,n}] \cdot \mathrm{P}[\underline{\mathbf{s}} \mid \underline{\mathbf{w}}_{1,n}] \quad . \tag{2.15}$$

This divides the problem nicely into two parts: one part deals only with word-level language, and the other part deals only with the acoustical aspects of speech.

n-grams

Unfortunately, a complete model $\mathrm{P}[\underline{\mathbf{w}}_{1,m}]$ for all word sequences is impossible to acquire. However, a simplifying assumption can make the construction of such a model feasible: assume

that the probability of an instance of any word depends only on the previous $n-1$ words. The model $\mathrm{P}[\underline{\mathbf{w}}_{1,m}]$ is reduced by this assumption to an n*-gram model* [Jelinek, 1990]. Typically, n is 2 (*bigrams*) or 3 (*trigrams*). An n-gram model is simply a $\|\mathcal{V}\|^n$ size table (where $\|\mathcal{V}\|$ is the size of the vocabulary from which the words of the sequence $\underline{\mathbf{w}}_{1,n}$ are drawn), in which the entry of the cell with coordinates $(i_1, i_2, \ldots, i_n)$ is the probability estimate $\hat{\mathrm{P}}[w_{i_n} \mid w_{i_1}, w_{i_2}, \ldots, w_{i_{n-1}}]$.

Such a model can be easily constructed by scanning large text corpora and computing statistics (probability estimates). One method involves calculating the relative frequency of the occurrence of the n-th word in the context of its $n-1$ predecessors:

$$\hat{\mathrm{P}}[w_{i_n} \mid w_{i_1}, w_{i_2}, \ldots, w_{i_{n-1}}] = \frac{\|\{(w_{i_1}, w_{i_2}, \ldots, w_{i_n}) \in \mathcal{C}\}\|}{\|\{(w_{i_1}, w_{i_2}, \ldots, w_{i_n-1}) \in \mathcal{C}\}\|} \quad . \tag{2.16}$$

As noted, n is typically very small, since an n-gram model requires $\|\mathcal{V}\|^n$ probabilities to be estimated. Models constructed from available text corpora reveal that n-gram models having reasonably low perplexity can be constructed.

Back-off n-gram Models

Sufficient data for building an accurate table of n-gram statistics is not always available. In fact, for many n-grams, our estimate based on relative frequency in a corpus may be zero. Predicting zero probability may fatally kill viable options in a search. Hence the ability to "back off" to available $(n-1)$-grams or beyond is desirable. This is typically accomplished by smoothing the straightforward n-gram model using the lower order estimates from the same corpus [Katz, 1987].

2.3.3 Phrase Structure Models

Other language models originate with Chomsky's theory of linguistic competence [Chomsky, 1957]. For example, phrase structure models are more appropriate than collocation-based models for modeling natural linguistic phenomena (*e.g.*, long-distance dependencies such as agreement constraints and wh-movement). Numerous models of phrase-structure, or grammar formalisms, exist (*e.g.*, [Jensen *et al.*, 1993; Allen, 1994]). We will only capture their spirit here without delving into the details.

A grammar implicitly defines a set of acceptable word sequences over a given vocabulary; this set comprises a formal language [Hopcroft and Ullman, 1979]. Two primary language classes have been used in spoken language systems: regular languages and context-free languages. A language in the first class can be described using a regular grammar (RG), often called a Finite-State Grammar (FSG) in the speech literature. An equivalent formalism used in past speech systems is Augmented Transition Network (ATN) grammar [Thorne *et al.*, 1968; Woods, 1970]. A context-free language can be described using a Context-Free Grammar (CFG). Equivalently, ATN grammars have been used for both SR search and NL parsing. Several formalisms for compactly describing CF languages have been based on the principle of unification (*c.f.* [Shieber, 1985]). Only unification-based grammars having a finite number of values for each feature are actually context-free.

Regular Grammar

In the first class, regular grammars are analogous to n-gram models. In fact, a bigram model can be called a "probabilistic regular grammar" or a "stochastic" RG. Like n-grams, they can be used to constrain the possible connections between word HMMs. However, the constraints are not as expressive as those provided by n-grams. In terms of conditional probability links, a finite-state grammar sets some links to zero and assumes that the remainder occur with equal probability. Some FSGs (and Transition Network Grammars) encode both syntax and phoneme-level information in one network as HARPY did. Such an arrangement is the non-probabilistic equivalent of HMMs embedded within an n-gram network.

Context Free Grammar

As for the second class, unlike FSGs, Context Free Grammars (and equivalent formalisms) require recursion and the generation of many hypotheses. Thus, efficiency is a potential problem. However, many constructs of natural language are too complex for simple FSGs. One possible compromise is a technique for construction of a FSG from a CFG [Pereira and Wright, 1991].

Formally, a CFG is a four-tuple $(\mathcal{V}, \mathcal{N}, N_1, \mathcal{R})$. The components are defined as follows:

1. $\mathcal{V}$ is a set of terminal symbols $\{w_1, w_2, \ldots, w_{\|\mathcal{V}\|}\}$ (*i.e.*, the vocabulary).

2. $\mathcal{N}$ is a set of non-terminal symbols $\{N_1, N_2, \ldots, N_{\|\mathcal{N}\|}\}$ (for our purposes, the set of lexical categories $\mathcal{L} \subseteq \mathcal{N}$).

3. N_1 is the starting non-terminal symbol.

4. $\mathcal{R}$ is a set of rules $R_{i,j}$ of the form $N_i \rightarrow \zeta_{i,j}$, where $\zeta_{i,j}$ is a string of terminals and non-terminals, i ranges over $\{1, \ldots, \|\mathcal{N}\|\}$ and j ranges over $\{1, \ldots, m_i\}$.

This model has weaknesses that the n-gram model did not; for example, local lexical preferences are not captured well. However, models have been devised that attempt to handle both common linguistic phenomena while providing a framework for quantitative preferences.

Probabilistic Context Free Grammar

One compromise model is *Probabilistic Context-Free Grammar* (PCFG). A PCFG is a generalization of a Context-Free Grammar (CFG) and is alternatively called a *Stochastic Context Free Grammar* (SCFG). Simply put, a PCFG is a CFG with probabilities for rule usage. Formally, a PCFG is a five-tuple $(\mathcal{V}, \mathcal{N}, N_1, \mathcal{R}, \mathcal{P})$ (*c.f.* [Charniak, 1993]) with the first four members defined as for a CFG and the final member defined as follows: $\mathcal{P}$ is a probability density over the set of rules in $\mathcal{R}$ such that all probabilities for the same non-terminal sum to 1. Thus, for each rule $R_{i,j}$ in $\mathcal{R}$, there is an associated probability P$[N_i \rightarrow \zeta_{i,j}]$, which we can think of as the probability of expanding the non-terminal N_i by $\zeta_{i,j}$ rather than $\zeta_{i,k}$ for some k such that $1 \leq k \leq m_i$ but $k \neq j$.

Thus, a PCFG assigns both a structure and a probability $\hat{\mathrm{P}}[\underline{\mathbf{w}}_{1,T}]$ to a string $\underline{\mathbf{w}}_{1,T}$. In other words, the probability density $\mathcal{P}$ induces another probability density over the set of all word sequences:

$$\sum_{\underline{\mathbf{w}}_{1,T} \in \mathcal{V}^T} \mathrm{P}[\underline{\mathbf{w}}_{1,T}] = 1 \quad . \tag{2.17}$$

This is clearly helpful for forming and ranking syntactic analyses in an NLU system, but it is also helpful in so far as it provides an estimate of the likelihood of the occurrence of a string $\underline{\mathbf{w}}_{1,n}$ which we can use to our advantage as shown above. For computing entropies of probabilistic grammars, see [Soule, 1974].

Given a CFG, a simple approach to constructing a corresponding PCFG involves counting the number of times each rule in $\mathcal{R}$ would be used to generate the trees provided in a parsed corpus $\mathcal{C}$, such as the Penn Treebank. These frequencies are used to construct estimates for $\mathcal{P}$. For example, consider a non-terminal N_i. This simple technique estimates the probability of using $R_{i,j}$ to combine $\zeta_{i,j}$ ($j \in \{1, \ldots, m_i\}$) in order to derive N_i by counting the number of times N_i occurs in the set of trees dominating only the nodes listed in $\zeta_{i,j}$ as follows:

$$\hat{\mathrm{P}}[R_{i,j}] = \left\| \{ N_i \in t^{\mathcal{C}} \mid N_i \text{ dominates only } \zeta_{i,j} \} \right\| \quad . \tag{2.18}$$

More recently, researchers have shown that untagged corpora can also be used in conjunction with a CFG to derive a corresponding probability density $\mathcal{P}$ [Kupiec and Maxwell, 1992].

Furthermore, as with HMMs, it is possible to process a corpus of pre-parsed ("tagged") text to induce a PCFG without a previously available CFG "backbone" [Baker, 1979; Lari and Young, 1990]. As with Baum-Welch reestimation, the training algorithm is an instance of EM and uses the Inside-Outside algorithm in a similar fashion to the Forward and Backward procedures. The training algorithm itself is often referred to as the Inside-Outside algorithm (*e.g.*, in [Lari and Young, 1990]).

Furthermore, as for n-gram models, we can make the same case for never assigning a zero probability to any string, because such a string may arise in practice. This approach is advantageous when working with the noisy and error-filled data that arises naturally. Smoothing may also be necessary to overcome problems with sparse training data.

PCFGs have well-understood algorithms, but they make independence assumptions unjustified by natural language. More sophisticated phrase-structure language models involving rule probabilities conditioned on parent rules, for example, have also been trained and applied in parsing [Bahl *et al.*, 1989; Black *et al.*, 1993]. Magerman has recently published work using decision-tree models for making statistical parsing decisions without unnecessary independence assumptions [1994; 1995].

Unification-based Grammar

With respect to the third class, many Unification-based formalisms have been constructed for use in NLU systems. A straightforward unification grammar is a simple extension of a CFG. Each rule has a context-free backbone with typed feature-structures augmenting the non-terminal and terminal symbols.

Recent work at Microsoft Research indicates that it is possible to use a well designed grammar similar to a unification-based formalism and to gather rule probabilities from an untagged text corpus [Richardson, 1994].

2.4 Search and Speech Recognition/Parser Interfaces

Where the SR system should end and the NLU system should start remains undecided, but there appears to be a region in the interpretation continuum that is most appropriate. In perhaps the most common integration the SR system hands off a word sequence transcription to the parser component of NLU. The issue of how best to couple SR and parsing is an important one, since using an appropriate language model to aid the recognition process has the potential to reduce the perplexity of the task and to reduce recognition errors.

There are three essentially orthogonal dimensions to a SR/NLU coupling: *complexity* of the search algorithm for the language model, *incrementality*, and *tightness*. The dimension of complexity ranges from the case where no language model is used to using a full parsing algorithm for a sophisticated phrase-structure model. The incrementality dimension is binary and indicates whether the speech recognizer interacts on-line with higher-level knowledge sources or simply feeds an entire hypothesis to higher-level processing all at once. Finally, Harper and others at Purdue [1994] have classified spoken language systems along the tightness dimension as *tight*, *loose*, and *semi-tight* couplings. [6] These couplings are illustrated in Figure 2.5. They focus on categorizing couplings

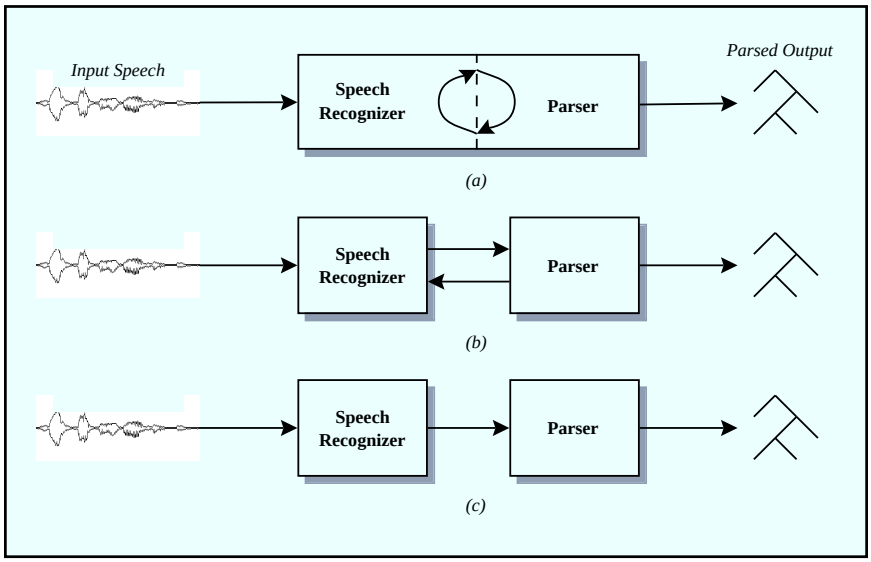


Figure 2.5: *Couplings Categorized by Tightness: (a) Tight; (b) Semi-Tight; (c) Loose.*

between acoustic models and language models, and they criticize tight couplings as "intractable." This criticism is misleading, since we shall see that some of the most effective and relatively inexpensive acoustic/language model couplings are in fact tight. To clarify the issue, however, it

[6] The authors actually say that a system is either "tightly coupled," "loosely coupled," or "semi-coupled," but I have attempted to nominalize the categories.

should be noted that those language models that have been tightly coupled with acoustic models in an effective manner are very simple and must be coupled, in turn, with an additional higher-level language model in order to yield at least a syntactic analysis. Looking along the tightness dimension, Table 2.1 (adapted from [Harper *et al.*, 1994]) summarizes the categories.

	Tight Coupling	Semi-Tight Coupling	Loose Coupling
Motivation	Psycholinguistic evidence	Compromise	Software engineering
Modularity of Knowledge Sources	All KSs integrated in single model	KSs can be removed but not isolated	Each KS in stand-alone module
Inter-module Communication	N/A	Typically two-way	Typically one-way (pipelined)
Scalability	Not easy	Reasonable	Easy
Computation Complexity	Feasible only with simple language models	Feasible for models of moderate complexity	Feasible for large problems and complex language models
Examples	Acoustic-level syntactic parser	LR parser + phone verification; n-grams ($n > 2$)	Constraint-based system; Blackboard model

Table 2.1: *Spectrum of Speech Recognition/Natural Language Understanding Interfaces Along the Tightness Dimension.*

2.4.1 Tight Coupling

Optimally, the coupling between a SR system and a NLU system would be tight in the sense that input would be incrementally processed at all levels in real-time. The reasons are at least three-fold. First, humans seem to perform acoustic and linguistic analysis in real-time and incrementally (for psycholinguistic evidence, see *e.g.*, [Spivey-Knowlton *et al.*, 1994]). Second, higher-level processing modules, such as the parser and the Dialogue Manager (DM) could constrain the SR module using syntactic, semantic, and pragmatic knowledge in order to increase recognition accuracy. Third, as each word is recognized, it would be morphologically decomposed, used to extend a partial syntactic analysis, and composed with a growing semantic analysis. The final result would be a meaning representation built incrementally from the incoming acoustic signal. Thus, if a partial analysis of an utterance is formed *while it is heard*, then higher-level modules can, for example, detect misunderstandings or interrupt when necessary.

As indicated in the table, a tightly coupled system integrates all of the knowledge sources for speech processing into a single interdependent model that cannot easily be untangled. Architecturally, the language and acoustic models are not separable. Unfortunately, for complex language models, the efficiency of a tight coupling is abominable.

HMMs and the Viterbi Algorithm

As an example of a feasible tight coupling, consider the integration of a HMM acoustic model with a simple language model. Typically for continuous speech, a vocabulary of words is selected in advance. One HMM is then trained for each word. Often word-level training simply consists of 1. training an HMM for each phoneme or phoneme-in-context (a phoneme occuring in a specific phonemic context) and 2. forming each word HMM by linking copies of the appropriate phoneme models into a network that allows for alternate legal pronunciations (*c.f.* [Bahl *et al.*, 1983]). For the sake of discussion, these word-level HMMs are then embedded in a larger network, in which each individual HMM is linked to every other by an arc labeled with the appropriate bigram probability (the estimate of how likely the second word will follow the first word in unseen text or speech). If we assume for the sake of illustration that each word can be modeled by an HMM having the topology shown in Figure 2.4, then Figure 2.6 depicts two stages of the composite HMM. In reality,

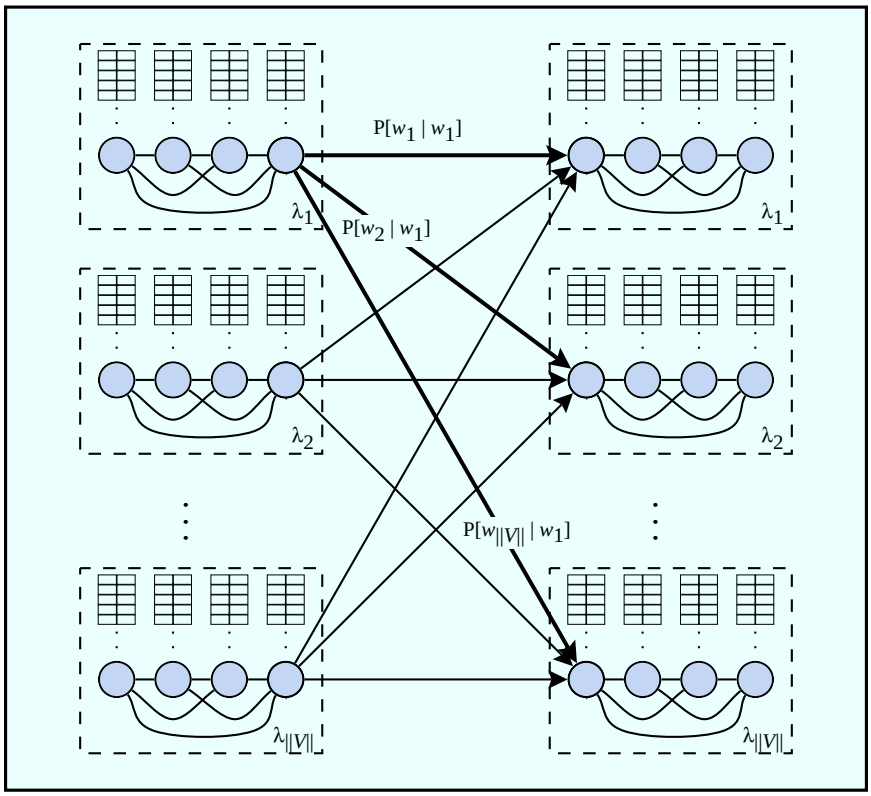


Figure 2.6: *Composite HMM Acoustic Model and Bigram Language Model.*

for a large vocabulary it is impractical to explicitly construct this composite HMM; rather, during the search, transitions from one word HMM to the next are hypothesized on-the-fly using a bigram model. We will return to this point once we have examined possible search algorithms more closely.

Given an acoustic utterance as a sequence of acoustic parameters (or parameter vectors), the straight-forward Viterbi algorithm [Viterbi, 1967; G. E. Forney, 1973] searches for the most likely state sequence in the integrated HMM/bigram model (our composite HMM). Hence, in a typical HMM-based speech recognizer, this algorithm, or some variation thereof, is the heart of the de-

coder [Rabiner and Juang, 1986; Poritz, 1988; Rabiner, 1989]. It "decodes" the input into the most likely state-sequence and thereby identifies a unique word sequence. Thus, the search effectively segments the utterance and performs homophone disambiguation.

To conceptualize the operation of the Viterbi algorithm, consider a two-dimensional lattice indexed by discrete time on the horizontal axis and unique HMM states on the vertical axis. Thus, each node in the lattice is a unique HMM state at a single time-step; we will call it a state/time-step node. The Viterbi algorithm is a dynamic programming algorithm on partial paths through this state/time-step lattice. Importantly, the dynamic programming is justified by the Markov Property of the HMM.

To illustrate, we choose a vocabulary of size $||\mathcal{V}||$ and suppose for simplicity's sake that each word in the vocabulary can be adequately modeled by our small HMM from Figure 2.4. Next, we construct the composite HMM using these word models and a bigram model. We then construct the corresponding state/time-step lattice as described above and eliminate all nodes that have no possible input arcs. Figure 2.7 depicts a subset of this state/time-step lattice. In the figure, the dark arcs indicate inter-word arcs and would be labeled with bigram probabilities.

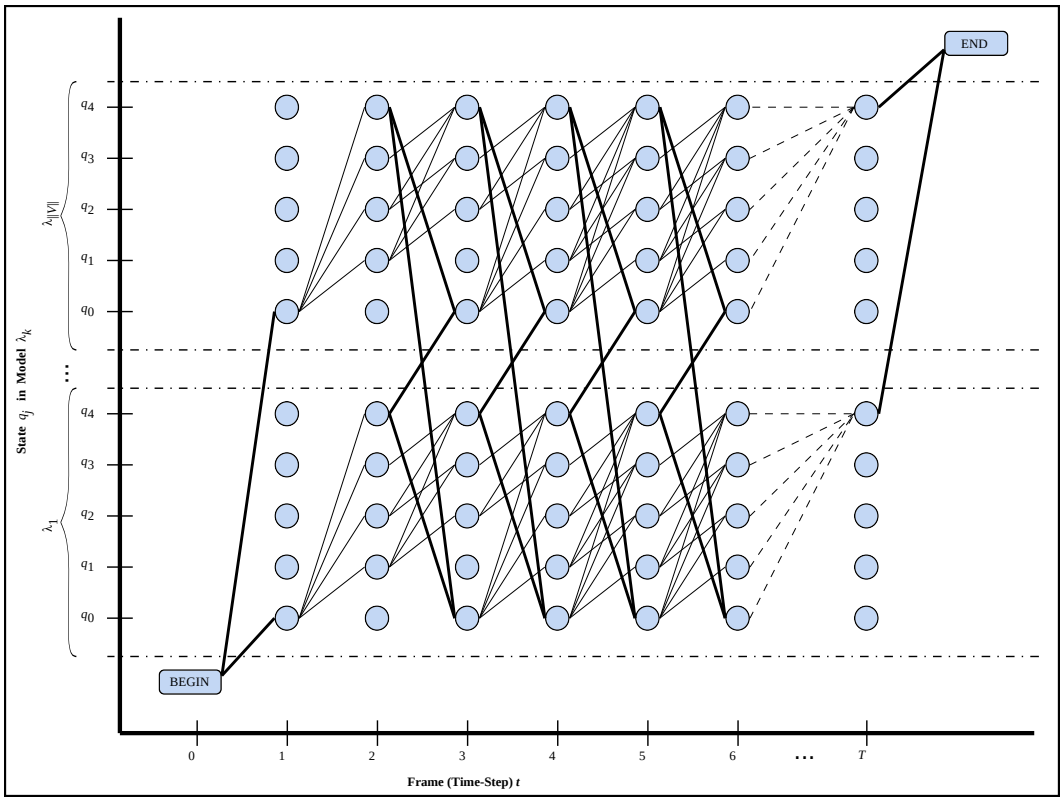


Figure 2.7: *A State/Time-Step Lattice Constructed from a Composite HMM/Bigram Model.*

This search generates the single best transcription of the input that is optimal in the maximum-likelihood sense. As mentioned before, in order to deliver a syntactic analysis some representation of meaning, a second higher-level language model must be coupled in a looser fashion with this technique.

Returning to the issue of explicitly constructing the composite HMM and the state/time-step lattice, we should note that it is sensible to prune at each time step those paths in the state/time-step lattice that do not exceed a minimum threshold, often referred to as the "beam width." Only those paths with likelihoods greater than the beam width survive. Construction of the lattice is thus performed time-synchronously only in promising regions. This beam search, as it is usually called, cannot guarantee that the optimal path (and corresponding transcription) will be found.

Also, at the conclusion of the algorithm, it is possible to do more than just follow the back-pointers to derive the top hypothesis. We can also search backward through the lattice explored by the beam search in order to find a ranked list of paths. The A* algorithm has been used in this setting at many sites (*c.f.* [Huang *et al.*, 1993a]). Search algorithms that generate the n-best hypotheses [Chow and Schwartz, 1989; Schwartz and Chow, 1990] do precisely this. Alternative variants of the beam-search generate a condensed version of the explored sub-lattice rather than the n-best hypotheses. Figure 2.8 sketches examples of these available alternatives.

HMMs and Viterbi Parsing

A related though far less feasible tight coupling involves a PCFG, a more sophisticated, phrase structure language model. Assuming that word HMMs have been trained as discussed above, we construct a composite HMM/PCFG by analogy with the composite HMM/bigram model [Lari and Young, 1990]. The HMMs are simply used in place of the word terminal nodes, and the HMM output symbols (acoustic parameters) can be thought of as the possible terminals of our new compound model.

Given a sequence of acoustic parameters (or parameter vectors), a straight-forward search is very much like the Viterbi algorithm. It searches for the most likely parse tree permitted by the integrated HMM/PCFG using a chart to track possible parse theories. We can think of this algorithm as a probabilistic acoustic parser. It parses the input sequence into the most likely structure permitted by the grammar and thereby identifies a unique word sequence. Thus, the search not only segments the utterance as did the HMM/bigram model, but it also parses it syntactically through a single unified mechanism. For an example, see the integrated models used in BeRP [Jurafsky *et al.*, 1994]).

As before, it is practical to prune those parse theories pursued during the search that do not exceed a minimum threshold that we can also call a "beam width." Only those theories with likelihoods greater than the beam width survive. Again, this beam search cannot guarantee that the optimal theory (and corresponding transcription) will be found. However, every hypothesis that survives pruning but can be covered by the root non-terminal will be generated; thus, this algorithm can generate ranked parses.

Dynamic Time Warping and CYK Parsing

Yet another tight coupling is the non-probabilistic analogue to the HMM/PCFG coupling. It integrates pattern template acoustic models and a CFG language model. Such an integrated acoustic/language model has been used in research by Ney. He augmented a CYK parser for acoustic input by exhaustively finding all possible endpoints for every terminal and scoring the endpoints using Dynamic Time Warping [Ney, 1991].

However, although using a CFG or PCFG on-line may be linguistically appealing, the possible parsing algorithms are impractical due to their computational cost, which is at best $O(n^3)$, where n is the number of input symbols [Lari and Young, 1990]. When parsing acoustic parameter sequences for utterances several seconds in length, n can be in the hundreds.

2.4.2 Loose Coupling

As we have seen, in a usable spoken language understanding system built from existing technologies and algorithms, a tight coupling with complex language models is not feasible. Furthermore, given SR and NLU systems that were designed largely independently, a tight coupling between the two may not be feasible, but a loose coupling may be appropriate. In a loose coupling, each component resolves those ambiguities in its domain of expertise and forwards other ambiguities to later components that can apply higher-level knowledge sources (including local and conversational context) to the problem. It is imperative to bring to bear the top-down constraints available in the NLU system but without bogging down the SR system. In other words, there is a tangible trade-off between the complexity of the top-down constraints and performance.

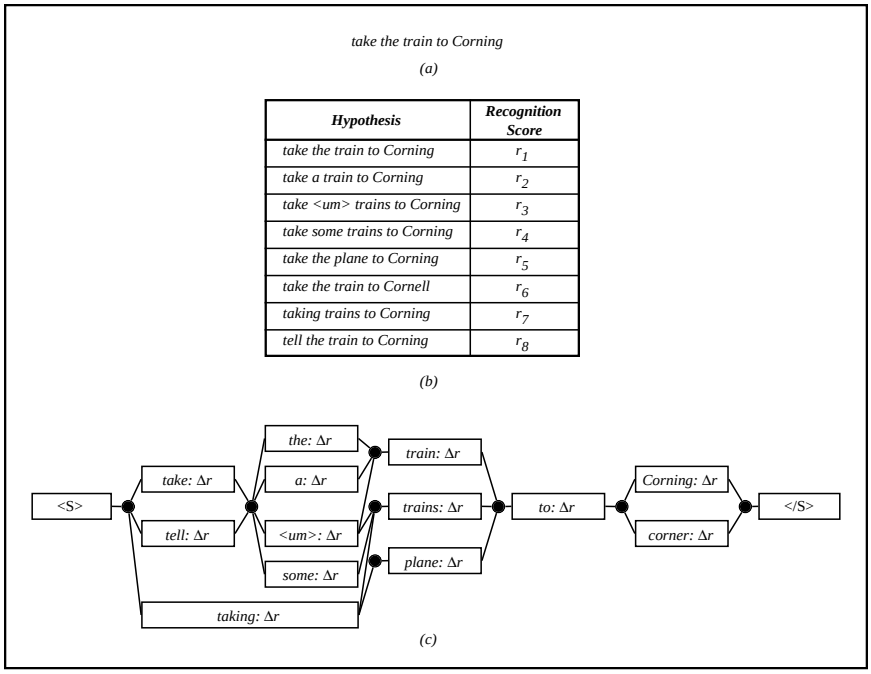
A loosely coupled system isolates the knowledge sources into relatively independent communicating modules. A communication scheme among modules must be designed. Each module uses "level-appropriate" information to make its job less expensive and to avoid combinatorial explosion. For example, a loose coupling avoids making acoustic decisions in the syntactic module. Typically a loose coupling can be identified by its pipelined architecture, with each module post-processing the output of the prior module. Real-time performance is possible even in complex tasks. Also, the modularity makes design, debugging, and understandability more straightforward (the social dynamics of developing independent modules are also more tractable).

In a loose coupling, acoustic matching (and possible usage of simple language models) precedes the application of higher-level language models. In the simplest loose coupling, the SR system provides a single transcription to the parser. In a more sophisticated scheme the SR system provides output in one of two possible formats: either ranked alternatives (the n-best) [Schwartz and Austin, 1991; Schwartz *et al.*, 1993] or a word lattice [Lowerre and Reddy, 1986]. A word lattice is a directed acyclic graph with scored word hypotheses on the arcs; such a lattice compactly captures possible sentence hypotheses. Figure 2.8 depicts all three possibilities.

Ranked Alternatives

Given ranked alternatives, a parser can *rescore* the list of hypotheses to yield a reordered list [Rayner *et al.*, 1994]. This may yield an improvement over the SR system alone in finding the top-best transcription hypothesis and eliminating some recognition errors.

Many modern spoken language understanding systems use this "n-best" approach for coupling with higher-level knowledge sources. For example, the BBN Byblos system [Chow *et al.*, 1987] was the first to generate a list of the n-best hypotheses [Chow and Schwartz, 1989]. It uses tightly coupled HMM acoustic models and a simple n-gram language model. Ostendorf [1992] and others coupled Byblos with higher level knowledge sources via an n-best coupling. The BBN Hark system extends on this work and applies syntactic and semantic rules using a chart parser [Bates *et al.*, 1993].



Hypothesis	Recognition Score
take the train to Corning	r_1
take a train to Corning	r_2
take <um> trains to Corning	r_3
take some trains to Corning	r_4
take the plane to Corning	r_5
take the train to Cornell	r_6
taking trains to Corning	r_7
tell the train to Corning	r_8

Figure 2.8: *SR Output: (a) 1-best; (b) n-best (with scores); (c) Word Lattice (with scores).*

As a second example, CMU's ATIS system consists of the Sphinx-II SR system and the Phoenix parser [Ward, 1990]. Sphinx-II, as described earlier, uses HMM acoustic models and an n-gram language model (coupled tightly); Phoenix performs frame based parsing using a semantic phrase grammar. Ward and Issar report that in the most recent version of the CMU ATIS system, the ability to rescore n-best output using the Phoenix parser contributes to reduction in recognition error rate [Ward and Issar, 1995].

Another example is MIT's ATIS (and Pegasus) system. MIT's system consists of an n-best loose coupling between the SUMMIT speech recognizer and the TINA natural language parser. TINA was designed specifically for processing spoken natural language [Seneff, 1992]. Inspired by the PCFG and ATN formalisms, a set of context-free rules is written and converted to a network structure. Then, transition probabilities on all arcs are trained on a corpus of example sentences. The parsing algorithm uses a stack decoding search strategy, with top-down control flow, and it includes a feature-passing mechanism to deal with long-distance movement, agreement, and semantic constraints. TINA also provides an interface between syntactic and semantic analysis. It should be noted that TINA has been designed with the ability for use in both loose and semi-tight couplings since it can be used to analyze completed theories, or it can filter partial theories during the search process. Most recently, however, use of TINA for the purpose of increasing SR accuracy has not been successful. The team reports:

> We have included an alternate system (MIT_LCS4), which differs from our standard system only in that the final step of filtering by the TINA NL system was omitted. In the pre-adjudicated results, TINA gave a small advantage in performance, but this advantage disappeared after adjudication. It is still our belief that NL constraint should

> be useful for recognition, but will probably not be significantly effective unless we can incorporate the NL component earlier in the search, fully utilizing its probabilities rather than simply filtering on a hard decision based on the presence or absence of a full parse. [Glass *et al.*, 1995]

Word Lattices

A loose coupling via an n-best list simplifies the task of higher-level processing, in so far as higher level modules need only consider a sentence at a time. However, this approach is deficient in so far as most of the utterances generated have many word sequences in common and are typically only slight variations of one another; thus, significant work is duplicated. The word-lattice is often preferred since it contains significantly more information than an n-best list of comparable size. Experiments by Harper *et al.* [1994] have shown that word graphs are approximately 83% more compact than n-best lists. Also, recent results from the Cambridge speech recognition group indicate that under specific conditions word lattices contain the correct word sequence roughly 93.5% of the time [Woodland *et al.*, 1995].

For example, Harper *et al.* [1994] integrate SR, prosodic analysis, and a parser using a loose coupling. The SR is built from HMM acoustic models and produces a word lattice called a Spoken Language Constraint Network (SLCN). The word lattice is constrained by acoustic matches, an n-gram language model, and stress and duration patterns provided by a prosody module. Also, the prosody module annotates the word lattice with prosodic cues that can be used by higher-level processing to further prune the lattice.

Blackboard

At the "loosest" end of the loose coupling category is the blackboard model used by the Hearsay II system [Lesser *et al.*, 1975]. In the blackboard model, each knowledge source is represented as an independent process that can acquire information from and submit information to the blackboard. The blackboard itself is a multi-level network that permits generation and connection between alternative hypotheses at all levels. This approach is very flexible and ultimately modular. As Harper *et al.* [1994] observe, "One reason this approach has not come back into favor may be that the blackboard approach is too loosely coupled.".

2.4.3 Semi-Tight Coupling

Semi-tight couplings fall in between the other two. A knowledge source can be used to guide a lower level search in the system. Thus, the removal of the knowledge source impacts the lower level search and can therefore not be considered completely independent.

HMMs and the Viterbi Algorithm Revisited

The most successful integration of an acoustic model and a language model has been the HMM/n-gram combination. As described above in the discussion of tight couplings, when $n = 2$ we actually have a tight coupling. When $n > 2$, the composite HMM/n-gram model is not explicitly

constructed; rather, during the search, transitions from one word HMM to the next are hypothesized on-line using the n-gram model and the previous $n-1$ word hypotheses. Since the n-gram model is used to guide the attempted acoustic matches, this approach is a semi-tight coupling.

Unification-based Grammar Parsing

Since Unification-based Grammars are an extension to CFGs, it is not difficult to see that their use in a tight coupling would also suffer from prohibitively expensive searches. In fact, parsing for Unification Grammars is more expensive than for CFGs [Haas, 1989]. Hence, a tight coupling is not currently feasible, whereas semi-tight couplings are possible. In fact, work funded by the German VERBMOBIL project has investigated possible semi-tight couplings of an HMM-based SR and a parser using a Unification-based Grammar [Hauenstein and Weber, 1994]. The authors of this work refer to their couplings as "tight," but their own descriptions of the couplings satisfy the definition of a semi-tight coupling as defined by Harper *et al.*. The fact that their SR and parser modules run as distinct processes (with the ability to run on separate machines) attests to their separability. The authors' motivation for using the term "tight" seems to draw from the fact that each of their couplings is time-synchronous.

Their first coupling involves a time-synchronous bottom-up coupling. The decoder produces a word lattice using only the HMM acoustic models. The lattice is sent incrementally to the parser as it is produced. The (chart) parser searches from left to right with the aid of an n-gram language model and the unification grammar. (This one is actually a loose coupling, although it is on-line.)

In the second coupling, the decoder produces word hypotheses based on acoustic score. The decoder consults the parser for each hypothesis. In response the parser provides 1. a language model score from the n-gram language model and 2. a binary decision based on the unification grammar. The decoder uses this information to rescore its hypotheses.

In the third coupling, the same parser takes the lead by producing LR predictions from left to right using the unification grammar and the n-gram model together. These top-down restrictions constrain the acoustic match by the decoder.

Such LR prediction by the parser has been tried before using a Tomita parser [Kita *et al.*, 1989; Hanazawa *et al.*, 1990]. The researchers at ATR use the parser to predict phones that are verified by phone-level HMMs. Words are built from phones using rules in the grammar. As in the experiment by Hauenstein and Weber, the acoustic and language models are not entirely separable since the acoustic model receives its focus from the language model.

2.5 Robust Parsing Strategies

Successful analysis of spontaneous spoken language is significantly different than processing text. For the interpretation of spoken language, there is a tension between providing robustness and imposing constraints. A robust system attempts to construct an accommodating interpretation, especially in the presence of performance errors by the speaker and SR errors. In contrast, in order to provide language constraints to a speech recognizer, a parser should detect that a recognized string is not an English sentence and disprefer that hypothesis. As we have seen, for a tight coupling between the parser and the SR module, the parser may enforce as many constraints on

partial utterances as possible. Furthermore, to handle the incremental structure of spontaneously spoken utterances, we require methods for segmenting a given utterance into "chunks." This section reviews phrase-structure and heuristic approaches to this problem. We first consider one family of approaches based on pattern matching techniques. Second, we consider a family of approaches based on phrase-structure grammar parsing techniques.

2.5.1 Robust Pattern Matching

The first approach to robust parsing is based on the idea that spontaneous utterances are noisy and not worth the trouble of a full analysis. Techniques subscribing to this approach use simple phrase models to find regions of usable text and discard any segment that does not appear to be usable. The term "skip parser" has been used to refer to such systems. The simple phrase models are often based on regular grammars, finite-state automata, and ATNs.

For example, SRI's FASTUS system is a system for extracting information from free text in English, for entry into a database. Its parser is a set of cascaded, nondeterministic finite state automata [Hobbs *et al.*, 1993]. It has performed well on information extraction tasks established by the "Message Understanding Conference" (MUC) community. It is important to note the limitations of a system such as FASTUS. FASTUS is most appropriate for information extraction tasks, rather than full text understanding. That is, it is most effective for text-scanning tasks where 1. only a fraction of the text is relevant and 2. there is a pre-defined, relatively simple, rigid target representation that the information is mapped into.

The SpeechActs conversational system at Sun Labs supports interaction with applications on a workstation over the phone [Yankelovich, 1994; Yankelovich *et al.*, 1995]. Motivated by the FASTUS approach, it uses a parser called Swiftus [Martin and Kehler, 1994]. It translates the output of the BBN Hark SR system into feature-value pairs. Thus, in essence, Swiftus fills slots in frames. The CMU ATIS system couples the Sphinx-II recognizer with the Phoenix parser. Phoenix also fills slots in frames using ATNs to identify promising phrases [Ward, 1990].

Another example is the HWIM system, developed at BBN [Woods, 1983] as part of the first ARPA speech recognition initiative [Wolf and Woods, 1980]. HWIM employed a unique search strategy called island-driven parsing. This strategy uses an ATN grammar to scan an utterance for "islands" of word-sequences that are most likely to be correct. The objective is to grow these most confident islands until an entire sentence has been hypothesized if possible. Rather than parsing from left-to-right, island parsing goes counter to the intuition that the signal should be analyzed in the order in which a listener would receive it.

At CMU, Young and Ward coupled Sphinx-II with a robust parser in a loose fashion [Young and Ward, 1993]. Given an utterance of spontaneous speech, the SR system generates the single, best-scoring word transcription hypothesis. The RTN-based NLU system parses the output string and uses higher-level knowledge sources to identify possible misrecognitions. Consequently, the potentially misrecognized segment of the utterance is reprocessed by a RTN speech decoder using a subset of the networks used previously by the parser. [7]

One downfall of these techniques is that the phrase patterns are very specific to the task at hand and cannot be easily ported. They also do not yield a substantial syntactic analysis.

[7] On an analogous problem, Srihari and colleagues at Buffalo have investigated similar re-analysis techniques for using syntax to refine handwriting recognition [Srihari and Baltus, 1993].

2.5.2 Robust Grammar-based Parsing

The second approach to robust parsing uses phrase structure models that are capable of providing full-fledged parses and semantic analyses whenever possible. When a sentence-level hypothesis cannot be found, such techniques are resilient and construct alternatives.

Chart Parsing

Most often, such techniques rely on the technique called chart parsing. Earley described an efficient algorithm for parsing CFGs top-down using a data-structure called a "chart," hence the name *chart parser* [1970]. Other techniques have been devised for bottom-up chart parsing. Chart parsing can be considered dynamic programming on partial phrase structures. It is possible to use a chart parser in an incremental fashion, which makes it a better candidate than a more traditional parser for use in a SR system (*c.f.* [Paeseler, 1988]). Because the chart contains partial phrase structures, it is possible to glean them from the chart in the event that no sentence-level constituents are formed. For example, the Microsoft Research parser and its predecessors (*c.f.* [Jensen *et al.*, 1993]) construct a "fitted parse" in the event that a sentence cannot be identified. Constituents are selected from the chart in such a way that they cover the input.

The Microsoft Research Persona system [Ball and Ling, 1994] integrates speech recognition with the parser just described (as well as 3-D animation and speech synthesis) to create a conversational assistant that interacts with a user in a spoken dialogue. One primary goal of the system is to allow users as much flexibility as possible in expressing their requests in the syntax they find most natural, even though the assistant currently handles requests in a very limited domain. Ball and Ling explain that using a broad-coverage parser makes porting the system to new application domains a trivial task from the point of view of parsing. Since the Microsoft parser generates logical forms that must be canonicalized for their domain, only the module that normalizes them must be ported. For this reason, they claim, "The use of a general purpose natural language understanding system is a better strategy for conversational interfaces than the creation of specialized application languages."

Gemini is SRI's natural language understanding module developed for spoken language applications. Gemini tightly constrains language recognition to limit over-generation. The parser is all-paths bottom-up, uses subsumption checking, and is on-line. However, it also extends the language analysis to recognize certain characteristic patterns of spoken utterances not generally included in a grammar and to recognize specific types of performance errors by the speaker [Dowding *et al.*, 1993]. This is accomplished by an utterance grammar and utterance parser that are used to post-process the chart. The SRI ATIS system incorporates this robust interpretation component that constructs queries out of grammatical fragments when an utterance cannot be analyzed as a single phrase or utterance [Moore *et al.*, 1995].

2.6 Prosodic Analysis

To address the incrementality of spontaneously spoken utterances, we require methods for segmenting a given utterance into "chunks" representing individual thoughts. The previous section has reviewed phrase-structure and heuristic approaches to this problem. However, alone, they are

unable to cope successfully with the problem. They neglect an important resource for solving this problem, namely prosody. This section provides a brief overview of prosodic features followed by a discussion of how they have been used by other researchers to aid in parsing.

2.6.1 Prosodic Features

In need of a unit of comprehension for spoken language, we cannot use a sentence or a complete turn, as we reasoned in the introduction. However, Halliday [1967] and Gee and Grosjean [1983] reasoned that prosody should be the main source of information for deriving such a unit. Which prosodic features are most useful for identifying these phrasal units? The main prosodic attributes are *phrasing*, *prominence*, and *intonation*. Prosodic phrasing refers to the perceived groupings of words in an utterance; they can be measured in terms of break indices. Phrasal prominence refers to the perceived strength or emphasis of some syllables in a phrase. Prominence is also referred to in some work as stress, accent, or emphasis. Intonation refers to the tonal patterns of an utterance. Wightman and Ostendorf [1994] claim that phrasing and prominence are most useful for the task of finding structure in an utterance.

Phonological research has investigated abstract prosodic structure. within the linguistic literature, numerous researchers have posited a number of prosodic phrasing constituents with hierarchical relationships such as word, minor phrase, major phrase, and utterance [Wightman, 1992]. Research in phonetics has analyzed the acoustic cues that mark these units. The cues that have been claimed to mark prosodic phrase boundaries are preboundary lengthening, pauses, breaths, boundary tones, and speaking rate changes [Wightman and Ostendorf, 1994]. The cues that have been claimed to mark phrasal prominence include duration lengthening, pitch accents, and increased energy. They are determined using techniques for measuring F0 (fundamental frequency contour), duration, and energy.

These cues (correlates) provide the basis for automatic detection of the linguistic constituents. Wightman and Ostendorf [1994] have developed an algorithm that can automatically find a mapping between the cues and the perceptually labeled prosodic patterns. This mapping can then be used on-line to detect prosodic phrasing and prominence. However, because the mapping is designed for and trained on speech from professional FM radio news announcers reading scripts, its performance is limited. Consequently, there is a need to extend the current algorithm for spontaneous speech.

2.6.2 Prosodic Features Aid Parsing

At the lowest level, Waibel demonstrates that prosodic cues alone can be used to constrain legal word hypotheses in continuous speech [Waibel, 1987; Waibel, 1988]. For our present purposes, we focus on the fact that parsing can use prosodic features derived from the speech signal in order to enhance the speed and accuracy of the analysis [Pierrehumbert and Hirschberg, 1990; Rowles and Huang, 1992]. As we have seen, many current robust parsing strategies typically rely only on phrase structures and heuristics, neglecting prosodic features. Simply put, the available information is not being used.

By using the available prosodic cues, the prosodic features of phrasing and prominence can be detected and used for syntactic disambiguation. Using the Wightman and Ostendorf algorithm, Ostendorf, Wightman, and Veilleux [1993] have reported using automatically detected prosodic

phrasing to evaluate (rescore) possible parses. Their technique can be said to perform syntactic disambiguation *after* the parsing stage. For a given utterance and hypothesized word-transcription and a particular parse of that sequence, their score is the probability that the acoustic features coincide with the hypothesized word sequence given the syntactic parse; this score is based on acoustic and "language" (prosody/syntax) models. Veilleux and Ostendorf [1993] have extended the algorithm to deal with speech in the ATIS task and have shown that the prosody/parse score improves the average rank of the correct sentence hypothesis. Unfortunately, this approach follows parsing, where significant influence can be had directly.

Others have shown that prosody can constrain syntactic hypotheses or provide anchor points for parsing strategies [Rowles and Huang, 1992]. Work by Harper *et al.* [1994] actually integrates prosodic constraints into a constraint-based parsing framework. Prosody is also known to correlate with the pragmatic and semantic content of an utterance, but there is little record of work in the literature that deals with how to apply this information.

Chapter 3

Speech Recognition in the Trains Domain

In this chapter, I discuss preliminary work involving speech recognition in the Trains domain, including a speech recognizer trained on the TRAINS Dialogue Corpus. I also briefly discuss the TRAINS-95 system and its SR, language models, and parser components. Numerous dialogues have been collected using a preliminary version of the TRAINS-95 system; I also report on these efforts here. Finally, I comment on the potential for improved dialogue understanding in the Trains domain.

3.1 HTK-based Speech Recognizer Trained from Trains Dialogue Corpus

This section describes the TRAINS Dialogue Corpus (TDC) and experiments with a speech recognition system trained and tested on utterances from the TDC.

3.1.1 The TRAINS Dialogue Corpus

The TRAINS project is a long term research project to develop an intelligent planning assistant that is conversationally proficient in natural language [Allen, 1994]. In particular, in its 1993 implementation, the TRAINS agent helps a person (the manager) construct and monitor plans about a railroad freight system. The manager is responsible for assigning cargo to trains and scheduling shipments, scheduling various simple manufacturing tasks, and for revising the original plan when unexpected situations arise during plan execution. The system aids the manager in all aspects of this task by interacting in natural language.

As part of that project, Peter Heeman and others have collected a corpus of spontaneous problem solving dialogues. The dialogues involve two human participants, one who is playing the role of manager, and another playing the role of the system [Gross *et al.*, 1992; Heeman and Allen, 1995]. The participants are aware of one another; this was not a wizard-of-Oz exercise, since natural spontaneous dialogue was desired. The entire corpus consists of 98 dialogues totaling six and a half hours in length and containing about 55,000 transcribed words and 5900 speaker turns [Heeman and

Allen, 1995]. These dialogues have been transcribed and aligned at the word- and phoneme-levels and include several prosodic cues. They have also been segmented into utterance files. An utterance file can be defined in the following terms: they are sequences of words that capture local phenomena, respect the turn-taking mechanism, and have relatively short length [Heeman and Allen, 1994a].

3.1.2 SR Prototype

I have constructed a simple SR system using the HTK toolkit and the TRAINS Dialog Corpus (TDC). HTK is a toolkit for building HMM-based SR systems.

As of the time that this SR experiment was performed (December 1994), the phoneme-aligned corpus consisted of 3110 utterances containing 29831 transcribed word tokens (83412 transcribed phoneme tokens) and 707 distinct words (including noise words such as coughs, sneezes, laughs, breaths, etc.). [1] For the purpose of building this SR system, we divided the phoneme-aligned corpus into a training set and a test set: we used 15 of the dialogues, consisting 2853 utterances for the training set; and for testing, we chose 5 of the dialogues (at random), containing 257 utterances.

3.1.3 Evaluation

I trained HMM acoustic models for each phone in the phone set. I then performed recognition at the word-level, using 548 word HMM acoustic models built from the phone models and using a bigram language model built from the training set alone. The word recognition results are shown in Table 3.1. Note that only 11.46% of the utterances were recognized entirely correctly.

	%	#
Words		2027
Correct	52.2%	1057
Total Errors	76.3	1545
Substitutions	42.3%	857
Insertions	28.4%	575
Deletions	5.6%	113
Word Accuracy	**23.8%**	

Table 3.1: *Word Recognition Performance.*

What is the significance of these results? First of all, the recognition accuracy is poor for such a medium-sized vocabulary. By contrast, some systems have been reported to perform in the high 90s for vocabularies of this size; however, it is critical to note that such results have been reported on read speech and on human-computer speech. However, they have not been reported on truly spontaneous human-human speech as we have done here. Part of the significance of these results is that they attest to the truly spontaneous nature of the speech in the TDC. As the examples in

[1] Since that time, the entire corpus has been phoneme-aligned, has been placed on CD-ROM, and is available through the LDC [Heeman and Allen, 1995].

the introduction attested, spontaneous speech can be erratic. Recognizing truly spontaneous speech using traditional techniques is difficult.

There are also several directions that *could* be pursued in order to improve this bare-bones system. One area in which the current SR system can be improved is its language model. Several options for improving the language model are feasible. First, an improved (word-)bigram language model can be built from another, larger corpus. Any corpus used for building our language models should have a similar character to ours. It should at least be transcribed dialogue and would preferably include prosodic cues similar to the TDC. Possible candidates include the various ATIS speech corpora. Second, if the HTK `HVite` tool is enhanced to use word-trigram language models, we will be able to immediately take advantage of this feature and thereby further constrain the perplexity of our recognition task. Furthermore, the HTK tool-set does not currently include any mechanisms for generating a word-lattice. It uses a simple Viterbi decoder that generates the top transcription hypothesis. Finally, context-independent phone models are used currently. A straightforward enhancement would involve training context-dependent phone models.

3.2 Language Models for Sphinx-II in TRAINS-95

3.2.1 The TRAINS-95 System

Within the context of the TRAINS project and since the TRAINS-93 implementation, we are building a system that can interact and collaborate with humans in problem solving using spoken language. By problem solving, we include a wide range of applications, including interactive planning and scheduling, task-related information retrieval, and command and control applications. Specifically, we have been focusing on mixed-initiative planning about transportation scheduling [Allen *et al.*, 1994; Ferguson, 1995]. This domain is used as a vehicle for studying two capabilities that we believe are inextricably linked: human-machine dialogue and mixed-initiative planning. First, we believe it makes little sense to study human-machine dialogue independently of a class of tasks that needs to be accomplished. Second, we believe that the notion of natural language dialogue provides a powerful metaphor for mixed-initiative planning. Our notion of dialogue is not restricted to natural language, however. Speech is just one of several modes of communication available to the user and the system. Other modalities include map displays, mouse actions, menus and keyboard driven commands. The word "dialogue" refers to the entire human-computer interaction across all communication modes.

As a step to building the full TRAINS system, we have constructed a smaller-scale system from which to boot-strap, called the TRAINS-95 system [Allen *et al.*, 1995]. The goal of TRAINS-95 is to use the simplest scenarios and tasks that would produce interesting dialogue behavior. The task for TRAINS-95 is simple route planning problems. Typically, a goal may involve moving two or three trains to specific destinations starting from a randomly generated initial situation. The current version of the TRAINS-95 system consists of the following components, which are logically organized as shown in Figure 3.1:

- the Sphinx-II speech recognition system [Huang *et al.*, 1993b] trained on ATIS data, giving 1-best output;
- the TRUETALK speech synthesis system;

- a fairly comprehensive unification-based grammar and lexicon built from the TRAINS Dialogue Corpus;
- a robust understanding system based on a bottom-up chart parser with monitors;
- a speech-act based dialogue model (used for both understanding and generation);
- a graphical display; and
- a deliberately weak route planner.

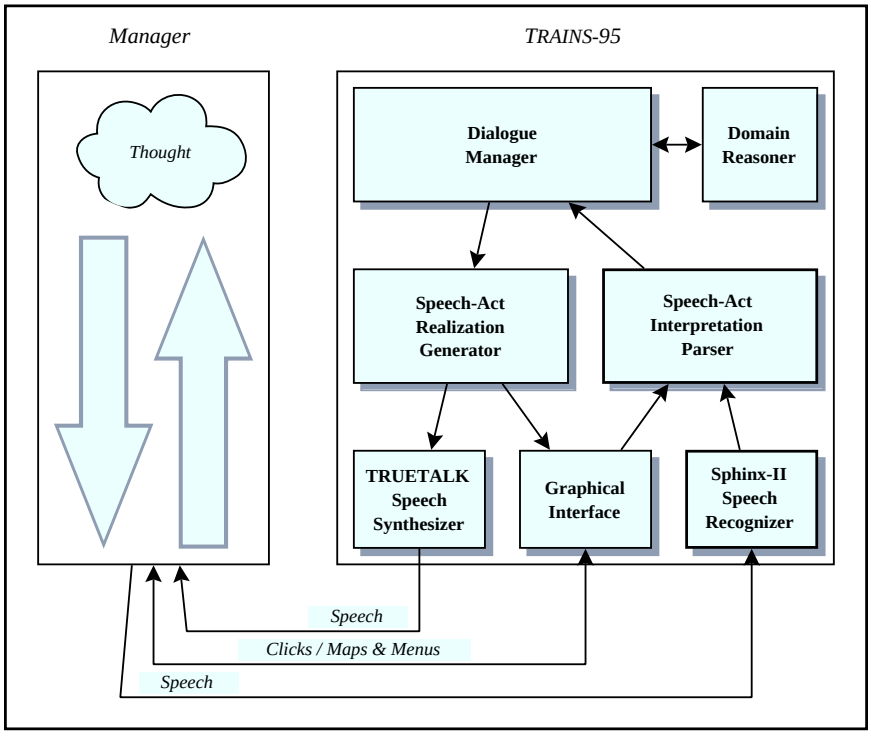


Figure 3.1: *Architecture of the* TRAINS-95 *System (adapted from* [Allen *et al.*, 1995]).

The route planner is deliberately weak in order to encourage interaction. Otherwise, as soon as the system identified the goal, it could provide the optimal solution. The planner is restricted so that it cannot plan routes between cities more than four hops apart, and it randomly generates a route from all possible routes (of length 4 or less) for cities that are close enough. Thus, the user has to incrementally develop her plan throughout the dialogue. The system has a basic conversational capability: it generates appropriate confirmations and participates in the give-and-take of the route development. It also integrates language generation with graphic output. The system can robustly operate even in the presence of significant speech recognition errors.

3.2.2 Speech Recognition and Parser

The speech recognition component of TRAINS-95 is based on the Sphinx-II recognizer from CMU using a back-off bigram language model trained on transcribed ATIS dialogues. The interface

currently provides a "push-to-talk" mechanism and displays the results of the recognition as they are computed by Sphinx-II. Words are output incrementally as they are recognized, with control tokens used when Sphinx-II "backs up" its hypothesis. Hence, we have a loose, on-line coupling between Sphinx-II and the parser that results in a quicker response for the user.

The TRAINS-95 parser must be both robust against SR errors yet able to produce a detailed semantic analysis of the input when possible. It does this by combining ideas from several different approaches. First, for effective disambiguation, the unification-based grammar classifies each constituent both by syntactic and semantic categories. Thus, rules in the grammar mix syntactic and semantic constraints, with some rules being primarily syntactic, while others are primarily semantic. While our grammar shares many of the advantages of semantic grammars, few of our semantic categories are domain specific. Rather, they deal with semantic constructs that underlie much of everyday language, including semantic concepts such as agents, events, actions, times, locations, units and measures, static and movable objects, plans and goals, and so on. As a result, the grammar could be adapted relatively easily to handle other problem solving situations.

The parser is a bottom-up chart parser. Robust interpretation is accomplished by a set of monitors. Monitors are procedures that are invoked when constituents matching certain patterns are added to the chart. The monitors extract key information about the parsed constituent and let the parser continue. The output of the parser includes the following information regardless of whether the input completely parsed or not:

- Objects - the interpretations of the noun phrases in the utterance.
- Speech-Act - the speech-act type realized by the utterance.
- Action - the principle action involved.
- Semantics - the logical form.
- Paths - any paths mentioned.

Because a single utterance may involve several speech-acts spoken incrementally, the output of the parser is actually a series of analyses containing the above information, one for each speech-act. Utterances are interpreted as actions (*speech-acts*) belonging to one of several available categories. These action categories belong to a well-defined hierarchy and are used for systematic interpretation of the speaker's intentions. The speaker acts by way of her utterances to accomplish various effects with the system. Thus, utterances are related to the mental state of the speaker [Grice, 1957; Austin, 1962; Searle, 1969]. The actions include such things as suggesting, rejecting, and requesting. These actions are especially useful when used in a plan-based approach to dialogue understanding. This idea was first suggested by Bruce [1975], and formalized by Cohen and Perrault [1979], who developed a system that takes an agent's goal and finds a speech action that will accomplish it, and by Allen and Perrault [1980], who try to determine the agent's goal, given an observed action,

The robust parser pays particular attention to extracting information about speech-acts. There are two basic mechanisms for recording information about speech-acts. The full parser produces a speech-act interpretation using rules that associate speech-acts with the syntactic and semantic structure of the utterances. Second, the speech-act monitors look for lexical clues for the speech-act intended. For example, an utterance that contains "please" would be flagged as a request, and an

utterance that begins with "I want to ..." is flagged as a goal statement. If the full parser produces a speech-act interpretation, this reading may be further specialized by the lexical clues. If there is no full parse, the lexically-based pattern may still successfully identify the speech-act. A good example of this occurs in one of the early videotaped sessions. The user actually said "Let's try going via Cleveland and Syracuse instead," but the speech recognizer produced the output "Let's try going feed equivalent answer reduced instead." While virtually none of this input is interpretable, the monitors detected the use of the phrase "Let's" and the word "instead" at the end of the utterance, which suggests that the speech-act was a rejection (or partial rejection) of the previous proposal. The system responded based on this interpretation by offering an alternate route, quite a good continuation given how little of the utterance was correctly recognized.

This demonstrates an important point about robustness. If the system can identify the intended speech-act, it can often produce a reasonable continuation even if little of the content of the utterance is understood. If nothing else, it will be able to generate an intelligent clarification request since it has a model of what the user was trying to do even though it doesn't know the details.

3.2.3 TRAINS-95 Dialogue Collection, Segmentation, and Transcription

TRAINS-95 has been running for several months. During that time, we have demonstrated the system to a diverse set of people who have consented with our use of logs and data collected during their sessions with the system. Table 3.2 is a summary of the current corpus of collected dialogues.

Dialogues	Utterances	Words
258	3313	20332

Table 3.2: *Current Corpus of* TRAINS-95 *Dialogues.*

The early dialogues required segmentation; however, segmenting the dialogues did not require application of the TDC utterance-file heuristic, since the interchange between manager and system occurs in turns. Reference transcriptions have been written for each utterance in each dialogue for the purpose of evaluating the performance of Sphinx-II using various language models. These transcriptions will also be used in training and testing the post-processor models to be described in chapter 4.

3.2.4 New Language Model

In order to introduce vocabulary appropriate to the TRAINS-95 task, I built a custom language model using the transcribed utterances from ATIS2, ATIS3, and the TDC. I wrote tools for conditioning the TDC in order to make its formatting consistent with the ATIS corpora. For constructing the bigram back-off model, I used Roni Rosenfeld's Language Model Toolkit [Rosenfeld, 1995]. The word-accuracy of Sphinx-II when tested on TRAINS-95 utterances is higher with the ATIS-only model than with our custom language model because the ATIS model was built using lexical classes, which give the model more consistent and thorough coverage. This technique will soon be available to us also. Table 3.3 illustrates this with utterance-level recognition results, and Table 3.4 provides word-level results.

	ATIS-only LM		ATIS + TDC LM	
	%	#	%	#
Utterances		3313		3313
Correct	**34.3%**	1136	**26.6%**	882
With Errors	65.7%	2177	73.4%	2431
With Substitutions	59.9%	1986	66.1%	2190
With Insertions	24.2%	803	23.1%	765
With Deletions	15.9%	526	27.3%	904

Table 3.3: *Utterance Recognition Performance.*

	ATIS-only LM		ATIS + TDC LM	
	%	#	%	#
Words		20332		20332
Correct	74.8%	15218	69.3%	14085
Total Errors	31.0%	6294	35.9%	7305
Substitutions	21.7%	4406	24.5%	4976
Insertions	5.8%	1180	5.2%	1058
Deletions	3.5%	708	6.3%	1271
Word Accuracy	**69.0%**		**64.1%**	

Table 3.4: *Word Recognition Performance.*

3.3 Conclusions

The TRAINS-95 system is an excellent foundation upon which to build in order to explore the issues of this thesis. The domain is rich enough to encourage interesting dialogues but simple enough to avoid unnecessary complexity. Speech recognition rates are usable but not stellar; hence, significant progress can be made by improving the coupling between SR and NLU components. Also, the robust parsing strategy is state of the art in most respects; however, it lacks guidance where other state of the art parsers also require improvement, namely in the realm of chunking spontaneous utterances. Hence, we expect that dialogue understanding can be significantly enhanced by the efforts described in the next chapter.

Chapter 4

Improving the Loose Coupling

As we have seen, the available computational methods for analyzing speech do not perform as well as we would hope on spontaneous speech. Even a state of the art recognizer such as Sphinx-II achieves only slightly worse than 70% word accuracy on fluent speech collected from sessions with the TRAINS-95 system, and attempts at recognizing speech recorded from human-human conversations yield even more discouraging results. The first section of this chapter presents my plans for overcoming several kinds of speech recognition errors.

We have also seen that transcribed spontaneous speech is challenging to process because it is more incremental than written language. Utterances are often composed of brief phrases and fragments that come in installments and refinements. The second section presents my plans for coping with spontaneous phrase structure while parsing.

Subsequent sections discuss experiments for validating the work, efforts for continuing data collection, a work schedule, and conclusions.

4.1 Correcting Speech Recognition Errors

Various SR/NLU couplings have been used to prevent or to correct errors. Tight couplings with language models capable of performing syntactic analysis (and even higher-level analyses) while constraining the speech recognition process are too expensive for the tasks we are interested in. Semi-tight couplings allow for interaction with some higher-level knowledge sources, but they require that the knowledge sources be carefully designed to cooperate. Loose couplings are feasible for large vocabulary tasks. Best of all, loose couplings combine relatively independent modules in a pipeline, which is especially desirable in light of the increasing use of speech recognizers as black boxes.

In the TRAINS-95 system, we have an effective loose coupling between the independently developed Sphinx-II recognizer and the TRAINS-95 robust parser. The examples in the introduction provide ample evidence that there is room for improvement in this coupling. Improvements will be manifest by fewer recognition errors and, of course, fewer overall understanding errors.

To address the issue of reducing speech recognition errors, I regard the channel from the user to the output of the SR module as a noisy channel, and I adopt statistical techniques (some of

them borrowed from statistical machine translation) for modeling that channel. In this section, I first provide more motivation for post-processing; second, I mention the motivation for turning to statistical machine translation for inspiration; third, I describe the nature of the post-processor component and present preliminary results; fourth, I flesh out the model; and fifth, I mention related issues to be investigated.

4.1.1 Motivation

Why reduce recognition errors by post-processing the SR output? Why not simply better tune the speech recognizer's language model for the task? By taking a look into modern SR technology, we find that several human speech phenomena are poorly modeled and that recognition is accordingly impaired. This suggests that the SR module does indeed belong as a component of the noisy channel. One such phenomenon is assimilation of phonetic features. Some SR engines attempt to model phonemes in a context-dependent fashion, and some attempt to model cross-word co-articulation effects. However, as speaker speeds vary, the SR's models may not be well suited to the affected speech signal. Such errors can be handled through post-processing.

It should also be noted that the model's content could be utilized by the speech recognizer itself in the decoding algorithm if the recognizer were suitably modified. When using an existing SR component as a general-purpose black box, however, this is not an option. Furthermore, the statistical model will be trained on utterances in a particular domain; therefore, the post-processor truly does belong outside of the recognizer.

Furthermore, as we have seen, existing n-gram models alone cannot represent the substitutions that we require for correction. The primary advantage to this approach over existing approaches for overcoming SR errors lies in its ability to introduce options that are not available in the SR module's output. As mentioned earlier, existing rescoring tactics cannot do so. However, one disadvantage to the post-processing approach is the need to actually train the additional statistical models described below.

4.1.2 Statistical Machine Translation

Both qualitative and quantitative models for machine translation (MT) of human language have been designed and implemented. A statistical model for translating individual sentences proposed by Brown *et al.* [1990] at IBM is relevant to our problem of correcting speech recognition errors. As we discussed earlier in the case of speech recognition, their model is based on the communication engineer's concept of a channel through which information is transmitted and where errors are possibly introduced. They consider the generation of a sentence in a foreign language, such as French, to be the transmission of an English sentence through a noisy channel. The English sentence is the original "undefiled" signal, and the French sentence is the same signal at the other end of the channel. The task of the MT system then is to recover the original English given the French. This model may sound a bit far-fetched, and it does have its critics.

However, if we consider the process of understanding a spoken utterance, we can adapt the same idea and posit the existence of a string of English words ($\underline{\mathbf{e}}_{1,n}$) in the mind of the speaker. Those words are uttered and transmitted to the listening system's microphone. The sounds are then transcribed as a string of English words ($\underline{\mathbf{e}}'_{1,n'}$) by the SR component of the system. The channel

beginning at the speaker's mind and ending at the output of the SR module is a noisy channel, in which errors are frequently introduced in all segments of the channel, including the SR module, essentially at the word-level. We can apply the statistical MT techniques to recover the original string of words and thereby correct some of the errors introduced in the channel. Perhaps this scenario is more acceptable to the reader than the mutation of English into French. The details of the approach have been adapted below for the correction of SR errors.

4.1.3 Correcting Recognition Errors Using a Post-Processor

To model the channel, I adopt a model that describes some of the effects on utterances in a specific task domain when they pass through the speech recognizer. Specifically, it accounts for frequent errors such as simple word/word confusions and short phrasal and segmentation problems (*e.g.*, one-to-many word substitutions and many-to-one word concatenations). In addition to the model, I will design a suitable search algorithm to use the model to find the most likely correction for a given word sequence from the SR module. I will build a component that uses the statistical model and the search algorithm and wedge it into the interpretation pipeline just behind the SR module. This post-processor will correct some of the errors introduced in the noisy channel in order to provide input containing fewer errors to the parser, making its analysis more reliable in turn.

I divide the SR module's vocabulary into two parts: a task-specific vocabulary and a general-purpose vocabulary. The model will be trained to repair utterances consisting of words from the task-specific vocabulary. The SR module can still recognize words outside of this specific vocabulary. In fact, if an out-of-domain word arises, then the post-processor will simply forward the unknown word to subsequent components (assuming that that word was not observed as a misrecognition in the current context in the training data). If, however, that word is known to be frequently misrecognized, then the post-processor will correct it to the appropriate in-domain word.

Why is post-processing the domain-specific parts of the output from a general-purpose speech recognizer a valid path to take? I will adapt a point made by Ball and Ling [1994]: using a broad-coverage module allows a system to deal with diverse kinds of utterances; then for domain-specific tuning, the output of the general-purpose module can be suitably post-processed. Although porting the system to new domains requires tuning the post-processor, the general-purpose component can be used with little or no change.

4.1.4 The Model

Brown *et al.* delineate their approach into three parts: a translation model, a language model, and a search among possible source word sequences. We will describe each.

Figure 4.1 illustrates the relationship of the speaker, the channel, and the error-correcting post-processor. By applying Bayes' rule, we derive a simple expression for the most likely pre-channel sequence $\underline{\hat{\mathbf{e}}}_{1,n}$. The derivation is identical to the derivation of the statistical approach to SR:

$$\underline{\hat{\mathbf{e}}}_{1,n} = \operatorname*{argmax}_{\underline{\mathbf{e}}_{1,n}} \mathrm{P}[\underline{\mathbf{e}}_{1,n}] \cdot \mathrm{P}[\underline{\mathbf{e}}'_{1,n'} \mid \underline{\mathbf{e}}_{1,n}] \quad . \tag{4.1}$$

The first factor, $\mathrm{P}[\underline{\mathbf{e}}'_{1,n'} \mid \underline{\mathbf{e}}_{1,n}]$, models the behavior of the channel. Following Brown *et al.*, we call this the *channel model*. The second factor, $\mathrm{P}[\underline{\mathbf{e}}_{1,n}]$, models the formation of English utterances by

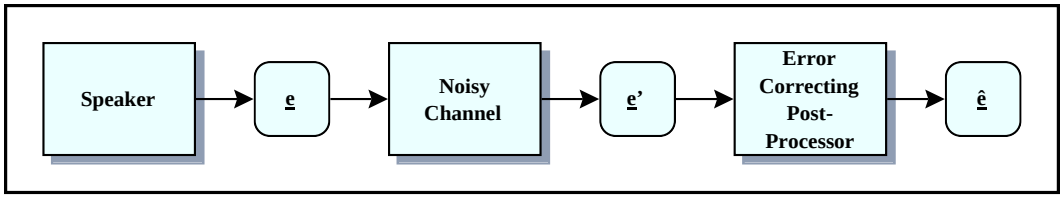


Figure 4.1: *Recovering Word-Sequences Corrupted in a Noisy Channel.*

the speaker. Brown *et al.* call this the *language model*; it is actually the listener's model of the speaker's language as in the formulation of statistical SR.

Word sequence $\underline{\mathbf{e}}'_{1,n'} = (w'_1, w'_2, \ldots, w'_{n'})$ is composed of words from a vocabulary known to the listener, and $\underline{\mathbf{e}}_{1,n} = (w_1, w_2, \ldots, w_n)$ contains possibly unknown words that the post-processor will be unable to recover. Furthermore, for a sizable vocabulary, adequately estimating the probability distributions that model the channel and the speaker's language requires mammoth amounts of data; therefore, it is necessary to approximate through independence assumptions. Several assumptions are possible, and we will begin with a basic set of assumptions before suggesting others.

First Approximation

For the language model, we assume as a first-cut that each word in $\underline{\mathbf{e}}_{1,n}$ is dependent only on its predecessor (a word bigram model):

$$\hat{\mathrm{P}}[\underline{\mathbf{e}}_{1,n}] = \prod_{i=0}^{n-1} \mathrm{P}[w_{i+1} \mid w_i] \quad . \tag{4.2}$$

Here, w_0 is the START symbol, so that $\mathrm{P}[w_1 \mid w_0]$ is the probability that word w_1 begins an utterance.

For the channel model, a first-cut independence assumption is that the production of each word in $\underline{\mathbf{e}}'_{1,n'}$ is only the transmitted version of the word with corresponding position in $\underline{\mathbf{e}}_{1,n}$. Thus,

$$\hat{\mathrm{P}}[\underline{\mathbf{e}}'_{1,n'} \mid \underline{\mathbf{e}}_{1,n}] = \prod_{i=1}^{n} \mathrm{P}[w'_i \mid w_i] \quad . \tag{4.3}$$

We say that a word is aligned with the word it produces.

As we saw in the introduction, function words in utterances are frequent victims of speech recognition errors. In all fairness to the speech recognizer, human performance in uttering function words is often sloppy even though function words are important for distinctions in meanings. Prepositions, articles, short verbs, pronouns, and conjunctions all contribute significantly to the meaning of spoken utterances in a task-oriented dialogue, yet they are often short and hidden between larger words. Function words are not the only victims of substitution errors on the part of SR. Content words are often affected as well.

We also require a method for searching among possible source utterances $\underline{\mathbf{e}}_{1,n}$ for the one that yields the greatest value of $\mathrm{P}[\underline{\mathbf{e}}_{1,n}] \cdot \mathrm{P}[\underline{\mathbf{e}}'_{1,n'} \mid \underline{\mathbf{e}}_{1,n}]$. The search method we propose builds possible source sequences $\underline{\mathbf{e}}_{1,n}$ one word at a time by relying on the order of words in $\underline{\mathbf{e}}'_{1,n'}$ and constructing

possible corrections using the one-for-one assumption. While building the source hypotheses, each hypothesis is scored according to the language model and the channel model so that the most promising hypotheses can be pursued first. To make the search efficient, dynamic programming on partial source sequence hypotheses is used in conjunction with pruning hypotheses falling below threshold.

Here is a toy example of how word substitution errors can be handled by this model and search algorithm. Consider a small vocabulary for the recognizer: $\{$*and*, *am*, *me*$\}$. We assume of course, for the sake of discussion, that the recognizer would have a tough time accurately recognizing utterances composed only of these words. We build a bigram language model by analyzing a corpus of transcribed utterances from this vocabulary and come up with Table 4.1. Each entry is of the form $\mathrm{P}[w_{i+1} \mid_{\text{follows}} w_i] = p$, indicating the probability that word w_{i+1} will follow w_i.

$\mathrm{P}[and \mid_{\text{follows}} \texttt{START}] = 0.85$
$\mathrm{P}[and \mid_{\text{follows}} and] = 0.01$
$\mathrm{P}[and \mid_{\text{follows}} am] = 0.2$
$\mathrm{P}[and \mid_{\text{follows}} me] = 0.79$
$\mathrm{P}[am \mid_{\text{follows}} \texttt{START}] = 0.1$
$\mathrm{P}[am \mid_{\text{follows}} and] = 0.33$
$\mathrm{P}[am \mid_{\text{follows}} am] = 0.33$
$\mathrm{P}[am \mid_{\text{follows}} me] = 0.33$
$\mathrm{P}[me \mid_{\text{follows}} \texttt{START}] = 0.05$
$\mathrm{P}[me \mid_{\text{follows}} and] = 0.8$
$\mathrm{P}[me \mid_{\text{follows}} am] = 0.05$
$\mathrm{P}[me \mid_{\text{follows}} me] = 0.15$

Table 4.1: *Bigram Language Model for Post-Processor Search Example.*

We run the same training utterances through the speech recognizer and analyze the one-for-one substitution errors made by the recognizer to come up with channel model in Table 4.2. Each entry is of the form $\mathrm{P}[w'_i \mid_{\text{replaces}} w_i] = p$, indicating the probability that word w_i is recognized as w'_i.

Now we speak the utterance "and me": $\underline{\mathbf{e}}_{1,n}$ = (*and*, *me*) (this is the source or pre-channel sequence). The recognizer thinks we said "am me": $\underline{\mathbf{e}}'_{1,n'}$ = (*am*, *me*) (this is the post-channel sequence). $\underline{\mathbf{e}}'_{1,n'}$ is passed to the post-processor, and the search algorithm builds the list of hypotheses and scores shown in Table 4.3 in an incremental fashion (we will not trace the incremental construction or the beam pruning here). Each row begins with a hypothesized word sequence (w_1, w_2) in the first column and an expression in the second column of the form

$$(\mathrm{P}[w'_1 \mid_{\text{replaces}} w_1] \times \mathrm{P}[w'_2 \mid_{\text{replaces}} w_2]) \times (\mathrm{P}[w_1 \mid_{\text{follows}} \texttt{START}] \times \mathrm{P}[w_2 \mid_{\text{follows}} w_1]) = p \quad . \tag{4.4}$$

This value is the probability that word sequence (w_1, w_2) is a valid correction according to our model. Thus, the search successfully finds the correct pre-channel word sequence, since it has the best score.

$\mathrm{P}[and \mid_{\text{replaces}} and] = 0.6$
$\mathrm{P}[and \mid_{\text{replaces}} am] = 0.39$
$\mathrm{P}[and \mid_{\text{replaces}} me] = 0.01$
$\mathrm{P}[am \mid_{\text{replaces}} and] = 0.3$
$\mathrm{P}[am \mid_{\text{replaces}} am] = 0.6$
$\mathrm{P}[am \mid_{\text{replaces}} me] = 0.1$
$\mathrm{P}[me \mid_{\text{replaces}} and] = 0.01$
$\mathrm{P}[me \mid_{\text{replaces}} am] = 0.1$
$\mathrm{P}[me \mid_{\text{replaces}} me] = 0.89$

Table 4.2: *Simple Channel Language Model for Post-Processor Search Example.*

Hypothesis Sequence	Score
(*and, and*)	$(0.3 \times 0.01) \times (0.85 \times 0.01) = 0.0000255$
(*and, am*)	$(0.3 \times 0.1) \times (0.85 \times 0.33) = 0.008415$
(*and, me*)	$(0.3 \times 0.89) \times (0.85 \times 0.8) = \mathbf{0.18156}$
(*am, and*)	$(0.6 \times 0.01) \times (0.1 \times 0.2) = 0.00012$
(*am, am*)	$(0.6 \times 0.1) \times (0.1 \times 0.33) = 0.00198$
(*am, me*)	$(0.6 \times 0.89) \times (0.1 \times 0.05) = 0.00267$
(*me, and*)	$(0.1 \times 0.01) \times (0.05 \times 0.79) = 0.0000395$
(*me, am*)	$(0.1 \times 0.1) \times (0.05 \times 0.33) = 0.000165$
(*me, me*)	$(0.1 \times 0.89) \times (0.05 \times 0.15) = 0.0006675$

Table 4.3: *Scored Hypotheses in Post-Processor Search Example.*

Enhancements to the Models

To improve the language model, we can use higher-order n-grams, thereby assuming that each word in $\underline{\mathbf{e}}_{1,n}$ is dependent on its $n-1$ predecessors. We can also use back-off n-gram models for combating the problem of sparse training data.

For the channel model, we relax the constraint that replacement errors be aligned on a word by word basis, since not all recognition errors consist of simple replacement of one word by another. Some errors appear as the break-up of one word into shorter words. Other errors involve the erroneous concatenation of two words to make a longer word. We will use utterance d95-3:utt17 from the collected TRAINS-95 dialogues as an example.

```
REF:  GO FROM CHICAGO TO TOLEDO
HYP:  GO FROM CHICAGO TO TO LEAVE AT
```

Following Brown *et al.*, we refer to a picture such as Figure 4.2 as an *alignment*. An alignment indicates the source words in the REF sequence for each of the words in the HYP sequence. For alignments, we use the following notation. We write the post-channel transcription ($\underline{\mathbf{e}}'_{1,n'}$) followed by the

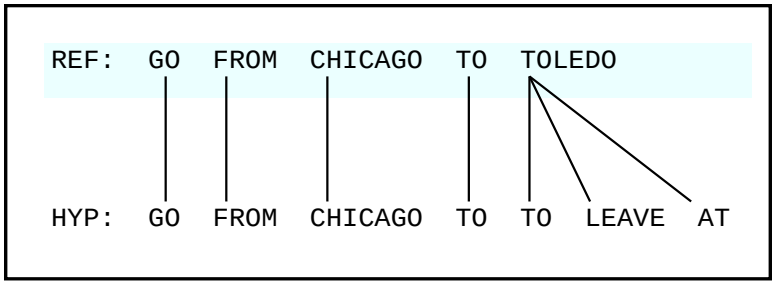


Figure 4.2: *Alignment of a Hypothesis and the Reference Transcription.*

pre-channel transcription ($\underline{\mathbf{e}}_{1,n}$) separated by a vertical bar and enclosed in parentheses. Following each of the pre-channel words, we provide its fertility in the current alignment in parentheses. Alignments are easily computed using a dynamic programming algorithm for word sequence alignment. Returning to d95-3:utt17 as an example, we have the alignment: `(GO FROM CHICAGO TO TO LEAVE AT | GO(1) FROM(1) CHICAGO(1) TO(1) TOLEDO(3))` We also refer to the number of post-channel words produced by a pre-channel word in a particular alignment as the *fertility* of that pre-channel word. The alignment problem in MT is potentially more convoluted than in our situation. In MT, words can appear quite far from the pre-channel word that produces them; Brown *et al.* refer to this problem as *distortion*; however, we will not need to concern ourselves with a model of this phenomenon since we can assume that order does not change.

To augment our channel model, we require a *fertility model* $\mathrm{P}[k \mid w]$ that indicates how likely each word w in the pre-channel vocabulary will have a particular fertility k. When a word's fertility k is an integer value between two and five, it indicates that the pre-channel word resulted in additional post-channel words. When a word's fertility is one, then the word accounts for exactly one post-channel word. When a word's fertility is a fraction $\frac{1}{n}$ (for $2 \leq n \leq 5$), then the word and $n-1$ neighboring words (for a total of n) have grouped together to result in a single post-channel word. I call this situation "fractional fertility." For example, $k = \frac{1}{3}$ indicates the situation in which three source words contribute to one word in the hypothesis; *i.e.*, each word accounts for one-third of the post-channel word. When a word's fertility is a fraction $\frac{m}{n}$ (for $2 \leq m \neq n \leq 5$), then the word and $n-1$ neighboring pre-channel words have grouped together to result in m post-channel words. The latter case can be used to handle segmentation errors. For example, $k = \frac{3}{2}$ indicates the situation in which two source words contribute to three words in the hypothesis; thus, we can imagine each word accounting for three-halves of the post-channel words. A concrete example of this alignment is `(TO LEAVE DOING | TOLEDO(3/2) IN(3/2))`.

To understand how fertility models are used, we need to extend the search algorithm sketched above. As before, the algorithm searches for an optimal source utterances $\underline{\mathbf{e}}_{1,n}$ in the maximum likelihood sense. This extended search builds possible sequences $\underline{\mathbf{e}}_{1,n}$ one word at a time using $\underline{\mathbf{e}}'_{1,n'}$ for guidance as before. Each word in $\underline{\mathbf{e}}'_{1,n'}$ is exploded (or collapsed with neighbors) using all possible combinations. The hypotheses are scored according to 1. the LM and 2. the channel model for one-for-one replacements or the fertility model for other kinds of replacements. As before, dynamic programming on partial source sentences and beam pruning will make the search efficient.

Observe that the fertility model scores only *the number of words* used to replace a particular word. It actually relies on the language model to score the contents of the replacement. This is

motivated by the related approach of Brown *et al.*, who appear to have taken this direction in order to avoid the problems of gathering statistics from hopelessly sparse data.

4.1.5 Preliminary Results

The first-draft technique has been implemented together with a bigram back-off language model. The existing corpus of TRAINS-95 dialogues was transcribed by Sphinx-II to yield post-channel transcriptions. The corpus and the post-channel corpus were divided into five partitions. Models were trained from four out of five of the partitions and tested on the remaining partition. This experiment was repeated for each of the five possible configurations in order to yield a cross-validated performance result. The performance results are depicted in Table 4.4. The first point

System	Word Accuracy	# Training Utterances	# Words
Sphinx II (alone)	69.0 %	—	—
Sphinx II & PP	73.048 % $\pm$ 0.606 %[1]	3313	20576

Table 4.4: *Preliminary Results for First Draft Post-Processor (PP).*

to note is the improvement in word-accuracy achieved by the post-processor over Sphinx II alone: over 4 %.

4.1.6 Other Issues

The question of whether there is a steep linear relationship between the amount of training data and improvement in word accuracy rates. This issue must be investigated further using existing dialogue data and new data as it is collected.

I must address the issue of what form the input and output of the post-processor should take. I will perform experiments involving the word-sequence and word-lattice representations with and without probabilistic score tags. For the word-lattice configuration, the post-processor and parser must be modified to process the alternatives in the lattice. One point to consider here is the width of the lattice (*i.e.*, the number of alternatives at a given point in the utterance). This factor can reflect the confidence of the SR in its hypotheses and may be useful as a parameter in the correction process.

Another option to be explored is whether the post-processor should receive input from the SR module on-line. The post-processor could also communicate with the parser in an incremental fashion.

In addition to the purely statistical mechanisms for recovering pre-channel word sequences outlined above, other cues may augment the search. For example, syllables and vowel nuclei may be usable for aligning pre-channel and post-channel words and phrases. Such alignments may be useful for further constraining the search algorithms and yielding better corrections.

Another possible area of investigation is whether the post-processor can be used for user adaptation. If the speech recognizer was not designed to adapt to a user as the user speaks, then the post-processor is a natural place to build this functionality.

4.2 Parsing Spontaneous Utterances

Various approaches have been used to parse spontaneous utterances robustly. They are motivated by the need to overcome the speech recognizer's transcription errors and the spontaneous performance of the speaker. (Note that these approaches do not correct the SR errors; they simply attempt to tolerate them.)

The robust parser in the TRAINS-95 system is an excellent example of the state of the art. It is able to extract useful speech-acts from input word sequences in the face of errors, as we have seen. However, its performance can be improved.

To further address the problem of dealing with spontaneous utterances while parsing, I will integrate two sources of information that have been shown to contribute to effective syntactic analysis: probabilistic scores and prosodic features. In this section, I first discuss the planned use of probabilistic scores briefly. Second, I discuss my plans for using prosodic features.

4.2.1 Probabilistic Scores

With the goal of robustly generating speech-acts, I note that in the TRAINS-95 parser, ambiguities arise that could be resolved through probabilistic preferences. For example, the parser can generate multiple utterance segmentations; however, only the first one to be formed (essentially a random choice) is used. To make the choice more rational, I will upgrade the parser to use 1. context-independent rule probabilities, 2. context-dependent lexical (category assignment) probabilities, and 3. word plausibility probabilities from the post-processor. I have already modified a version of the TRAINS-95 parser to invoke the forward algorithm on input sentences before parsing. The context-dependent lexical probabilities that it generates are used in conjunction with context-independent rule probabilities to yield the most likely parse first.

From the bottom-up perspective, a rule in the grammar carries a context-independent rule probability to indicate how likely its left-hand side will form the non-terminal on the right-hand side *without* regard to the surrounding context. Context-dependent lexical probabilities indicate how likely a word can be assigned to a particular lexical class (part-of-speech) while taking into consideration the left context (preceding words). Plausibility probabilities are simply the scores that we have already seen as they are generated by the post-processor. They indicate how likely a particular word-sequence is the true correction of the SR output. Thus, my investigation includes the use of the post-processor to jump-start the probabilistic parsing process.

4.2.2 Prosodic Features

With regard to prosody, simple prosodic cues such as silence, filled pauses, and noise words can be provided by the SR module. I will investigate how well these simple cues can be used to aid the post-processor. For example, I will use intervening silence to inhibit the post-processor from merging two separate words.

I will perform experiments to determine how useful the following criteria are in segmenting the input while parsing: prosodic phrasing and prominence, phrase-structure rules, probabilistic scores, and key-word spotting. In addition to the probabilistic techniques mentioned above, I will

augment the phrase-structure grammar with prosodic features and constituents to guide the parser in preferring more probable and prosodically plausible sequences of phrases.

I will wedge Wightman and Ostendorf's prosodic analysis tool between the error-correcting post-processor and the robust parser as shown in Figure 1.2. For a given utterance of spontaneous speech, this tool will take the acoustic waveform and the post-processor output to generate prosodic phrasing and prominences. Necessary modifications to the tool for handling spontaneous speech will be investigated, possibly with Colin Wightman at the New Mexico Institute of Mining and Technology. The output of the prosodic analysis will go to the parser, where prosodic constraints have been demonstrated to provide clear leverage in making syntactic disambiguation decisions.

4.3 Evaluation

Each step of the implementation must be validated through experiments. I will measure the ability of the modified TRAINS-95 system to perform on three levels:

1. Word recognition accuracy;
2. Speech-act generation accuracy;
3. End-to-end performance.

Word recognition accuracy will indicate how well the post-processor aids Sphinx-II (and to what extent the post-processor impedes recognition as well). Speech-act generation accuracy will measure whether the new components aid in successful interpretation. These metrics can be applied off-line using the dialogue database. The end-to-end performance benchmark will be useful for measuring whether the new components aid in the overall problem of spoken language understanding. As we discussed at the onset, understanding is measured in terms of useful responses. Quantitative measures such as how many turns a user requires to complete a task by conversing with the system may indicate how useful the system's responses are. This metric must be applied on-line in experiments with users, since conversations require a user's ability to respond to the system.

4.4 Data Collection

I will also continue to gather TRAINS-95 dialogues using version 1.0 (and higher) of the TRAINS-95 system. This version of the system does not require that a human segment the dialogue into utterances. I will also transcribe the utterances. This data will be used with the data already collected to build the TRAINS-95 language, channel, and word fertility models to be used by the post-processor. I will also continue to use the data for cross-validated performance tests.

4.5 Conclusions

This thesis will contribute models and methods for overcoming speech recognition errors and the incremental nature of spontaneous speech. These ends will be reached by the following means:

- a model and working implementation of a post-processor to correct recognition errors for an arbitrary third-party speech recognition system;
- modifications to the Wightman and Ostendorf prosodic analysis algorithm for robust behavior on spontaneous speech; and
- a prosody-wise probabilistic parser and grammar for handling the incremental nature of spontaneous speech in general settings.

One conceptual contribution of these models is evidence for the fact that modern speech recognition components can be used successfully as a black-box for robustly interpreting utterances in a dialogue with a human. A second contribution is that accurate spontaneous utterance segmentation can be facilitated by statistics, prosodic cues, and robust parsing techniques. The final contribution is a working conversational planning assistant capable of effective spoken language understanding.

Acknowledgments

I would like to thank my advisor, James Allen, for his help in selecting this thesis topic and for his guidance in focusing the exposition. I would also like to thank James, George Ferguson, and Brad Miller for their ongoing collaboration on the TRAINS-95 project (initially known as the "TOYTRAINS project"). Thanks also go to Len Schubert and Dean Richard Aslin for their helpful comments in conjunction with the presentation and defense of this proposal.

Special thanks go to Alex Acero, Fil Aleva, Bill Dolan, Olac Fuentes, Peter Heeman, George Heidorn, Xuedong Huang, Mei-Yuh Hwang, Karen Jensen, Andrew McCallum, Colm O'Riain, Joseph Pentheroudakis, Steve Richardson, Len Schubert, Sunil Issar, and Lucy Vanderwende, for helpful feedback on this and/or earlier related projects and for enlightening conversations. For Sphinx-II and the Statistical Language Modeling (SLM) Toolkit, thanks also go to Alex Rudnicky, Sunil Issar, Ron Rosenfeld, and all those who have contributed to Sphinx/Sphinx-II at CMU throughout the years.

I also wish to thank Brian Fuller for his help in transcribing many of the dialogues collected using TRAINS-95.

I reserve my most heartfelt thanks for my wife, Kirsti, for her unwavering love and support.

References

[Abney, 1991] Steven P. Abney, "Parsing by Chunks," In Robert C. Berwick, Steven P. Abney, and Carol Tenny, editors, *Principle-Based Parsing: Computation and Psycholinguistics*, pages 257–278. Kluwer Academic Publishers, Boston, 1991.

[Aho and Ullman, 1977] Alfred V. Aho and Jeffrey D. Ullman, *Principles of Compiler Design*, Addison-Wesley, Reading, Massachusetts, 1977.

[Allen, 1994] James Allen, *Natural Language Understanding, Second Edition*, Benjamin Cummings, 1994.

[Allen *et al.*, 1995] James F. Allen, George Ferguson, Brad Miller, and Eric Ringger, "Spoken Dialogue and Interactive Planning," In *Proceedings of the ARPA Spoken Language Systems Technology Workshop*, San Mateo California, January 1995. ARPA, Morgan Kaufmann.

[Allen and Perrault, 1980] James F. Allen and C. Raymond Perrault, "Analyzing Intention in Utterances," *Artificial Intelligence*, 15:143–178, 1980, (Reprinted in [Grosz *et al.*, 1986]).

[Allen *et al.*, 1994] James F. Allen, Lenhart K. Schubert, George Ferguson, Peter Heeman, Chung Hee Hwang, Tsuneaki Kato, Marc Light, Nathaniel G. Martin, Bradford W. Miller, Massimo Poesio, and David R. Traum, "The TRAINS Project: A case study in defining a conversational planning agent," TRAINS Technical Note 94-3, Department of Computer Science, University of Rochester, Rochester, NY, 14627, 1994, To appear in the Journal of Experimental and Theoretical AI, 1995.

[ARPA, 1995] ARPA, editor, *Proceedings of the ARPA Spoken Language Systems Technology Workshop*, San Mateo, California, January 1995. ARPA, Morgan Kaufmann.

[Austin, 1962] J. L. Austin, *How to Do Things with Words*, Oxford University Press, New York, 1962.

[Bahl *et al.*, 1983] L. R. Bahl, F. Jelinek, and R. Mercer, "A Maximum Likelihood Approach to Continuous Speech Recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 5(2):179–190, March 1983.

[Bahl *et al.*, 1989] Lalit R. Bahl, Peter F. Brown, Peter V. de Souza, and Robert L. Mercer, "A Tree-Based Statistical Language Model for Natural Language Processing," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 37(7):1001–1008, July 1989, Reprinted in [Waibel and Lee, 1990]: 507-514.

[Baker, 1979] James K. Baker, "Trainable Grammars for Speech Recognition," In Jared J. Wolf and Dennis H. Klatt, editors, *Speech Communication Papers for the 97th Meeting of the Acoustical Society of America*, pages 547–550, Cambridge, Massachusetts, 1979. Acoustical Society of America, MIT.

[Ball and Ling, 1994] J. Eugene Ball and Daniel T. Ling, "Spoken Language Processing in the Persona Conversational Agent," Submitted to UIST '94, April 1994.

[Bates *et al.*, 1993] M. Bates, R. Bobrow, P. Fung, R. Ingria, F. Kubala, J. Makhoul, L. Nguyen, and R. Schwartz, "The BBN/HARC Spoken Language Understanding System," In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 111–114. IEEE, 1993.

[Baum, 1972] Leonard E. Baum, "An Inequality and Associated Maximization Technique in Statistical Estimation for Probabilistic Functions of Markov Processes," In Oved Shisha, editor, *Inequalities–III (September 1969)*, pages 1–8, New York, 1972. Academic Press.

[BBN HARK Systems, 1994] BBN HARK Systems, "The Hark Recognizer," World Wide Web, 1994, Universal Resource Locator: `http://www.bbn.com/hark.html`.

[Bellman, 1957] Richard E. Bellman, *Dynamic Programming*, Princeton University Press, 1957.

[Biermann *et al.*, 1985] A. Biermann, R. Rodman, D. Rubin, and J. Heidlage, "Natural Language with Discrete Speech as a Mode for Human-to-Machine Communication," *Communications of the ACM*, 28(6), 1985.

[Black *et al.*, 1993] E. Black, F. Jelinek, J. Lafferty, D. Magerman, R. Mercer, and S. Roukos, "Towards History-based Grammars: Using Richer Models for Probabilistic Parsing," In *Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics*, pages 31–37. ACL, 1993.

[Brown and Yule, 1983] Gillian Brown and George Yule, *Discourse Analysis*, Cambridge University Press, Cambridge, 1983.

[Brown *et al.*, 1990] Peter F. Brown, John Cocke, Stephen A. Della Pietra, Vincent J. Della Pietra, Fredrick Jelinek, John D. Lafferty, Robert L. Mercer, and Paul S. Roossin, "A Statistical Approach to Machine Translation," *Computational Linguistics*, 16(2):79–85, June 1990.

[Brown *et al.*, 1992a] Peter L. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, Jennifer C. Lai, and Robert L. Mercer, "An Estimate of an Upper Bound for the Entropy of English," *Computational Linguistics*, 18(1):31–40, March 1992.

[Brown *et al.*, 1992b] Peter L. Brown, Peter V. deSouza, Robert L. Mercer, Vincent Della Pietra, and Jenifer C. Lai, "Class-Based *n*-gram Models of Natural Language," *Computational Linguistics*, 18(4):467–479, December 1992.

[Bruce, 1975] Bertram C. Bruce, "Generation as a Social Action," In *Theoretical Issues in Natural Language Processing (TINLAP-1)*, pages 64–67, 1975, (Reprinted in [Grosz *et al.*, 1986]).

[Carbonell and Hayes, 1983] J. Carbonell and P. Hayes, "Recovery Strategies for Parsing Extragrammatical Language," *AJCL*, 9(3-4):123–146, 1983.

[Charniak, 1993] Eugene Charniak, *Statistical Language Learning*, MIT Press, Cambridge, Massachusetts, 1993.

[Chomsky, 1957] Noam Chomsky, *Syntactic Structures*, Mouton, 1957.

[Chow *et al.*, 1987] Y. L. Chow, M. O. Dunham, O. A. Kimball, M. A. Krasner, G. F. Kubala, J. Makhoul, S. Roucos, and R. M. Schwartz, "BYBLOS: The BBN Continuous Speech Recognition System," In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 89–92. IEEE, April 1987, Reprinted in [Waibel and Lee, 1990]: 596-599.

[Chow and Schwartz, 1989] Y.S. Chow and R. Schwartz, "The N-Best Algorithm: An Efficient Procedure for Finding Top N Sentence Hypotheses," In *Proceedings of the Second DARPA Workshop on Speech and Natural Language*, pages 199–202, San Mateo, California, October 1989. DARPA, Morgan Kaufmann.

[Church and Mercer, 1993] Kenneth W. Church and Robert L. Mercer, "Introduction to the Special Issue on Computational Linguistics Using Large Corpora," *Computational Linguistics*, 19(1):1–24, March 1993.

[Church and Patil, 1982] Kenneth W. Church and Ramesh Patil, "Coping with Syntactic Ambiguity or How to Put the Block in the Box on the Table," *Computational Linguistics*, 8(3-4):139–149, 1982.

[Cohen and Perrault, 1979] Philip R. Cohen and C. Raymond Perrault, "Elements of a Plan-based Theory of Speech Acts," *Cognitive Science*, 3(3):177–212, 1979, (Reprinted in [Grosz *et al.*, 1986]).

[Dowding *et al.*, 1993] J. Dowding, J. Gawron, D. Appelt, J. Bear, L. Cherny, R. Moore, and D. Moran, "Gemini: A Natural Language System for Spoken-Language Understanding," In *Proceedings of the ARPA Human Language Technology Workshop*, pages 43–48, San Mateo, California, March 1993. ARPA, Morgan Kaufmann.

[Earley, 1970] Jay Earley, "An Efficient Context-Free Parsing Algorithm," *Communications of the ACM*, 13(2):94–102, February 1970, Reprinted in [Grosz *et al.*, 1986].

[Ferguson, 1995] George Ferguson, *Knowledge Representation and Reasoning for Mixed-Initiative Planning*, PhD thesis, University of Rochester, Rochester, NY, 14627, 1995, Technical Report 562.

[Fink and Biermann, 1986] Pamela K. Fink and Alan W. Biermann, "The Correction of Ill-Formed Input using History-Based Expectation with Applications to Speech Recognition," *Computational Linguistics*, 12(1):13–36, January-March 1986.

[G. E. Forney, 1973] Jr. G. E. Forney, "The Viterbi Algorithm," In *Proceedings of IEEE*, volume 61, pages 266–278. IEEE, 1973.

[Garside and Leech, 1987] Roger Garside and Geoffrey Leech, *The Computational Analysis of English*, Longman, 1987.

[Gee and Grosfean, 1983] J. Gee and F. Grosfean, "Saying What You Mean in Dialogue: A Study in Conceptual and Semantic Co-ordination," *Cognitive Psychology*, 15(3), 1983.

[Gee and Grosjean, 1983] J. Gee and F. Grosjean, "Performance Structures: a Psycholinguistic and Linguistic Appraisal," *Cognitive Psychology*, 15:411–458, 1983.

[Glass *et al.*, 1995] J. Glass, D. Goddeau, L. Hetherington, M. McCandless, C. Pao, M. Phillips, J. Polifroni, S. Seneff, and V. Zue, "The MIT ATIS System: December 1994 Progress Report," In *Proceedings of the ARPA Spoken Language Systems Technology Workshop*. ARPA, ARPA, January 1995.

[Gray, 1984] R. M. Gray, "Vector Quantization," *IEEE ASSP Magazine*, 1(2):4–29, April 1984, Reprinted in [Waibel and Lee, 1990].

[Grice, 1957] H. P. Grice, "Meaning," *Philosophical Review*, 66:377–388, 1957.

[Gross *et al.*, 1992] D. Gross, J. Allen, and D. Traum, "The TRAINS 91 Dialogues," TRAINS TECHNICAL NOTE 92-1, Department of Computer Science, University of Rochester, Rochester, New York, 1992.

[Grosz *et al.*, 1986] B. Grosz, K. Sparck Jones, and B. Webber, editors, *Readings in Natural Language Processing*, Morgan Kaufmann, San Mateo, 1986.

[Haas, 1989] Andrew Haas, "A Parsing Algorithm for Unification Grammar," *Computational Linguistics*, 15(4):219–232, 1989.

[Halliday, 1967] M. A. Halliday, "Notes on Transitivity and Theme in English: Part 2," *Journal of Linguistics*, 3, 1967.

[Hanazawa *et al.*, 1990] T. Hanazawa, K. Kita, S. Nakamura, T. Kawabata, and K. Shikano, "ATR HMM-LR Continuous Speech Recognition System," In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 53–56. IEEE, 1990.

[Harper *et al.*, 1994] M. P. Harper, L. H. Jamieson, C. D. Mitchell, G. Ying, S. Potisuk, P. N. Srinivasan, R. Chen, C. B. Zoltowski, L. L. McPheters, B. Pellom, and R. A. Helzerman, "Integrating Language Models with Speech Recognition," In *Procedings of the Workshop on the Integration of Natural Language and Speech Processing at the 12th Annual Natuional Conference on Artificial Intelligence*. AAAI, July 1994.

[Hauenstein and Weber, 1994] Andreas Hauenstein and Hans Weber, "An Investigation of Tightly Coupled Time Synchronous Speech Language Interfaces Using a Unification Grammar," In Paul McKevitt, editor, *Proceedings of the AAAI '94 Workshop on Integration of Natural Language and Speech Processing*, pages 42–49, Seatle, Washington, July 1994. AAAI.

[Heeman and Allen, 1994a] P. A. Heeman and J. Allen, "Dialogue Transcription Tools," (Cited in [Heeman and Allen, 1994b]), 1994.

[Heeman and Allen, 1994b] P. A. Heeman and J. Allen, "Tagging Speech Repairs," In *Proceedings ARPA Human Language Technology Workshop*, pages 182–187. ARPA, March 1994.

[Heeman and Allen, 1995] Peter Heeman and James F. Allen, "The TRAINS 93 Dialogues," TRAINS Technical Note 94-2, Department of Computer Science, University of Rochester, Rochester, NY, 14627, March 1995.

[Hindle and Rooth, 1993] D. Hindle and M. Rooth, "Structural Ambiguity and Lexical Relations," *Computational Linguistics*, 19(1):103–120, March 1993.

[Hobbs *et al.*, 1993] J. R. Hobbs, D. E. Appelt, J. S. Bear, D. J. Israel, and W. M. Tyson, "FASTUS: A System for Extracting Information from Natural-Language Text," Technical Note 519, SRI, Palo Alto, California, 1993.

[Hopcroft and Ullman, 1979] John E. Hopcroft and Jeffrey D. Ullman, *Introduction to Automata Theory, Languages and Computation*, Addison-Wesley, Reading, Massachusetts, 1979.

[Huang *et al.*, 1992] X. D. Huang, F. Alleva, H. W. Hon, M. Y. Hwang, K. F. Lee, and R. Rosenfeld, "The Sphinx-II Speech Recognition System: An Overview," *Computer, Speech and Language*, 1992.

[Huang *et al.*, 1993a] X. D. Huang, F. Alleva, H. W. Hon, M. Y. Hwang, K. F. Lee, and R. Rosenfeld, "The SPHINX-II Speech Recognition System: An Overview," *Computer, Speech and Language*, 1993.

[Huang *et al.*, 1993b] X. D. Huang, F. Alleva, M. Y. Hwang, and R. Rosenfeld, "An Overview of the SPHINX-II Speech Recognition System," In *Proceedings of the ARPA Human Language Technology Workshop*, San Mateo, California, March 1993. ARPA, Morgan Kaufmann.

[Huang and Jack, 1989] X. D. Huang and M. A. Jack, "Semi-continuous hidden Markov models for speech signals," *Computer Speech and Language*, 3:239–251, 1989.

[Huang *et al.*, 1995] Xuedong Huang, Alex Acero, Fil Alleva, Mei-Yuh Hwang, Li Jiang, and Milind Mahajan, "Microsoft Windows Highly Intelligent Speech Recognizer: WHISPER," In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*. IEEE, 1995, To Appear.

[Hull, 1992] Jonathan J. Hull, "Combining Syntactic Knowledge and Visual Text Recognition: A Hidden Markov Model for Part of Speech Tagging In a Word Recognition Algorithm," In *Proceedings of AAAI Fall Symposium on Probabilistic Approaches to Natural Language Processing*, pages 77–83. AAAI, 1992.

[Jelinek, 1985] Fred Jelinek, "The Development of an Experimental Discrete Dictation Recognizer," In *Proceedings of the IEEE*, pages 1616–1624, November 1985, Reprinted in [Waibel and Lee, 1990]: 587-595.

[Jelinek, 1990] Fred Jelinek, "Self-Organized Language Modeling for Speech Recognition," Reprinted in [Waibel and Lee, 1990]: 450-506, 1990.

[Jensen *et al.*, 1983] K. Jensen, G. Heidorn, L. Miller, and Y. Ravin, "Parse Fitting and Prose Fixing: Getting a Hold on Ill-Formedness," *AJCL*, 9(3-4):147–160, 1983.

[Jensen *et al.*, 1993] Karen Jensen, George Heidorn, and Stephen Richardson, editors, *Natural Language Processing: the PLNLP Approach*, Kluwer Academic Publishers, Dordrecht, Netherlands, 1993.

[Jurafsky *et al.*, 1994] Daniel Jurafsky, Chuck Wooters, Gary Tikeman, Jonathan Segal, Andreas Stolcke, and Nelson Morgan, "Integrating Experimental Models of Syntax, Phonology, and Accent/Dialect in a Speech Recognizer," In Paul McKevitt, editor, *Proceedings of the AAAI '94*

Workshop on Integration of Natural Language and Speech Processing, pages 107–115, Seatle, Washington, July 1994. AAAI, AAAI.

[Katz, 1987] Slava M. Katz, "Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognizer," In *IEEE Transactions on Acoustics, Speech, and Signal Processing*, pages 400–401. IEEE, IEEE, March 1987.

[Keenan *et al.*, 1991] F. G. Keenan, L. J. Evett, and R. J. Whitrow, "A Large Vocabulary Stochastic Syntax Analyser for Handwriting Recognition," In *Proceedings of the First International Conference on Document Analysis*, pages 794–802. IDCAR, 1991.

[Kita *et al.*, 1989] K. Kita, T. Kawabata, and H. Saito, "HMM Continuous Speech Recognition Using Predictive LR Parsing," In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 703–706. IEEE, 1989.

[Klatt, 1977] Dennis H. Klatt, "Review of the ARPA Speech Understanding Project," *The Journal of the Accoustical Society of America*, 62(6):1324–1366, December 1977, Reprinted in [Waibel and Lee, 1990]: 554-575.

[Kupiec and Maxwell, 1992] J. Kupiec and J. Maxwell, "Training Stochastic Grammars from Unlabelled Text Corpora," In *AAAI-92 Workshop Program on Statistically-Based NLP Techniques*, pages 14–19, San Jose, CA, 1992. AAAI.

[Lari and Young, 1990] K. Lari and S.J. Young, "The Estimation of Stochastic Context-Free Grammars using the Inside-Outside Algorithm," *Computer Speech and Language*, 4:35–56, 1990.

[Lavie, 1994] Alon Lavie, "An Integrated Heuristic Scheme for Partial Parse Evaluation," In *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics*, pages 316–318. ACL, June 1994.

[Lee, 1990] K. F. Lee, "Context Dependent Phonetic Hidden Markov Models for Continuous Speech Recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, April 1990, Reprinted in [Waibel and Lee, 1990].

[Lee *et al.*, 1990] K. F. Lee, H. W. Hon, and R. Reddy, "An Overview of the SPHINX Speech Recognition System," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, January 1990, Reprinted in [Waibel and Lee, 1990].

[Lee, 1989] Kai-Fu Lee, *Automatic Speech Recognition: the Development of the SPHINX System*, Kluwer Academic, Boston, London, 1989.

[Lesser *et al.*, 1975] V. R. Lesser, R. D. Fennell, L. D. Erman, and D. R. Reddy, "Organization of the Hearsay-II Speech Understanding System," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 23:11–23, 1975.

[Levinson *et al.*, 1983] S.E. Levinson, L.R. Rabiner, and M.M. Sondhi, "An Introduction to the Application of the Theory of Probabilistic Functions of a Markov Process to Automatic Speech Recognition," *The Bell System Technical Journal*, 62(4), April 1983.

[Lowerre and Reddy, 1986] Bruce Lowerre and Raj Reddy, "The Harpy Speech Understanding System," In *Trends in Speech Recognition*. Speech Science Publications, Apple Valley, Minnesota, 1986, Reprinted in [Waibel and Lee, 1990]: 576-586.

[Luenberger, 1979] David G. Luenberger, *Introduction to Dynamic Systems: Theory, Models, and Applications*, John Wiley and Sons, New York, 1979.

[Magerman, 1994] David M. Magerman, *Natural Language Parsing as Statistical Pattern Recognition*, PhD thesis, Stanford University, Palo Alto, California, February 1994.

[Magerman, 1995] David M. Magerman, "Statistical Decision-Tree Models for Parsing," In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*. ACL, 1995.

[Martin and Kehler, 1994] Paul Martin and Andrew Kehler, "SpeechActs: A Testbed for Continuous Speech Applications," In Paul McKevitt, editor, *Proceedings of the AAAI '94 Workshop on Integration of Natural Language and Speech Processing*, pages 65–71, Seatle, Washington, July 1994. AAAI, AAAI.

[Moore *et al.*, 1995] R. Moore, D. Appelt, J. Dowding, J. Gawron, and D. Moran, "Combining Linguistic and Statistical Knowledge Sources in Natural-Language Processing for ATIS," In *Proceedings of the ARPA Spoken Language Systems Technology Workshop*. ARPA, ARPA, January 1995.

[Ney, 1991] Hermann Ney, "Dynamic Programming Parsing for Context-Free Grammars in Continuous Speech Recognition," *IEEE Transactions on Signal Processing*, 39(2), February 1991.

[Nguyen *et al.*, 1993] Long Nguyen, Richard Schwartz, Francis Kubala, and Paul Placeway, "Search Algorithms for Software-Only Real-Time Recognition with Very Large Vocabularies," In *Proceedings of the ARPA Human Language Technology Workshop*, San Mateo, California, March 1993. ARPA, Morgan Kaufmann.

[Ostendorf *et al.*, 1992] M. Ostendorf, A. Kannan, S. Austin, O. Kimball, R. Schwartz, and J. R. Rohlicek, "Integration of Diverse Recognition Methodologies Through Reevaluation of N-Best Sentence Hypotheses," In *Proceedings ARPA Human Language Technology Workshop*, pages 83–87, San Mateo, California, March 1992. ARPA, Morgan Kaufmann.

[Ostendorf *et al.*, 1993] M. Ostendorf, C. Wightman, and N. Veilleux, "Parse scoring with Prosodic Information: An Analysis/Synthesis Approach," *Computer Speech and Language*, 7(2), 1993.

[Oviatt, 1994] Sharon L. Oviatt, "Interface Techniques for Minimizing Disfluent Input to Spoken Language Systems," In *Proceedings of CHI '94*, pages 205–210, Boston, Mass., 1994. ACM Press.

[Paeseler, 1988] Annedore Paeseler, "Modification of Earley's Algorithm for Speech Recognition," In H. Niemann *et al.*, editor, *Recent Advances in Speech Understanding and Dialog Systems*. Springer-Verlag, Berlin, 1988, Reprinted in [Waibel and Lee, 1990]: 515-518.

[Pereira and Wright, 1991] Fernando C. N. Pereira and Rebecca N. Wright, "Finite-State Approximation of Phrase Structure Grammars," In *Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics*, pages 246–255. ACL, 1991.

[Pierrehumbert and Hirschberg, 1990] Janet Pierrehumbert and Julia Hirschberg, "The Meaning of Intonational Contours in the Interpretation of Discourse," In Philip R. Cohen, Jerry Morgan, and Martha E. Pollack, editors, *Intentions in Communication*, pages 271–312. MIT Press, Cambridge, Massachusetts, 1990.

[Poritz, 1988] A. B. Poritz, "Hidden Markov Models: A Guided Tutorial," *Proceedings of IEEE ICASSP*, pages 7–13, April 1988.

[Rabiner and Juang, 1986] L. R. Rabiner and B. H. Juang, "An Introduction to Hidden Markov Models," *IEEE ASSP Magazine*, 3:4–16, January 1986.

[Rabiner and Juang, 1993] Lawrence Rabiner and Biing-Hwang Juang, *Fundamentals of Speech Recongition*, PTR Prentice Hall (Signal Processing Series), Englewood Cliffs, NJ, 1993.

[Rabiner, 1989] Lawrence R. Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," *Proceedings of the IEEE*, 1989, Reprinted in [Waibel and Lee, 1990]: 267-296.

[Rayner *et al.*, 1994] Manny Rayner, David Carter, Vassilios Digalakis, and Patti Price, "Combining Knowledge Sources to Reorder N-best Speech Hypothesis Lists," In *Proceedings ARPA Human Language Technology Workshop*, pages 212–217. ARPA, March 1994.

[Richardson, 1994] Stephen Richardson, "Bootstrapping Statistical Processing into a Rule-based Natural Language Parser," Submitted to ACL-94, February 1994.

[Rosenfeld, 1995] Ronald Rosenfeld, "The CMU Statistical Language Modeling Toolkit and its use in the 1994 ARPA CSR Evaluation," In *Proceedings of the ARPA Spoken Language Systems Technology Workshop*, San Mateo, California, January 1995. ARPA, Morgan Kaufmann.

[Rowles and Huang, 1992] Chris Rowles and Xiuming Huang, "Prosodic Aids to Syntactic and Semantic Analysis of Spoken English," In *Proceedings of the 30st Annual Meeting of the Association for Computational Linguistics*, pages 112–119. ACL, 1992.

[Rozanov, 1977] Y. A. Rozanov, *Probability Theory: A Concise Course*, Dover Publications, New York, 1977.

[Rudnicky *et al.*, 1994] Alexander I. Rudnicky, Alexander G. Hauptmann, and Kai-Fu Lee, "Survey of Current Speech Technology," *Communications of the ACM*, 37(3):52–57, 1994.

[Sakoe and Chiba, 1978] Hiroaki Sakoe and Seibi Chiba, "Dynamic Programming Algorithm Optimization for Spoken Word Recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 26(1):43–49, February 1978, Reprinted in [Waibel and Lee, 1990].

[Schwartz and Austin, 1991] R. Schwartz and S. Austin, "A Comparison of Several Approximate Algorithms for Finding Multiple n-best Sentence Hypotheses," In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*. IEEE, May 1991.

[Schwartz *et al.*, 1993] R. Schwartz, S. Austin, F. Kubala, J. Makhoul, L. Nguyen, and P. Placeway, "New Uses for the n-best Sentence Hypotheses within the Byblos Speech Recognition System," In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*. IEEE, April 1993.

[Schwartz and Chow, 1990] R. Schwartz and Y-L. Chow, "The n-best Algorithm: An Efficient and Exact Procedure for Finding the n Most Likely Sentence Hypotheses," In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, April 1990.

[Searle, 1969] J. R. Searle, *Speech Acts: An Essay in the Philosophy of Language*, Cambridge University Press, Cambridge, 1969.

[Seneff, 1992] Stephanie Seneff, "TINA: A Natural Language System for Spoken Language Applications," *Computational Linguistics*, 18(1):61–86, March 1992.

[Shannon, 1948] Claude Shannon, "The Mathematical Theory of Communication," *Bell Systems Technical Journal*, 30:50–64, 1948.

[Shieber, 1985] Stuart M. Shieber, "An Introduction to Unification-Based Approaches to Grammar," In *CSLI Lecture Notes*, page No. 4. Chicago University Press, Center for the Study of Language and Information, Stanford, California, 1985.

[Soule, 1974] Stephen Soule, "Entropies of Probabilistic Grammars," *Information and Control*, 25:57–74, 1974.

[Spivey-Knowlton *et al.*, 1994] Michael Spivey-Knowlton, Julie Sedivy, Kathleen Eberhard, and Michael Tanenhaus, "Psycholinguistic Study of the Interaction Between Language and Vision," In Paul McKevitt, editor, *AAAI 1994 Workshop on the Integration of Natural Language and Vision Processing*, pages 189–192. AAAI, August 1994.

[Srihari and Baltus, 1993] Rohini K. Srihari and Charlotte M. Baltus, "Incorporating Syntactic Constraints in Recognizing Handwritten Sentences," In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI-93)*, pages 1262–1267, Chambery, France, August 1993.

[Stallard and Bobrow, 1992] David Stallard and Robert Bobrow, "Fragment Processing in the DELPHI System," In *Proceedings of the Speech and Natural Language Workshop*, pages 305–310, San Mateo, California, February 1992. DARPA, Morgan Kaufmann.

[Taylor and Karlin, 1994] Howard M. Taylor and Samuel Karlin, *An Introduction to Stochastic Modeling*, Academic Press, Boston, 1994.

[Thorne *et al.*, 1968] J. Thorne, P. Bratley, and H. Dewar, "The Syntactic Analysis of English by Machine," In D. Michie, editor, *Machine Intelligence 3*. Edinburgh University, Edinburgh, 1968.

[Veilleux and Ostendorf, 1993] N. Veilleux and M. Ostendorf, "Prosody/Parse Scoring and its Application in ATIS," In *Proceedings of the ARPA Human Language Technology Workshop*, pages 335–340, San Mateo, California, March 1993. ARPA, Morgan Kaufmann.

[Viterbi, 1967] Andrew J. Viterbi, "Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm," *IEEE Transactions on Information Theory*, IT-13(2):260–69, April 1967.

[Waibel *et al.*, 1989] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K. Lang, "Phoneme Recognition Using Time-Delay Neural Networks," In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*. IEEE, May 1989, Reprinted in [Waibel and Lee, 1990].

[Waibel, 1987] Alex Waibel, "Prosodic Knowledge Sources for Word Hypothesization in a Continuous Speech Recognition System," In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 20.16.1–20.16.4. IEEE, 1987, Reprinted in [Waibel and Lee, 1990]: 534-537.

[Waibel, 1988] Alex Waibel, *Prosody and Speech Recognition*, Morgan Kaufmann, San Mateo, California, 1988.

[Waibel and Lee, 1990] Alex Waibel and Kai-Fu Lee, editors, *Readings in Speech Recognition*, Morgan Kaufmann, San Mateo, 1990.

[Walker, 1978] D. Walker, editor, *Understanding Spoken Language*, Elsevier North-Holland, New York, New York, 1978.

[Ward and Issar, 1995] W. Ward and S. Issar, "The CMU ATIS System," In *Proceedings of the ARPA Spoken Language Systems Technology Workshop*. ARPA, ARPA, January 1995.

[Ward, 1989] Wayne Ward, "Modeling Non-verbal Sounds for Speech Recognition," In *Proceedings of the Second DARPA Workshop on Speech and Natural Language*, pages 47–50, San Mateo, California, October 1989. DARPA, Morgan Kaufmann.

[Ward, 1990] Wayne Ward, "The CMU Air Travel Information Service: Understanding Spontaneous Speech," In *Proceedings of the DARPA Speech and Natural Language Workshop*, pages 127–129, San Mateo, California, June 1990. DARPA, Morgan Kaufmann.

[Weischedel *et al.*, 1993] R. M. Weischedel, M. Meteer, R. Schwartz, L. Ramshaw, and J. Palmucci, "Coping with Ambiguity and Unknown Words Through Probabilistic Models," *Computational Linguistics*, 19(2), 1993.

[Wightman, 1992] C. Wightman, "Segmental Durations in the Vicinity of Prosodic Phrase Boundaries," *Journal of the Acoustic Society of America*, March 1992.

[Wightman and Ostendorf, 1994] C. Wightman and M. Ostendorf, "Automatic Labeling of Prosodic Patterns," *IEEE Transactions on Speech and Audio Processing*, 2(4):469–481, October 1994.

[Wolf and Woods, 1980] J. Wolf and W. Woods, "The HWIM Speech Understanding System," *Lea*, pages 316–339, 1980.

[Woodland *et al.*, 1995] P. Woodland, C. Leggetter, J. Odell, V. Valtchev, and S. J. Young, "The Development of the 1994 HTK Large Vocabulary Speech Recognition System," In *Proceedings of the ARPA Spoken Language Systems Technology Workshop*. ARPA, ARPA, January 1995.

[Woods, 1970] W. Woods, "Transition Network Grammars for Natural Language Analysis," *Communications of the ACM*, 13(10):591–606, 1970.

[Woods, 1983] W. A. Woods, "Language Processing for Speech Understanding," In *Computer Speech Processing*. Prentice-Hall International, Hemel Hempstead, England, 1983, Reprinted in [Waibel and Lee, 1990]: 519-533.

[Yankelovich, 1994] Nicole Yankelovich, "SpeechActs & The Design of Speech Interfaces," In *Proceedings of the CHI '94 Workshop on the Future of Speech and Audio in the Interface.* ACM, April 1994.

[Yankelovich *et al.*, 1995] Nicole Yankelovich, Gina-Anne Levow, and Matt Marx, "Designing SpeechActs: Issues in Speech User Interfaces," In *Proceedings of the CHI '95 Conference on Human Factors in Computing.* ACM, May 1995.

[Young and Ward, 1993] Sheryl R. Young and Wayne Ward, "Semantic and Pragmatically Based Re-recognition of Spontaneous Speech," In *Proceedings Eurospeech 1993*, 1993.

[Young and Woodland, 1993] Stephen J. Young and Philip C. Woodland, *HTK: Hidden Markov Model Toolkit*, Entropic Research Laboratory, Washington, D.C., 1993.