Subword Lexical Modelling for Speech Recognition

by

Raymond Lau

S.M., Massachusetts Institute of Technology (1994)
S.B., Massachusetts Institute of Technology (1993)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 1998

Author .....
Department of Electrical Engineering and Computer Science
March 31, 1998

Certified by .....
Stephanie Seneff
Principal Research Scientist
Department of Electrical Engineering and Computer Science

Accepted by .....
Arthur Smith
Chairman, Departmental Committee on Graduate Students

Subword Lexical Modelling for Speech Recognition

by

Raymond Lau

Submitted to the Department of Electrical Engineering and Computer Science
on March 31, 1998, in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

Abstract

In this work, we introduce and develop a novel framework, ANGIE, for modelling subword lexical phenomena in speech recognition. Our framework provides a flexible and powerful mechanism for capturing morphology, syllabification, phonology, and other subword effects in a hierarchical manner which maximizes sharing of subword structures. ANGIE models the subword structure within a context-free grammar and an accompanying probability model. We believe that our framework has several advantages: The sharing mechanism allows training data to be pooled amongst instances of the same word substructure even when they occur across different words in the lexicon. Further, knowledge of this substructure can be extended to filler models in a word-spotter, new words added incrementally to a recognizer's vocabulary, and potentially in support of new word detection. The context-free foundation allows for ease of research and experimentation with varying subword representations, and also facilitates integration with a natural language understanding system. Finally, the availability of subword structural information in a recognition system enables exploration of prosodic models which use this information.

In this thesis, we demonstrate ANGIE's feasibility and efficacy in a variety of applications. Using ATIS corpus data, we show that ANGIE results in performance improvements on phonetic recognition, reducing error rate from 39.8% to 36.1% as compared to a phone bigram baseline. We show its competitiveness in the task of word-spotting, where we also report on a comparative study of different subword lexical models for the filler space. The FOM results ranged from 85.3 for a phone bigram to 89.3 for a system using the full ANGIE parse tree and a lexicon of 1200 words. We also discuss an implementation of a competitive continuous speech recognition system based on ANGIE, which achieves a recognition error rate of 18.8% on our test set as compared to a baseline error rate of 18.9%, both using a word bigram. Finally, we explore the integration of ANGIE with a natural language understanding system, resulting in a fully coupled system, based on context-free frameworks for both phonological and linguistic modelling. The integrated system achieves a recognition error rate of 14.8% on the same test, an improvement of 21.6%. We will also discuss two pilot studies, one on handling dynamic vocabulary updates within a continuous speech recognizer and the second on hierarchical duration modelling within a word-spotter. Both studies showed promising results.

Thesis Supervisor: Stephanie Seneff
Title: Principal Research Scientist

Acknowledgments

Much credit for the underlying design of the ANGIE subword lexical modelling framework that is the focus of this thesis belongs to my thesis supervisor, Stephanie Seneff. Originally inspired by the doctoral dissertation work of Lee Hetherington, a member of the Spoken Language Systems Group back when I first joined as a graduate student in 1994 and still a member today, albeit not in one continuous sitting, I sought out to take a stab at building a system which can detect new, out-of-vocabulary words. Early on, I knew that I needed a different framework for modelling subword structures and word pronunciations, other than a simple pronunciation graph. At around the same time, Stephanie was creating the second generation of a hierarchical, layered subword model which she christened ANGIE. (The first generation was the bidirectional letter-to-sound/sound-to-letter *layered bigrams* framework used in the doctoral disseration work of Helen Meng, another SLS alumnus who deserves special mention for her intellectual contributions.) Many of the underlying designs for ANGIE, including the context-free framework and the basic bottom-up, left-to-right parsing strategy, were the result of Stephanie's handiwork. I teamed up with Stephanie, and we have worked together since, building the engineering infrastructure up to the point that it can now support various speech recognition engines. We never did get around to tackling new word detection, but we hope that our work will help someone get much closer than previous attempts!

Victor Zue deserves special mention, not only because he is a member of my thesis committee, but also because of his encouraging and supporting attitude as head of SLS which has made the group a truly vibrant and amenable place to work. I also owe a great deal to Eric Grimson, who supported my graduate career in many ways. Besides making time to sit on my thesis committee, Eric also served as chairman of both my oral qualifying examination and area examination committees. An MIT graduate student can only pray to be blessed with such fortune as to have Eric's repeated guidance.

I do not want to forget the other members of SLS, particularly my officemates, past and present, who have endured my idiosyncracies throughout my four years here, and also the folks who make the place run, both in terms of computing and administrative infrastructure.

Finally, I want to thank my family, both immediate and extended, for their never ending support, almost to the point of toleration, of the many decisions made in life that

ultimately led me to this stage. Without their support and guidance through the many trials and tribulations of left, I would not have the opportunity to be here today. Perhaps my deepest grattitude goes to my dearly departed grandfather, who early in my childhood revered in the possibility of seeing a doctorate in his family, a first amongst his siblings and descendants, and instilled upon me the value of education. If not for his strong education ethos and supervision over my school work in my formative years, I do not know if I would have had the courage and perseverance to pursue a Ph.D.

Raymond Lau
mailto: mail@raylau.com
http://www.raylau.com
Cambridge, Massachusetts
March 30, 1998

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Spoken language has become accepted as a natural method for human-machine interaction. One of the fundamental challenges of developing a spoken language system is the development of a speech recognition component. Research in speech recognition has been ongoing for approximately three decades. Much progress has been made during that time span. We started with very small vocabulary, speaker dependent, isolated word recognition systems. These systems recognized only a small number of words, such as the ten digits, which were trained for a particular speaker or set of speakers only, and which required pauses between words ([59], chap. 1). Today, we have large vocabulary systems, capable of recognizing from 20,000 to upwards of 100,000 words. The systems are now speaker independent, working out of the box for any speaker, and in some cases even speaker adaptive, learning the peculiarities of a person's speech over time. Isolated speech has long yielded to continuous speech in the research environment, and more recently, in the commercial marketplace as well, with the introduction of systems by IBM and Dragon ([23]). Error rates have been reduced dramatically.

Nevertheless, despite all the progress that has occurred, many challenges remain. The primary focus of our work will be to introduce and implement a novel model for subword lexical modelling. Subword lexical modelling refers to a model for capturing the variability in the pronunciations of words. Such variability may exist because words may have multiple underlying pronunciations, different speakers have varying styles, and because of contextual effects. A speech recognition system must be able to handle the variability for it to be a robust recognizer. In this introduction, we will review, in greater detail, the role of subword

modelling and the most prevalent forms of it in existing speech recognition systems. Then, we will motivate our model, ANGIE, by introducing phonological and linguistic developments which suggest modifications to the existing models, particularly in terms of creating a hierarchical framework for subword lexical modelling.

While the direct focus of our research is on subword lexical modelling, we would like to mention several major open issues upon which our work touches. They are the problem of new words, the use of prosodic information in the speech recognition process, and the integration of subword phonological processing with higher level linguistic processing. We briefly introduce these issues, along with the potential applications of our work in their contexts. Finally, we will provide some background on recognition and on our experimental framework. We will conclude this introductory chapter with an overview of the remainder of this thesis.

1.1 Sublexical Modelling

In this section, we will present some background material on current approaches to sublexical modelling. Most moderate-to-large vocabulary continuous speech recognition systems in existence today model speech as a concatenation of subword units. The use of subword units helps to overcome two major disadvantages of whole-word models, the prior norm. One, we need a very large amount of training data to capture the appearance of each word in various phonetic contexts which may affect the realization of the word, especially the beginning and end, in speech. Two, because many words share common substructures which behave similarly in similar phonetic contexts, to have a separate model for each context of each word would be an extremely inefficient representation. There are several possible choices for the subword units to model ([59], chap. 8): phone-like units, demisyllable-like units, syllable-like units, and other acoustic units. If phone-like units were chosen, then there are typically 50 to 70 different context-indepedent units which may be used for the English language. For demisyllable-like units, the number of units increases to the order of several thousand, and for syllable-like units, the number increases to the order of 10,000, rapidly becoming intractable in terms of computational complexity and amount of training data needed. (The term acoustic units used here refers to an inventory of units selected through some computational method such as a clustering technique.)

For any particlar inventory of subword units, we need a way to associate a sequence of these units with a particular word. The mechanism by which we accomplish this must handle phenomena such as alternate pronunciations for a word (e.g., "either" may be pronounced as /iy dh ax r/ or as /ay dh ax r/[1] and phonology (e.g., how a word is realized acoustically as phones), particularly phonological variation[2] (e.g., "you" is usually realized as [y uw] but in certain contexts, such as in "did you," it may be realized as [jh uw]). The phonemic variants can typically be handled by multiple entries in the lexicon because the number of alternate pronunciations is typically small. The phonological phenomena are more difficult because they are typically context dependent[3] and would also require a large number of entries in the lexicon to handle all possibilities. It is the process of handling the possibilities of differing phonetic realizations for a given word within a speech recognition system that forms the crux of the proposed framework. For ease of exposition, we will concentrate on phone-like units. We will refer to modelling the sequence of phones permitted for different sequences of words as *subword lexical linguistic modelling* or simply *sublexical* or *subword modelling*[4]. There are two dominant approaches to sublexical modelling. They are to either use an explicit *pronunciation graph* or to employ some form of *implicit modelling*.

1.1.1 Pronunciation Graph

A few systems, including the very early Harpy system ([44]), MIT's SUMMIT system ([78], [79]), LIMSI's speech dictation system ([20]), and Cohen's work with SRI's DECIPHER system ([11]), attempt to model phonological variations and other sublexical phenomena by means of an explicit pronunciation graph[5]. In the case of the SUMMIT system, one or more phonemic baseforms are input for each word in the vocabulary. A pronunciation graph is then generated for each word. A pronunciation graph consists of a set of nodes and arcs. Associated with each arc is a permitted phone, along with a weight for traversing the arc.

[1] We have adopted the convention of indicating phones with a single pair of enclosing []'s and phonemes with enclosing //'s unless the context is unambiguous. The symbol set we are using is a modified ARPAbet set as described in [19], sec. 4.3.

[2] Hereinafter, we will simply speak of "phonological variation" to refer to all phonological effects, including constraints such as the observation that the phoneme /t/ may or may not be aspirated, but it is nearly always aspirated in a syllable initial position

[3] In this case, we mean to include both higher level sublexical context such as being in a syllable initial position, etc., as well as the adjoining phonetic context.

[4] The term *lexical access* is also used in the literature (e.g., [11]).

[5] The terms *pronunciation network* and *allophone network* are also used in the literature to refer to the same concept.

The pronunciation graphs generated from the baseform are expanded through the application of phonological rules to admit different phonological variations. The weights are trained through an iterative training process.

Instead of using phonological rules to generate a pronunciation graph and then iteratively training the weights of the arcs, it is also possible to generate the pronunciation graph statistically from the phonemic baseforms. Phoneme to phone n-grams or a decision tree can be used (e.g., [61]).

We see two major, related drawbacks to typical pronunciation graph implementations. One is the lack of sharing of common word substructure amongst different words. For example, the words "fly," "flying," "flight," and "flights" all share an initial phoneme sequence, /f l ay/, which is likely to be realized phonetically in similar manners. The lack of sharing means that training data for a given word can only be used to train the pronunciation graph weights for that word. If the common substructure can be shared, then training data from different words which have the same common substructure can also be shared, increasing the robustness of the training process. For example, the /l/ in this environment is likely to be devoiced, and the probability of devoicing can be jointly learned for all of these words. We have seen one attempt to address this problem. Cohen ([11]) ties arcs for certain subword units together across different words and pools their training data. Specifically, he looks for subword units which share a common phone sequence and also where the arcs for that phone sequence result from the application of the same set of phonological rules. The purpose of the latter restriction is to separate subword units which may be heavily context dependent. Interestingly enough, Cohen also suggests that a mechanism based on syllable structure may be productive, but, because he did not have a mechanism for identifying syllables, he abandoned that approach. The framework we will propose will incorporate the syllable as a core feature.

A related problem is that the addition of new words to the recognizer's vocabulary is difficult with a pronunciation graph approach. This is so because the arcs supporting the new word must be newly introduced into the graph, and all the arc weights in the added word's prounciation graph have to be trained with instances of that new word. While neutral weights can be used, we feel that it would be more desirable if the substructures, which the added word shares with existing words, can inherit probabilities already learned from related words.

1.1.2 Implicit Sublexical Modelling

Many systems do not use pronunciation graphs to model sublexical phenomena. Instead, they absorb the sublexical modelling into a combination of the structure/parameters of a hidden Markov model (for systems which are based on HMMs) and the acoustic modelling of the subword inventory units. This statement is best illustrated by example.

A successful early continuous speech recognition system was CMU's SPHINX system ([39], [41]). In the development of SPHINX, Lee started with an inventory of context-independent phonetic units and an assumption that each word had a single pronunciation. He started with baseform pronunciations from a dictionary but then decided to replace the dictionary baseforms with the most likely realized pronunciation for each word. At this point, the sublexical modelling was straightforward (and inflexible). Each word was associated with a single series of phones and each phone had a single acoustic model.

Lee then recognized the existence of certain sublexical phenomena which were not captured by this straightforward model. For example, he cites the following ([41], section B):

> For example, the first /d/ in "did" is always released while the last /d/ may not be released. Also, closures before stops are optional.

Lee chose to model these two types of phone deletion implicitly in the parameters of the hidden Markov model framework used in SPHINX by allowing a state transition that skips the phone. Lee and other researchers (e.g., [67]) also recognized that the articulation of phones is heavily dependent upon context. Lee chose to include *triphone* models, models which are specific to certain left and right phone contexts, in his system as well as *function-word-dependent phone* models, word specific phone models in the case of function words only. Lee also studied *generalized triphones*, which are basically a clustering of triphones so as to avoid sparse data problems for rarely occurring triphones. In this case, the acoustic models for the context dependent phones implicitly absorb the phonological variations specific to those contexts.

There are a myriad of approaches to capturing phonological phenomena through the selection of more specific phones in the lexical inventory. The term *allophone* (e.g., [12], section 1.5.2) is sometimes used in the literature to refer to models of phones which are context-dependent. Besides triphones (also diphones) and word-specific phones, some form of automatic clustering can be used to select an appropriate inventory. *K-means* clustering

([2]) or *classification and regression trees (CART)*, also known as *decision trees* ([4], [28]) have been used to accomplish this.

1.2 Towards a Hierarchical Representation

In the approaches to subword modelling we have thus far presented, there is an underlying assumption that words are best modelled as flat sequences of phone-like units. In the pronunciation graph approach, the phonological rules are applied to sequences of phones to generate possible variations. This is reminiscent of the rewrite rule approach in the early seminal works on generative phonology such as the *Sound Pattern of English* ([7]). In the implicit modelling approach, the use of context-dependent phones attempts to capture phonological effects through examining what the neighboring phones are, which can be thought of as a statistical method of capturing such rewrite rules.

Kahn ([33]) first suggested that there needs to be a unit which is larger than a phone, but smaller than a word to help explain various phonological processes. Kahn posits (Ibid, pp. 20):

> (a) that there exists, on the phonetic level, a well-defined unit of perception and production larger than the segment and smaller than the word, and (b) that this unit plays a very significant role in conditioning distributional statements, sound changes, synchronic phonological rules, etc., i.e., that it is of general phonological significance. The unit is of course the syllable.

Defining exactly what a syllable is has been a matter of controversy in the literature. However, most speakers will readily agree on the number of syllables in a given word, although where the syllable boundaries are not as clear. Most phonological definitions (e.g., [68]) hypothesize a *sonority scale*, which ranks the various sounds of English according to the extent which the sonority feature is present. Sonority is what typifies sounds which are produced primarily via vocal-tract excitation at the glottis with little obstruction to the air flow. Thus, vowels would rank high on the sonority scale, whereas voiceless stops, which are produced entirely via obstruction with no vocal tract excitation, would rank at the bottom. Given a sonority scale, a syllable is defined such that a *well-formedness rule* is enforced: within a syllable, there is a sonority peak with surrounding segments of decreasing sonority. Given this requirement, certain other rules help in fixing syllabification, such as a

maximum onset principle which tries to maximize the number of consonants in the onset, or beginning, position of a syllable, and a *stress resyllabification principle* which prefers to have segments assigned to a preceeding syllable if it were stressed.

Kahn's work takes a set of syllabification rules which he develops and shows that they "can be used to condition many phonological rules of English in a simple and natural way." (Ibid, chap. 2) Kahn even further posits that the syllable may play a role in phonotactic constraints, limiting what are permitted phonetic sequences, but stops short of giving a complete phonotactic theory. Subsequent to the work of Kahn, Selkirk ([68]), and others, many phonologists now recognize that

> the syllable is at the heart of phonological representations. It is the unit in terms of which phonological systems are organised... The syllable has received a very considerable amount of attention from phonologists, especially in recent years...

(Katamba, chap. 9, [34]). Thus, guided by the work of Kahn and other phonologists, we believe that a subword modelling framework should not be flat as in existing frameworks. Rather, it should have at least a layer for syllabification; hence, we adopt a hierarchical approach, which we will describe in greater detail in Chapter 2.

The work of Kahn, Selkirk, and others has contributed greatly to phonological theory, and particularly to the role of the syllable. However, their work is solely theoretical. For the purposes of a speech recognition system, we need a computational framework. A pioneer in terms of creating a subword computational model, which accounted for the syllable, was Church ([10]). Church implemented a system based on Earley's parser ([18]). It accepted phonetic transcriptions, produced by a linguistic consultant, as input and produced a syllabification from which words can be decoded.

Church's work influenced us in several ways. He showed that phonological constraints are actually sources of information and not noise, and that it was possible to discover syllables in a bottom-up manner from allophonic and phonetic cues. As we will discuss later, we share Church's view that subword processes are governed very much by a bottom-up philosophy. Further, Church's success at bottom-up discovery of syllables gives us hope that our framework can be used to address the new word detection problem, which we will describe later in this chapter. His work also introduced to us the idea of hierarchical

representations, which have two advantages according to Church (Ibid, sec. 1.4.1.1 and sec. 1.4.1.2): (1) improved performance due to sharing and (2) having a lexicon free of allophones. The second advantage refers to organizing "the lexicon so that the common shared sequences correspond to natural linguistic constituents ... not just arbitrary sequences of segments. Thus, for example, the prefix /r iy/ ought to be shared in words like *reduce* and *retry* where it is a linguistically motivated constituent, but it should not be shared in words like *read* and *real* where it is merely a common subsequence of segments." (Ibid, pp. 31) This is in contrast to the "put everything into context-dependent phones" approach pioneered by Lee and discussed earlier. Our position is that we agree strongly with point (1) and although we do not disagree that having linguistically motivated constituents is preferable, we are not necessarily opposed to context-dependent acoustic models. We agree on point (2) in that there is merit in a sublexical framework which has a more linguistically motivated organization than context-dependent phones as currently defined; however, we do not interpret this to mean that the inclusion of context-dependent phones or other more specific acoustic units into such a framework is to be excluded. A final idea we adopt from Church is that of using a context-free grammar to describe subword phenomena. The traditional linguistics community has used primarily a framework of context-sensitive rewrite rules, which are computationally expensive to parse. However, our context-free framework, discussed in Chapter 2 will differ from Church's in at least one important manner. Church marks his non-terminal categories in such a manner as to capture much of the context-sensitive information. We will rely on an additional probability model to capture much of the context-sensitivity.

Church has made numerous contributions, but he stopped short in one area. He did not extend his work into a system which can handle the large, errorful phone graphs created by the front-end of a typical speech recognition system. Church includes a chapter entitled "Robustness Issues" (Ibid, chap. 8) where he begins to explore such an environment. He notes that his parser works with the segmental lattice produced by a skilled spectrogram reader[6], but when he attempted to process phone graphs produced by a speech recognition front-end from BBN, he met with little success (Ibid, sec. 8.2). Church does, however, suggest that a probabilistic framework may help in such an environment (Ibid, sec. 8.5). This is exactly the approach we will adopt in our framework.

[6]Who marks a spectrogram with hypothesized phones for various segments

Such a probabilistic framework was pursued in the work of Meng ([49] and [47]). Meng's layered bigram framework actually plays a very significant role in the evolution of our framework. However, although probabilities were included in her work, like Church, she focused on a task where relatively error-free inputs were involved. The task was bidirectional letter-to-sound/sound-to-letter generation. In either direction, the input is relatively free of errors and the question of whether a hierarchical subword model can work with the noisy input from a speech recognizer's front-end remains unaddressed.

1.3 Acoustics of Supra-segmental Units

Thus far, our discussion of the history leading to a supra-segmental subword unit, that is, a unit larger than a phone unit, and a subword hierarchy has focused primarily on textual phonetic strings, rather than the actual processing of any acoustic data. The work of Randolph ([60]) provides a comprehensive study of the actual realizations in large acoustic data corpora and established empirically the relevance of the syllable in explaining many of the realizations, particularly when it comes to stop consonants. Randolph's analysis was conducted with various statistical classification methods on several corpora, including TIMIT ([37]).

Some researchers have also considered accounting for supra-segmental units within acoustic models in an actual recognition system, but keeping the lexical model essentially flat. Hu et al. ([29]) create syllable-like units, basically merging phones for which the segment boundary is difficult to discern, and achieved recognition accuracies which were claimed to be comparable to that achieved with phone-like units on a small vocabulary twelve-word month recognizer (95.5% vs. 96%). Jones ([32], [31]) created a syllable recognizer for a thirteen hundred word vocabulary task involving read speech. One of the problems Jones encountered was the lack of training data for certain syllables. He varied the number of mixtures depending on the amount of training data to compensate for sparse data problems. Ultimately, Jones claims that word recognition accuracies using syllable units were competitive with those obtained using phone like units.

A drawback with using larger acoustic units, as the work of Jones verifies, is the lack of training data. Namely, as we move from phones to syllables, the number of possible units increases from around sixty to several thousand. One possible approach would be to have

"a hybrid system, where the most common syllable HMMs would be used in conjunction with whole-word and phoneme models." ([32]).

This was the approach followed with *context freezing units (CFUs)* ([72]). CFUs are an attempt to capture more and more higher level context-dependent information in a single HMM. Each word is decomposed into multiple layers[7]: phones, phonemes, clusters, demi-syllables, syllables, words and compound words. Acoustic models are trained for various units as the amount of training data allows. An HMM for a larger unit will include paths with each possible sequence of smaller units for which acoustic models exist and also a path with the larger unit if an acoustic model exists for it. The example given in the paper for the German demisyllable "_urk" includes one path with just the model for "_urk" and another path composed of the cluster "_ur_" and the phoneme /k/. The "_ur_" model also includes paths for the further decomposition into phonemes as well as for the "_ur_" model itself, etc.

The use of CFUs or syllable acoustic units certainly captures some information implicitly about syllable structure. Even the use of context-dependent diphones and triphones arguably captures some supra-segmental information, which may implicitly include knowledge of syllable structure information. For example, certainly, some triphone sequences are much more common in certain syllable positions than others. However, these views still result in a flat framework where the syllable knowledge is hidden in the acoustic models, and is unnatural and difficult to control. For example, phenomena such as always releasing the first /d/ in "did" are not something we can control easily. This information is encoded into the particular acoustic model for the corresponding syllable. We are persuaded by the Church view that we should lean towards a linguistic, rather than acoustic, organization which corresponds more naturally to linguistic units, and thus, we lean towards a hierarchical sublexical framework. We do not doubt that there is a value in having larger supra-segmental acoustic units, especially when sufficient training data exist. We intend to incorporate such acoustic models into our system at some future date. However, we believe there is great value in a lexical model which accounts for syllable information, independent of the choice of acoustic models. Thus, we focus, in the present work, on the task of a subword lexical model and confine ourselves to context-independent acoustic phone models.

[7] A layered approach is one which we share in our ANGIE framework, as described in chapter 2.

1.4 Our Goal: A Computational Framework for Recognition

Kahn and others have introduced the concept of a hierarchical subword framework. Church has implemented an actual computational machinery based on such a framework. However, Church worked with relatively error free inputs. The few references we found relating to the use of syllables for recognition in the literature[8] were primarily focused on using larger acoustic units, such as syllables. The goal of our work is to further extend the lexical line of development of Kahn and Church to a system which attempts at modelling sublexical phenomena in a hierarchical, syllable aware manner, and which can work with errorful inputs, such as those generated by the acoustic front-end processor of a speech recognition system. In order to do this, we will extend the categorical approach pursued by Church with a probabilistic framework, much like the work of Meng, and attempt to implement a hierarchical, probabilistic, sublexical modelling framework which can deal with large, errorful, phonetic segmentation graphs typically encountered with speech recognition tasks. Naturally, a major part of this effort will involve engineering design considerations, involving search strategies and computational resource management (e.g., memory, time, etc.) Also, much of the work will have an empirical flavor rather than the theoretical flavor found in the phonology literature.

1.5 Possible Applications of Our Model

The goal of this thesis is not solely to demonstrate that such a hierarchical subword modelling framework can be made to work. That certainly would represent an important accomplishment but we would like to accomplish more than an intellectual aspiration. We are inspired to pursue such a subword model because we believe that it can better address some of the difficult, open issues in speech recognition research. We review three areas where we feel that our subword model will prove advantageous.

1.5.1 New Words

New words pose a continuing problem for speech recognition systems. Despite the tremendous growth in vocabulary sizes over the decades, there are still a significant portion of

[8] For English that is. There is quite a bit more work for Chinese, where syllables are more prominent lexical units, and hence are more comparable to using words in an English recognizer.

words which remain outside of even a large 100,000 word vocabulary. This poses some clear problems for tasks such as the application of information retrieval to speech data, where new words will not be indexable and searchable. Worse yet, even if we were willing to accept some unrecognizable words, new words also tend to cause recognition errors when they occur and may cause other problems in a spoken language system if the misrecognitions go undetected. Hetherington ([25], sec. 2.9) found that:

> In fact, it can take very large vocabularies, on the order of 100,000 words or more, even to get the new-word rate down to 1% for some types of tasks. We showed that although a new-word rate of 1% may seem low enough, it can correspond to 17% of sentences containing one or more new words. Having nearly one in five utterances containing a new word is almost certainly an unacceptably high rate. Because of the misrecognition and misunderstanding that a new word could cause, a sentence rate as high as one in five would likely interfere with a user's interaction with a spoken language system.

There are two main issues involved with new words. The first is how to have a recognizer detect the presence of a new word, instead of misrecognizing a new word as some other word in the vocabulary, which would be the natural outcome in many of today's maximum likelihood frameworks. The second is, how do we easily support the expansion of recognizer vocabularies, perhaps dynamically at run time, as in the case of a conversational system returning a list of items of interest, for example, from an information resource such as a web site.

Our work involves developing a novel framework for modelling subword lexical phenomena in a speech recognition system. Through extensive hierarchical sharing, we hope that our framework can better support generalizing learned subword knowledge to new word additions. Also, by providing a more detailed lexical model of subword structures, we aspire to leverage off these models for new word detection. In the work of De Mori and Galler ([16]),"pseudo-syllable" acoustic units were used in an attempt to address issues of vocabulary independent recognition, a goal which we share in principle. A novel feature of the work was that a first pass search generated a syllable graph, then a second pass search decoded it into words through the use of a series of HMM phonotactic models. Although no concrete results were reported, we are encouraged that some success was alluded to in terms

of decoding words from syllable-like units. The authors also reported on a very small pilot experiment on detecting new words via the use of a syllable recognizer and mentioned some success. We envision our framework supporting a conversational system which can detect new words and ask the user for clarification, have a dynamically modifiable vocabulary, perhaps based on dialogue context, and even generate pronunciations from the spellings of words through the underlying linguistic framework's dual phone/letter nature[9].

1.5.2 Prosody

The inclusion of prosodic information during the recognition process has been an often cited, but elusive goal ([15]). Our subword modelling framework, by providing hypothesized subword structural information during the recognition process, can be exploited by a prosodic modelling system. Further, as discussed in the next section, we also explore the possibility of integrating higher level linguistic information, which can also potentially be employed by a prosodic model. While actual work on prosodic models is beyond the scope of the present thesis, we will mention one pilot study involving the incorporation of a colleague's duration model, which employs our interpretation of subword structure, ([9]) into our word-spotter in Chapter 6. Durational patterns and pauses are prosodic cues generally believed to convey information in natural speech. Duration is influenced by several factors, some of which ANGIE can potentially use to provide relevant information to assist in the construction of a duration model. Summarized from [8], these components include:

Phonological Phones have an inherent duration distribution, e.g., voiced stops are shorter than voiceless stops and are also influenced by contextual effects, e.g., when two identical adjacent phonemes are realized as a single geminant phone.

Lexical The position of a phone within a word can greatly influence its duration. Here, ANGIE can potentially provide valuable information. For example, stressed vowels are generally longer than unstressed vowels and consonants in pre-stressed positions are often longer than those in unstressed and post-stressed positions.

Syntactic The phrasal pattern of a sentence often affects phone durations. For example, vowels in syllables preceding phrase boundaries are often longer than those in

[9]Details on this last capability will be discussed briefly in Chapter 2 but are otherwise beyond the scope of this thesis.

non-phrase-final syllables. Also, segments preceeding a pause are often lengthened. More important to our work, this prepausal lengthening is often accompanied by a lengthening of syllables.

Speaking Style Naturally, a person's speaking style has a tremendous influence on duration. Speaking rate is an important factor which has numerous phonological influences as well (e.g., faster speakers tend to flap and palatalize more whereas slower speakers tend to insert glottal stops more frequently). However, because it is a continuous variable, it is difficult to measure reliably.

Semantic Some researchers have suggested that speakers tend to slow down at the end of a conceptual unit while emphasis and constrastive stress tends to increase. The integration of higher level linguistic processing into our ANGIE framework, which we will discuss next, has the potential of providing relevant information for the detection of semantic dependencies.

We will incorporate Chung's duration model into our word-spotter and report on its performance. Further work in this area is beyond the scope of this thesis.

1.5.3 Integrating Subword Phonological Processing with Higher Level Linguistics

Many spoken language systems employ a higher level language understanding component in addition to a speech recognition component. The interface between the two components is best characterized as a feed forward only process, with either an N-best list (the top N full sentence hypotheses from the recognizer) or a word graph (a graph representation of the top scoring hypotheses from the recognizer, as in [71]) being passed from the recognizer to the understanding component. The understanding component then either rescores the hypotheses or chooses the highest scoring one that parses. Very little progress has been made in terms of feeding knowledge in the reverse direction, from the understanding component to the recognition component. An understanding component is needed to obtain useful results from a spoken language system, where honoring the user's request rather than recognition is the aim. However, attempts at leveraging off the understanding component for better recognition have met with only limited success (e.g., [52], [77]).

Our subword model is based on an underlying context-free framework. Context-free grammars also underly numerous natural language understanding systems, including the TINA system from MIT ([69]). Part of our work will be to explore whether our subword framework can be integrated with a natural language understanding system more tightly, so that knowledge feeds in both directions, allowing the NL system to help filter unpromising hypotheses early.

1.6 Recognition and Choice of Lexical Unit

So far, we have been referring to work as focused upon subword modelling. However, the choice of words as being the ideal lexical unit is not preordained. A natural question to ask is, what is the role of the lexical unit? We view the recognition process as divided by the lexical units into two parts as illustrated in Figure 1-1. In the sublexical portion, we believe bottom-up and sharing concepts tend to dominate. For example, at the very bottom, we have some selection of phone-like units which are shared by all words. Similarly, we have phonemic units which are also shared by all words and phonological processes which govern the derivation of phones from phonemes. Bottom-up sharing dominates here as exemplified by the observation that occurences of the same phonemes in different words, but in similar contexts, tend to be governed by similar phonological rules. Above the lexical level, we believe that top-down ideas tend to dominate. The most striking example of this in the English language is the occurrence of gaps, where entire phrases are moved, usually forward, from one part of a sentence to another, as is the case when the top-level sentential structure is a wh-query (e.g., "What meals does this flight serve *[trace]*[10]?").

The role of the lexical unit in a recognition system is to serve as a common point around which to organize the recognition search process. As mentioned in a preceeding section, many systems, which include higher level language understanding components, pass on an N-best list to the understanding system. The N-best list is pruned at the lexical level and the search processes in such systems are isolated by the lexical level. We will address what choice of lexical units facillitates both subword modelling, higher-level linguistic processing, and their integration. The word is one choice. Syllables and morphemes are others.

[10]The trace is a category which encodes the base position for a moved constituent. In this example, the moved wh-constituent is "What meals."

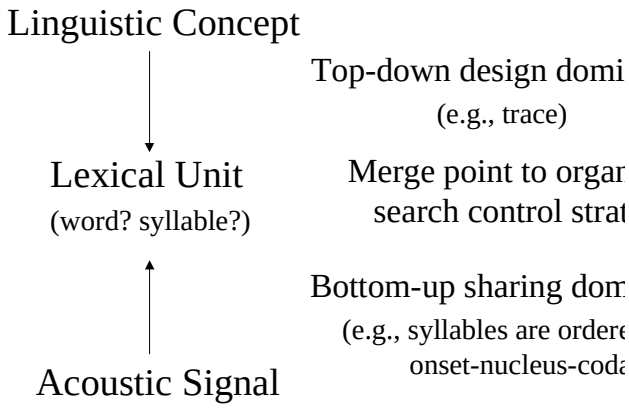

Figure 1-1: Recognition process above and below the lexical unit.

1.7 Experimental Framework: ATIS Corpus

Our experiments will be conducted on data from the *Air Travel Information System* (ATIS) corpus ([27]), including the ATIS-3 additions ([14]). The main addition in ATIS-3 was the expansion of the city list from 11 cities to 46 cities. We will collectively refer to ATIS (0-2) and ATIS-3 as simply ATIS. The ATIS domain was the former common evaluation task for ARPA spoken language system developers. The speech data consist of user inquiries related to air travel planning to solve specific scenarios presented to the user. A typical scenario (from [27], figure 1) might be:

> You have only three days for job hunting, and you have arranged job interviews in two different cities! (The interview times will depend on your flight schedule.) Start from City-A and plan the flight and ground transportation itinerary to City-B and City-C, and back home to City-A.

A typical user query might be:

> Show me flights from Pittsburgh to Boston on September fourth in the morning.

The corpus consists of approximately 27,500 utterances of continuous and spontaneous speech from 732 different speakers. Some reasons for selecting ATIS include:

Continuous and spontaneous speech We feel that modern recognition systems should support naturally spoken speech, which dictates a preference for continuous over isolated speech and for spontaneous over read speech.

Selection of keywords The various city names and airline names provide an excellent selection of keywords for planned word-spotting experiments.

ATIS-3 city additions The addition of cities to create ATIS-3 provides ample data where a set of new words has been added to the vocabulary. This facilitates planned experimentation involving the handling of new words.

Compatibility with previous work A very extensive study of word spotting within the ATIS domain was performed by Manos in [45]. Similarly, Hetherington in [25] has studied the impact of new words on recognition by treating the additions in ATIS-3 as new words for a baseline ATIS-0 to ATIS-2 recognizer. The availability of results from the same corpus eases the process of performance evaluation.

We will be working with a subset of the ATIS data to facilitate speed of experimentation. Our subset will consist of around 5,000 utterances for acoustic and language model training, 10,000 utterances for subword linguistic training, the designated development set for development, and the December 1993 test set for testing. We will use context-independent acoustic models for our experiments, since, as we mentioned, our focus is on the linguistic side and not on acoustic modelling. We report our results on a segment-based recognizer, which we will build throughout this thesis, based on the same front-end that the MIT SUMMIT system ([79]) uses.

1.8 Summary of Goals

In short, the goals of this thesis are to introduce a novel framework, ANGIE, for sublexical modelling, which we believe possesses several desirable features, and demonstrate the feasibility and efficacy of our framework in various speech recognition tasks. Some of these desirable features include an integrated probabilistic rule-based model for phonological effects, ease of altering subword representations, sharing of subword structures, which we hope can help with the problems of coping with new words and flexible vocabularies, provision of subword structural information for prosodic modelling, and the ability to be integrated with higher level linguistic modelling and understanding. Feasibility of our framework will be demonstrated via implementation in phonetic recognition, word-spotting and full speech recognition systems, all using ATIS as an experimental data corpus.

1.9 Overview

Now that we have presented some of the broader issues motivating our work along with some background on sublexical modelling, we next present an overview of the remainder of the thesis. In Chapter 2, we describe our ANGIE framework for subword lexical modelling. We will discuss its motivations, context-free structure, probability model, and some initial perplexity evaluations. The next three chapters describe our empirical work in implementing our ANGIE framework within three speech recognition tasks and also mention some subword lexical modelling studies we conducted while pursuing this work. Chapter 3, presents an overview of a phonetic recognition system based on ANGIE, including some background material about the important issue of search, along with some promising initial results using our recognizer in the task of phonetic recognition. The results will show that the ANGIE-based system is superior to the baseline system in terms of error rate. Chapter 4 presents some background material on the task of word-spotting, i.e., finding keywords in speech, and describes the implementation of a word-spotting system based on ANGIE. Also included is a description of a series of experiments where we varied the subword model within the word-spotter and the results of such variations. We will show that ANGIE can support a competitive word-spotter and that generally, the more subword constraints on the filler model, the better the word-spotting performance becomes. After that, we discuss the implementation of a full recognition system based on ANGIE in Chapter 5. Our system will be shown to be competitive with a baseline system which uses a pronunciation graph. Here, we also explore an integrated recognition system where higher level natural language constraints are combined with the subword constraints from ANGIE in a tightly coupled search process, and present several respectable comparisons arguing in favor of such a combination. In Chapter 6, we present two pilot experiments involving the ANGIE framework, one involving work done jointly with a colleague on duration modelling and another on the incorporation of new words into a recognition system based on ANGIE. We will show that ANGIE does support the incorporation of new words into the vocabulary, that this process is arguably simpler with ANGIE than with other sublexical models, and that the inclusion of a natural language component in the ANGIE framework also supports dynamic vocabularies. In our final chapter, we conclude with several closing remarks along with suggestions for future related research.

Chapter 2

ANGIE: The Proposed Framework

In this chapter, we present a probabilistic framework for sublexical modelling which we have named ANGIE ([70]. The ANGIE framework is a descendant of the framework described in Meng ([47], [49]). The motivations behind the framework include creating a sublexical model which:

- captures various sublexical phenomena, including phonology, syllabification and morphology, in a unified framework;
- is probabilistic in nature;
- promotes sharing of common sublexical structures among different words in the vocabulary, words introduced into the vocabulary, and in principle, new out-of-vocabulary words;
- proceeds in a bottom-up manner, reflecting our bottom-up sublexical philosophy mentioned in Chapter 1, our desire to share and our aim to model word-like structures for the background filler in word-spotting and new words;
- provides a single framework for multiple tasks, including recognition oriented tasks and letter-to-sound/sound-to-letter generation; and
- shares a common context-free framework with many natural language understanding systems, permitting an integrated system for both phonological and linguistic modelling.

At the heart of ANGIE are a context-free grammar, a parser, and a probability model associated with parses generated. We describe each in turn.

2.1 Context-free Grammar

The context-free grammar is hand written and is used to obtain a hierarchical representation of the sublexical phenomena of interest. There has been some work in the literature which tries to automatically learn a grammar describing subword structures (e.g., [66]). However, we fear that such an approach will migrate back in the direction of implicit subword modelling, where the actual subword relationships are hidden, not easily controllable, and not motivated by linguistic organizations. Therefore we have chosen to maintain explicit control over the structural organization.

A typical parse tree is shown in Figure 2-1. The hierarchical representation has a very regular *layered hierarchical* structure. We have previously stated some of the motivations for a hierarchical subword structure in Chapter 1. Here, we would also like to point out that, from a computational framework point of view, the choice of a hierarchical structure allows us access to information available at each layer of the hierarchy. A set of transformations effected by linear rewrite rules, which is typically used in the linguistics literature, obscures this information. The root SENTENCE node is currently realized as a sequence of WORD nodes. Presently, these two categories act as place holders, but could later be replaced with other alternatives such as topical units, syntactic units, or both. The layers beneath the WORD node capture, from top to bottom:

1. Morphology
2. Syllabification
3. Phonemics
4. Phonetics

The very regular, layered structure of our parse trees is mandated by the organization of our grammar. The categories in the grammar are grouped into separate sets for each of the layers. The productions are written such that the left hand category belongs to the set from a given layer and the right hand categories all belong to the set from the layer immediately below. No productions are permitted to "skip" a layer. Thus, each parse tree permitted by the grammar has precisely the layers described above.

The phonetics layer includes nodes for each possible phone in our phone set. The phones are associated with acoustic models applied to the speech signal. Our phone set is given in

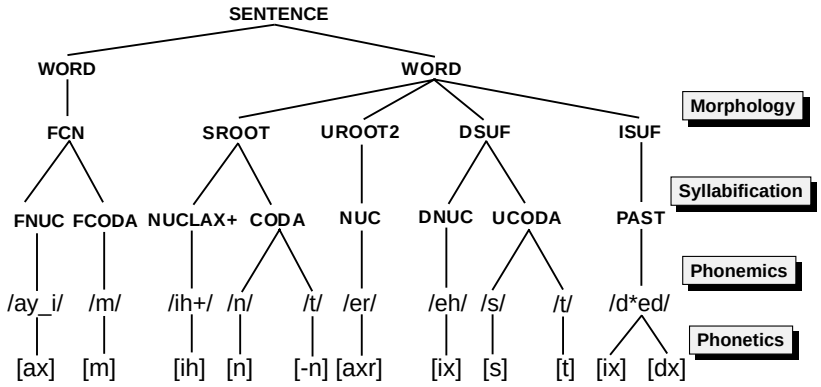

Figure 2-1: Sample parse tree for the phrase "I'm interested."

Table 2-1. The phonemics layer includes nodes for our set of approximately 100 different phoneme-like units, which are listed in Table 2-2. Our lexicon is organized either as a single level structure, with phonemic baseforms for each word, or as a two level structure, with syllables for each word and phonemeic baseforms for each syllable. The syllables in the two layered organization are marked for position, e.g., prefix. Our set of phoneme-like units includes stressed (marked by "+") and unstressed vowels, onset (marked by "!") and non-onset consonants, some morpheme-specific units (e.g., /d*ed/ for the past tense morpheme "ed"), some function word-specific units (e.g., /uw_to/ for the /uw/ in "to"), and several pseudo-diphthongs (e.g., /aar/). We include onset and stress markings so that this information is readily available to the probability model at the phone layer, since both properties are phonologically significant. Function words are more frequently reduced in realizations, hence, we felt the need to separate out their phonological effects from non-function words. The few remaining cases of special markings were similarly arrived at and validated through numerous development iterations. These iterations consisted of examining sample forced alignment and recognition outputs and tuning the set of phoneme-like units. The goal of such tuning was to capture information which we felt would be helpful and to eliminate sparse data problems.

The syllabification layer includes various syllable parts, such as ONSET, NUC+ (stressed nucleus), and CODA. Although the role of the syllable was initially somewhat controversial, as we have noted in our introductory chapter, it has received much subsequent attention in the linguistic community and has recently gained wider acceptance. The introduction of the syllable greatly simplifies many phonological rules ([33]) and it also acts as the basic phonotactic unit (c.f. [34], sec. 9.4.1). The latter reason, allowing the inclusion of phonotactic constraints, is particularly important to us if we want to be able to constrain the recognition process, especially with respect to unknown words in our system[1].

However, despite the acceptance of a supra-segmental syllable unit in phonology, the rules of syllabification are far from a settled issue. We have mentioned some syllabification guidelines in Chapter 1, such as well-formedness in terms of the sonority hierarchy, maximum onset, and stress resyllabification, but there are numerous examples for which a clear division into syllables is unclear. In those cases, we take the position that the precise

[1]Church recognized the power of phonological rules as providing constraints instead of the then dominant view that they are a source of noise ([10], sec. 1.2.1.2).

Standard Phones			
aa	bOtt	iy	bEEt
ae	bAt	jh	Joke
ah	bUt	k	Key
ao	bOUght	kcl	k closure
aw	bOUt	l	Lay
ax	About	m	Mom
axr	buttER	n	Noon
ay	bIte	ng	siNG
b	Bee	ow	bOAt
bcl	b closure	p	Pea
ch	CHoke	pcl	p closure
d	Day	q	glottal stop
dcl	d closure	r	Ray
dh	THen	s	Sea
dx	flap as in muDDy	sh	SHe
eh	bEt	t	Tea
epi	epenthetic silence	tcl	t closure
er	bIRd	th	THin
ey	bAIt	uh	bOOk
f	Fin	uw	bOOt
g	Gay	ux	tOOt
gcl	g closure	v	Van
hh	Hey	w	Way
hv	aHead	y	Yacht
ih	bIt	z	Zone
ix	debIt		

R-colored and Nasalized Vowels			
aar	as in *are*	aor	as in *for*
aen	as in *can*	ehr	as in *fare*

Non-standard Phones	
fr	f r combination as in *from*
hl	devoiced l in *flight*
scl	captures noisy closure following fricatives, usually between s and t
ti	i in the context of a preceeding alveolar stop which may have been deleted by our segmentation algorithm, as in *city*
tr	retroflexed t burst as in *trip*
ts	combination of a t burst and s

Pauses	
pause	sentence initial and sentence final pause
iwt	inter-word silence
iwt2	noisy interval within a *pause* or iwt

Table 2-1: Phone set for ANGIE

Normal	Stressed	Special	Normal	Onset	Special
aa	aa+		b	b!	
aar	aar+		ch	ch!	
ae	ae+		d	d!	d*ed
ah	ah+	ah_does	dh	dh!	
ao	ao+	ao_on	f	f!	
	aol+		g	g!	
aor	aor+		h	h!	
	aw+		jh	jh!	
ay	ay+	ay_i	k	k!	
eh	eh+		l	l!	
ehr	ehr+		m	m!	
el	el+		n	n!	
		en_and	r	r!	
er	er+				ra_from
ey	ey+	ey_a	p	p!	
ih	ih+		s	s!	s*pl
ihr	ihr+		sh	sh!	
		ing	sil		
		ix_in	v	v!	
iy	iy+	iy_the	t	t!	
ng			th	th!	th*s
ow	ow+		w	w!	
oy	oy+				wb
q			y	y!	
	uh+		z	z!	z*pl
uw	uw+	uw_to			
		ux_you			
yu	yu+				

Table 2-2: Set of phoneme-like units for ANGIE

Our set of phoneme-like units includes stress (+) and onset (!) markings for many units along with several word-specific units (e.g., ah_does), several special suffix-morpheme-specific units (e.g., d*ed and ing), and the word boundary (wb) and silence (sil) units.

syllabification is not what is most important, per se, but rather that we have consistent syllabification across words in our lexicon. This is enforced via two mechanisms. One is that our set of phoneme-like units includes markings for onset consonants, which disambiguates many syllabifications. The other is the inclusion of a probability model, which we will describe later. The probability model will naturally favor syllabifications which are used by words with similar sublexical structure in our training data, creating a self-reinforcing consistency.

The morphology layer serves to divide a word into morpheme units. This is motivated by two factors. One is that much work in generative phonology, including *Sound Pattern of English* ([7]) and more recent work in lexical phonology (e.g., [35], [36]), recognize the interaction between a decomposition of the lexicon into morphemes and phonological processes. The other is that the rules of word formation are typically morphological ones. Because we want to recognize "word-like" structures, for the purpose of modelling new words, etc., we felt that a layer representing morphemes would be helpful.

Unlike the framework described in [49] and [47], we do not have *broad class* and *stress* layers. Elimination of the broad class layer had been found to help performance in the letter-to-sound/sound-to-letter tasks in [49]. However, the same work also suggested that elimination of a stress layer will be detrimental. Instead of an explicit stress layer, we distribute the stress information across the morphology, syllabification, and phonemics layers. Thus, for example, we have both *unstressed root* (UROOT) and *stressed root* (SROOT) in the morphology layer, and, as already discussed, we have stressed and unstressed markings in our set of phoneme-like units. The rationale for doing this is that because we have chosen trigram bottom-up probabilities in our probability model (see sec. 2.3 *infra*), having stress as a separate layer above syllabification, as in [49], would render the stress information inaccessible at the lower phonemics and phonetics layers.

Of particular interest in our grammar are the rules going from the phonemics to phonetics layer. These rules govern what phonological processes are permitted. Phonological variations are typically specified in terms of context-sensitive or even full Turing rewrite rules[2]. However, a parser for context-sensitive rules is not known to be efficient (i.e., not known to be implementable within a polynomial time bound) and a parser for full Turing rules is, naturally, known to be undecidable. Thus, we want a framework for which efficient

[2]Worse yet, the rules are usually specified in an explicit linear ordering (c.f. [7]).

parsers are known to exist. As suggested by the work of Church ([10]), our framework uses only context-free rules. We will rely on a combination of the hierarchical structure, our choice of nonterminals, and the probability model to learn any context dependency in our phonological rules[3]. A typical rule might be[4]:

$$/\text{p}/ \;\Rightarrow\; \$\text{pcl}\ [\$\text{p}]$$

indicating that the phoneme /p/ (in a non-onset position) may be realized as the phone $pcl followed by an optional $p release. Here, we adopt the convention that brackets indicate optional elements. For a /p!/ (in an onset position), the rule is:

$$/\text{p!}/ \;\Rightarrow\; [\$\text{pcl}]\ \$\text{p}\ [\$\text{hh}]$$
$$\$\text{pcl}$$

As we see with the /p!/ phoneme-like unit, markings for onset position are not carried over into the phone layer with distinct onset and non-onset phones. Instead, we merely indicate through our rules that an onset /p!/ may be aspirated, hence the optional $hh phone, but a non-onset /p/ may not. The probability model then learns the probability of aspiration for an onset /p!/. We do something similar for stressed and unstressed vowel phonemes and function-word-specific phoneme-like units. Instead of separate stressed and unstressed vowel phones, we expect that an unstressed vowel phoneme will more likely be reduced to a schwa phone. Presently, our phone set consists of around 65 different generic phones. Our phone set is constantly evolving. The current rationale for having only very

[3]Church argues for a hierarchical phrase-structured grammar as a computationally feasible alternative to linear rewrite rules for representing phonological phenomena. Our framework shares these ideas. However, Church relies heavily on absorbing context-dependencies into the choice of non-terminals in his grammar. Some would argue, then, that in fact, phonological rules are really inherently context-free, but that phonologists have only chosen a context-sensitive representation for conciseness, for familiarity, or for some other motive. We take no position on whether phonological rules require context-sensitivity or not. Rather, we adopt the practical position that since we also include a probability model, we hope that it will learn whatever context sensitivities are required. Thus, there is a compelling argument why we *should* use a context-free grammar, namely, the existence of efficient parsers, and there is not a compelling argument against this choice.

[4]When we illustrate context-free rules, we will use $ for phone terminals. We depart from the [] convention for phones because [] is usually used to indicate optional components when expressing context-free rules.

generic phones is to avoid the possibility of splitting limited training data across multiple phonetic units with very similar properties.[5]

Thus far, the bottom-most layer is indicated as being the phonetics layer. This is true for our work, but the same framework can also be used for sound-to-letter and letter-to-sound generation if we replace the phonetics layer with a graphemics layer ([49], [47], [70]). The same context-free rules can be used, except that we have rules going from the phonemics to phonetics layer in one case and rules going from the phonemics to graphemics layer in the other case. Having a single framework for different tasks is a satisfying objective.

The context-free grammar dictates which parses may be generated by the parser. Any parse not licensed by the grammar is not allowed in our framework. Thus, the grammar must either generate exactly, or *overgenerate* the set of allowed parses. Naturally, it is very difficult, if not impossible, to construct a grammar which generates exactly the set of parses that occur in a natural language. Thus, we err towards overgeneration. The probability model is then relied upon to score the resulting parses, assigning low scores to parses which are permitted by the grammar, but which are not likely to occur in English. For the interested reader, an example set of context-free rules used by ANGIE is included in Appendix A.

2.2 Parser

The parser takes as input a sequence of phone terminals and tries to generate one or more parse trees. In the process, it also applies the probability model described in the next section to score the parses. Our parser proceeds in a bottom-up left-to-right manner. Because of the very regular nature of our rules, we can consider the following alternate visualization of a parse tree. We can view the parse tree as a table. Each layer is a row in the table. Along the bottom-most row are the phones. Each column in the table represents a path from the root of the tree to a phone at a leaf node. The parse tree from Figure 2-1 is shown in table format in Figure 2-2.

Our parser proceeds as follows. We start from the lower left-hand corner of the parse

[5] As mentioned in our introduction chapter, we are not opposed to having more context-specific, and possibly supra-segmental acoustic units when the training data permits. However, for the purposes of the present thesis, we want to concentrate on lexical modelling. We feel that without performing much work on determining what context-sensitive units to choose, the training data will not support a blind move to such units. Thus, we will work with generic context-independent units for the purposes of this thesis.

SENTENCE											
WORD		WORD									
FCN		SROOT			UROOT2	DSUF				ISUF	
FNUC	FCODA	NUCLAX+	CODA		NUC	DNUC	UCODA			PAST	
/ay_l/	/m/	/ih+/	/n/	/t/	/er/	/eh/	/s/	/t/		/d*ed/	
[ax]	[m]	[ih]	[n]	[-n]	[axr]	[ix]	[s]	[scl]	[t]	[ix]	[dx]

Figure 2-2: Table representation of parse for the phrase "I'm interested."

table. We see if any application of rules permits the first phone. Then we climb our way up the first *column*, generating different theories for all the possible phoneme-like units, syllable units, morpheme units, etc. When we are done with the first column, we move on to the second column, then the third, etc. Thus, our parser proceeds in a bottom-up, left-to-right manner. Because we always advance all theories from one column to the next, at any given point in time, the frontier of all active theories involves the same position within the phone string being parsed. Thus, our parser can be said to be working in a breadth-first manner.

The bottom-up design naturally supports a system which shares as much of the lower level structures as possible. A top-down design would dictate that we hypothesize distinct higher level structures, such as morpheme units, and then their decompositions. Such a design would not allow common lower level structures to be shared during the search process for different higher level structures. Different linguistic theories, which may share a common initial sublexical structure and a common initial phone string, can share a common set of partial parses during the search process. Furthermore, a bottom-up approach would help in modelling new out-of-vocabulary words for a new word detector and in modelling the background filler for a word spotter. The breadth-first design is needed during recognition because we will be working with partial phonetic hypotheses, and we need to obtain partial scores for all possible partial parses of the hypothesized phone sequence. A more detailed algorithmic description of our breadth-first parser design is included in Appendix B.

2.3 Probability Model

There are two motivations for including a probability model in ANGIE. We had mentioned that ANGIE is based on a context-free grammar. However, many phonological phenomena are typically believed to be of a context-sensitive manner. Typically, phonological rules are

expressed relative to left, right, or both left and right contexts. ANGIE hopes to capture this information as part of its probability model. Also, unlike the work of Church cited earlier ([10]), our framework needs to support a task which generates many errorful input phone hypotheses. We believe that a probabilistic framework is needed to cope with such a scenario. The probability model in ANGIE consists of two types of probabilities:

Advancement probabilities These are the conditional probabilities of a leaf node in the parse tree (that is, a phone terminal in the bottom-most layer) given its immediate left column. We term these advancement probabilities because we can think of ANGIE performing a left-to-right parse, advancing to a new phone to start the next column. Most of the probabilistic constraints are captured by this probability.

Trigram bottom-up probabilities These are the conditional probabilities of an internal node in the parse tree (that is, a nonterminal in any layer other than the bottom-most layer) given its left sibling and its child. We call these bottom-up probabilities because we can think of the ANGIE parser generating a possible nonterminal as it climbs a column bottom-up from a phone to the root. These are trigram probabilities because the probability is conditioned upon two other nodes, namely the child and left sibling.

The conditional probability of a column is the product of the conditional advancement probability and the various conditional trigram bottom-up probabilities up to the point where a column merges with its left column. For example, referring back to Figure 2-2, the conditional probability of the second column is:

$$\begin{aligned} Pr(\text{2nd column} \mid \text{1st column}) \quad = \quad & Pr_{advance}([\text{m}] \mid \text{SENT, WORD, FCN, FNUC, /ay\_l/, [ax]}) * \\ & Pr_{bottom-up}(m \mid /\text{ay\_l}/, [\text{m}]) * \\ & Pr_{bottom-up}(\text{FCODA} \mid \text{FNUC}, /\text{m}/) \end{aligned}$$

The probability of a parse table is the product of the conditional probabilities of the columns in the table. Our probability model is essentially a phone bigram-like model if we consider that adjacent phones anchor adjacent columns, but one which captures longer distance information through the upper layers of our parse tree. The upper layers consist of units

which are often larger than phone units. This model resembles the one in [49] and [47], except that there, the bottom-up probabilities depend on the entire *left history*, defined as the nodes in the left column extending from the child of the left sibling all the way up to the root, and the current child.

A good example of how the rules and probability model work together is illustrated in our sample parse shown in Figure 2-1. Here, the /t/ in "interested" is deleted and is represented in the parse tree by the special phone [-n]. The deleted /t/ is in a cluster with /n/ in a coda position with falling stress. This information is captured by the combination of the left column, which gives us the /n/ cluster and coda information, and the bottom-up probabilities, which would presumably permit the climb from the deletion to the /t/ (another possibility would be /d/). A brief description of the memory structures used to implement the probability model in the parser is included in Appendix B.

2.3.1 Smoothing

While the preceeding discussion describes our probability model in general, there are a few specific details involving smoothing and the handling of word boundaries about which to be concerned. Generally, the trigram bottom-up probabilities do not suffer from many sparse data problems because of their limited context. However, this is not the case for the advancement probabilities. There are certain column-phone advancements which are very rare in training data and thus their probabilities cannot be reliably estimated. To cope with this, we have implemented some backoff smoothing. Specifically, let:

LC be the current left column context

$p_1 \ldots p_P$ be our set of phones

$c(LC, p_i)$ be the count of training data occurrences of phone p_i following context LC

$c(LC) = \sum_{i=1}^{P} c(LC, p_i)$

$z(LC)$ be the number of different p_is for which $c(LC, p_i)$ is zero

Then for $z(LC) > 0$, we let:

$$Pr(p_i|LC) = \begin{cases} \frac{c(LC,p_i)}{c(LC)+k} & \text{if } c(LC, p_i) > 0, \\ \frac{k}{z(LC)*(c(LC)+k)} & \text{if } c(LC, p_i) = 0. \end{cases}$$

where k is some non-negative constant, currently set to 20. If $z(LC) = 0$, we do not do any smoothing, that is, $Pr(p_i|LC) = c(LC, p_i)/c(LC)$.

At a word boundary, the issue of sparse data becomes serious for the advancement probability to the first phone of a new word (because many different word endings for the previous word are possible and only a small number of them will have occurred in training data). To mitigate this problem and also to address a difficulty with an overly expensive search, which we will discuss later in this chapter, we have decided to condition the advancement probability for the first phone after a word boundary *only* upon the last phone of the previous word. In other words, we ignore the upper level structure of the left column in this case. This change also makes it possible to merge many linguistic theories at putative word boundaries during the speech recognition search process. Despite the pooling, we nevertheless smooth these across-word-boundary advancement probabilities as before, but letting $k = 70$. The choices for k were obtained through experimentation comparing language model performance on development data for various values of k.

2.4 Lexical Constraints

Thus far, we have described how ANGIE models subword lexical structure. What is missing is how ANGIE goes from a subword structure to an actual word (or some other lexical unit). As alluded to earlier, we accomplish this via either a single lexicon of permissible phonemic sequences for each word, or two lexicons, one of permitted morphemic units for each word and another of permissible phonemic sequences for each morphemic unit. Our earlier experiments use the first method, which is simpler, but our later recognition and some word-spotting experiments use the second method. The second method, although more complex, has the advantage of supplying more constraint to the recognition search process. While both methods constrain words to the same possible phonemic sequences, the second method additionally constrains permitted parse trees to those which have the morph boundaries in appropriate locations with respect to the entries in the lexicon.

2.5 Training Procedure

Our training procedure proceeds as follows. Start with a context-free grammar and an arbitrarily initialized probability distribution. Process the set of training sentences, where

each sentence is a string of words and phones, as follows:

1. Parse the first training sentence, ignoring any zero probabilities encountered.
2. Take the most likely parse, breaking ties arbitrarily, and increment the counters associated with the probabilities used to score the parse.
3. Recompute the probability model given the updated counters.
4. Repeat these steps with each successive sentence.

Our set of training orthographic/phonetic sentence pairs was created by computing a set of forced alignments for approximately 10,000 utterances with the SUMMIT ATIS recognizer ([80]). Because the SUMMIT phone set differs from the phone set used by ANGIE and because the phonological rules admitted by the two systems are also different, a series of text processing steps were applied to the SUMMIT outputs to create our training set. The text processing steps translated phone sequences allowed by SUMMIT but not by ANGIE into corresponding ones acceptable to ANGIE, and also translated what we judged to be less preferable sequences into ones which we found more agreeable. Also, since we evolved our phone and rule sets over time as we developed our system, analyzing failed parses and ANGIE forced recognition (to be discussed further in Chapter 3) outputs, further processing steps were applied to the training set to match it to the evolution of the ANGIE phone set.

2.5.1 Perplexity Experiments

We have conducted some initial experiments on the ANGIE linguistic model in isolation, that is, not as a part of a recognizer. In this experiment, no constraints on phonemic sequences were employed and the training set was that described in the previous paragraph. Then we ran ANGIE's forced alignment recognizer on a set of 895 test sentences drawn from the ATIS December 1993 test set, not part of the 10,000 training sentences. We used ANGIE's forced alignment recognizer to generate the phonetic transcriptions for the test set rather than SUMMIT because we know that it will generate phonetic transcriptions consistent with ANGIE's rule and phone sets. We evaluated the per phone *perplexities*[6]

[6] *Perplexity (PP)* is a common evaluation measurement for language models. It is defined as $\lg\left(\sum_x Pr_{data}(x) \lg Pr_{model}(x)\right)$. It corresponds roughly to the average "branch-out" factor of the model. Thus, a per phone PP of 7 would mean that the model gives roughly the same constraint as if only 7 phones were possible and there is no model. A lower PP is better than a higher PP because it indicates that the model is more constraining.

Set	ANGIE	Phone Bigram	Phone Trigram
Training set	6.07	9.90	4.98
Test set	7.15	14.91	9.20

Table 2-3: Per phone perplexity of ANGIE linguistic model vs. phone n-gram models.

of our ANGIE linguistic model and compared them to traditional phone n-grams. The results are summarized in Table 2-3. ANGIE's sublexical linguistic model appears to be twice as constraining as the phone bigram model when evaluated on the basis of perplexity. Even more promising, it appears to outperform a phone trigram model on test data by a substantial margin.

2.5.2 Maximum Likelihood

A question to ask is whether our training procedure results in a probability model that has the maximum likelihood of generating the training data (in the absence of smoothing, of course). That is, of all the probability assignments possible, are the parameters we arrive at ones which will maximize the probability of generating the training data. The answer in our specific case is, unfortunately, no. The reason is that in step 2, we choose the *most likely* parse and increment counters associated with it. However, which parse is most likely depends on the sequence of training sentences seen so far (but not on the remaining training sentences). Thus, we will not neccessarily arrive at a global maximum likelihood solution.

We see several ways to correct this drawback. One is to train our probability model based on "correct" parse trees instead of on the most likely parse. Such "correct" parse trees can be hand generated or automatically generated and hand verified and corrected. However, because we intend our system to be working with a large amount of training data, this approach is not feasible. We can also go with a partial solution by first initializing the probability model with a small number of "correct" parse trees and then running the training algorithm as stated above. While this will still not result in a maximum likelihood solution, it may result in a better solution because we bootstrap the probability model with known correct data. Finally, another compromise we can make is to settle for what we will call a *locally optimal* solution, that is, we get an initial model from the training algorithm above and then iteratively increase the likelihood of the model until we reach

a local optimum. This can be done in a straightforward manner with the *E-M algorithm* ([17]).

We have tried the locally optimal approach. The resulting model was only slightly better than the model trained with the algorithm as originally stated. From this, we conclude that the grammar we have is "tight" enough, that is, it does not overgeneralize much, to result in a probability model that, although not provable as being maximum likelihood, is close enough to a local optimum to not require E-M iterations during training. Thus, we have adopted the basic training procedure for all future experiments.

2.6 Engineering Issues

In this section, we discuss some engineering issues involving our ANGIE parser. The issues primarily involve maintaining a tractable system, in terms of both time and space complexity. We address in the next three subsections steps employed to prune the search for valid parses, our strategy for managing the memory requirements of the search, and some search design issues involving integration of ANGIE into a recognition system.

2.6.1 Pruning

We employ several pruning mechanisms in our parser's search engine. A *beam prune* restricts the possible theories attached to a given phone string by limiting the maximum number of live theories after each column advancement. We also perform what we have dubbed *siamese twin* pruning in the case where we only want a single most probable parse for a given phone string. If there were two partial theories who agree in their final columns, then the lower scoring one is pruned. Because all future parsing and probabilities only depend on the final column, the pruned theory will always be inferior, so there is no need to keep it around[7].

At word boundaries, we prune much more aggressively. As may be recalled from our description of the probability model, only a phone context is carried across word boundaries. We mentioned two reasons for this. One is to mitigate sparse data problems. The other is to limit the size of the search. We found that the ability to prune based on final phone

[7]We call this *siamese twin* pruning because the two theories are not necessarily identical twins but yet as far as the future is concern, they may as well be, except for the score difference.

contexts at word boundaries narrowed the search space tremendously, especially when ANGIE is incorporated into a recognition engine of some type.

Finally, in the case of training and forced alignment, when we know the word sequence corresponding to the phone string being parsed, we can take advantage of this knowledge by pruning away theories whose parses yield a series of phoneme-like units that do not form legal sequences corresponding to the string of words.

2.6.2 Memory Management

A serious problem with a naive implementation of the parser relates to memory management and usage. During training and forced alignment, this is not too much of an issue because of the filtering by phoneme-like units described in the previous section. However, during recognition, the number of putative phone strings for which we need ANGIE linguistic scores grows very rapidly. Pruning can keep the number of active theories under control, but a mechanism is needed for reclaiming the memory used by pruned theories and also theories for which further advancement is no longer possible. As already mentioned, if two phone strings share a common initial sequence, then they also share a set of common initial partial parses. This serves to reduce memory requirements because of the sharing, but it also serves to complicate memory reclamation. When a partial theory is pruned, we cannot simply reclaim all the memory used by the partial theory because an initial portion of it may be shared with other partial theories which are still alive. To solve this dilemma, we have implemented a *reference counting* memory management systems. Each partial theory is extended in terms of columns going left to right and in terms of nodes going bottom-up within a column. Each node includes a reference counter tracking the number of extensions which depend on that node. The situation is similar for the columns. Whenever a node (or column) structure is disposed of, we decrement the counters for any ancestors to which it refers. If any of those counters are decremented to zero, then we reclaim their memory and decrement the counters of their ancestors, etc. An example showing how the reference counting garbage collector works in the context of our parser is included in Appendex B.

2.6.3 Search Design

We have already mentioned some of the issues surrounding the design of our search within ANGIE. Namely, we suggested that it should be left-to-right in order to support partial

hypotheses during recognition, and that it needs to be bottom-up to support sharing and discovery of word-like structures for handling new words, etc. We would like to bring up several other issues which relate to how ANGIE's parser search interacts with an ultimate recognition engine's search over the acoustic space. If we consider the interaction for a moment, we realize that there are actually two search spaces which need to be merged together: namely, an acoustic space and a lexical space. A given string of lexical phone units can be associated with numerous paths through an acoustic phone graph. Similarly, a given acoustic sequence can correspond to numerous lexical hypothesis, since multiple phonemic sequences can correspond to the same phonetic realization. It is even possible to have multiple words realized as the same phonetic sequence. This is the case not only with homonyms, but also with highly reduced realizations of function words. An issue is how to tie the acoustic and lexical theories together. We have adopted a strategy of simultaneously pursuing multiple lexical theories which correspond to the same phonetic string, and bundling them into one unit. The score used to represent this unit is taken to be that of the highest scoring theory bundled in the unit. Certainly, this policy of letting lexical theories ride "piggy-back" introduces search errors. Later on in this thesis (Chapter 5), we will discuss how to represent the interaction of word theories with subword theories. In that case, we will find that an attempt to bundle leads to decreased performance. However, to represent the entire cross-product of possible lexical theories and possible acoustic theories proves too unwieldy because of the bushiness of the search spaces for these theories. Some practical choice is mandated, and the solution presented there is the one we selected.

A natural optimization we would like to make is to also have multiple acoustic paths which share the same phone sequence use the same bundle of lexical theories, as opposed to having duplicate bundles. This seems obvious and uncontroversial, in that it does not introduce any further search errors, but this also has a ramification for the design of the ANGIE probability model. Namely, if we recognize that the final output desired of a recognizer is some type of word sequence, we realize that once words have been proposed and the subword score accounted for, we are no longer concerned with the exact ANGIE parses which generated the word hypothesis.[8] Thus, if the subword structure were discarded upon arrival at a word ending, we can increase our sharing dramatically. Namely, if two theories

[8]This is not strictly true. A long range prosodic model may actually care about subword structure over several words. However, for the purposes of the work presented in this thesis, this simplification holds.

share the same phone string hypothesis, *for the final word currently being pursued in the recognition search*, then they can share the same ANGIE lexical-theory bundle. This represents a dramatic increase in sharing as compared to only permitting the sharing along the entire phone string hypothesis since the beginning of the utterance match. Our original ANGIE design did not take this sharing into account and created a totally intractable search when we attempted to deploy the framework in an actual recognition task.

We had mentioned that this sharing has an implication for the design of ANGIE. The implication is that we should be designing our framework to maximize sharing at word boundaries. One of the simplifications we presented under the section on pruning (sec. 2.6.1) was actually a second generation design we incorporated to support this. Obviously, we are referring to the reduction of the advancement probability's conditional context to a single phone, rather than an entire column, at word boundaries. This reduction allows us to discard all subword information except the final phone, once a word has been proposed during the recognition search process. All acoustic theories which represent the same phone sequence for the next word being searched, and which share the same ending phone, can then share the same ANGIE bundle, yielding a savings in both memory requirements and computational requirements[9].

2.7 Summary

In this chapter, we introduced the ANGIE framework for sublexical linguistic modelling. ANGIE is a unified computational framework capturing morphology, syllabification and phonology through a layered hierarchical structure. At the heart of ANGIE are a context-free grammar, a bottom-up breadth-first left-to-right parser and a probability model. The probability model consists of two types of probabilities, advancement probabilities and bottom-up trigram probabilities. Our hope for the probability model is for it to capture both some of the context-dependencies commonly believed to govern phonological processes and also to account for variability. We believe that accounting for variability is crucial in a system that needs to handle errorful inputs.

Lexical units in ANGIE are tracked via a listing of legal phonemic sequences or two

[9]The ANGIE bundle is only computed once rather than repeatedly for each acoustic theory, reducing computational needs.

listings, one of legal morphemic sequences and another of legal phonemic sequences for the morphemic units. We have explored some of the implementational issues involved, including time and space complexity concerns. We discussed pruning, memory management, and issues relating to organization of theories during the search process.

Finally, we have achieved favorable perplexity results when comparing ANGIE to a phone bigram and phone trigram. ANGIE achieves a test set perplexity of 7.15 as compared to a phone bigram's 14.91 and a phone trigram's 9.20. We also observed that ANGIE surpassed the phone trigram on testing data but not on training data, suggesting that while the phone trigram may actually learn the training data better, ANGIE's model generalizes better to unseen test data. While perplexity results do not necessarily lead to improved recognition performance, there is widely believed to be some correlation between the two, and, nevertheless, the positive result is promising.

Chapter 3

Phonetic Recognition

In this chapter, we describe the initial implementation of a recognizer based on the ANGIE sublexical framework. The recognizer will be focused on the task of phonetic recognition. We feel that phonetic recognition is an appropriate task for an initial evaluation of ANGIE for several reasons. It is a less computationally intensive task than continuous speech recognition of words, but it does have the benefit of being correlated with word recognition performance so it is a realistic indication of the framework's efficacy. Also, since ANGIE is first and foremost a subword modelling framework, evaluation on a task that does not involve the presence of a word lexicon seems an appropriate starting point. Any defects in the design of the framework can be more easily isolated in the absence of words.

Building this recognizer involves several challenges. We need a set of acoustic models for each of the phones in our phone sets. To accomplish this, we need a system which can perform forced alignment. With such a system, we can seed the acoustic models from an existing recognizer (SUMMIT was used in our case), and then iteratively improve the acoustic models by performing a forced alignment pass and then retraining the models. We also need front-end acoustic signal processing for our system. However, the main issue underlying both the forced alignment system and the phonetic recognition system will be the search strategy needed to combine the information from the acoustic models and the ANGIE sublexical model and output a reasonable hypothesis. In the sections which follow, we will address all of these points and present some numeric results of our phonetic recognition system. The goals at this point are to develop an ANGIE based system for training acoustic models, for helping to identify possible problems with our sublexical modelling framework

and in our search strategy, and to provide a first test of the ANGIE system in an actual, recognition task involving acoustic data.

We will show that on the task of phonetic recognition within the ATIS domain, our ANGIE-based system outperforms a baseline system implemented using the MIT SUMMIT ([79]) recognizer employing a phone bigram sublexical model. The recognition error rate decreases from 39.8% to 36.1%, with approximately 1.6 percentage points of the decrease attributable to the addition of the upper layers in our ANGIE framework and the remainder to the more pure acoustic models trained with the aid of an ANGIE forced alignment system. In the previous chapter, our positive results with perplexity experiments empirically contributed to the validation of our phonological framework. The favorable outcome in this chapter, involving an actual deployment within a recognition system, supports our belief that we have a workable framework for recognition.

3.1 Front-End

Our system has the same basic front end as the segment-based MIT SUMMIT system. The preemphasized input audio samples undergo a short time Fourier transform (STFT), and are passed through a bank of 40 triangular filters ([48]) to produce Mel-frequency spectral coefficients (MFSCs), which are then transformed via a discrete Fourier transform (DFT) into 14 Mel-frequency cepstral coefficients (MFCCs). The speech signal is also segmented into a segmentation graph (c.f. section on acoustic segmentation in [79]). Acoustic measurements are computed for each segment. Each segment in the graph may be a proposed phone. Our set of measurements are very generic: Averages of the MFCCs across the first, middle and final thirds of each segment, derivatives slightly in from the left and right boundaries, and absolute segment duration, a total of seventy-one measurements. A principal component analysis is performed on the measurements. Context-independent mixture diagonal Gaussian acoustic models are created for each unit in our phonetic inventory. Each phone is actually associated with two mixture models, one for the absolute duration measurement and another for the remaining measurements. We have five and ten mixtures for each, respectively. We also process the acoustic data through a causal energy based silence detector, similar to that described in [38]. We do this because our segmentation algorithm generates a very densely populated graph around silence regions, which tends to slow down our search

considerably. We used the information from the silence detector to do some extra pruning in detected silence regions during the early development of our system. As we progress to the next chapter on word-spotting, the silence detector will actually be discarded as the acoustic models will have been adequately trained by iterative training to render the silence detector extraneous.

3.2 Recognition Search

An important module of any speech recognition system is the search component. We have already discussed the generation of a segmentation graph, with phone scores for each segment in the graph. However, in order to generate an output hypothesis, be it a string of words, as in a normal speech recognition system, or a string of phones, as in our phonetic recognition system, we need to find a "good" path through the segmentation graph[1]. In determining what is "good," we need to take into account both the phone scores computed as described in the previous section along with the constraints on valid phone sequences and their scores, which will be provided by ANGIE in our case. For small enough segmentation graphs, an exhaustive search is possible, yielding an *optimum* result, that is, finding a path which is highest scoring. For reasonably sized graphs, such a search, sometimes referred to in artificial intelligence literature as a *British Museum search*, will not be computationally tractable. Nevertheless, optimal search algorithms do exist if certain constraints can be imposed on the organization of the search graph and also on the scoring model. We do not mean to imply, however, that optimal search is necessary. In many cases, a suboptimal search will find an adequate solution.

Almost every speech recognizer has some kind of a search strategy. Some systems use a single pass search; others use a multiple pass search, with one pass generating a set of hypotheses to be filtered by the next pass. Each pass may consist of a single search algorithm or may consist of multiple algorithms operating in a dovetail manner — that is, one search algorithm activates another search algorithm operating in parallel. At the most basic level, most of the search algorithms used in speech recognition can be subdivided into

[1]While we focus our exposition on searching through segmentation graphs because we are using a segment-based framework, many of the comments apply to frame-based systems as well, except the search graph will include an assignment of frames to putative phones, hence conceptually creating an analogous segmentation graph.

two broad subcategories: *dynamic programming approaches* and A^* variants, of which the widely used *stack decoder* ([30], [56], [55], [57]) is one.

3.2.1 Dynamic Programming Approaches

Several recognizers employ a dynamic programming approach ([13], chap. 16) in the search component. In order to employ a dynamic programming approach, the search problem must be cast into a form such that there is an optimal substructure property, that is, the optimal solution of a larger problem involves finding the optimal solutions to smaller problems; and an overlapping substructure property, that is, the solution to a smaller subproblem is used repeatedly. The most widely used technique in this class for speech recognition is the *Viterbi* search ([74]). MIT's SUMMIT ([79], [80]) system and Philips' research system ([53]) are examples of systems which use a Viterbi search in an early pass. With a Viterbi search, we have a graph, frequently known as a *trellis* or a *lattice*, where there are a series of nodes associated with each time boundary and there are edges associated with phonetic units[2]. For expository purposes, let us assume that we are working with a system where each edge can connect only nodes associated with adjacent time boundaries. Our trellis is then a very regular two dimensional table with time on one axis and possible phonetic units on the other.

Let $t \in 1..T$ be the time boundaries and $p \in 1..P$ be the phonetic units. Denote a node in the graph as $n_{t,p}$ for the node corresponding to time t and phonetic unit p. An edge from n_{t,p_1} to n_{t+1,p_2} corresponds to phonetic unit p_1 having been spoken to get to time boundary t and phonetic unit p_2 to get from time t to $t+1$. Let $E_{t,p_1,t+1,p_2}$ be the score of this edge computed from the acoustic model. Similarly, let $S_{t,p}$ represent the best score of a path from the beginning to node $n_{t,p}$. The Viterbi algorithm then implements the following dynamic programming recursion:

$$S_{t,p} = \max_{1 \le i \le P} S_{t-1,i} + E_{t-1,i,t,p}$$

Observe that in this example, all the scores for time t must be computed before we can compute the scores for time $t+1$. For this reason, the Viterbi search is referred to as a *time synchronous* search. In general, this strict time synchronicity can be relaxed somewhat

[2]Or perhaps syllable or word units.

depending on the structure of the trellis. For example, in a segment-based system like MIT's SUMMIT ([79]), the edges are associated with segments which may skip certain time boundaries. Nevertheless, the search proceeds in a generally time synchronous fashion.

The Viterbi search, as described, is optimal in that the backtrace of the highest scoring path is indeed one of the highest scoring paths. In general, this optimality is guaranteed only if the score of each edge in the path depends only on that edge. Thus for example, if we were searching through a word trellis, the use of word bigram scores satisfies this requirement because each edge identifies the two words needed to compute the bigram score. However, if we were to use trigram scores, we would have to expand the trellis such that each node represents a two-word context in order to use a Viterbi search. This is one of the major drawbacks of Viterbi search. Any attempt to incorporate longer distance information will increase the search complexity greatly, in terms of both time and space, because of the need to add many nodes to adequately capture context. Another significant drawback of the Viterbi search is that it can only yield the best scoring answer and not a list of top scoring answers. A list of top scoring answers, also known as an N-best list, would be useful if we were performing a multi-stage search where the later stage either resorts or filters the list.

The Viterbi search runs in $\theta(TP^2)$ time, which is one of its greatest attractions. The *stack decoder* search and other variants of an A^* search (described in the next section) do not have a nice polynomial bound on time complexity. In practice, some form of beam pruning is usually employed to accelerate the Viterbi search further. Typical approaches might be to eliminate all nodes which fall below a certain absolute score threshold or to eliminate the nodes at a given time boundary which are not within a certain threshold of the best scoring node.

As mentioned earlier, the classic underlying common element in a dynamic programming approach is the organization into a search lattice such that there is an optimal substructure property and that there are overlapping subproblems. In speech recognition, these two criteria are met when we organize the search along a time line and restrict a scoring model to considering only local information. Thus, in the case of the Viterbi search, finding the optimal solution at time t involves finding the optimal solutions to each possible node at time $t-1$. In this case, the time $t-1$ problem is a subproblem of the time t problem. The local constraint requirement insures an optimal substructure property. Without it, at time t, we may have an optimal solution which does not involve an optimal solution to

any time less than t. For example, a trigram model may favor a given three unit sequence tremendously. The optimal one or two unit sequences are of little consequence once we reach a third unit which completes the favored sequence. Finally, because many nodes need to be considered at time t, for the different possible phone extensions from time $t-1$, we have to look at the optimal solutions at time $t-1$ on multiple occassions, hence, we have overlapping subproblems.

3.2.2 A^* Search

The basic A^* search proceeds as follows ([76], pp. 94, modified to suit the environment of a speech recognition system):

1. Form a one-element priority queue with an initial path consisting only of the start node
2. Until the highest scoring path in the priority queue reaches the end of the utterance or the queue is empty:
   (a) Remove the highest scoring path from the priority queue
   (b) Find all possible extensions of the path
   (c) Insert the extended paths back into the priority queue, with a score consisting of the partial path score plus the *future estimate*, an estimate of the highest score until the end of the utterance.
3. If we reach the end of the utterance, we have succeeded, else, we have failed

If the *future estimate* were always an underestimate, that is, it never estimates a score higher than the highest actual score to the end of the utterance, then the A^* search results in an optimal solution, in that the path we have found is a highest scoring path. (This is often referred to as an *admissible* search.) If we have no future estimate, then the search becomes a *best-first search.*

The primary benefit of an A^* approach over a dynamic programming approach is that it allows for the incorporation of arbitrarily long-distance constraints and that, by continuing to run the search after obtaining the first answer, we can get multiple answers in declining order of score, yielding an N-best list. However, we pay a great cost in terms of time

complexity. Unlike the Viterbi search, there is not a nice polynomial bound on an A^* search. In practice, much pruning needs to be done and the result is often some type of beam search. Also, in speech recognition, it is usually not possible to find an admissible future estimate. Some type of suboptimal heuristic estimate is frequently used. In some cases, no future estimate is used, resulting in a best-first search, which was actually how we implemented our phonetic recognition system[3]. The most popular A^* variant in speech recognition is the stack decoder, whose description we defer until the chapter on word-spotting (Chapter 4).

3.2.3 Combinations of Searches

In speech recognition systems, multiple searches are frequently used. For example, in the MIT SUMMIT system ([79], [80]), a forward Viterbi search is used to generate a word-graph, which is then searched backwards via an A^* search. In that implementation, the Viterbi scores are used as the future estimates during the second pass. In the 1994 BBN BYBLOS speech recognition system [54], a five pass search is performed, with some passes going forwards and others going backwards! Multiple pass searches are typically employed so that a faster running search can be used in earlier passes to generate a pruned graph or an N-best list, with more detailed but slower running searches deferred until a later stage.

3.3 Our Best-First Search

For our recognizer, we have implemented a single pass best-first search of the segmentation graph. The score of each recognizer theory is the combination of the acoustic scores associated with the phones and segments in the theory, the ANGIE score[4], with some heuristic adjustments, and some heuristic weights.

The heuristic weights are necessary to overcome a common problem with best-first search strategies being employed in a system where scores are tied to probabilities attached to each segment in a path. Because probabilities are always less than or equal to one, and because

[3] In the absence of a future estimate, it may be necessary to introduce other heuristic mechanisms to normalize the scores of long vs. short theories, particularly in systems based on probabilistic models since longer theories will tend to have lower probabilities. We will discuss our set of heuristics when we describe our best-first search.

[4] Recall from chapter 2 that we can have multiple ANGIE parses for a given string of phones. We take the score of the highest scoring parse and use that in our recognizer theory.

logs of probabilities are typically used as scores[5], the scores are always negative. As a path becomes longer, there is an ever increasing negative bias on the scores. The net result is that a best-first search will always tend towards extending the shorter paths, making very slow progress to the end of the utterance. In an A^* search, a reasonable future estimate can help counteract this bias. In our case, we introduce several heuristic rewards to mitigate the situation. We include:

Phoneme reward A constant is added for each phoneme in the theory.

Time reward To favor progress along the time line in the theory, a value proportional to the length of the theory, in terms of units of time, is added.

Thus, as a path becomes longer, it will get an extra "boost" fpr having more phonemes and more time accounted for. Of course, if the rewards were not chosen carefully, we may end up "rushing ahead" with an inferior theory. By this, we are referring to the case when an inferior theory happens to account for more time or more phonemes and is favored at the expense of a potentially better theory which happens to be shorter at the time. This then becomes a self-fulfilling prophecy as the longer, inferior theory is extended.

Despite the addition of these heuristic weights, we had discovered that by themselves, they did not prove adequate empirically to enable our recognizer to succeed. We explored numerous other heuristics and settled upon the following scoring adjustments in our recognizer:

Mean-sigma acoustic score normalization Normalize the scores from the mixture diagonal Gaussian models by subtracting the mean and dividing by the variance of that model as scored on a forced alignment of training data.

Mean entropy ANGIE **score normalization** Modify the ANGIE probabilities by subtracting the mean training data entropy over all phones following a specific left phone context.

The idea behind these rewards is to give theories *a score of zero on average*, so that those which are better than average get positive scores and those which are worse than average

[5]Computational precision is much better with addition of log probabilities than multiplication of probabilities. The latter tends to be rounded down to zero after a few multiplications.

get negative scores. The net result of the *mean-sigma acoustic score normalization* is such that on a forced alignment run over training data, the acoustic models will be normalized to have an average score of zero and a variance of one. The *mean entropy normalization* results in an average linguistic score of zero following every phone on ANGIE parses of the training data. We have experimented with several normalizations of the ANGIE scores. We have also considered offsetting by the mean entropy following a left-phone context and also for no context but for the specific phone being predicted. Of all of these, we have found offsetting by the mean of the left phone context to be the most effective.

We are not particularly pleased with the use of so many heuristics in the search strategy. In the next chapter, when we discuss word-spotting, we will switch to a different search strategy which both proves much more effective and requires fewer heuristics.

3.3.1 Pruning

As mentioned earlier in this chapter in the introductory material about search, various pruning strategies, both admissible and inadmissible, are often used to speed up the system. We implemented several pruning mechanisms in our search:

Prune inferior theories at same time boundary If we have multiple recognizer theories which end at a given time boundary and they share the same ANGIE linguistic theory, then we prune all but the highest scorer. (The different theories may result from different paths through the segmentation graph.)

Acceptable acoustic scores Do not consider theories where a hypothesized phone has an unacceptably low score. Require a minimum threshold on acceptable acoustic scores.

Maximum queue size During our best-first search, limit the number of live theories by limiting the total size of the queue. If the queue size exceeds the limit, prune the lowest scoring theories until it returns within the limit.

Maximum paths per time boundary At any time boundary, after encountering a certain number of theories, only consider another theory at that boundary if its score exceeds the average score of the theories already seen. We actually have two thresholds. An absolute one for all boundaries and another smaller one for boundaries which

are more than a predetermined amount of time behind the theory which has advanced the most timewise.

The first of these is an obvious admissible pruning strategy in that it will not result in a suboptimal search. Actually, since our ANGIE probability model only relies on the left phone context across word boundaries, we can compare only the final phones at word boundaries and still have an admissible pruning strategy. We adopt this improvement.[6]

The others are inadmissible and result in a type of beam pruning. We have tried a variety of strategies to make our search run under acceptable computational resources. For example, we had also tried a variation of the "maximum paths per time boundary" pruning where we consider instead the number of theories which *cross* a boundary[7]. We have found that alternative to be ineffective.

A more detailed description of the best-first search implementation used in our phonetic recognizer is given in Appendix C.

3.3.2 Forced Alignment

Thus far, the recognizer we have described works for both the forced alignment task as well as the phonetic recognition task. However, during forced alignment, because we know the correct orthographic transcriptions of the utterances, we can apply additional constraints during the search process to prune away theories which do not correspond to the given orthographies. As we discussed in section 2.6.1, we can prune paths whose phone sequences do not correspond to linguistic theories with the correct sequence of phonemes for the given orthography. Furthermore, as we arrive at each word ending, we can check to see if the words associated with the linguistic theory match the orthography, and, if not, we can prune the path. A sample forced alignment output is shown in Figure 3-1.

[6]Originally, our first ANGIE implementation carried the full column context across word boundaries. However, driven by both the need to speed up the search process, and to mitigate sparse data problems, we reduced the context to be left phone only. Without this simplification, we had great difficulty controlling both the time and memory resources required by the search process.

[7]Because our segmentation graph has overlapping segments, not all theories will have to include a certain time boundary. Some theories may include a segment that crosses but does not end or begin at a given boundary.

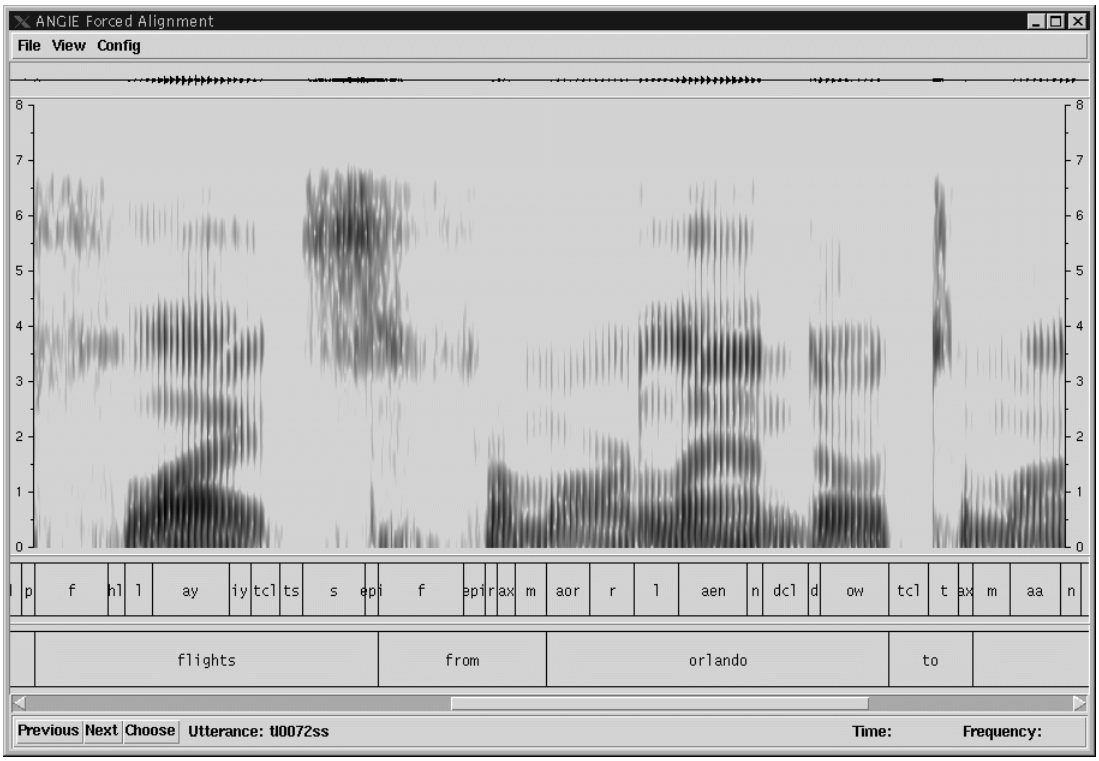

Figure 3-1: Sample output of forced alignment process. Shown are the word and phone alignments for the partial utterances "...flights from Orlando to Mon[treal]..."

3.4 Phonetic Recognition

Our first experiment in evaluating the feasibility of ANGIE as a sublexical model for a speech recognizer is on a phonetic recognition task. In Chapter 2, we had observed that ANGIE achieves respectable performance in terms of phone perplexity as compared to bigram and trigram phone model (7.15 vs. 14.91 and 9.20). The strong language model constraints provided by ANGIE will hopefully result in better recognition. The experiment here will be the first test to determine whether this is indeed the case or not.

To use ANGIE within a phonetic recognition system, we permit ANGIE to propose word-like structures, which we will call "pseudo-words," based on its set of rules and probabilities. Thus, no explicit word (or syllable) constraints are employed. ANGIE can, and does, propose novel and non-sensical syllables, although their probabilities will be lower because they are not well supported by training data. However, there is an ordering constraint at the syllable layer, so that for example, if a bottom-up ANGIE parse results in a prefix syllable, it would be permitted only in the prefix position of a pseudo-word.

Because the ATIS corpus is not phonetically labeled, we choose to use the forced align-

ment outputs of our own recognizer as the correct reference answers. We believe this to be a reasonable approach because the ultimate goal is to evaluate the feasibility of our framework for a full word recognition system, and it is the case that if we can engineer a phonetic recognizer to output the phone strings of the forced alignment process, then we would have perfect recognition. Thus, measuring how close we can get to that point, in terms of *phone error rate*[8] is a reasonable approximation. This approach has been used before for evaluating phone recognition on corpora without expert-created phonetic transcriptions. For example, Ljolje ([43]) used this technique with the Resource Management corpus ([58]).

In this experiment, we choose as our training data a 5000 utterance subset of the ATIS-3 corpus. As our test data, we use the ATIS December '93 test set of just under 1000 utterances. We train our ANGIE probabilities on training data that includes the orthographies, but during the actual phonetic recognition process, we do not impose any constraints on what constitutes a "word" other than that from the ANGIE grammar and probabilities. In other words, we use our ANGIE model to guide the recognizer search process in proposing word-like structures, but we do not require that proposed pseudo-word boundaries result in a sequence of actual words in our lexicon. For example, if the system is able to propose a word boundary after the phone sequence [s ae n f r ae n], which may correspond to the beginning of the word san_francisco, then we will permit it to do so although no baseform in the lexicon permits a word boundary at that location. The system is also able to propose totally nonsensical words, such as the phone sequence [ae f th], which might be written in letters as "afth," as shown in the Figure 3-2. For comparison, we created a baseline phone recognizer based on the SUMMIT architecture, using a phone bigram as the sublexical model.

There are two possible baseline comparisons. One is to take the phone set and acoustic models from the SUMMIT ATIS recognizer ([80]) along with the forced alignments from the full SUMMIT ATIS recognizer as the correct reference phone strings to score against. In this comparison, we are exploring the effects of both ANGIE's higher level modelling, that is, its inclusion of higher level sublexical structure beyond a simple phone bigram, and of ANGIE's phonological flexibility. We believe that ANGIE's representation allows for better phonological modelling than a pronunciation graph generated through the application of

[8] *Accuracy* is defined as the percent correct minus the percent of deletions and insertions. The substitutions, deletions and insertions are calculated based on an alignment of the reference and hypothesis strings which maximizes the accuracy. The term *error rate* is defined as 100% minus the accuracy.

phonological rules along with iteratively trained arc weights[9]. This allows us to have cleaner acoustic models for the phones because we can absorb more of the phonological variation into the sublexical model and avoid polluting the acoustic models.

On the other hand, we can run the SUMMIT recognizer using the exact same phone set and acoustic models as ANGIE, so that the only difference is in the sublexical models[10]. This comparison will isolate the improvement due to the higher level structure modelled by ANGIE but not by the phone bigram.

The first comparison is probably more worthwhile if we take the view that phonetic recognition is just a stepping stone to a full word recognizer and we are comparing against a baseline for a full working ATIS recognizer. However, the second comparison is nevertheless informative because it helps indicate whether the additional layers in ANGIE's hierarchical model are doing anything useful. Unfortunately, neither comparison allows us to isolate the effects of our best-first search process as compared to SUMMIT's Viterbi search. A final complication arises in that the different phone sets will make a direct comparison difficult. However, we can always project both phone sets to a single phone set.

An example of the output from our phonetic recognition system is shown in Figure 3-2. In this example, several phones are correctly hypothesized such as the [d ey y iwt q ae f] sequence. Also included in the figure are hypothesized pseudo-word boundaries, that is, places where the ANGIE model finds it probable to terminate a pseudo-word. These are not used to score the phonetic recognition results, but are included in the figure to provide the reader with a sense of what pseudo-word boundaries are proposed by ANGIE. Some of the hypothesized pseudo-words correspond to correct words in the utterance, e.g., "[Wed]nesday." Others correspond to reasonable real words in English, but incorrect words for this particular utterance, e.g., "the" and "fine." Finally, there are also totally nonsensical words in English, which are considered word-like by ANGIE's rules and probabilities, e.g., "afth."

The results of the three experiments are summarized in Table 3-1. We have mapped all the phones to the *CMU 39 phone set* ([40]), commonly used in the literature, for comparison purposes. The complete mapping of ANGIE phones to CMU 39 phones is shown in Table 3-2.

[9]The generic pronunciation graph framework can model all the variations that ANGIE can. However, the lack of the upper sublexical layers may force us to create more lax, that is, more overgenerating, phonological rules than we would in ANGIE.

[10]SUMMIT is very flexible in terms of which set of acoustic models and phones it uses.

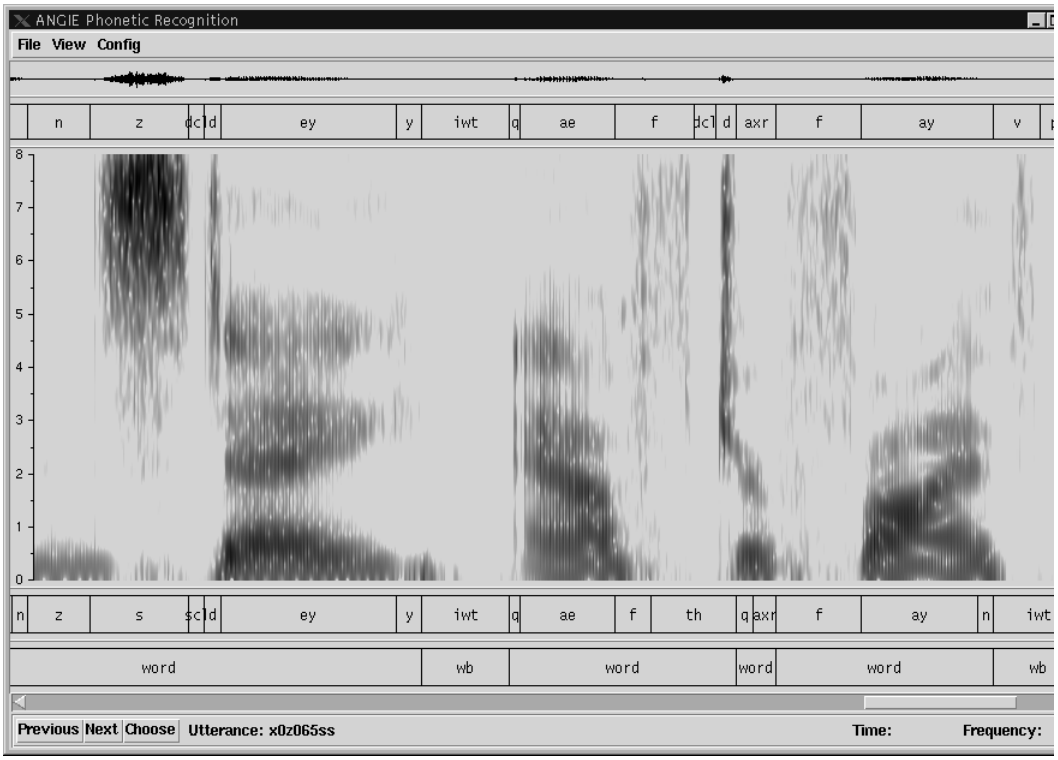

Figure 3-2: Sample output of the phonetic recognition system. Above the spectrogram is the forced alignment system's output, taken to be the reference answer. Below the spectrogram are the phonetic recognizer's phone string output and its pseudo-word boundary hypotheses. The utterance shown is "[Wed]nesday after five." The phonetic recognition system hypothesized word boundaries which correspond to "[Wed]nesday afth er fine." The phones shown are from the actual ANGIE phone set, before conversion to the CMU 39 phone set.

Recognizer	Sub	Del	Ins	Overall Error
Phone bigram (SUMMIT ATIS phones/models)	19.2%	10.6%	10.1%	39.8%
Phone bigram (ANGIE phones/models)	18.7%	11.5%	7.6%	37.7%
ANGIE	20.8%	7.9%	7.4%	36.1%

Table 3-1: Phonetic recognition results

We are encouraged by the positive results of the ANGIE recognizer and sublexical model.

3.4.1 Analysis

The overall error rate decreased from 39.8% with the SUMMIT ATIS context-independent baseline to 36.1% with ANGIE. We hypothesize that there are two primary factors contributing to the improvement. One is that, because ANGIE provides a more flexible phonological model, it can choose more precise acoustic segments, allowing the acoustic models for the phones to absorb less variation, thus, becoming more pure. The other factor is the strong language model that describes probabilistically the syllable structure of English. This is manifest in the low perplexity ANGIE realizes on test data. The combination of of these two factors was effective in reducing the error rate by 9.3%. To better tease apart the contributions of each of these factors, we can examine the experiment with the phone bigram and ANGIE phones/models. That experiment resulted in an intermediate error rate of 37.7%. This suggests that the cleaner acoustic models were responsible for 2.1 percentage points of the error rate reduction and that the other 1.6 was contributed by the additional linguistic information provided by the upper layers of the ANGIE framework.

Table 3-3 summarizes the top errors in each category: substitutions, deletions and insertions. The substitutions for all three experiments look like reasonable confusions, such as vowel confusions, with perhaps the exception of the [cl] $\rightarrow$ [dh] substitution in the ANGIE case. We suspect an explanation may be that the function word "the," which is often sloppily realized, is being inserted to fill in noisy silence regions. Thus, the noisy silence regions, included under the [cl] phone, are misrecognized as [dh], a phone very prevalent in "the." The high incidence of insertion of [dh] is probably due to the same reason, insertions of "the." We have noticed other indications that function words may be overweighted in ANGIE as well, and will attempt to address this problem in subsequent work. The other top errors do not appear to be remarkable.

aar, aor, aar r, aor r → aa r	k → k
ae → ae	l → l
aen, aen n → ae n	m → m
ah, ax → ah	n → n
ow ix, ow iy, ao ix, ao iy → oy	ng, ng n → ng
ao, ow, aa → aa	p → p
aw → aw	r → r
ay, ay iy → ay	sh → sh
b → b	th → th
ch → ch	dx ti → dx iy
d → d	tcl ti → cl iy
dh → dh	tcl t ti → cl t iy
dx → dx	ti → dx iy
eh → eh	tr r, tr w → t r
ehr, ehr r → eh r	t, tr → t
ey → ey	kcl ts, kcl ts s → cl k s
ey y → ey y	ts, ts s → t s
f → f	s → s
fr, fr r → f r	uh → uh
g → g	uw, ux → uw
hh → h	v → v
hl, hl l → l	w → w
ix, ih → ix	y → y
iy, iy y → iy	z → z
jh → jh	hv, scl, bcl, pcl, dcl, tcl, gcl, kcl, q, epi, iwt, iwt2, *pause* → cl

Table 3-2: Mapping of ANGIE phones and phone sequences to phones and sequences in the CMU 39 set. "cl" designates the closure phone in the CMU 39 phone set.

Phone Bigram (SUMMIT ATIS phones)			Phone Bigram (ANGIE phones)			ANGIE		
Sub	Del	Ins	Sub	Del	Ins	Sub	Del	Ins
z → s	cl	cl	aa → ah	cl	cl	ix → eh	cl	cl
k → t	ix	t	m → n	ix	n	ah → eh	n	n
m → n	n	n	eh → ix	n	z	m → n	l	ix
d → t	ah	f	n → m	ah	l	ah → aa	ix	v
l → w	l	r	k → t	l	t	cl → dh	ah	dh

Table 3-3: Top five phonetic recognition errors

3.5 Summary

In this chapter, we discussed the design of a phonetic recognizer utilizing the ANGIE framework as its sublexical model. We presented some background on the issue of search and discussed the implementation of a best-first search strategy in our recognizer. We used our recognizer to both iteratively train acoustic models (in forced alignment mode) and as a phonetic recognition system. We provided some preliminary baseline comparisons for our phonetic recognition system. The comparisons were performed on outputs normalized into the CMU 39 class phone set. Because we do not have human created phonetic transcriptions for ATIS, forced recognition transcriptions were used as references. The early results were promising, showing that ANGIE is competitive with a phone bigram in phonetic recognition. The ANGIE-based system acheived an error rate of 36.1%, lower than the baseline phone bigram system's error rate of 39.8%. A close evaluation suggested that improved phonological modelling accounted for roughly 2.1% of the gap and the more powerful language model due to the longer distance, upper layer information from the ANGIE framework accounted for the other 1.6%. An analysis of the phonetic recognition results suggested that our recognizer may suffer from excessive insertions of function words, particularly "the." Care will have to be taken to address this problem in subsequent work.

Chapter 4

Word-Spotting

In this chapter, we describe a word-spotting system based on ANGIE. The task of word-spotting is to detect the presence of a set, typically a small set, of keywords in speech. We will be testing our ANGIE-based word-spotter on trying to detect the city names in ATIS. Our goals in implementing a word-spotter are several. We want to further evaluate the feasibility of ANGIE. We want to work towards a continuous speech recognition system. One of the pieces missing from the phonetic recognition system was the lack of lexical constraints on what phonemic sequences form legitimate lexical units, typically words, in the lexicon. A word-spotting platform allows us to experiment with including such constraints in the search process in a more controlled manner than attempting full recognition. In the limit, we can think of a word-spotter with many keywords as approaching a full recognizer with the keywords being the vocabulary. Word-spotting is a task which typically includes a combination of both known words, the keywords, and unknown words, usually modelled via filler models. This combination allows us to experiment with differing sublexical constraints on the unknown word space.

We will begin this chapter with some background information on the standard evaluation methodology, which we adopt, for word-spotters, along with a brief summary of prevalent approaches to the word-spotting task. Next, we will describe our implementation of a word-spotter using the ANGIE framework. Finally, we conclude with the results from a series of experiments involving a range of subword constraints on the filler model, from simple phone bigram to full word recognition, conducted within the ANGIE-based word-spotter.

4.1 Background

4.1.1 Evaluation Methodology

In a word-spotting task, the goal is to detect the appearance of one or more keywords in an utterance. However, detection percentage by itself is insufficient as an evaluation measure. We would not want a system which frequently claims that a keyword is present when in fact, the keyword is not present. Thus, we must take into account the number of *false alarms* as well. The two obvious extremes in the tradeoff between detection and false alarms are to have no detections/no false alarms or 100% detection and lots of false alarms. Naturally, neither of these operating points are desirable. Instead we want a point in between the extremes. Better yet, we want to be able to choose an operating point along a curve of possible operating points.

The curve to which we have referred is known as the *receiver operating characteristics (ROC)* curve and is a fairly common way of characterizing a word-spotter's performance. Typically, we have detection rate on the vertical axis and false alarms per keyword per hour of speech (fa/kw/hr) on the horizontal axis. Both axes are usually reported on a percentage basis, so for example, 5 fa/kw/hr refers to 0.05 false alarms per keyword per hour. Detection rate at n fa/kw/hr ($p(n)$) is computed as follows. For each keyword, order the hypothesized detections by descending score. Mark each hypothesized detection as a hit or a false alarm. A hit is defined as the midpoint of the hypothesized keyword time span falling within the time span of an actual occurrence of that keyword. $p(n)$ is then the number of keywords scored above the n'th false alarm[1] divided by the number of keyword occurrences in the test set. A more detailed description of this procedure is given in [73].

To facillitate comparison of various word-spotting systems, a summarized *figure of merit (FOM)* ([73], [6]) is typically cited. The FOM is the detection rate averaged over one through ten fa/kw/hr. The calculation of the FOM accounts for boundary data points on the ROC curve and is given by:

$$FOM = \frac{1}{10T}\left(\alpha p(N+1) + \sum_{j=1}^{N} p(j)\right) \tag{4.1}$$

$$\alpha = 10T - N \tag{4.2}$$

[1]This is summed across all the lists for all the keywords. The n'th false alarm is the n'th time a false alarm occurs in each list.

$$T = number\ of\ hours\ of\ speech \tag{4.3}$$

$$N = first\ integer\ \geq 10T - \frac{1}{2} \tag{4.4}$$

For our evaluation, we will be considering the set of city names in ATIS as our keywords. The list of keywords along with their frequency of occurrence in training and test data are given in Table 4-1. We chose this particular keyword set because work on word-spotting with this set of keywords and in the ATIS domain[2] has been performed in the area of varying acoustic units by Manos ([45]), and we can use the results there as an approximate guideline for judging the competitiveness of our system.

4.1.2 Typical Approaches

There are two primary approaches to word spotting. One of them is to perform full continuous speech recognition and then detect the presence of the keywords in the recognized speech ([75], [65]). While deceptively primitive and computationally expensive, this approach seems to yield the best results in many cases ([45]). The competing approach is to model the keywords and the background "garbage" or "filler" separately. Either a frame-based HMM approach (e.g., [62]), a segmental approach (e.g., [45]), a neural network (e.g., [1]), or a combination (e.g., [42]) can be used for the word spotter. While this approach does not typically outperform the continuous speech recognition approach, it does have the benefit of requiring less computation. Also, there is the possibility of combining background filler models with a continuous speech recognition approach. Such an approach has been found to be not too promising when whole word filler models are used ([75]) but much more promising when subword filler models are used ([64]).

4.2 ANGIE Word-Spotter

For our word-spotter, we adopt a filler model approach. We use ANGIE as the subword lexical model for both the keyword and filler space, the sole difference being that for the keywords, we introduce an additional constraint on the permitted phonemic baseform, whereas for the filler space we allow any phoneme sequence. For the score to associate with each keyword hypothesis, we take the difference in path score at the beginning and end of the hypothesized

[2]But with a much larger data set than the subset we have selected

Keyword	Training Count	Testing Count
atlanta	74	4
baltimore	120	11
boston	116	16
burbank	42	27
charlotte	134	20
chicago	164	46
cincinnati	87	17
cleveland	145	18
columbus	82	3
dallas	70	20
denver	210	26
detroit	82	11
houston	119	19
indianapolis	144	32
kansas_city	184	23
las_vegas	158	34
long_beach	64	1
los_angeles	125	17
memphis	115	34
miami	243	39
milwaukee	219	53
minneapolis	93	6
montreal	89	16
nashville	73	10
new_york	294	44
newark	162	17
oakland	23	18
ontario	30	7
orlando	197	51
philadelphia	30	5
phoenix	150	34
pittsburgh	89	12
saint_louis	110	27
saint_paul	65	4
saint_petersburg	68	11
salt_lake_city	109	35
san_diego	109	14
san_francisco	177	7
san_jose	87	12
seattle	149	30
tacoma	81	10
tampa	72	15
toronto	141	17
washington_d_c	57	7
westchester_county	24	4

Table 4-1: Our set of keywords and their frequencies in training and test sets.

keyword. At first, we thought that the simple addition of lexical constraints to our phonetic recognition system from Chapter 3 was all that was needed. However, this implementation performed poorly.

A closer examination revealed that a major contributor to the poor results was a misbehaving search strategy. Specifically, the problem of comparing short vs. long theories exhibited the following phenomenon, which manifested itself in the word-spotter. Suppose an utterance has a low scoring region within it. As the paths approach this region, their scores would naturally drop, pushing them to the bottom of the stack. At the bottom of the stack, the paths encounter the very real risk of being pruned due to low scores compared to other paths in the stack. In fact, such pruning does tend to occur. So the situation is, the best paths entering the bad region are the ones most likely to be pruned. Now, we are left with the lower scoring paths entering the bad region. Since the stack only has lower scoring paths by now, these are the ones which survive past the bad region. One obvious solution would be to increase the stack size to reduce the effects of pruning. Unfortunately, with our implementation, this results in unacceptable running times. One natural question to ask is what happened to the balancing heuristics we used in phonetic recognition and why our search strategy worked for phonetic recognition but not for word-spotting. Our belief is as follows. In the task of phonetic recognition, getting a small number of phones wrong is not disastrous. However, when we have to decode the phones into real keywords, the impact is substantially greater since we need to hypothesize longer, accurate consecutive strings of phones in order to decode successfully into keywords. Of course, the probability of making no errors decreases when a longer string must be correctly hypothesized, in some cases exponentially, with the length of the string. The previously successful heuristics empirically proved insufficient in our word-spotter. We experimented with a variety of other heuristics to try to salvage the system, but eventually we decided to explore alternate search organizations. The one which proved most successful for us was the *stack decoder*.

4.2.1 Stack Decoders

The basic stack decoder was introduced by IBM in the 1970s ([30]). A typical stack decoder operates as follows, as described by Doug Paul ([56], [55], [57]):

1. Initialize the stack with a null theory.

2. Pop the best (highest scoring) theory off the stack.

3. Perform acoustic and language-model fast matches to obtain a short list of candidate word extensions of the theory.

4. For each word on the candidate list:

   (a) Perform acoustic and language-model detailed matches and add the log-likelihoods to the theory log-likelihood.

      i. if (not end-of-sentence) insert into stack.

      ii. if (end-of-sentence) insert into stack with end-of-sentence flag = TRUE.

5. Go to 2.

The description above describes a procedure that is essentially identical to a best-first search if the *stack*[3] were a priority queue. However, a typical implementation of the stack differs from a priority queue in one crucial respect. Instead of sorting the theories solely by decreasing order of score, we first sort the theories by increasing order of time, and then by decreasing order of score. Thus, when we remove a theory from the stack, we are always removing a theory which is one of the theories furthest behind in time. This also has the effect of causing all theories which end at a particular point in time to be explored together as a group.

The *fast matches* referred to in step 3 are computationally inexpensive scoring mechanisms for reducing the number of word extensions which must be checked with the more expensive *detailed matches* ([3], [5]).

A feature of the stack decoder organization worth mentioning is that there tends to be an "active window" over which the search algorithm is actively pursuing at any given time. This is enforced by the primary sort of the theories by time. This has a potential advantage in that the search progresses in a regular, left-to-right manner, which is conducive to pipelined implementations, easing the implementation of systems with features such as "always live" microphones. This also has a potential disadvantage in that very promising theories are not allowed to progress much faster than other theories, as would be the case with a best-first search.

[3]Note that the use of the term stack here is borrowed from the speech recognition literature and bears very little relationship to the use of that term in computer science literature to describe a LIFO data structure.

4.2.2 ANGIE's Stack Decoder Implementation

For our word-spotter, we adopt a variant of the stack decoder. We use no fast-match and we advance by the phone instead of by the word. Also, we employ the pruning strategies we had used with our phonetic recognition system (Chapter 3) with two modification: 1) At word boundaries, we have to match not only the last phone, but also the identity of the last word before we can prune, and 2) Instead of constraining the total queue size and the number of paths crossing a given boundary, we now limit the number of paths ending at each time boundary. We have no overall restriction on total stack size, but because each hypothesized segment can span only a small number of boundaries, there is a *de facto* limit on overall stack size, hence, keeping memory usage manageable. We have not chosen to have the causal silence detector in our word-spotter. After some experimentation, we discovered that our acoustic models have successfully learned what silence is through the many iterations during our phonetic recognition work, and are actually better at classifying silence than the causal silence detector.

The use of the stack decoder implementation yields much more acceptable results. We believe that the effectiveness of this particular search organization derives from a critical statement we made earlier, namely, that *all theories which end at a particular point in time are explored together as a group.* Thus, the theories competing against each other during the search process all cover *the exact same acoustic space.* As the reader may recall from our earlier discussion of search in phonetic recognition, the problem of balancing short vs. long theories is a major one. With the stack decoding strategy, the theories cover the same time span, and hence, require less normalization before they can be effectively compared. In our final word-spotter implementation, we eliminated the mean-entropy normalization and time rewards used in the phonetic recognizer. With roughly comparable theories, the mean-entropy normalization actually degrades our word-spotter in our experiments. The time reward is, of course, irrelevant since all theories compared will have the same time reward. We did not remove the phoneme reward because there remains the issue of how to compare theories of differing linguistic lengths; that is, one theory may hypothesize a long phonetic sequence and another a short one even though they both end at the same point in time. An alternative way of organizing the stack decoding strategy might be to sort the stack first by the number of linguistic units in each theory rather than by time, in

which case, we can eliminate the phoneme reward instead of the time reward. We did not explore this alternative because we were satisfied with our word-spotter's performance with the current search strategy.

While we were satisfied with having removed a major heuristic, the mean entropy normalization, we did have to add one new heuristic. However, this heuristic is a simple set of constants and is easier to understand and justify than the mean entropy normalization. We included a reward for the detection keywords to encourage their selection. This is a common technique (e.g., it was used in [45]) to improve performance, since predicting extra occurrences of keywords does not increase the false alarm rate unless the scores are not sufficiently discriminating. As long as the false detections have lower scores than actual detections, introducing them does not negatively impact either the ROC curve or the FOM. The keyword rewards were optimized on development data in our final system. We defer the actual performance comparison until we present our comparison of possible subword lexical filler models.

4.3 Varying Subword Lexical Model for Fillers

With our basic ANGIE word-spotting system implemented, we decided to conduct a series of experiments in which we varied the subword lexical constraints on the filler model. There has been quite an amount of previous work in terms of exploring the effects of acoustic modelling on word-spotting, in terms of keyword-dependent acoustic models, etc., (e.g., [63], [64], and [45]), but relatively little in the area of lexical models. Few researchers have held the acoustic models constant and focused on the lexical constraints. The only reference we came across in this area was Meliani and O'Shaughnessy [46] where the authors considered implementing the filler model as an alteration of the lexical models only, and not of the acoustic models. In that work, the authors looked at using syllables to model the filler space. We will mention some more details comparing our results to those of Meliani and O'Shaughnessy when we discuss our experimental outcome.

In our work, we will vary the subword lexical model for the filler space from a relatively simple phone bigram to a highly constraining full word model. ANGIE's framework provides us a great deal of flexibility in the design of the filler model. Our hypothesis is that the more constraining the model, the better the performance. The success of using full recognition for

word-spotting supports this viewpoint. However, we will also try to quantify exactly how much various constraints contribute to performance along with determining running speed vs. performance tradeoffs. The subword lexical models we have looked at are as follows:

Phone Bigram A phone bigram model constrains the probability distribution of phones in the filler space. This model is roughly comparable to the context-independent phones as fillers configuration in [45]. We will use this as a baseline.

Pseudo-words ANGIE is trained with full knowledge of the ATIS lexicon, but during word-spotting, the word constraints for the non-keywords are removed. Thus, similar to our phonetic recognition setup (Chapter 3), the system proposes possible subword structures for the filler space and governs where the proposed "pseudo-words" can end. Recall from the discussion on phonetic recognition that nonsensical and novel syllables are permitted, but a constraint is imposed on syllable order. An example of a pseudo-word might be *afth*: [ae f th] (the example we saw in Chapter 3).

Syllables ANGIE has access to a lexicon of permissible syllables and proposes combinations of these for the filler space. Since the syllable is the highest level unit, we simplify the probability model across syllable boundaries to be based only on left-phone context, as is the case for word boundaries. Also, we do not restrict the ordering of syllables, so nonsensical orderings are permitted. An example of a syllable sequence is *ciscofran*: [s ih s kcl k uh f axr n].

Morph Constraints ANGIE has a lexicon of morph-like units, which are essentially syllables but with additional designations for relative ordering, as in the pseudo-word case. These morph-like units are combined to propose pseudo-words. We use the full ANGIE probability model within each morph-like unit; however, other than the constraint on ordering, novel and non-sensical combinations are not prohibited. An example of a valid pseudo-word with morph constraints is *conflighting*: [kcl k aa n f l ay tcl t iy ng].

Known Words plus Pseudo-Words ANGIE has a lexicon of approximately 1200 known words. These words are governed by the full ANGIE probability model. However, we additionally permit pseudo-words, which we implemented as in the morph constraints

case. Our original thinking was that the use of pseudo-words can help model out-of-vocabulary words in the filler space and hence, lead to improved performance. Note that we do not allow for cross-word language models, such as a word n-gram, because we want the comparison to be focused on subword models. (There is, of course, the left-phone context advancement probability in ANGIE.) We do not use a word unigram model either, except for the keyword boosts as described earlier.

Known Words Only As in the preceeding case, but no invention of pseudo-words is permitted.

The list above is presented roughly in the order of increasing constraints, with the exception of the syllables and morph constraints cases, which are not directly rankable in this dimension. Each of the above cases can be implemented via a configuration of the ANGIE grammar or a setting of ANGIE parameters. For example, to implement a phone bigram, we replace the normal context-free rules, of which examples were given in Chapter 2, with very simple ones. We associate one phoneme-like unit with each phone. Each phonemics-to-phonetics rule is simply the phoneme-like unit non-terminal on the left-hand-side and the phone terminal on the right-hand-side. The rules governing the upper layers consist of one non-terminal for each layer, with a rule for each layer generating the next, and the syllabification layer generating all possible phoneme non-terminals. Of course, these rules and non-terminals are in addition to our regular set used to model the keywords. In all cases, cross-word language constraints were avoided, to focus the comparison on the subword lexical model.

4.3.1 Experimental Results

We summarize the results of our filler model experiments in Table 4-2 and Figure 4-1. We also include an entry for running the full SUMMIT recognizer to perform word-spotting to establish an upper limit on performance, since full recognition is generally believed to be one of the best approaches to word-spotting. We also include relative execution times of the various constraint sets. The times are normalized to the pseudo-words case rather than to real time because we have not invested the engineering effort to bring our ANGIE word-spotter up to competitive speeds. We omit the time for the phone bigram case because that can be implemented an order of magnitude faster using a Viterbi search, so a comparison

Filler Model	**FOM**	**Relative Time**
Phone Bigram	*85.3*	-
Pseudo-Words	86.3	1.00
Syllables	87.7	0.56
Morph Constraints	88.4	0.79
Known Words + Pseudo-Words	88.6	0.79
Known Words Only	89.3	0.74
Full Recognition w/Word Bigram	*93.9*	-

Table 4-2: Word-spotting FOMs for various filler models. Smaller relative times indicate *faster* running times. The pseudo-words case is normalized to have a relative time of 1.00. Word-level statistics are excluded from all but the last system.

based on the current implementation would not be a fair presentation.

The general trend is that performance improves with increasing sublexical constraints being imposed upon the filler model. This is as we predicted, that having a more detailed model leads to better performance. However, we should point out that our original hypothesis that having both known words and pseudo-words is desirable, since the pseudo-words can model the out-of-vocabulary words, was not validated empirically. Not having any pseudo-words at all resulted in better performance. We suspect that we may not have enough out-of-vocabulary words in our test set. It turns out that with a 1200 word vocabulary, the occurrence of out-of-vocabulary words was only 0.82% in the test set. However, cutting the vocabulary down to 400 words[4], which raises the out-of-vocabulary fraction to 8%, did not change the relative rankings. We had to trim the vocabulary size down to 200 words, with an out-of-vocabulary fraction of 14%, before the pseudo-words begin to help. We do not have a conclusive explanation for this observed behavior, but we suspect that having out-of-vocabulary words, while found to be a significant problem in recognition (e.g., [25]), is not as much of a problem in word-spotting, since making mistakes in the filler space is not penalized by the scoring methodology, with the exception of hypothesizing a false alarm. However, allowing pseudo-words creates possible competition for the keywords, since keywords can be modelled by pseudo-words as well, albeit, with possibly lower scores.

Something unexpected shows up in the speed comparison as well. We typically attained

[4]Taking the 400 most frequent words, as determined by training data statistics, plus the keywords as the vocabulary.

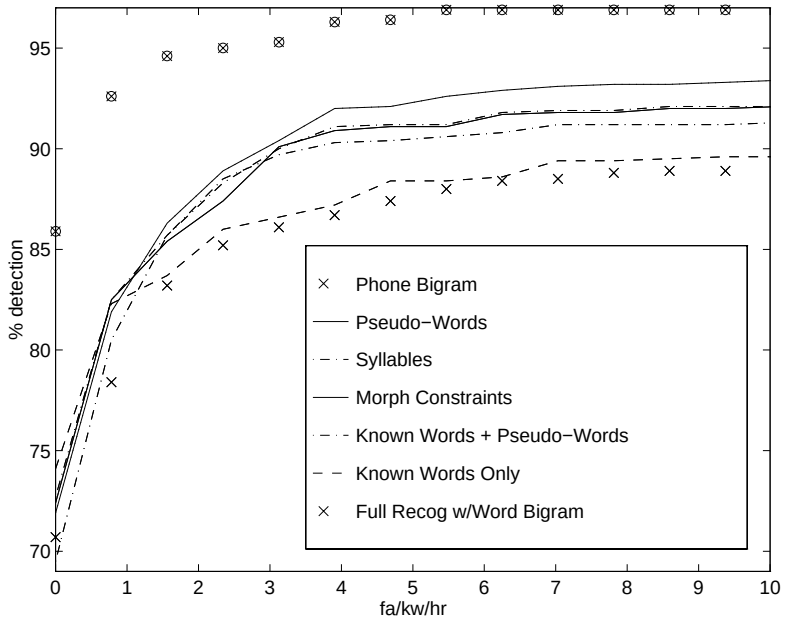

Figure 4-1: ROCs for various filler models.

improved speed along with improved FOM performance. We believe the reason for this positive tradeoff is because the more constraining models, although more computationally expensive to run the parser on, also narrow down the search space more by eliminating unreasonable theories. In our case, the smaller search's benefits far outweigh the extra work we have to do in the parser. Another interesting observation to make about running speeds is the extremely fast operation of the syllables system. The reason for this is that at the syllable boundaries, we need only compare left-phone contexts before pruning since that is all the ANGIE model carries across the syllable boundary. Syllable units are shorter than word units; hence, we have many more opportunities to prune. This general theme of finding more opportunities to collapse theories and prune is quite prevalent in our implementation experience. We first encountered it with our phonetic recognition system (c.f., Chapter 3) where we reduced the ANGIE context across word-boundaries to that of a left-phone context in order to promote merging of theories at word boundaries. We will encounter this issue again when we discuss integrating an ANGIE recognizer with the TINA natural language processing system in a later chapter.

The results presented in this section are difficult to compare directly to other results in

the literature. The most direct comparison we can find is with a result presented in the work of Meliani and O'Shaughnessy ([46]). In that work, the authors developed a word-spotting system with strictly lexical fillers, that is, the only difference between keyword and filler models were in their lexical, and not acoustic modelling, much as is our case here. In that work, with context-dependent acoustic phone models, the authors found that syllablic fillers outperformed phonetic fillers. The other work we would like to mention is that of Manos ([45]). Manos' focus was roughly a mirror image of our focus. He studied a range of filler models, but in terms of acoustic rather than lexical fillers. His work reached a conclusion similar to ours with respect to more detailed models leading to improved performance, but in the acoustic instead of lexical domain. Although Manos used a larger data set than we, our results for the phone bigram filler when compared to the full SUMMIT recognition result showed a comparable difference to Manos' comparison of his "CI fillers" system, which is essentially the same as our phone bigrams, and his full SUMMIT baseline. Our FOM dropped from 93.9 to 85.3 with the phone bigram. Manos' FOM dropped from 89.8 to 81.8 with his CI fillers. Our FOM drop was 8.6 and Manos' was 8. These are not directly comparable on account of different data sets, but because both cases involved ATIS and city names as keywords, there is some relationship between the two experiments. We find the similarity in drop vs. the baseline reassuring insofar as we believe our ANGIE framework to be competitive.

4.4 Summary

In this chapter, we discussed our word-spotting task within the ATIS domain. We choose to evaluate our ANGIE word-spotter on the task of spotting the city names in ATIS, since the choice of cities as keywords is documented in the literature. We started by discussing evaluation methodologies and the trade-off between detection percentage and false alarms. We presented the receiver operating characteristics (ROC) curve as a standard representation of the trade-off from the literature, along with the figure of merit (FOM) summary statistic. Next, we described our implementation of an ANGIE-based word-spotter, particularly our problems with the best-first search used successfully in our phonetic recognition system from the last chapter. We switched to a stack decoder strategy, which proved much more effective. We believe that the primary reason for the stack decoder's better performance is

that it always compares theories ending at the same boundary, hence theories which cover the same acoustic space. This mitigates the difficulties in balancing scores for theories of different lengths.

With a functional word-spotter, we described a range of experiments where we varied the subword lexical constraints for the filler model. We considered, in order of increasing constraint, phone bigram, pseudo-words invented by ANGIE, syllables, morph-like units (essentially syllables with ordering constraints), full words plus pseudo-words, and full words only. We discovered, as anticipated, that performance generally improved with increased constraints. However, we were surprised to see that having pseudo-words in addition to known words, in an attempt to model out-of-vocabulary words, hurt the process, even if we lowered the vocabulary coverage of the test set to 92%. To make the pseudo-words helpful, we had to lower coverage further to 86%. We believe that in word-spotting, having the pseudo-words potentially cover the keyword space is detrimental unless there is a high percentage of unknown words, which would benefit from the pseudo-words. We also discovered, perhaps surprisingly, that speed actually tends to improve with the more constraining models, so there is a positive speed vs. FOM trade-off. We believe that the extra constraints narrowed down the search sufficiently to offset the added computational cost of their implementation. Of all the filler models compared, the syllables model ran exceptionally fast. This was probably due to the additional pruning possible at syllable boundaries. This may suggest that the syllable could form a productive unit for bottom-up processing into an intermediate representation for further consideration by a linguistic processing system, such as a natural language understanding system. We will explore this possibility in our work on continuous speech recognition (Chapter 5). However, we will discover that the apparent advantages of syllables are offset by other disadvantages, at least for the English language.

Chapter 5

Continuous Speech Recognition

In this chapter, we describe the implementation of a full speech recognition system based on the ANGIE framework. We consider word recognition to be the final and most difficult test of the feasibility of ANGIE as a sublexical modelling infractructure. We will explore using both words and syllables as lexical units in our system. Our final results show that we can implement a recognizer using ANGIE that is competitive in performance, though not in speed, with a baseline system using SUMMIT. In both cases, a word bigram provides the word language model constraints. More interestingly, we will also explore the integration of the ANGIE framework with a context-free grammar based natural language understanding system, TINA ([69]). We will show that the ANGIE framework can indeed be integrated in a tight manner, without the use of N-best lists or word graphs, and that this integration leads to a 21% drop in recognition error rate as compared to a word bigram, whereas N-best resorting with TINA leads to only a small improvement.

5.1 Experimental Framework

In this chapter, we will be evaluating our ANGIE based recognizer on the task of word recognition. However, because our system is still in the developmental stages, and because we want to speed up the development/test iteration cycle, we will be operating our recognizer on phone graphs which have been pruned down to a more manageable size rather than the full phone graph hypothesized by the front-end from the raw waveforms. The pruning is done by using the baseline SUMMIT recognizer with a word bigram language model and keeping only the phones which occur on the higher scoring paths. We will consider the

the paths that correspond to the top N (100 in our case) theories. This technique is an adaptation of the common practice of working from word graphs in the word language modelling community. There, a recognizer generates a word graph from the higher scoring paths encountered during recognition using some baseline configuration. This word graph is then searched with the language model under development. Since we are dealing with subword modelling, we naturally extend this development and evaluation paradigm to a graph of subword units, in our case, phones, because phones are what the ANGIE framework takes as initial inputs. We use this technique to reduce both the time and space complexities of our recognizer, which, in its current state, has not been sufficiently engineered into a system that operates in an acceptably efficient manner when processing raw waveforms[1]. It may indeed turn out that some form of first stage "fast match" recognition may be an appropriate design for a practical system.

As in our phonetic recognition work (Chapter 3), we will be using error rate as the evaluation function, except that instead of measuring phone error rate, we will be considering word error rate. We will also be using the same training, development and test sets used previously in phonetic recognition and word-spotting.

5.2 Basic Recognizer Implementation

The implementation of a basic recognizer using ANGIE for its sublexical model and using a word bigram for the word level language model is relatively straightforward. The same fundamental setup from our word-spotting work (Chapter 4) extends naturally and effectively to a word recognizer. We incorporate the word bigram statistics at the end of a putative word. The stack decoding strategy from our word-spotter proves workable for word recognition. The pruning mechanisms from the word-spotter also prove to be safe for our recognizer. They took into account a single word left-context when considering whether a pruning operation is safe or not in order to distinguish keywords from each other and keywords from fillers. Because we are using a word bigram for the word level statistics, a single word left context is a safe mechanism for the current task as well. If we were using longer context n-grams, then the pruning would need to be changed to examine longer left-contexts. Two changes were needed, however, and we mention them next. The first

[1] We leave such engineering optimizations to future work.

change is the elimination of the keyword boosts used in the word-spotter. The other change is more substantial and we devote a detailed discussion to it because, like our switch to a stack decoder when we implemented the word-spotter (Chapter 4), we learned a lesson about engineering search organizations that is not readily apparent except through empirical experimentation.

5.2.1 Homonym Related Search Issues

The other change addresses a familiar but delicate trade-off that has to do with homonyms. In the word-spotter, we did not encounter any homonyms because our keyword set did not have any homonyms. Even with words for the fillers, we could safely ignore homonyms, since errors in the filler space do not penalize us and because no cross-word constraints were used. The issue with homonyms in a bottom-up system, such as ANGIE, is what to do with the two or more theories being proposed bottom-up for each homonym. We can either keep the two theories bundled together riding "piggy-back" or we can split them into different paths in the top-level search. We have encountered this issue once before, in handling multiple ANGIE parses for a given phone sequence (Chapter 2). There, we have decided to keep the theories bundled together and to take the highest score as the representative score for the bundle. However, theories which lead to word terminations are split out into a bundle separate from the partial word theories. The main factor affecting this decision is an efficiency vs. search accuracy consideration. With the theories bundled, because we are taking a representative score instead of the actual scores, we introduce the potential for suboptimality in the search. However, bundling the theories reduces the number of paths which must be pursued independently, and improves both the time and space requirements of the search. Whether the suboptimality which results from bundling is detrimental enough to offset the time and space benefits of bundling is an empirical issue.

In the case of multiple ANGIE parses for the same phone sequence, bundling the theories was almost a necessity because of the large number of possible parses for many phone sequences. However, when it comes to recognizing words, the potential search errors of bundling pose a much greater risk. Our initial attempt at implementing a recognizer with bundling proved disappointing enough that we reimplemented the search with the different homonyms separated out. We believe the reason for this is the importance of the word bigram statistics, which in the bundled case, are taken from the best scoring homonym.

Recognizer	Total	Sub	Del	Ins
SUMMIT	18.9%	11.7%	4.9%	2.3%
ANGIE Syllable Units	26.6%	14.4%	8.7%	3.5%
ANGIE Word Units	18.8%	11.7%	4.2%	2.9%

Table 5-1: Recognition error rates for SUMMIT and ANGIE based recognizers employing a word bigram.

However, when we reach the end of the next putative word, there is a very real, and empirically damaging, possibility that the greedy choice of the best previous word is no longer appropriate. The net result appears to be that the distorted scores cause many more correct theories to be dropped from the stack due to pruning. Thus, we modified the search to unbundle homonyms into separate theories, but only after the acoustic scores have been consulted on a shared bottom-up theory. The concept of bundling vs. unbundling will be revisited again when we describe TINA natural language integration. We believe it to be an important, albeit seldom enunciated, engineering issue when organizing search strategies for recognizers.

5.2.2 Basic Recognition Results

The word error rate results of our ANGIE based recognizer are summarized in Table 5-1. We include the results for both a baseline SUMMIT system and our ANGIE-based system. We also include the results of an ANGIE-based system where the lexical unit is a syllable instead of a word; that is, ANGIE's lexicon consisted of a list of syllables and their baseforms. A separate lexicon of words and their decomposition into syllables was used to decode words from the syllables. No intermediate syllable graph is created. Rather, the word decoding constraints are applied on-the-fly as hypotheses with syllables are generated bottom-up by the ANGIE recognizer.

As can be seen from the table, the use of words in an ANGIE-based system is competitive, but the use of syllables falls short. A likely explanation is that having the strong constraints at the syllable boundaries is beneficial to performance, and therefore, their removal results in a large degradation. As we had previously observed, the use of syllables as fillers provided a tremendous speed advantage in our word-spotter (Chapter 4), but their use actually slows down our recognizer when we have to decode into words. A possible reason for this is

ANGIE			SUMMIT		
Sub	Del	Ins	Sub	Del	Ins
flight → flights	the	the	flight → flights	the	a
flights → flight	a	and	a → the	a	the
a → the	is	to	flights → flight	to	to
list → is	in	a	which → what	i	and
fly → flight	of	on	in → and	of	in

Table 5-2: Top five errors for ANGIE and SUMMIT word recognition systems

that in word-spotting, there is no need to decode into words, so pruning can happen at syllable boundaries. For word recognition, we do not have this advantage. Worse yet, the removal of the stronger constraints at syllable boundaries leads to a much bushier search space, increasing the time requirements. We should caution, however, that the poor performance of syllables in English does not necessarily imply that for languages where syllables are the natural lexical unit (e.g., Chinese), syllables will not perform better. One of our main motivations for attempting syllables was that we wanted to integrate a natural language understanding system together with our ANGIE recognizer without having to ever decode into a word representation, so that the same arrangement can carry over naturally into such syllable-based languages. A syllable-based approach can also facilitate a vocabulary-independent recognizer implementation. However, after our disappointment here, and continued disappointment when we attempted NL integration, we decided to stay with words when dealing with English.

The top five errors in each category (substitutions, insertions, and deletions) for the two systems are given in Table 5-2. A closer comparison of the ANGIE and SUMMIT systems does not show any remarkable differences. In both cases, the top three substitution errors were the same and the next two were words common in the ATIS task. The top deletion and insertion errors in both cases were short function words, which is to be expected since they are both frequent in occurrence and incur little acoustic and linguistic penalty when deleted or inserted. ANGIE tends to have extra insertions over deletions, however, unlike the flagrant case with the [dh] phone in our phonetic recognition system (Chapter 3), we do not consider the percentage difference, nor the evidence gathered from a random examination of several recognized utterances, to warrant concern.

5.3 Natural Language Integration

Another major goal of our work in ANGIE recognition is the integration of natural language understanding constraints in a tight manner, without having to generate either an intermediate N-best list or a word graph. As has been noted by other researchers (e.g., [52]), the incorporation of NL constraints has generally not resulted in much improved recognition performance when measured by word error rate. However, we observe that as the integration becomes tighter, for example, when word graphs are used instead of N-best lists, the results are more promising (e.g., [71]). Since ANGIE's recognizer uses a search strategy that supports parsing and long distance constraints, it should be relatively straightforward to integrate the NL constraints directly into the recognition search process.

We would like to mention two prior attempts at integrating NL into the recognition process itself, as opposed to a feed forward process involving N-best lists or word graphs. In Zue et al. ([77]), one of the approaches taken was to compile an NL grammar into a word n-gram. While this does permit some NL constraints to be applied during the recognition process, a word n-gram is substantially less powerful than a context-free grammar with probabilities. For example, it is possible for this approach to admit a sentence which cannot be parsed by the grammar. Finally, this approach cannot not directly lead to a meaning representation at the end of recognition process. Another interesting approach is that of Goddeau ([21]). There, a shift-reduce parser is integrated into an A^* search in a manner very similar to how we will integrate an NL system into our stack decoder. However, the parser used was not truly an NL system in that it was unable to generate a meaning representation. Further, the important issue of robust parsing, which we will address later in this section, was handled in a manner that cannot be used to derive a meaning representation. Specifically, in the event of a parse failure, Goddeau's system backed off to a word n-gram. Finally, we should note the author reported only small performance improvements, citing as a possible culprit a problem with the future estimate used in the A^* search. The author further suggested as a future possibility the use of a stack decoder, which does not require a future estimate. In our work, we will be pursuing such a strategy.

5.3.1 TINA and Top-Down Processing

For our NL integration work, we used MIT's TINA system ([69]). Like ANGIE, TINA is also based on a context-free framework. The context-free rules used by TINA are also written by hand, as is the case with ANGIE. TINA has several other features worth mentioning:

Constraints Constraints are used for certain syntactic features such as number agreement and verb tense enforcement. These are items which would be unwieldly to express in a pure context-free grammar because it would require categorizing non-terminals by their features, resulting in a dramatic expansion of the grammar, especially since many non-terminals can have multiple feature categorizations. Constraints are also used to handle gaps which are fairly frequent in wh-queries in English. For example, the wh-query "What meals does this flight serve?," can be thought of as corresponding to the same deep structure as the normal ordered sentence, "Does this flight serve dinner?" Constraints are used to enforce where moved constituents may be "absorbed." A more detailed discussion can be found in [69].

Probabilistic Framework As in the case of ANGIE, TINA also has a probability model to better select among multiple errorful inputs presented to it by a recognizer, or among ambiguous parses for a single hypothesis, or among a combination of both. As in ANGIE, the framework is organized to easily generate a probability for the next *word* given the preceeding partial parse tree.

Automatic Training The probabilities in TINA are automatically derived from training data as in ANGIE. This permits TINA to learn information from training data, in addition to the information provided by the human engineered rules and constraints.

Robust Parsing Ungrammatical sentences are frequently encountered, either due to careless verbalization, or due to recognition errors. A natural language system needs to handle such sentences without completely failing to parse. TINA includes such a mechanism for *robust parsing*.

Instead of pursuing a bottom-up strategy, TINA is primarily a top-down driven system. So, instead of starting with the first word in a sentence, and pursuing possible theories which start with the given first word, TINA starts with the sentence, and pursues possible

derivations from that until it finds theories which can start with the given word. The different top-down TINA and bottom-up ANGIE strategies reflect a design philosophy which we had earlier enunciated. Namely, as we discussed in the introduction to this thesis, below the level of the word (or other lexical unit), we want bottom-up sharing. However, above the lexical level, we find it much easier to capture phenomena such as gaps in English via a top-down strategy. It is possible to capture gaps bottom-up, however, it is conceptually trickier to visualize the process. For example, we will have to be constantly proposing possible acceptor locations, with little constraint, for each hypothesized trace and carrying the proposals towards the top of the parse tree to verify or reject them. The alternative, to pursue a uniform top-down strategy, even below the lexical level, is also possible. However, we feel that certain desirable features, such as widespread sharing of word substructure, and also the support of new word theories, which we ultimately want ANGIE to be able to handle, will be difficult to implement top-down.

Given that we are keeping a top-down design for the supralexical parser and a bottom-up design for the sublexical parser, the natural organizational point is at the level of the lexical units. Since we are using a stack decoder, which supports long distance constraints, this is not too difficult. Essentially, we have ANGIE propose the words bottom-up. At that point, the stack decoder consults TINA for an NL score, much as it previously consulted a word bigram, merges the two scores, and puts the new theory back onto the stack. An illustration of this process is shown in Figure 5-1. Because there is pruning at three levels, first in terms of a maximum number of ANGIE theories per phone sequence, then in terms of a maximum number of TINA theories per word sequence, and finally, in terms of a maximum number of recognizer theories on the stack, this organization is not as integrated as a single massive search that prunes at only one location. As an example, perhaps a poor ANGIE theory may be offset by a very good TINA score, but it will still get pruned in our organization since the ANGIE theories are pursued separately from the TINA theories. However, our integration is still much tighter than a word graph organization. The reason is that this separate scoring gets unified together at each word boundary. With a word graph, the entire word graph is generated without any information from the NL system, hence, the separate scoring is only integrated at the end of the utterance, resulting in a much greater possibility of errors due to the pruning involved in generating the word graph.

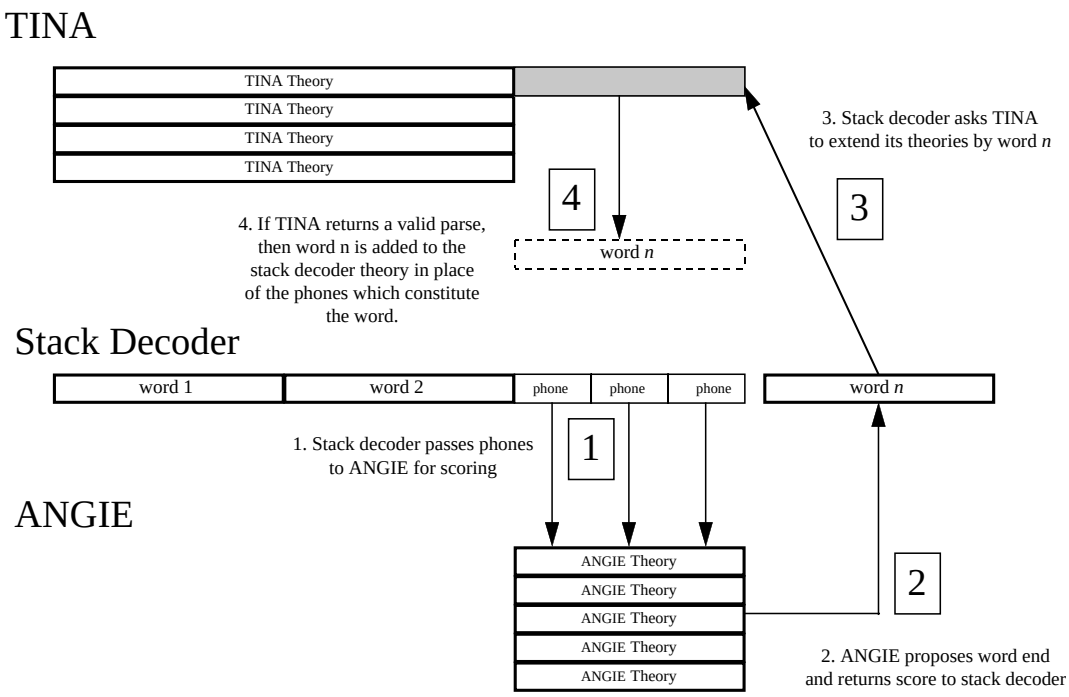

Figure 5-1: Integration of TINA into the search process.

5.3.2 Robust Parsing

One issue with which we have to deal in our NL integration is robust parsing. Robust parsing typically allows an utterance to be parsed as fragments, in an attempt to increase a natural language's coverage. It is difficult to have the grammar itself have a very high coverage due to both sloppy verbalization and also recognition errors. We started with the same TINA grammar for ATIS that was used in the word graph NL integration work of [69]. In that system, robust parsing is handled by having top level grammatical rules whose net effect is similar to[2]:

$$\text{sentence} \Rightarrow \text{skip\_word}^* \ [\text{full\_parse}] \ (\text{skip\_word} \mid \text{partial\_parse})^*$$

The idea is that we want to find a full parse, embedded in some combination of skip words and partial parses. A full parse is the most desirable type of parse. It is a well formed grammatical clause which our NL system can analyze and generate into a semantic frame. The partial parses and skips are to absorb random surround, such as false starts, filled pauses, and other disfluencies. Only non-content words are allowed to be skipped. We discovered that this method of supporting robust parsing results in an extremely bushy search process, because the NL parser has to hypothesize the possibility of ending a partial parse or inserting a skip after almost every word. Our initial attempt at using a grammar with this flavor of robust parse support took on the order of forty-five to sixty minutes to recognize a typical ATIS utterance on a Pentium Pro 200 MHz based machine, clearly an untenable situation. Even after we reduced the complexity of the grammar somewhat, the running time was still extremely slow.

We suspected the expense of this particular robust parsing mechanism as the culprit, so we pursued an alternate strategy. We decided to take the robust parse handling out of TINA and moved it into the recognizer search. The recognizer search takes the following strategy along a particular path hypothesis. Parse as many words as possible, supported by full parse theories only. When TINA fails, retrace backwards until we find a place where the TINA theory can end a "sentence." Then try starting a new parse from that point going forward. This is not as robust as the previous robust parse mechanism since we restrict the point at which to reparse to be only the one which requires the least backward retracement

[2]Here, the * refers to zero or more, the | refers to alternatives, and [] means optional.

and we do not even consider it at all unless triggered by a parse failure. The more expensive robust parsing strategies would have considered all possible breaking points, regardless of parse failure. However, in practice, this greedy strategy turns out to be adequate in our experiments. We also needed to include a heuristic penalty so that theories with excessive fragmentation are penalized when compared against comparably scoring theories with fewer (or no) fragmentation. This alternate implementation operated with much more reasonable speed, taking roughly 2.5 times longer than with a word bigram rather than two orders of magnitude longer, as in the full robust parse case.

5.3.3 Merging Theories to Increase Pruning Opportunities

With the introduction of an integrated NL system and hence, extremely long distance constraints, the opportunities to prune theories at word boundaries, taking only the last word in the theory into account, is diminished greatly. While we can merge theories which share the same word sequence, which occur when different acoustic hypotheses decode into the same word sequence, this results in a significant drop in pruning opportunities. One of the ideas we explored is to come up with merge categories, that is, to create a set of categories where the TINA theories may be pursued as a merged bundle. An example of a merge category in our system is a "direct-object." Say a given sequence of words form a direct-object. Regardless of where in a theory this sequence of words appears, it should always have the same score. Thus, once this sequence of words has been parsed as a direct-object, the parse information and score can be shared across all occurrences of the sequence. In essence what we are trying to do is to reintroduce bundling opportunities into the NL mechanism. Because of its extremely long distance constraints, it is more difficult to locate such opportunities when considering linear sequences of words. However, by considering subtrees with common categories, we feel that we may have opportunities to share computation.

We implemented such a strategy with approximately ten merge categories. However, we did not witness a noticeable impact on either running speed or error rate performance. A closer evaluation shows that our hypotheses generate relatively few merge candidates. Perhaps more fruitful merge categories can be chosen. Because our system ran acceptably in terms of demonstrating NL integration with ANGIE, we chose not to pursue this line of exploration further in this work. We will comment further about the general issues of

when to merge and bundle in our concluding remarks in the final chapter, as this theme is prevalent throughout this thesis.

5.3.4 Syllables vs. Words

One observation to make is that if our recognizer were intended to be deployed as part of a conversational system where understanding, and not word recognition, is the goal, then we are not really limited to choosing words as the lexical unit. We can, for example, choose syllables, which may be easier to deal with in terms of flexible vocabularies, etc. However, much as in the case of trying syllables with our basic recognizer, we were not very successful in the TINA intergrated recognizer either. Our suspicions already mentioned for the basic recognizer are repeated in this case as well. Given our poor track record with syllables and recognition, at least for English, we did not invest in further examination along these lines.

5.3.5 NL Integration Results

Table 5-3 summarizes the results when the TINA NL component is integrated into our ANGIE recognizer. Included as baselines are the results of SUMMIT without TINA, SUMMIT with TINA 100-best rescoring and ANGIE without TINA. As can be seen from the table, the integrated ANGIE plus TINA recognizer performs much better than the baseline bigram based systems. Morever, tight integration performs much better than an N-best rescoring attempt using the 100 best SUMMIT hypotheses. In that rescoring experiment, we actually combine the TINA score with the SUMMIT score (which includes both acoustics, sublexical model score, and word bigram score) with a weighted linear interpolation. If we try using the TINA score alone, the results were very poor. The weights were selected in a "best case" manner, via optimization on the test set, so the actual performance of the rescoring system may be lower. However, our goal is to show that the integrated ANGIE plus TINA recognizer performs substantially better, which we believe to be the case illustrated by the results provided.

The error rate reduction with TINA integration occurred mainly in the substitutions category. Insertion errors were also reduced but deletion errors actually increased slightly. The error rate breakdown for various systems are included in Table 5-3. The top five errors in each category for the two systems are given in table 5-4. The top fourth and fifth substitution errors in the ANGIE plus TINA case appear to be directly attributable to

Recognizer	Total	Sub	Del	Ins
SUMMIT w/Word Bigram	18.9%	11.7%	4.9%	2.3%
ANGIE w/Word Bigram	18.8%	11.7%	4.2%	2.9%
SUMMIT w/Word Bigram and TINA 100-best Resorting	18.2%	10.8%	5.5%	1.9%
ANGIE w/TINA Integrated	14.8%	8.7%	4.5%	1.6%

Table 5-3: Comparison of recognition error rates with incorporation of TINA NL processing system.

NL confusions. For example, in our grammar, "a" and "an" are generally interchangeable. Similarly, in most cases, "I" and "I'd" are also interchangeable since "I like" and "I'd like" are both well-formed. A few of the other top substitutions can also be explained by NL ambiguity. For example, the top sixth is "Newark" becoming "New," as in "New York," a much more common city name than Newark, but grammatically interchangeable with Newark in most cases and acoustically confusable with it. In this case, we would imagine that the word bigram should have a similar confusability. The reason this particular confusion pair shows up now is probably because we are giving TINA a much higher weight than we give the word bigram, because on held-out data, TINA was found to work better with a higher weight. This is not unexpected since we believe that TINA provides a superior linguistic model. The top seventh substitution is "only" becoming "all," also gramatically interchangeable. These last two examples are causes of concern, because, although they are gramatically valid substitutions, they result in the wrong semantic interpretation. The top deletions do not show anything extraordinary. The top insertions, however, seem to confirm that the Newark/New York problem is indeed a problem that may need to be address in a deployed system. It also helps explain why insertions is the only error category where the ANGIE plus TINA system did worse than the ANGIE plus bigram system.

5.4 Summary

In this chapter, we discussed the implementation of a full word recognition system based on ANGIE. We consider this to be the most challenging test of ANGIE's feasibility as a framework supporting speech recognition tasks. Because of engineering efficiency issues involving our developmental stage implementation, we decided to conduct our experiments on a pruned phone graph instead of the full phone graph as derived from acoustics. This practice for

ANGIE plus Word Bigram			ANGIE plus TINA		
Sub	Del	Ins	Sub	Del	Ins
flight → flights	the	the	a → the	the	and
flights → flight	a	and	flights → flight	a	the
a → the	is	to	flight → flights	i	york
list → is	in	a	a → an	to	to
fly → flight	of	on	i → i+d	and	a

Table 5-4: Top five errors for ANGIE word recognition systems with word bigrams and with TINA integration

language modelling is fairly common when dealing with word language models, where word graphs are frequently searched with the new model. We extend the paradigm to subword language models by working off a pruned phone graph to expedite experimentation.

The implementation of our recognizer was a straightforward extension of our word-spotter from the previous chapter. We did encounter an issue new to the present chapter, namely, what to do with homonyms. In word-spotting, even when using words as fillers, homonyms can be safely neglected because they were not part of our keyword set and their misrecognition in the filler space did not impact word-spotting performance. However, with recognition, we concluded that splitting homonym hypotheses into different theories in the search performs better than keeping them bundled in a "piggy-back" manner. Our ANGIE-based recognizer achieves an error rate comparable to a SUMMIT baseline system, also using a word bigram. The top errors in both cases were similar as well, suggesting that ANGIE is a competitive word recognition framework, but is otherwise unremarkable in this incarnation. We had also attempted using syllable units for the lexical units instead of word units, but that results in noticeably inferior performance.

Where the ANGIE-based recognizer produces a significant improvement, from 18.9% error rate to 14.8%, is when it is integrated with the TINA natural language processing system. Previous experience at bringing in an NL system resulted in only a small improvement when an N-best resorting paradigm was used ([52]) and our own validation experiment with using TINA to resort a 100-best list from SUMMIT mirrors that experience. In our experiment, we combine the SUMMIT score with a TINA score using an experimentally optimized weight. The TINA configuration used in our resorting experiment was the same one that was used in the integration experiment. A substantially better performance improvement has been

reported when the interface is tightened to that of a word graph ([71]). In our work, we take the integration one step further and merge the NL system into a combined search process. Because ANGIE, like the TINA NL system, is based on a context-free foundation, our search engine naturally supports the long distance constraints provided by TINA and can be expanded to achieve this integration.

Conceptually, it is possible to have one parser handle both the subword context-free rules of ANGIE and the supraword context-free rules of TINA. However, because we feel that the subword model is better handled in a bottom-up paradigm, to promote sharing and to permit generalization to new words, and the supraword model is better handled in a top-down manner, to more easily handle such phenomena as gaps without making the search too expansive, we did not choose this route. Instead, we decide to back off the integration by one step. In particular, we will continue to have ANGIE generate word hypotheses bottom up, but at the end of each word, we will bring in the TINA model. Thus, we have three search threads proceeding in a lock-step manner. This compromise still represents a much tighter integration than a word graph, because the TINA constraints are brought to bear at each word ending, whereas a word graph is generated without any knowledge of the TINA model. In word graph implementations, the TINA model is only consulted after the complete word graph for an utterance is generated.

A major problem encountered with our NL integration attempt is the bushiness of the search needed to support robust parse, which was expressed as context-free rules permitting partial parses and the insertion of "skip words" (fillers). In an N-best or word graph paradigm, the search space over which we need to pursue possible NL parses is much smaller and robust parse is not as much of a problem. To overcome this problem, we factor the robust parsing mechanism out of the context-free rules. Instead, our search strategy pursues an NL theory until it dies, then backs up until it can find a breaking point to break the theory into two theories, one ending at the break, and a new one starting there. While not as comprehensive as the original mechanism, the revised strategy proved sufficient in terms of achieving empirically good performance at acceptable speeds. We also experimented with the idea of merging NL theories at certain "merge categories," categories where we believe the work done within the subtree for the category can be shared with other occurrences of that subtree. An example of a merge category would be a "direct object." However, that attempt did not affect time complexity noticeably and had little impact on recognition

performance.

An analysis of the errors in our ANGIE-plus-TINA system shows that some within category confusions were introduced. For example, we saw many cases of "New York" being substituted for "Newark." Both are in the same NL category, "city name," and are acoustically confusable. Because New York is much more common, the confusion is natural. The reason why the problem is more prevalent with TINA than in the word bigram case is that, because TINA is a much more powerful linguistic model, it receives a higher weight than the word bigram in the final scoring. But TINA is not able to sufficiently distinguish Newark from New York, so, the increased linguistic model weight given to TINA leads to the confusion.

In our TINA NL integration work, we also attempted to use syllables instead of words as the lexical units, but without much success. Note that our work is in ATIS, which is English. Syllables may work better for other languages, which are more naturally syllable-based, such as Chinese.

We are satisified that ANGIE proved feasible as a full word recognition framework, and further, that the integration of the TINA NL system was successful. In the next chapter, we will report on some pilot studies we have conducted on trying to add new words, which we believe ANGIE to support well, and on prosodic modelling taking advantage of the ANGIE parse structure.

Chapter 6

Pilot Studies in New Words and Duration Modelling

In our motivating remarks regarding the ANGIE design, we had mentioned two possibilities which we will explore in terms of pilot studies in this chapter. One of our claims is that ANGIE can support flexible vocabulary changes, for example, incremental addition of words to the recognizer, better than other subword lexical models. The other is that ANGIE, because it provides a subword decomposition in terms of a parse tree, can potentially support novel models which make use of such information, for example, prosodic modelling. Our two pilot studies will begin to explore how well the ANGIE framework supports these two motivating goals. The first study is an attempt to determine whether ANGIE better supports the addition of new words to the vocabulary of a recognizer, because it can provide some subword support for the new word through sharing of substructures with existing words. Our results will show that ANGIE by itself performs comparably in the presence of these new word additions to our baseline SUMMIT system. The baseline SUMMIT system uses a pronunciation graph, so we will give the new words neutral pronunciation weights for the various arcs in the graph. However, because of ANGIE's support for TINA integration (Chapter 5), we find that the integrated ANGIE plus TINA system has a measurable performance lead, even in the presence of new words.

The second pilot study involves work done jointly with our colleague, Grace Chung. Chung has implemented an ANGIE-based hierarchical duration model which has been found to improve phonetic recognition performance over a simple phone duration model ([8]). We

will take her model and integrate it into our word-spotter (Chapter 4). We found that the inclusion of the Chung duration model into keyword scoring increases the performance of our best word-spotter configuration from a FOM of 89.3 to 91.6. We also found that on the acoustically confusable "New York" vs. "Newark" keyword pair, the duration model can cut the error rate by up to 68% if we only consider the error rate in terms of disambiguating those two keywords, treated specifically.

6.1 Incorporating New Words

We had stated early on that one of the goals of our ANGIE framework is to be able to support flexible vocabularies. By this, we mean that we would like to be able to easily change the vocabulary of our recognizer with a minimum, preferably no, retraining of the ANGIE probabilities or alteration of the context-free rules. ANGIE's bottom-up design, with its extensive sharing, is supposed to allow for generalization of learned probabilities from known words to newly added words. Our first pilot study attempts to determine to what extent we have achieved this goal.

6.1.1 Experimental Framework

We envision the following scenario as the backdrop of this pilot experiment. Assume that our ANGIE-based recognizer is used within a conversational system, such as MIT's GALAXY system ([22]). Such a system may retrieve information from a database in response to a user query. For example, the user may ask the system about flights from Boston to California and the system may respond with a list of cities in California. Naturally, we would expect the user to respond with one of the cities on the list. Now, imagine that the list includes both cities in the recognizer's vocabulary and cities not in the vocabulary. It would be nice at this point, if the vocabulary can be dynamically adjusted to include all the cities present in the list. This is precisely the setup we are assuming for the purposes of this pilot study. Although the particular choice of city names is somewhat contrived, the concept could be applied much more extensively, as for example, to a list of bookstore names in Cincinatti retrieved from the web.

How do we obtain pronunciations for the words to be added to the vocabulary? They may be found in a dictionary or generated by a letter-to-sound system. Since the ANGIE

framework supports letter-to-sound generation, we can even use ANGIE itself for this purpose. Some work on using ANGIE for letter-to-sound generation is reported in [70]. However, because letter-to-sound generation is beyond the scope of the present thesis, we will, for the purposes of this pilot study, assume that the pronunciations have been given to us.

To insure that we have a properly trained baseline system, in terms of the new words being added to the vocabulary, we will adopt the following experimental methodology. We will take the existing ANGIE-based and SUMMIT recognizers used in Chapter 5, and we will artificially remove a list of words from their vocabularies. The list of words will be the words whose addition to the vocabulary we are simulating. This insures that we have baseform pronunciations for the "new" words with a known performance in a fully trained system. The full recognizers will form the baseline comparisons, and we will simulate adding the words to the reduced recognizer to see how well the recognizers handle the addition of new words as compared to the baselines.

Our particular choice of simulated new words will be a list of some of the city names in ATIS. Recall that ATIS-2 had one list of city names, which was later expanded in ATIS-3. We view this as a natural choice for the simulated new words, namely, the new city names added in ATIS-3. There is a natural basis for this choice, for we can imagine having developed the recognizers initially for ATIS-2 and then deploying a system in an environment where the full set of ATIS city names show up when data are retrieved from a database. Moreover, this particular choice of simulated new words has been used previously in the literature in the new word characterization work of Hetherington ([24], [26]). Hetherington also followed a methodology for creating an artificially reduced vocabulary system very similar to ours. The list of ATIS-2 and ATIS-3 cities is given in Table 6-1.

6.1.2 SUMMIT Implementation

In the MIT SUMMIT system, which we consider to be one of our baseline comparisons, subword modelling is handled via a pronunciation graph. The pronunciation graph is generated by taking a list of baseforms for each word in the vocabulary, and then applying phonological rules to generate a variety of possible phonetic realizations for each word. These phonetic realizations are what is represented in the pronunciation graph. Moreover, each edge in the graph has a weight associated with it. The weight represents a reward or penalty for transitioning that edge, that is, for having that phone in the realization. The

City	Training Count	Testing Count
ATIS-2 and ATIS-3 Cities		
atlanta	74	4
baltimore	120	11
boston	116	16
dallas	70	20
denver	210	26
detroit	82	11
oakland	23	18
philadelphia	30	5
pittsburgh	89	12
san_francisco	177	7
washington_d_c	57	7
ATIS-3 Only Cities		
burbank	42	27
charlotte	134	20
chicago	164	46
cincinnati	87	17
cleveland	145	18
columbus	82	3
houston	119	19
indianapolis	144	32
kansas_city	184	23
las_vegas	158	34
long_beach	64	1
los_angeles	125	17
memphis	115	34
miami	243	39
milwaukee	219	53
minneapolis	93	6
montreal	89	16
nashville	73	10
new_york	294	44
newark	162	17
ontario	30	7
orlando	197	51
phoenix	150	34
saint_louis	110	27
saint_paul	65	4
saint_petersburg	68	11
salt_lake_city	109	35
san_diego	109	14
san_jose	87	12
seattle	149	30
tacoma	81	10
tampa	72	15
toronto	141	17
westchester_county	24	4

Table 6-1: City names in ATIS-2 and ATIS-3. The ATIS-3 only city names were the simulated new words in our pilot experiment on flexible vocabulary recognition.

current SUMMIT architectural implementation does not, per se, permit ease of additions to the vocabulary because the application of the phonological rules cannot be performed on incremental changes. However, we view this as a limitation of the present implementation, and not as a fundamental shortcoming of the pronunciation graph approach. For example, the work of Mohri on finite-state transducers ([50], [51]) can in theory be used to create a transducer representation of phonological rules, which can then be applied dynamically to a pronunciation graph. We are not aware of an actual existing implementation of such a system, nor of whether any engineering difficulties will arise so as to render such an approach impractical. In principle, we can envision such a possibility and for the purposes of our study, we will assume that a mechanism for incremental updates exist. The issue then is how to set the weights for the arcs of the newly added words. We will simply set their weights to zero, which corresponds to a neutral weight in the SUMMIT system.

For the purposes of the pilot study simulation, we take the full recognizer as the well trained recognizer. We modify the ATIS-3 city names in the pronunciation graph so that they have arc weights of negative infinity associated with them to simulate the stripped recognizer which only knows ATIS-2 cities. We choose to do this rather than remove the words from the vocabulary and attempt to train a new pronunciation graph because we do not have a mechanism for obtaining forced alignments for utterances with unknown words. Removal of the sentences containing ATIS-3 cities would leave us with a very small training set. Furthermore, development of a well trained recognizer takes numerous iterations and to train a true "small" recognizer would require a large investment in time and effort. Finally, we change these same arc weights to zero to simulate a recognizer with the ATIS-3 cities added to the vocabulary as new words.

A final issue is how to handle the simulated new words in the word bigram language model. Since our simulated environment assumes that we know the category of the new words, namely city names in our case, we can use a class bigram where all the city names form a class. Specifically, we let

$$Pr(city_n|word_{n-1}) = Pr(city_n|city\_class) * Pr(city\_class|word_{n-1})$$

For the $Pr(city_n|city\_class)$, we discovered that in a fully trained system, setting this probability to be $1/number\_of\_cities$ results in error rate equivalent to using the actual unigram within-class probabilities. Since this uniform within-class prediction makes handling the addition of city names straightforward and it does not detrimentally alter the baseline error

rate performance, we adopt this mechanism. Note that the same bigram language model issues arise for the ANGIE (with word bigram) recognizer and we will adopt the same approach there.

6.1.3 ANGIE Implementation

Recall from Chapter 2 that ANGIE consists of both a set of context-free rules and also a probability model. We believe that our rules are not specific to any lexicon. Thus, we do not see any need to alter our rules neither in the creation of the reduced recognizer nor in the subsequent addition of simulated new words to the recognizer. On the other hand, the probabilities are likely to be greatly impacted from not having the simulated new words in the training data. To simulate training in the absence of the new words, we will follow a procedure similar to the one we used for the SUMMIT case. Namely, we will keep the new words in the training set; however, when it comes time to update the ANGIE probabilities in the training process, we will not update those attributable to the new words. Thus, the reduced recognizer will have an ANGIE model trained without any contributions from the new words.

We had hoped that ANGIE's extensive sharing would be sufficient to support the new words once their baseforms were added to the lexicon, without further complications. Unfortunately, we discovered one serious problem. In paticular, our trained model had numerous zero advancement probabilities for many of the new words. This is not unexpected, as the advancement probabilities are conditioned on the entire left column context, which includes the upper layers of the subword structure as well as the lower ones. Given that our vocabulary size is limited, we can only expect to see a subset of all possible patterns for left column contexts. The bottom-up trigram probabilities did not prove to be a problem because the space consists of only three categories and was apparently well covered by training data.

We can suggest at least four ways to work around this problem. The first approach is to smooth the advancement probabilities in some manner. This was what was done in effect in the SUMMIT case, where the assignment of neutral arc weights can be thought of as "stealing" a small amount of probability mass for the new words. This may have a serious disadvantage in that zero probability events tend to provide a large amount of constraint and we do not want to lose them if possible. Thus, it would be desireable to only remove the zero probabilities when they affect the new words. This brings us to the second approach.

The second approach is to observe that a given phoneme generates only a small subset of possible one or two phone sequences. We can consider the phonemic baseform for the new word, and generate a list of possible phone sequences using a very simple model, such as a model which accounts for bigram probabilities at phoneme boundaries for the generated phone sequences. These phone sequences can then be parsed via ANGIE, and the appropriate probabilities updated.

However, since we have an excellent model of subword structure, namely ANGIE, why not use it to generate the phone sequences as well? This use of ANGIE is very similar to the sound-to-letter work reported in [70], except, here we are generating phones instead of letters. This was the approach we followed. We ran ANGIE in a phoneme-to-phone mode to generate possible phone sequences for the phonemic baseforms for the new words. We can do this by searching over possible phone sequences with ANGIE, and constraining the permitted phoneme sequence. We took the phone sequences obtained in this manner and added them to the training data to give some support for the new words when simulating their addition to the system. Examples of the phone sequences generated in this manner are shown in Figure 6-1. Our implementation ran in batch mode, however, it can be easily adopted to run online in an actual deployed system, perhaps after a list is retrieved from an information source.

A final possibility would be to mark the baseforms for the new words added to the system, and as we encounter those baseforms in the search process, modify any zero (or very low) probabilities encountered to permit the parse within the new word to proceed. The advantage of this approach is that only a few new words are likely to be postulated for a given utterance. This approach would only perform the computation needed to handle those words, rather than for the entire list of new words. The entire list can be quite long if a database retrieval results in many matches. We believe this solution to also be a relatively straightforward process, although we have not invested the effort into implementing it for the pilot study.

6.1.4 ANGIE-plus-TINA Implementation

The addition of new words to TINA can be handled in a manner similar to the mechanism for the word bigram. For TINA, it is actually easier because the TINA grammar already includes a non-terminal category for city names. We simply license the new words as terminals

New Word	Generated Phonetic Realizations
Charlotte	sh epi aar r l ax tcl t
	sh epi aar r l ix tcl t
	sh epi aar l ax tcl t
	sh epi aar r l ax t
	sh aar r l ax tcl t
	sh epi aar l ix tcl t
	sh epi aar r l ix t
	sh epi aar l ax t
	sh aar r l ix tcl t
	sh epi aar r l ax dx
Tampa	tcl t ae ah m pcl p ah
	tcl t ae ah m pcl p ix
	tcl t ae hv m pcl p ah
	tcl t ae hv m pcl p ix
	tcl t ae ah m pcl ah
	tcl t ae m pcl p ah
	tcl t ae m pcl p ix
	tcl t ae ah m pcl p ax
	tcl t ae hv m pcl ah
	tcl t ae hv m pcl p ax

Figure 6-1: Top ten phone sequences hypothesized by ANGIE for the new words "Charlotte" and "Tampa."

	SUMMIT	ANGIE	ANGIE-plus-TINA
Reduced Vocabulary	34.2%	31.2%	32.8%
Augmented Vocabulary	19.2%	19.2%	15.2%
Full Vocabulary	18.9%	18.8%	14.8%

Table 6-2: Error rates of different systems in the presence of simulated new word additions to the active vocabulary.

under the city name category and dynamically redistribute the probabilities under that category uniformly over all terminals in it. TINA also has certain "context-dependent start probabilities" based on a rule-external left context. Those probabilities were disabled for the city name category for the purposes of this experiment. Since our experimental setup supposed that we know the category of new words, inserting the new words under the city name category is a fair approach.

6.1.5 Results

We compared the incorporation of new words into the active vocabulary across several data points. We want to see how ANGIE-based subword models compare to a pronunciation graph model. We also want to see how TINA integration compares to the word class bigram language model. Finally, we want to know how far away our simulated new word enhanced systems are from a well trained system. The results are summarized in Table 6-2. Along one dimension is the vocabulary used: reduced (without ATIS-3 cities), augmented (with ATIS-3 cities added as simulated new words), and full. Along the other dimension is the particular system: SUMMIT (pronunciation graph with word class bigram), ANGIE (with word class bigram), and ANGIE-plus-TINA (with integrated TINA).

There are several points worth noticing. Perhaps the most disappointing to us is that the augmented ANGIE system does not perform any better than the augmented SUMMIT system. This may suggest that ANGIE's extensive sharing is not of immense benefit. A closer evaluation shows that the explanation is that in this test case, the lack of subword training does not result in a serious degradation in performance in the first instance. The full vocabulary ANGIE and SUMMIT achieved error rates of 18.8% and 18.9%, respectively. The augmented ANGIE and SUMMIT systems both achieved error rates of 19.2%, only 0.3% to 0.4% higher. Our results here are actually consistent with those obtained in Hetherington

([25], section 5.5.2):

> Overall performance was virtually identical between the two systems. Over the new words only, the word-error rate increased by only 0.4% (factor of 1.1). Evidently, the lexical arc weights were unimportant for the new words. Training the lexical models on examples of the new words, as in the baseline system, did not improve performance significantly. These results suggest that we do not need to worry about computationally expensive corrective training when adding new words.

However, they seem to contradict other findings in the literature which conclude that the lexical weights do matter (e.g., [78]). One possible explanation is that our choice of the simulated new words relies less on the subword model because our new words are relatively unambiguous in terms of pronunciations and occur in contexts which are well specified by the word language model. In the context of a conversational system retrieving a list of items from a database, the latter condition will likely be met on many occassions. However, further experimentation in cases where the first condition does not hold is warranted to better determine the importance of subword training for new word additions to the vocabulary.

Another item to note is that when the vocabulary is reduced, the ANGIE system performs better than the SUMMIT system. This may suggest that ANGIE is better at handling unknown words during recognition, however, without a more extensive study, we hesitate to state this as a concrete conclusion at this point.

When we look at the ANGIE-plus-TINA system's performance, we see that it suffers a small degradation (14.8% to 15.2% error rate) going from the full vocabulary to the augmented vocabulary. When we dissect this experiment by considering an augmented ANGIE and a fully trained TINA and vice-versa, we find that the degradation is split evenly into a slight increase (0.2%) for each. Thus, TINA suffers a very slight decline from having a well trained model of city name probabilities to one of uniform distribution over the city name category. Despite this, the augmented vocabulary ANGIE-plus-TINA system still performs much better than even the full vocabulary SUMMIT system. Thus the benefits of TINA integration exist even in the presence of new word additions. We should keep in mind that the TINA system has a much higher level of performance than the class bigram system; thus, it is not unreasonable to see TINA suffer slightly more when the available training data

are reduced.

One final point to mention is that the one case where TINA integration seems inferior to using a word bigram is in the reduced vocabulary case. Here, we see that ANGIE-plus-TINA achieves a 32.8% word error rate vs. 31.2% for ANGIE plus word bigram. A likely explanation for this is that with the reduced vocabulary, the frequent occurrence of unknown words makes it impossible for TINA to find a reasonable parse for many utterances, even with our robust parsing mechanism described in Chapter 5.

6.2 Hierarchical Duration Modelling

The Chung duration model ([9], [8]) is based on ANGIE and attempts to account for the durational relationships of sublexical units residing at various levels of the phonological hierarchy. For example, in extracting the duration pattern at the syllable level, the model tries to compensate for effects at lower levels in the hierarchy, such as at the phoneme level, through a process of normalizing the observed syllable distribution across different phonemic realizations. Similarly, phonemic durations are normalized across phonetic variations. Further, Chung's model is able to account for speaking rate variability. Her model calculates a speaking rate parameter from a single word occurrence by comparing the normalized duration for that word to the normalized duration for all words. The normalized duration already accounts of variability across different words and different realizations for the same word. For example, consider two occurrences of the word "butter" where the /t/ is flapped in one case and not in the other. If we considered only the absolute durations of the word "butter," we may wrongly conclude the the flapped realization reflects a faster speaking rate, even when the speaking rates are equal, because the flapped realization is of shorter duration. The normalization within her model would account for such phenomena to achieve a more accurate speaking rate calculation. Chung has implemented her model within the ANGIE framework and has found it to reduce phonetic recognition error rates within the ATIS domain by up to 7.7% (from 29.7% to 27.4%) in some cases. For our second pilot study, we obtained Chung's duration model and incorporated it into our word-spotter (Chapter 4) to see if it would lead to any improvement.

Our first experiment was to select a case where we expect duration modelling to be an immense help, and which proves to be quite difficult for our word-spotter to handle.

Specifically, we are referring to the "Newark" vs. "New York" confusion. One of the major negative factors affecting our word-spotter's performance is the system's tendency to output "Newark" as the spotted keyword where the actual keyword present in the acoustics is "New York." We took a subset of utterances consisting only of those with "Newark" or "New York" and evaluated them with a Chung duration model based post-processor. The initial results were quite promising. The original confusion error rate was 19%. After rescoring the regions where either "Newark" or "New York" were predicted with the duration model, the error rate was reduced to 6%. We should recall that the original front-end processor already has a simple phone duration model. The improvement was attributable to the contribution of Chung hierarchical duration model, made possible through the use of ANGIE, beyond the contribution of the simple phone duration model.

The first test was somewhat biased since we picked an area where we expect duration modelling to help greatly, namely an otherwise acoustically confusable pair. Inspired by the initially promising results, the other experiment we performed was to incorporate duration scoring for all the words in our keyword set and to integrate this score directly into the word-spotter's search rather than as a post-processor. After doing so, with word filler constraints, the FOM of our word-spotter increased from 89.3 to 91.6, which reduces the gap between our word-spotter and a full word recognition system (93.9 FOM) to half its former value.

Since we were conducting the experiments on the word filler constraints system, we also tried incorporating Chung duration scoring for the filler space as well as for the keyword space. However, our attempts at doing so were woefully unsuccessful. Performance deteriorated tremendously. Our suspicion is that the duration model is only effective when used on roughly correctly hypothesized words. Without cross word constraints, such as a word bigram, the filler space in our word-spotter, with whole word fillers, is undoubtedly highly errorful. While this does not hurt the word-spotter performance, we believe that the Chung duration model did not operate well when overwhelmed by the number of incorrect hypotheses. In other words, we suspect that the Chung duration model is good at distinguishing between several reasonable hypotheses, but it breaks down when applied to less reasonable hypotheses. Clearly, further work needs to be performed to better evaluate the situation. Some further analysis can be found in [8].

6.3 Summary

In this chapter, we described two pilot experiments we have conducted in an attempt to validate some of the perceived and claimed benefits of our ANGIE framework. Our first experiment tries to support our claim that ANGIE is better able to handle flexible vocabulary recognition than other frameworks. Namely, through extensive subword sharing and bottom-up parsing, the addition of words to the ANGIE vocabulary is relatively straightforward and does not require further lexical training, as in the case of lexical arc weights in a pronunciation graph based system. Our testing scenario is that of a conversational system retrieving a list of of items, some of which may not be in the recognizer's vocabuary, in response to a user query. The goal is to dynamically expand the vocabulary to support the new words. For our actual study, we simulated a system which knows only about the ATIS-2 cities, and not the ATIS-3 cities, by taking out the lexical data for the ATIS-3 cities from our models. For the SUMMIT system, this meant reducing the lexical arc weights for the ATIS-3 cities to negative infinity to simulate a reduced system, and resetting them to zero to simulate augmenting the reduced system with the ATIS-3 city names. For ANGIE, this meant not updating the counts attributable to ATIS-3 cities during training, and removing them from the lexicon during testing. For the purposes of our experiments, we assume that the correct phonemic baseforms for the simulated new words are provided to us already. An actual mechanism for doing so might be a dictionary, or a letter-to-sound system, perhaps based on ANGIE itself.

After adding the new words to the subword lexical modelling system, the other issue we have to worry about is the word language model. We considered two such language models, a word bigram and the TINA natural language understanding system. For the word bigram, we created a class bigram where "city name" appears as a class and the probabilities within the class are uniformly distributed. For TINA, there is already a non-terminal category for city name, so we merely altered the probability model to likewise be uniformly distributed over the city names.

If we actually try to run the augmented ANGIE-based system, we discover a problem with zero advancement probabilities. We can add a mechanism to smooth away these zero probabilities to enable the added words to be recognized and we have done so. When we we compare the ANGIE-based system to the SUMMIT pronunciation graph based system,

both using the same word class bigram, we discover that the performance was exactly the same, 0.2% to 0.3% worse for the augmented systems than for the fully trained versions. Thus, it would seem that in this respect, ANGIE's extensive sharing does not result in an observable performance advantage. However, we should note the limited conditions of our test, namely, that the choice of city names, which are relatively unambiguous and do not share much with existing words in the vocabulary, may bias the test towards not showing an advantage. The fact that performance overall only declines by 0.2% to 0.3% further suggests that our particular choice of new words does not benefit much from lexical training. The general experience in the literature is that lexical training does help in many cases (e.g. [78]). Further, while in theory, the implementation of dynamic application of phonological rules within a pronunciation graph approach can be done, for example, through the use of finite-state transducers, we do not know of an actual implementation of such. The engineering issues involved may be more complex that those involved with smoothing the zero advancement probabilities within ANGIE. We should also note that the reduced ANGIE system performs better than the reduced SUMMIT system 31.2% error rate vs. 34.2%). This may suggest that ANGIE is better able to handle recognition under the adversity of a high frequency of occurrence of unknown words.

Further, when we consider the addition of TINA to the ANGIE-based system, we find that the ANGIE-plus-TINA combination performs better in the augmented configuration than even the fully trained SUMMIT with word bigram configuration. The decline in performance as compared to a fully trained ANGIE-plus-TINA system is small, from 14.8% error rate to 15.2%. This decline is split about evenly between ANGIE and TINA, each contributing 0.2%. Overall, we are delighted to see that overall, the ANGIE-plus-TINA system performs very favorably in the presence of new words added to the vocabulary.

The other pilot study is an attempt to leverage off the ANGIE parse tree for prosodic modelling. Specifically, we incorporate Chung's hierarchical duration model ([9], [8]) into our word-spotter. An initial test suggests that the model, which makes extensive use of the relative duration of subword units between adjacent levels in the ANGIE layered representation, is very helpful at disambiguating two highly confusable words, "Newark" and "New York." Encouraged by the positive results, we integrated the Chung model into the word-spotter and found that if we include duration scoring for the keywords, we can improve the FOM from 89.3 to 91.6 for the known word filler configuration. We also tried using the

Chung model for the words hypothesized for the filler space as well, but performance deteriorated greatly. We suspect this is because the filler space is hypothesized with numerous errors and this causes the Chung model to break down.

The two pilot experiments reported here suggest that some of the perceived benefits of ANGIE we envision have indeed been realized, e.g., the ability to include a subword structure based duration model. However, some others, such as the promise of flexible vocabularies, have only been partially proven, e.g., in the case of the combined ANGIE-plus-TINA system. Further work needs to be done to see exactly how many promises ANGIE can fulfill. For example, the better ability to cope with a higher unknown word rate, along with the ability to integrate with an NL system, may facilliate the implementation of a new word detector with more extensive subword and higher-level linguistic support. Overall, we found the results of our pilot studies to be promising.

Chapter 7

Summary and Future Directions

In this thesis, we worked towards an integrated system for speech recognition motivated by an underlying bottom-up parsing philosophy beneath the word level and a top-down natural language understanding one above the word level. The subword lexical modelling framework, ANGIE, was designed in a manner to promote maximal bottom-up sharing, so that common substructures can be pooled during training and generalized to unseen instances. The ANGIE framework also attempts to model syllable structure explicitly, within a hierarchical structure. We combined ANGIE with a top-down linguistic component, the TINA natural language understanding system, to implement a single integrated system accompanied by improvements in recognition performance.

Below, we summarize the developments in this thesis. We start by introducing subword lexical modelling to capture phonology, both existing approaches and our innovations, and introduce our ANGIE framework. Next, we recapitulate the implementation of various speech recognition systems based on ANGIE, including some of the engineering issues encountered, and ending with a system that includes an integrated TINA natural language understanding component. Then we describe two pilot studies conducted in the area of adding new words to the system's vocabulary and duration modelling. Finally, we conclude with suggestions for natural extensions of our work.

7.1 Phonological Modelling

A typical speech recognition system needs to be able to handle variations in the pronunciations of various words. These variations may arise not only because of the possibility of

alternate pronunciations, as in the two manners of saying "either," but also because of the various phonological variations. For example, the /t/ sound in "butter" may be carefully pronounced, with the speaker placing the tip of her tongue to the back of her upper front teeth to generate a complete closure followed by an aspirant burst. However, it is also possible for the speaker to generate the same /t/ by quickly flapping her tongue to the roof of her mouth. Such variations in speech are quite common and must be accounted for by a recognizer. Most modern recognizers model acoustic sounds via units typically the size of a phone. The use of subword units mitigates training data sparsity by allowing a given subword unit to be shared across multiple words in the vocabulary. The recognizer has a collection of subword units and it needs to decode hypothesized strings of these units into words, taking into account the variations in pronunications of words. The component which makes this process possible is the subword lexical model, the focus of this thesis.

7.1.1 Existing Approaches

We can categorize the existing approaches to subword lexical modelling into two broad classes. The first is the explicit phone graph approach. A phone graph is used to represent the permissible pronunciations of a given word. A path through the graph corresponds to a pronunciation consisting of the phones associated with the edges traversed in the path. To capture the likelihood of various pronunciations, the edges are associated with weights, which represent the costs (or benefits) of traversing the respective edges. The other primary class of modelling techniques is to implicitly model the variations through the parameters of a hidden Markov model or through the parameters of the acoustic model or through both. In both of these cases, there is an underlying assumption that a word is best modelled solely as a sequence of the subword units. Specifically, the subword structure being modelled is *flat*.

7.1.2 Inspiring a Hierarchical Model

Much work in phonology suggests an alternative subword model, namely, one that is hierarchical and is aware of intermediate layer above the phones, but below the words, consisting of syllables. The work of Kahn ([33]), in particular, demonstrated the importance of syllables in explaining phonological processes. Further, Church ([10]) was actually able to implement a computational framework, which can take a string of phones, and discover

a sequence of syllables in a bottom-up manner. Church's success, although limited to an environment in which only primarily error-free phone strings were encountered, suggested to us that a bottom-up, hierarchical subword lexical model may possibly be productive in addressing the problems relating to new words, in terms of facilitating their detection and also of easing their incorporation into the system's vocabulary. Thus inspired, we to design our framework, named ANGIE.[1]

7.1.3 ANGIE

ANGIE is a hierarchical, layered, probabilistic subword modelling framework for speech processing. We presented the details of the framework in Chapter 2. In our model, subword structure is represented via a context-free grammar, presently generated by hand, and an automatically trained probability model. The ANGIE framework captures phonology, syllabification, and morphology in a unified framework. The ANGIE parser operates in a bottom-up manner, permitting the sharing of substructures amongst different words, and providing support for the discovery of word-like units. We believe a bottom-up design permits the system to best support flexible vocabulary recognition, since information learned for one word may be shared with other words which have similar subword structures, even if those words are new words being added to the vocabulary or new words being detected in speech. An important goal of this thesis was to develop the ANGIE framework and to demonstrate that the framework can support various speech recognition tasks, from phonetic recognition to word-spotting to full recognition.

7.2 Speech Recognition with ANGIE

In this thesis, we demonstrated the feasibility of ANGIE as a subword model for speech recognition by showing its effectiveness on several tasks. All the tasks were performed on a subset of the ATIS corpus. Approximately 5000 utterances were used as acoustic training data, and the standard December '93 test set (approximately 950 utterances) was used as test data. Because we wanted to focus our exploration on subword lexical modelling, the acoustic models were kept as simple as possible. Context-independent, mixture diag-

[1]The first version of the ANGIE framework was designed by Stephanie Seneff and partially inspired by the work of Meng ([49]). In this thesis, we developed the subsequent generations of the framework, along with the recognition systems that use it.

onal Gaussian models with a feature set consisting of averages of MFCCs across thirds of segments and derivatives across boundaries were used.

The first task we tackled was phonetic recognition. In phonetic recognition, we demonstrated that ANGIE's enhanced subword model led to an improved error rate of 36.1% as compared to 39.8% for a baseline system implemented with a phone bigram subword model. Further examination suggested that the longer distance linguistic information available via ANGIE's upper layers was responsible for approximately 1.6 percentage points of the 3.7 percentage point decrease in absolute error rate. The rest may be attributable to improved phonological modelling and other factors.

Our next challenge was the implementation of a word-spotter based on ANGIE. The ATIS city names were used as our set of keywords. Here, an assessment of our word-spotter's performance is complicated by the lack of a directly comparable baseline system. We implemented our ANGIE-based word-spotter with a variety of possible subword lexical models for the filler space, ranging from a not too constraining phone bigram arrangement, without any word constraints, to a very constraining, full ANGIE subword model which requires that only a given vocabulary of words are permitted within the filler space. Our word-spotter's performance ranged from a figure-of-merit of 85.3 for the phone-bigram filler to 89.3 for the known-words only model. The only baseline comparison we have is that of a full word recognition system, augmented by a word bigram, which achieves a FOM of 93.9. Full recognition is generally believed to represent the high end of word-spotting performance. We believe our word-spotter to be competitive because if we consider the results presented in Manos ([45]), the drop in performance from his full recognition baseline to his context-independent phone fillers system is similar to the drop from our full recognition baseline to our phone bigram system. His context-independent phone fillers system matches our phone-bigram system well, and, in both cases, ATIS data with city names as keywords were used. However, Manos had larger and more balanced data sets, so the comparison is not exact.

As we mentioned in the previous paragraph, we experimented with alternative subword lexical models for the word-spotter's filler space. The results of our experimentation validate the common intuition that the more constraints placed upon the system, the better the performance. We were surprised by three results though. One is that we generally do not incur a running speed penality as we increased the linguistic constraint. A likely explanation

is that the additional constraints help to control the bushiness of our search space. Another result is that we had anticipated a benefit from allowing the hypothesis of unknown words for the filler space, as opposed to permitting only words known in the vocabulary, for the configurations involving word models in the filler space. However, our experiments showed that permitting the unknown word to be hypothesized led to inferior performance. While our low out-of-vocabulary rate may have contributed to this outcome, we found that even with an out-of-vocabulary rate of 8%, permitting unknown words was detrimental. This led us to suspect that having out-of-vocabulary words appear in data to be word-spotted on is not nearly as detrimental as having them in data to be fully recognized (e.g., [25] found an average of 1.46 errors per new out-of-vocabulary word, substantially higher than the one error attributable to not recognizing the word correctly). Further, permitting an unknown word in filler space may allow the unknown word to be hypothesized in place of a poorly articulated keyword. Our final noteworthy discovery from the word-spotting filler model experiments was that using syllables to model the filler space represented an impressive speed vs. FOM trade-off. FOM performance was very respectable, but more importantly, the speed of this configuration was extremely fast.

Our final empirical test of ANGIE's feasibility was to implement a full word continuous speech recognition system. The extension of our word-spotting system to full word recgonition was not too difficult. Some adjustments to the search organization were needed along with the addition of support for a cross-word language model, in our case a word bigram. Our ANGIE-based recognizer achieved a recognition error rate of 18.8%, comparable to a baseline of 18.9% using MIT's SUMMIT recognizer. We also attempted to implement a syllable-based recognizer, inspired by the success of syllables in word-spotting, but were unsuccessful, at least with the ATIS corpus.

In our work on full recognition, we also experimented with integrating our ANGIE recognizer with a natural language understanding system based on TINA ([69]). Previous work suggests that an N-best rescoring approach yields only marginal improvement ([52]), but a more tightly integrated word graph approach ([71]) results in a much larger improvement. TINA is based on a context-free framework with automatically trainable probabilities, a robust parsing mechanism, and support for constraints to enforce agreements such as subject-verb. We felt that ANGIE's stack decoder recognizer implementation can easily incorporate TINA into its search process. In our implementation of an ANGIE-plus-TINA recognizer, the

stack decoder obtains a partial parse score from TINA whenever a word ending is proposed by ANGIE. The system had some initial intractibility difficulties which were found to be related to the bushiness of TINA's robust parse mechanism. We designed an alternate robust parse mechanism, which operated in a much greedier manner, to make the search practical. The inclusion of TINA in our recognizer increased performance dramatically. Error rate of the combined system fell to 14.8% from a baseline of 18.8% with ANGIE and a word bigram.

We feel that our successful implementation of ANGIE-based systems for three recognition tasks – phonetic recognition, word-spotting, and full word recognition – demonstrated empirically that our subword lexical modelling framework provides a feasible system for speech recognition. We are particularly pleased with the results obtained with TINA integration. We next describe some of the main engineering issues encountered as well as some attempts to demonstrate several advantages of our framework over existing ones.

7.3 Engineering Challenges of Search

During our implementation of various recognition systems based on the ANGIE framework, we faced critical engineering challenges relating to effective search strategies as well as to controlling the computational complexity of our system, both in terms of time and space. When it comes to overall search organization, the greatest problem encountered was in how to compare theories through a segmentation graph when the theories account for different segments. Short vs. long theories proved to be particularly problematic because longer theories tend to have lower scores in a probabilistic system, as log-likelihoods are typically employed in the scoring function. In our first recognition system, the phonetic recognizer, we used a best-first search and balanced off the short vs. long theories with a series of heuristic functions and weights. However, our subsequent work in word-spotting necessitated an alternate solution. We adopted a stack decoder approach, where the search queue is organized such that theories are sorted by time, the net result of which is that theories which cover the same time span, i.e., from time 0 to time t, are compared against each other. Of course, even paths which cover the same time span may have different numbers of lexical units accounted for, and that difference needs to be compensated for as

well. For that, we keep a phone reward heuristic.[2]

Besides the choice of the basic search strategy, we also repeatedly encountered a basic tradeoff between time/space requirements and the accuracy of the scoring model employed during the search process. This tradeoff takes many forms but it underlyingly represents a decision as to how much distinguishing information to maintain for each individual path hypothesis in our search. Obviously, the more information that is maintained, the greater the space requirements of our system will be. But more importantly, if we keep much distinguishing information for each path, then we have many fewer opportunities to prune "similar" paths, share similar paths, or both. An early example of this tradeoff was encountered while we were still building the ANGIE framework. The original implementation required a full ANGIE column to determine the advancement probability at all times. This implies that the only opportunity to prune was when two paths share the same ending column, i.e., same phone and upper layer categories. This implementation became unwieldy so we changed the design. In the new design, as soon as a word end is proposed in a path, we carry only the identity of the words in the path and the final phone. This simplification enables a much greater amount of pruning at word boundaries. Moreover, it allows us to share ANGIE theories for the next word encountered in the path with other paths which may be exploring similar word starting contexts, reducing space and time requirements by an order of magnitude. However, these advantages come at a cost. Specifically, we no longer have the detailed context information at the word boundaries. In this case, it worked to our advantage because we were also experiencing sparse data problems at word boundaries. Typically, the advantages and disadvantages have to be considered. In our experience, the choice usually comes down to one of pragmatics and experimentation.

Throughout our implementational experience, we also encountered the issue of "bundling" on numerous occassions. We have used the term "sharing" in this thesis to refer to organizing our searches such that computations needed by several different theories are only performed and stored once. Thus, sharing refers to a strategy to maximize memoization of the results of computing common subproblems. Our use of the term "bundling" refers to a related, but different concept. Frequently, in speech and natural language, we arrive at a logical unit which may have varying internal interpretations, but external to that unit, the

[2] Alternatively, we can sort our stack decoder queue by lexical units instead of acoustic time spans. We have not yet pursued this approach.

precise internal interpretation may not matter much. Rather, all that is needed is a representative internal interpretation. This phenomenon is reminiscent of the role of functional and data abstraction in computer program design. It appears that natural language exhibits some of the same designs! An example of this in our work would be that once a lexical unit, such as a word, is proposed bottom-up by ANGIE, the specifics of the word substructure is no longer useful information in our system. The only representative information we need about the word substructure is a score.

Alternatively, perhaps the precise internal interpretation does matter, but can be computed from the representative interpretation upon demand at low cost, thus favoring a bundled implementation from a time and space complexity point of view. For example, in our word-spotting work, even if we consider words as fillers, if a given part of the utterance, say time t_1 to t_2, can be hypothesized as one of several filler words, the precise word does not matter much to the word-spotter. Rather, all the word-spotter needs is the score of the best word hypothesis for the region. Thus, in our implementation of a word-spotter, we kept homonyms "bundled" together and took the score of the best homonym as representative. When we considered full word recognition with cross-word constraints, namely a word bigram, this bundling simplification is no longer appropriate.

We believe that a related opportunity, which we call "merging," may exist within our NL integration work, and we have discussed some initial experiments at creating merge categories. Specifically, we feel that certain NL categories, such as direct objects, may provide an opportunity for merging the alternatives within those categories to promote memoization. A given sequence of words may be encountered multiple times and hypothesized as a direct object. For example, this can occur with alternate theories for the remainder of the utterance. Since direct objects occur fairly frequently and have relatively self-contained linguistics, it would seem reasonable to avoid performing the computation to generate the same parse each time it is encountered. One way to implement this is to separate out the TINA theories whenever a direct object is proposed and group them into a collection. If the particular word sequence already has a parse in the collection, then we do not need to recompute the parse. Thus, computation along multiple divergent paths can be merged when one of these merge categories are encountered, and separated again when we exit the merge category. Our initial experiments along this line of exploration have not yet proved beneficial, but we believe that since natural language understanding systems provide few

opportunities to prune due to extremely long distance linguistic constraints (thus, it is very difficult to consider two theories equivalent because of the long distance context), finding and implementing the proper merging may reduce computational complexity significantly.

7.4 Pilot Studies

Thus far, we have presented our ANGIE framework for subword lexical modelling and have shown that it can be used successfully to support various speech recognition tasks. While we feel that this is a notable accomplishment, especially given that ANGIE represents a significant departure from existing approaches to subword lexical modelling, we have more auspicious goals for ANGIE. We designed ANGIE with the hope that it will be helpful in addressing several difficult challenges in speech recognition, including dealing with new, out-of-vocabulary words and prosodic modelling. We have already presented one experiment, namely the integration of ANGIE and TINA, so far in this summary, that went beyond a pure feasibility demonstration and touched upon new ground. Our actual motivation for the experiment was to work towards a system which can handle dynamic vocabulary updates, addressing partially the difficulties of dealing with new words. In the chapter on pilot studies (Chapter 6), we discussed an extended experiment on dynamic vocabulary updates and also an experiment addressing prosodic modelling in terms of an ANGIE-based duration model. We summarize these two pilot studies in the following subsections.

7.4.1 New Words

Our first pilot experiment involved the addition of new words to a recognizer's vocabulary without requiring any additional training data. The scenario contemplated is that of a conversational system retrieving a list from a database where some of the items on the list may not be in the system's vocabulary. The recognizer must add those items to its vocabulary. The assumptions for the scenario are that we can obtain pronunciation baseforms for the words to be added and that we also know their linguistic category. Our actual experiment divided the list of city names in ATIS into ATIS-2 and ATIS-3-only cities. We assumed that we had a recognizer well trained with ATIS-2 cities and treated the ATIS-3-only cities as new words to be added to the vocabulary. The challenge of dynamic vocabulary updates is that the recognizer needs to provide both subword lexical and word linguistic support for the

added words. Our first comparison was between a baseline SUMMIT system and an ANGIE-based system, both employing word class bigrams. In both systems, language model support was provided by redistributing the class probabilities for the "city name" class uniformly over the expanded vocabulary of city names. In the SUMMIT case, we simulated adding the new words to the lexicon with neutral lexical arc weights in the pronunciation graph. In the ANGIE case, we added the baseforms to the lexicon, relying on ANGIE's bottom-up sharing to provide support for the subword structures of the added words. However, we encountered several instances of zero probabilities in the ANGIE probability model, which we circumvented by applying a simulated smoothing procedure. Our results showed that both SUMMIT and ANGIE performed comparably, both achieving error rates of 19.2%, as compared to fully trained systems with error rates of 18.9% and 18.8% for SUMMIT and ANGIE, respectively. Thus, it appeared that ANGIE's sharing did not yield any improvement over SUMMIT; however, we should note that the actual decrease in performance from not having subword training was small, possibly suggesting that the city names added were distinct enough to be easily recognized without much subword support. Further, we should note that the simulated procedure for creating zero lexical arc weights in the SUMMIT pronunciation graph may actually be very difficult to implement in practice, whereas the only change required in ANGIE is the addition of the smoothing of zero probabilities, which we believe to be more straightforward. Finally, we should mention the performances of the reduced systems that did not know about the simulated new city names. The reduced SUMMIT system achieved an error rate of 34.2% as compared to the reduced ANGIE system's 31.2%, suggesting that in the presence of many out-of-vocabulary words, ANGIE performs better.

Our other evaluation of the addition of new words involved the ANGIE-plus-TINA system. For TINA, the simulated new words were added to the "city name" category and the probabilities within that category were redistributed over the expanded vocabulary, as in the word class bigram case. The combined system achieved an error rate of 15.2% with the augmented vocabulary, as compared to 14.8% for the same system trained on the full vocabulary. The degradation is attributable in even portions (0.2% point difference each) to reduced ANGIE and reduced TINA training. We should note that the expanded vocabulary performance exceeds the 18.8% error rate of a fully trained system using a word bigram instead of TINA by a substantial margin.

7.4.2 Duration Modelling

Our other pilot study was in the area of prosodic modelling. Since ANGIE provides a parse tree describing the subword structure of each word, we realized that this information can be leveraged for use in a prosodic model. Our colleague, Grace Chung, has been working with a hierarchical duration model ([8]) based on ANGIE parse trees. The only duration modelling we had in our original system was a simple phone duration model, whose score contributed to the acoustic score from the front-end processor. Chung's hierarchical duration model attempts to normalize durations across the various layers in the ANGIE hierarchy, e.g., in extracting the duration pattern at the syllable level, the model tries to compensate for effects at lower levels in the hierarchy, such as at the phoneme level, by normalizing the observed syllable distribution across different phonemic realizations. Her model can also extract a speaking rate parameter from a single word and normalize for differences in speaking rate based on it. Chung's model has been found to be effective in improving phonetic recognition.

We obtained Chung's model and incorporated it into our word-spotter. Recalling that the keywords "Newark" and "New York" were a major source of errors in our word-spotter, we evaluated the Chung model on this keyword pair. Without her model, our original confusion error for the two keywords was 19%. With it, the error dropped to 6%. Next, we merged the hierarchical duration score into our word-spotter's stack decoder search process. Using the hierarchical duration score only for the keywords in our best performing configuration, we increased the FOM from 89.3 to 91.6. We also attempted to extend the hierarchical duration scoring to the filler space as well, but that resulted in extremely poor performance. We suspect that Chung's model performs best when given reasonable hypotheses but may deterioriate on more errorful data.

7.5 Future Work

While we have made much progress in implementing a hierarchical subword lexical modelling framework, ANGIE, in this thesis, we feel that much more work needs to be done. The wide-ranging nature of this thesis reflects the need to adequately evaluate the ANGIE infrastructure on a variety of tasks to ascertain its effectiveness and also to uncover engineering issues involving the framework. This is a necessity whenever a new computational framework is introduced, especially one which takes a significant departure from existing

approaches. Having created a release 1.0 of the infrastructure, natural suggestions for future work fall into four categories: creating an improved release 2.0, borrowing from the lessons learned in this thesis and retrofitting them into previous systems, further investigating issues not completely resolved in this thesis, and pursuing explorations now enabled by the existence of the new platform.

In terms of release 2.0, the obvious area to address is efficiency. While our implementations run acceptably for a research platform in most cases, they run way too slowly and use too much memory for a real time deployment environment. Our full recognition engine ran unacceptably slowly even for a research platform without some preliminary pruning of the phone graph to be searched. Another area to improve from an engineering standpoint is to handle the smoothing of zero probabilities for new word additions in a more elegant manner than the one we used.

In category two, adapting our techniques to existing systems, we feel that natural language integration is one candidate. A second candidate is the work we have done in incorporating new words. Although our test case did not show the loss of a well trained sublexical model can lead to a large performance drop, nor did it conclusively show ANGIE's sharing to be beneficial, we feel that ANGIE may prove beneficial for other choices of words to be added to the vocabulary. Given this, we feel that it may be possible to use ANGIE to provide the information normally obtained from sublexical training for other approaches, such as for pronunciation graphs.

Discussion of the new word additions naturally brings us to category three. Why did the lack of sublexical training not result in poorer performance? We have seen evidence in the literature suggesting that it sometimes hurts ([78]) and it sometimes does not ([25]). We need to better characterize when performance is expected to suffer without better sublexical training. Further, we need to evaluate in those circumstances, whether ANGIE's sharing is beneficial or not, and if not, what other approaches can be taken. With respect to supporting dynamic vocabularies, we mentioned that we assumed known baseforms for new word additions. However, since ANGIE can also support letter-to-sound generation, a natural extension would be to use generated baseforms. Our colleagues have performed extensive work on evaluating letter-to-sound accuracy of ANGIE ([70]). However, the measure we would like to use is the efficacy of the generated baseforms on recognition performance. If the baseforms were slightly inaccurate, but not to the point of adversely impact recognition

performance, we would still be satisfied. This area needs to be explored.

Our mixed experiences with syllable lexical units also suggest an area for future work. Finally, we had mentioned our initial exploration into creating merge categories to streamline NL processing. While we have not met with success in this thesis, we nevertheless feel that the concept has merit and should be examined more carefully.

Also, the experiments reported in this thesis focused on using context-independent acoustic models but as we mentioned in our background comments on subword lexical modelling, an implicit modelling approach using context-dependent acoustic models to capture much of the variability in word realizations is a fairly common approach. Work needs to be done to better characterize the interaction between context-dependent modelling and using a framework such as ANGIE to capture variability in pronunciations. We do not know how well ANGIE performs either in conjunction with context-dependent acoustic models, or as compared to a system with such models.

In the final category of new work enabled by our platform, we see three broad areas which may prove promising. The first is in prosodic modelling. Our full recognition system, with TINA integration, provides much information about the speech being recognized, at both the subword levels and at the upper linguistic levels. This information can potentially be leveraged for use in prosodic modelling. The second area we have in mind is to further leverage the ANGIE subword structural information in a vein similar to our duration modelling experiments, but for the purposes acoustic modelling. Possibilities might include a different flavor of context-dependent acoustic models enabled by having the ANGIE parse information, such as models for a phone appearing in a certain syllable position, or models for larger units such as entire syllables. Such models may be applied during the search process to rescore a hypothesis as soon as one of the larger units is proposed bottom-up. In the prior paragraph, we had suggested a comparison involving traditional context-dependent models, such as generalized triphones. Perhaps that comparison can be expanded to include ANGIE-enabled context-depedent acoustic models as well. The third area which appeals to us is to explore the issue of new word detection, now that we have a system with subword support for new words. New word detection has proved to be a particularly elusive goal. We hope that our work can contribute towards progress in the area.

Appendix A

Example Set of Rules for ANGIE

Below are the context-free rules used by the ANGIE-based continuous speech recognition system described in Chapter 5. The inclusion of the rules here are intended to aid the reader in understanding the word substructures modelled by ANGIE. The rules are divided into two groups, high level and low level. The high level rules describe the derivation from the start symbol down to the phonemics layer. The low level rules govern the phonemics to phonetics transition and, together with the probability model, account for the phonological modelling within our framework. We should point out that these low level rules can also be replaced with a set of rules for letter terminals for use in letter-to-sound/sound-to-letter generation (c.f. [70]). Conventions for the rules are as follows:

- Lines starting with a semicolon (;) are comments.
- Rules are separated by blank lines.
- The left-hand symbol (LHS) of a rule appears on its own line, prefixed by a period (.).
- Lines following the LHS are alternative right hand sides. The alternatives are separated by either new lines or double vertical bars (||s).
- Alternative symbols are enclosed in parentheses (()s).
- Optional symbols are enclosed in brackets ([]s).
- Terminal symbols are prefixed by a dollar sign ($).

A.1 High Level Rules

```
;;*****************************************************************
;;layer 0 (start symbol)
.sentence
word

;;*****************************************************************
;;layer 1

.word
[fp] fcn
[fp] [pre] sroot [dsuf] sroot2 uroot [dsuf] [isuf]
[fp] [pre] sroot uroot [dsuf] sroot2 [dsuf] [isuf]
sroot [isuf] sroot2 [sroot3] [uroot]
sroot [dsuf] [isuf] || [fp] spre pre sroot

;;*****************************************************************
;;layer 2

.fp
pau [glottal] || glottal

.fcn
[fonset] fnuc [fcoda] [fsuf]

.fsuf
(v d*ed el s*pl m)

.fonset
(sh! b! dh! w! k! f! g! h! m! s! y! d! t! y! l!)

.fnuc
(ao ae aar ah ay uw aor eh ih ow iy er ehr ey el uh)
(iy_the ra_from ey_a en_and ix_in uw_to ux_you ay_i ah_does)

.fcoda
(t d m v z s n d v p ch th f)
```

```
.sroot
[onset] nuc_lax+ coda || [onset] nuc+ [coda] || [onset] lnuc+ lcoda || lnuc+

.spre
(^all ^ad ^com ^dis ^in ^ir ^non ^re ^sub ^un)

.sroot2
[onset] nuc_lax+ coda || [onset] nuc+ [coda] || onset lnuc+ lcoda || lnuc+

.sroot3
[onset] nuc_lax+ coda || [onset] nuc+ [coda] || onset lnuc+ lcoda || lnuc+

.dsuf
[uonset] dnuc [ucoda] || dnuc [umedial] nuc [ucoda]

.pre
[uonset] nuc [ucoda]

.uroot
[uonset] nuc

.isuf
^al (^pl ^ly ^ism) || ^ism [^pl] || ^ment [^pl] || ^th (^y ^pl)
(^th ^al ^ly ^past ^pl ^ing ^est ^er ^able ^ness ^ful ^less)
^past (^pl ^ly) || ^ing (^pl ^ly ^ful ^less ^ness)
^ly ^ness || ^ful (^ly ^ness) || ^er (^pl ^past)

.^ment
m! en t

.^day
d! ey

.^ence
en s
```

```
.^ant
en t

.^ate
ih t

.^ace
ih s

.^age
eh jh

.^ive
ih v

.^y
iy

.^ful
f! el

.^less
l! eh s


.^ness
n! eh s

.^al
el

;;;*************************************************************
;;layer 3

.pau
```

```
wb

.glottal
q

.ambi_cons
(t! v! m! p! n!)

.onset
s! (p t k) [r] || s! (p k) l || s! k (w y) || s! (p t k m n l w)
(p! t! k! b! d! g! r! y! h! l! n! w! m!)
(p! t! k! b! d! g!) r || (p! k! b! g! v!) l || (d! t! k! g! v!) w
t! y || (f! v! m! p! k! b! d! h! s!) y || (f! th!) [r]
(dh! s! sh! z!) || sh! (r w) || s! (f l w) || f! (l r) || (ch! jh! zh! v!)


.uonset
s! (p t k) || s! t r || (p! f!) r || k! w
(p! t! k! b! d! g! r! y! h! l! n! w! m! dh!)
(s! sh! zh! z! th!) || (ch! jh! v! f!)
(f! v! m! p! k! b! d! h! s! n!) y || (p! t! k! b! d! g!) r
(b! p! g! k!) l || (t! k! g!) w || t! y

.dnuc
(eh aa ah ow iy ih ao yu aor uw er em ehr) || (en el ay ae ey ing)

.nuc
(ae eh aa ah ow iy ih ao yu uw er)
(en em el aar ay ey aor) || (ay_i ix_in)

.lnuc+
(ey+ ow+ ay+ iy+ uw+ yu+ aa+) || ey+ ow+


.spre_nuc
(aol+ ihr+)
```

```
.nuc_lax+
(ae+ eh+ ih+ uh+ ao+ ah+ aa+)

.nuc+
(el+ oy+ aw+ ao+) || (aol+ ehr+ aor+ aar+ ihr+ er+) || (ux_you ay_i iy_the)

.coda
t || (m n ng l r) || (m l) (b p f) || (l n) (t d ch) || n (z s)
l (t k g m v) || ng (k g) || (sh ch jh z v n l) || (f v th dh zh)
[s] (p k t) || d s t || (b d g) || (f p k) t || (p k t) s || n jh || (z dh) m

.umedial
(s! b! p! t! l! h! n! k! g! z! v!) || s t! [r] || k (t! s!) || d s! || n m!

.ucoda
(m n l r) || (sh ch jh s v n l ng z) || (f v th dh) || dh m || (p k t b d g)
n (s d t jh) || m (p f) || (k t) s [t] || k t || p t

.lcoda
dh || (m n ng l) || n (t d jh ch) || (z ch jh v n l r) || (f v th s sh zh)
(p k t b d g) || [k] s t || l (d t)



;the function word set

.^indef
ey_a

.^def
dh! iy_the

.^and
en_and d

.^does
```

```
d! ah_does z

.^on
ao n

.^in
ix_in n

.^from
f! ra_from m

.^to
t! uw_to

.^you
ux_you

.^self
ay_i || ay_i (el d m v)

;;;the inflexional suffix set.

.^able
ah b! el || ah b! l iy

.^al
el

.^ly
l! iy

.^er
er

.^ton
t! en
```

```
.^son
s! en

.^ham
h! ae m

.^y
iy

.^th
[eh] th

.^past
d*ed

.^pl
s*pl

.^ing
ing

.^est
eh s t

.^ism
ih (dh z) m

;;the stressed prefixes

.^all
aol+

.^ad
ae+ (d b)
```

```
.^com
k! aa+ m

.^dis
d! ih+ s

.^in
ih+ n

.^ir
ihr+

.^non
n! aa+ n

.^re
r! iy+

.^sub
s! ah+ b

.^un
ah+ n
```

A.2 Low Level Rules

```
;;layer 4

.p
$pcl [$p]

.p!
[$pcl] $p [$hh] || $pcl
```

```
.t
($tcl $scl) ($t $tr) || ($-s $dx $-n $tcl $scl $t $ti) || $-hv

.t!
[$dcl] $d || [$tcl] $t [$hh] || ($tcl $scl) $t
[$tcl] $tr || ($dx $dcl $scl $tcl)

.k
$kcl $k [$hh] || $kcl [$k]

.k!
[$kcl] $k [$hh]

.b
[$bcl] $b || $bcl [$b]

.b!
[$bcl] $b || $bcl [$b]

.d*ed
$scl [$d] || $tcl [$t] || $dcl [$d] || [$ix] $dx || ($ix $eh) $dcl [$d]

.d
[$dcl] $d || $dcl [$d] || ($dx $-n)

.d!
($scl $dcl) $d || ($dx $d)

.g
[$gcl] $g || $gcl [$g]

.g!
[$gcl] $g || $gcl

.m
```

```
$m
$m $hv

.m!
($m $-m)

.em
($eh $ih $ix $ax) $m [$hv] || $m

.en
($eh $ih $ix $ax) $n [$hv] || $n

.en_and
($aen $ix $eh) $n || ($n $aen)

.n
($n $-aen) || $n $hv

.n!
($n $-n)

.ing
($ix $iy $ti) $ng ($n $hv) || ($ix $iy $ti) $ng

.ng
$ng || [$ng] $n [$hv]

.s
($s $sh $ts) || ($ts $s) $epi || ($z $ts) $s

.s!
($s $sh $-s $ts) || ($ts $s  $-s) $epi || $ts $s

.sh
$sh [$epi]
```

```
.zh
$sh

.zh!
$sh [$epi]

.sh!
$sh [$epi] || $-sh

.ch
$tcl $ch [$epi]

.ch!
($tcl $scl) $ch || $ch

.jh
[$dcl] $jh [$y]

.jh!
[$dcl] $jh

.f
$f [$epi] || $pcl $f

.ra_from
$r ($ah $ax) || ($-fr $axr)

.f!
($f $-f) [$epi] || $fr || $pcl $f

.v
($m $v)

.v!
$v
```

```
.th
[$tcl] $th || $dh

.th!
[$tcl] $th

.dh
$dh

.dh!
[$dcl] $dh || $-n || $-s

.uw_to
$-tr || $ix $hv || ($ux $uw) [$w] || ($ax $ix $uh)

.ux_you
($ux $-ux) || $jh $ux

.uw
($ux $uw $ix $axr)

.er
($r $er $axr $ax)

.ae
($ae $aen $ax $ix $-axr $eh) || ($ae $ix) $hv

.ey_a
$ey [$y] || ($axr $ax $ix $ah)

.ey
($ey $iy) || $ey ($y $hv)

.ay_i
($ay $ax $ix) || $ay $iy
```

```
.ay
$ay [$iy]

.oy
($ao $ow) $ix [$iy]

.ehr
$axr [$r] || $ehr [$r]

.eh
($eh $ih $ix $ax $-axr)

.ix_in
($ix $ih)

.ih
($ih $ix $-axr $eh $-s $axr) || $ih $hv

.ah_does
($ax $ix $ah)

.ah
($ah $ax $axr $ix) || $ah $hv || $-axr

.uh
$uh

.aor
$axr || $aor [$r]

.ao
($ao $aa $ax $ah) || $ao $ah

.aar
$axr || $aar $r
```

```
.aa
($aa $ah $ix)

.ow
($ow $uh $l $axr) || $ow $w

.iy_the
($iy $ix $ax $ah $-dh $l $axr)

.iy
$iy ($y $hv) || ($ti $iy $ix $-axr $-ti) || $y

.uw+
($uw $-ux) || $ux [$uw] || $uw $w

.er+
($axr $er)

.ae+
($ae $aen) || $ae ($hv $ah)

.ey+
$ey ($y $hv) || $ey

.ay+
$ay $iy || ($ay $ey $ae) [$hv]

.oy+
$ow ($iy $ix)

.ehr+
$ey $ehr || ($eh $ehr $axr) || $ehr ($axr $r)

.eh+
($eh $ah $ix)
```

```
.ihr+
$ih $axr

.ih+
($ih $ix $uh)

.ah+
$ah [$hv] || $ax

.uh+
$uh

.aor+
($aor $axr) [$r] || [$aor] $axr [$hv]

.aol+
$ao [$l] [$hv]

.el+
$eh $l || $ah $l

.ao+
$ao [$hv] || $aa

.aar+
$aar [$r]

.aa+
$aa [$hv]

.ow+
$ow [$w] || $uh

.aw+
$aw [$hv]
$ah
```

```
.iy+
$iy [$y] [$hv]

.r
($r $-tr $axr)

.r!
$r
($-axr  $axr)

.l
$l [$hv] || $hl [$l]

.l!
($l $-l)

.el
[$ax] $l

.w
$w

.w!
$w

.y
($y $-sh $jh $-ch)

.y!
($ch $y $-sh) || $-sh $epi

.yu+
[$y] ($uw $ux) [$w] || $jh $ux

.yu
```

```
($y $jh) $ux [$w] || $ux

.h!
($hh $hv)

.s*pl
[$z] $s [$epi] || ($ix $eh) $z $s || $ts $s || ($ts $sh) [$epi]

.z
[$z] $s [$epi] || [$z] $sh || $z

.z!
[$z] $s [$epi]

;;word boundary
.wb
$iwt || $iwt $em $iwt

;; glottal
.q
$q
```

Appendix B

Some Details on the Parser Implementation

In this appendix, we describe some of the details of how we implemented our ANGIE parser. The description is not meant to be the provide an optimal implementation, but more as an aid to help the reader understand the framework design and as a starting point for other implementations.

B.1 Parser Implementation

Recall from Chapter 2 that the ANGIE parser proceeds in a bottom-up, left-to-right manner and that a path from a terminal to the root of the tree is called a column in our terminology. At any given position in the input string, our parser takes as an input a collection of possible partial parse trees for the phone string prior to the given position and also the phone at the given position. (If we are at the beginning of a word, the partial parse is empty.) We extend the partial parse to include the phone at the given position as follows:

1. For each partial parse in the collection of partial parses:

   (a) Create a single node representing the terminal phone at the given position. The node is marked with layer information and also with a pointer to the partial parse, also known as the left context, so far.

   (b) Check to see that the advancement probability for this phone. If it is zero, then this partial parse cannot continue and can be garbage collected.

(c) Create a linked list, l, with only this node.

(d) Add the nodes in the result of Climb(l) to the set of possible parse extensions.

2. Consider the set of possible parse extensions as the new collection of partial parses for processing the next input position.

Our collection of possible parses is maintained as a linked list sorted in order of decreasing score. We impose a limit on the maximum number of entries in this list, presently set at fifteen. This limit was referred to in the beam pruning discussion in Chapter 2 and was optimized on development data.

The Climb(l) function is implemented as follows:

1. If l is empty, return an empty list.

2. If the nodes in l are already at the top-most layer, that is, the root, then stop and return l.

3. Initialize a new linked list, l_{new}, to be empty.

4. For each node, n, in the linked list l:

   (a) Consider all possible climbs up the column by checking to see if a given symbol is permitted by the rules and has a non-zero bottom-up trigram probability. Note that the regular nature of our rules, namely that the left-hand side and right-hand side are on adjacent layers, makes this a straight-forward process. For each permitted climb:

      i. Create a new node, n_{new}. This new node should inherit the left context from n and should also contain a pointer back to n so the parse tree information is available from a root node.

      ii. Add n_{new} to l_{new}.

   (b) If there are no possible climbs, then n can be garbage collected.

5. Recurse and return the results of Climb(l_{new})

B.2 Garbage Collection

As we can see from the previous section, there is substantial sharing that makes memory reclamation difficult. Consider the case of climbing a column from a given phone. The phone node is shared by all possible upward extensions above it. The same holds true for any other node created in the process. Thus, say the possible climbs from a given phone node are known to all reach dead ends until the Climb function reaches the morphology layer. It is not safe to garbage collect the particular phone node until the Climb function is called on the linked list at the morphology layer and discovers that no climbs are possible. At that point, the Climb function will garbage collect all the nodes at the morphology layer. The garbage collection process must then know that the nodes in the syllabification layer are also safe to garbage collect, since there are no possible climbs at the morphology layer. As mentioned in Chapter 2, this can be determined by maintaining a reference count. In this case, each time the Climb function creates n_{new}, which points to n, it should increment the reference counter for n. When a node is garbage collected, all the nodes pointed to should have their reference counters decremented, and should any reach zero, a recursive invocation of the garbage collection routine should occur. This same strategy can be followed for garbage collecting partial parses, where the sharing arises from multiple possible extensions to a given partial parse.

B.3 Probability Model Implementation

Recall from Chapter 2 that ANGIE has two types of probabilities, advancement and bottom-up trigram probabilities. The advancement probability is conditioned upon the left context, which consists of six nodes from the leaf to the root of the parse tree. We store these probabilities in a tree-like structure, where the root corresponds to layer 0 of an ANGIE parse tree (the root), and has entries for all possible symbols that can occur on layer 1. Retrieving the record for a specific symbol, s, provides a set of entries which correspond to possible symbols at the next layer under s. At the bottom is a structure with all possible phones and the advancement probabilities for them. A separate table holds the start of word probabilities, which are conditioned only upon the left phone context. Storing the bottom-up trigram probabilities is straight-forward in our implementation. Because the number of possible trigrams is small, we store these probabilities in a three dimensional

array. Probabilities are trained by tabulating frequencies of occurrence in a large corpus of parsed training sentences, and normalizing all counts leaving each context condition to sum to 1.0.

Appendix C

Phonetic Recognizer's Best-First Search Details

A general outline of the best first search used in our phonetic recognition system is given below:

1. Insert an empty path theory onto the stack
2. Remove the highest scoring path theory from the stack. Also, prune on maximum stack size as needed.
   (a) Check maximum paths per time boundary to see if this path theory gets pruned or not. If so, go to step 2.
   (b) If the linguistic ANGIE theory associated with this path theory has not been fully computed, finish computing it. If ANGIE returns a zero probability, prune the path and go to step 2.
   (c) Consider all possible phones and segments to extend this path theory. For each one that has an acceptable acoustic score, see if a corresponding linguistic ANGIE theory has already been explored.
      - If so, take the score from that. Check to see that no other path theory with the same linguistic theory covers the same time boundary with a better score. Insert the newly extended path into the stack.
      - If not, partially build the linguistic theory. If ANGIE returns a zero probability, prune the extended path, otherwise insert the extended path into the

stack.

3. Go to step 2

We compute the total ANGIE linguistic score in two steps. The reason for this is one of computational efficiency. Recall from Chapter 2 that the ANGIE probability model consists of two types of probabilities. It turns out that the advancement probabilities are easy to compute and are where most of the probability discrimination comes from. On the other hand, the trigram bottom-up probabilities are less important although somehat more expensive to compute. By only computing the advancement probability, a partial ANGIE score, we can eliminate some poor scoring theories, which have an unacceptable advancement probability, early on because the low score will cause those theories to drop down in priority on the stack and be pruned.

We also see that there is a constant distinction between the linguistic ANGIE theory and the path theory. A linguistic theory only knows about the sequence of phones and higher level partial parses of the phones. It knows nothing about the alignment of the phones to the acoustic data along the time line. A full path theory needs to know the actual path through the segment graph as well and hence knows about time. The upshot is that there can be many different path theories which share the same linguistic theory. Our implementation takes advantage of this by sharing the same linguistic theory amongst different path theories.

As in the case of the ANGIE parser described in Chapter 2, memory usage is a serious issue to contend with in our best-first search. We adopt the same reference counting memory reclamation technique as the ANGIE parser in our best-first search. We will also perform the same garbage collection in our stack decoder, described in Chapter 4 as well.

Bibliography

[1] J. Alvarez-Cercadillo, J. Ortega-García, and L. Hernández-Gómez, "Context modeling using RNN for keyword detection," in *Proc. ICASSP '93*, Minneapolis, MN, pp. I–569 – I–572, Apr. 1993.

[2] M. Bacchiani, M. Ostendorf, Y. Sagisaka, and K. Paliwal, "Design of a speech recognition system based on acoustically derived segmental units," in *Proc. ICASSP '96*, Atlanta, GA, pp. 443–446, May 1996.

[3] L. R. Bahl, S. V. de Gennaro, P. S. Gopalakrishnan, and R. L. Mercer, "A fast approximate acoustic match for large vocabulary speech recognition," *IEEE Trans. Speech and Audio Processing*, vol. 1, no. 1, pp. 59–97, 1993.

[4] L. R. Bahl, P. V. de Souza, P. S. Gopalakrishnan, D. Nahamoo, and M. A. Picheny, "Decision trees for phonological rules in continuous speech," in *Proc. ICASSP '91*, Toronto, Canada, pp. 185–188, May 1991.

[5] L. R. Bahl, F. Jelinek, and R. L. Mercer, "A maximum likelihood approach to continuous speech recognition," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. PAMI 5, no. 2, pp. 179–190, 1983.

[6] E. I. Chang, *Improving Wordspotting Performance with Limited Training Data.* Ph.D. thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, May 1995.

[7] N. Chomsky and M. Halle, *The Sound Pattern of English.* New York, NY: Harper & Row, 1968. republished in paperback, Cambridge, MA: MIT Press, 1991.

[8] G. Chung, *Hierarchical Duration Modelling for a Speech Recognition System.* Master's thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, May 1997.

[9] G. Chung and S. Seneff, "Hierarchical duration modelling for speech recognition using the ANGIE framework," in *Proc. Eurospeech '97*, Rhodes, Greece, pp. 1475–1478, Sept. 1997.

[10] K. W. Church, *Phrase-Structure Parsing: A Method for Taking Advantage of Allophonic Constraints.* Ph.D. thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, Jan. 1983.

[11] M. H. Cohen, *Phonological Structures for Speech Recognition.* Ph.D. thesis, Department of Electrical Engineering and Computer Science, University of California, Berkeley, CA, 1989.

[12] R. A. Cole, J. Mariani, H. Uszkoreit, A. Zaenen, and V. Zue, eds., *Survey of the State of the Art in Human Language Technology.* Cambridge, UK: Cambridge University Press, 1997.

[13] T. H. Cormen, C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms.* Cambridge, MA: MIT Press, 1990.

[14] D. A. Dahl, M. Bates, M. Brown, W. Fisher, K. Hunicke-Smith, D. Pallett, C. Pao, A. Rudnicky, and E. Shriberg, "Expanding the scope of the ATIS task: The ATIS-3 corpus," in *Proc. ARPA Human Language Technology Workshop '92*, Plainsboro, NJ, pp. 45–50, Mar. 1992.

[15] N. Daly, *Acoustic-Phonetic and Linguistic Analyses of Spontaneous Speech: Implications for Speech Understanding.* Ph.D. thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, May 1994.

[16] R. De Mori and M. Galler, "The use of syllable phonotactics for word hypothesization," in *Proc. ICASSP '96*, Atlanta, GA, pp. 877–880, May 1996.

[17] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society*, vol. 39, no. 1, pp. 1–38, 1977.

[18] J. Earley, "An efficient context-free parsing algorithm," *CACM*, vol. 13, no. 2, 1970.

[19] J. S. Garofolo, L. F. Lamel, W. M. Fisher, J. G. Fiscus, D. S. Pallett, and N. L. Dahlgren, "DARPA TIMIT acoustic-phonetic continuous speech corpus CD-ROM," 1993. National Institute of Standards and Technology, NISTIR 4930.

[20] J. L. Gauvain, L. F. Lamel, G. Adda, and M. Adda-Decker, "Speaker-independent continuous speech dictation," *Speech Communication*, vol. 15, pp. 21–38, Oct. 1994.

[21] D. Goddeau, "Using probabilistic shift-reduce parsing in speech recognition," in *Proc. ICSLP '92*, Banff, Canada, pp. 321–324, Oct. 1992.

[22] D. Goddeau, E. Brill, J. Glass, C. Pao, M. Phillips, J. Polifroni, S. Seneff, and V. Zue, "Galaxy: A human-language interface to on-line travel information," in *Proc. ICSLP '94*, Yokohama, Japan, pp. 707–710, Sept. 1994. URL http://www.sls.lcs.mit.edu/ps/SLSps/icslp94/galaxy.ps.

[23] N. Gross and P. Judge, "Let's talk," *BusinessWeek*, pp. 61–80, Feb 23 1998.

[24] S. Hayamizu, K. Itou, and K. Tanaka, "Detection of unknown words in large vocabulary speech recognition," in *Proc. Eurospeech '93*, Berlin, Germany, pp. 2113–2116, Sept. 1993.

[25] I. L. Hetherington, *The Problem of New, Out-of-Vocabulary Words in Spoken Language Systems.* Ph.D. thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, Oct. 1994.

[26] I. L. Hetherington and V. Zue, "New words: Implications for continuous speech recognition," in *Proc. Third European Conference on Speech Communication and Technology*, Berlin, Germany, Sept. 1993.

[27] L. Hirschman, M. Bates, D. Dahl, W. Fisher, J. Garofolo, K. Hunicke-Smith, D. Pallett, C. Pao, P. Price, and A. Rudnicky, "Multi-site data collection for a spoken language corpus," in *Proc. DARPA Speech and Natural Language Workshop '92*, Harriman, New York, pp. 7–14, 1992. distributed by San Mateo, CA: Morgan Kaufmann Publishers.

[28] H. W. Hon and K. Lee, "CMU robust vocabulary-independent speech recognition system," in *Proc. ICASSP '91*, Toronto, Canada, pp. 889–892, May 1991.

[29] Z. Hu, J. Schalkwyk, E. Barnard, and R. A. Cole, "Speech recognition using syllable-like units," in *Proc. ICSLP '96*, Philadelphia, PA, vol. 2, pp. 1117–1120, Oct. 1996.

[30] F. Jelinek, "Continuous speech recognition by statistical methods," *Proc. IEEE*, vol. 64, 1976.

[31] R. J. Jones, *Syllable-based word recognition.* Ph.D. thesis, Department of Electrical and Electronic Engineering, Univeristy of Wales Swansea, Wales, U.K., 1996.

[32] R. J. Jones, S. Downey, and J. S. Mason, "Continuous speech recognition using syllables," in *Proc. Eurospeech '97*, Rhodes, Greece, pp. 1171–1174, Sept. 1997.

[33] D. Kahn, *Syllable-based Generalizations in English Phonology.* Ph.D. thesis, Department of Linguistics and Philosophy, Massachusetts Institute of Technology, Cambridge, MA, 1976.

[34] F. Katamba, *An Introduction to Phonology.* London, England: Longman Group, 1989.

[35] P. Kiparsky, "From cyclic phonology to lexical phonology," in *The Structure of Phonological Representations (Part I)*, pp. 131–175, Dordrecht, Holland: Foris, 1982.

[36] P. Kiparsky, "Lexical morphology and phonology," in *Linguistics in the Morning Calm: Selected papers from SICOL-1981* (The Linguistic Society of Korea, ed.), pp. 3–91, Seoul, South Korea: Hanshin Pub. Co., 1982.

[37] L. Lamel, R. Kassel, and S. Seneff, "Speech database development: design and analysis of the acoustic-phonetic corpus," in *Proc. DARPA Speech Recognition Workshop*,, pp. 100–109, Report No. SAIC-86/1546, Feb. 1986.

[38] L. F. Lamel, L. R. Rabiner, A. E. Rosenberg, and J. G. Wilpon, "An improved endpoint detector for isolated word recognition," *IEEE Trans. Acoustics, Speech and Signal Processing*, vol. ASSP-29, no. 4, pp. 777–785, Aug. 1981.

[39] K. Lee, *Automatic Speech Recognition: The Development of the SPHINX System.* Boston, MA: Kluwer Academic Publishers, 1989.

[40] K. F. Lee and H. W. Hon, "Speaker-independent phone recognition using hidden Markov models," *IEEE Trans. Acoustics, Speech and Signal Processing*, vol. ASSP-37, no. 11, p. 1641, 1989.

[41] K. Lee, H. Hon, and R. Reddy, "An overview of the SPHINX speech recognition system," in *Readings in Speech Recognition*, pp. 600–610, San Mateo, CA: Morgan Kaufmann Publishers, 1990.

[42] R. P. Lippmann and E. Singer, "Hybrid neural-network/HMM approaches to wordspotting," in *Proc. ICASSP '93*, Minneapolis, MN, pp. I–565 – I–568, Apr. 1993.

[43] A. Ljolje, "High accuracy phone recognition using context clustering and quasi-triphonic models," *Computer Speech and Language*, vol. 8, pp. 129–151, 1994.

[44] B. Lowerre and R. Reddy, "The Harpy speech understanding system," in *Readings in Speech Recognition*, pp. 576–586, San Mateo, CA: Morgan Kaufmann Publishers, 1990.

[45] A. S. Manos, *A Study on Out-of-Vocabulary Word Modelling for a Segment-Based Keyword Spotting System.* Master's thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, Apr. 1996.

[46] R. E. Meliani and D. O'Shaughnessy, "Accurate keyword spotting using strictly lexical fillers," in *Proc. ICASSP '97*, Munich, Germany, pp. 907–910, Apr. 1997.

[47] H. M. Meng, S. Hunnicutt, S. Seneff, and V. Zue, "Reversible letter-to-sound / sound-to-letter generation based on parsing word morpology [sic]," *Speech Communication*, vol. 18, pp. 47–63, 1996.

[48] H. M. Meng, *The Use of Distinctive Features for Automatic Speech Recognition.* Master's thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, May 1991.

[49] H. M. Meng, *Phonological Parsing for Bi-directional Letter-to-Sound / Sound-to-Letter Generation.* Ph.D. thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, June 1995.

[50] M. Mohri, "Finite-state transducers in language and speech processing," *Computational Linguistics*, vol. 23, no. 4, 1997.

[51] M. Mohri and M. Riley, "Weighted determinization and minimization for large vocabulary speech recognition," in *Proc. Eurospeech '97*, Rhodes, Greece, pp. 131–134, Sept. 1997.

[52] R. Moore, M. Cohen, V. Abrash, D. Appelt, H. Bratt, J. Butzberger, L. Cherny, J. Dowding, H. Franco, J. M. Gawron, and D. Moran, "SRI's recent progress on the ATIS task," in *Proc. Spoken Language Systems Technology Workshop '94*, Plainsboro, NJ, pp. 72–75, Mar 1994.

[53] H. Ney, R. Steinbiss, R. Haeb-Umbach, B. Tran, and U. Essen, "An overview of the Philips research system for large-vocabulary continuous-speech recognition," *International Journal of Pattern Recognition and Artificial Intelligence*, 1994.

[54] L. Nguyen, T. Anastasakos, F. Kubala, C. LaPre, J. Makhoul, R. Schwatz, N. Yuan, G. Zavaliagkos, and Y. Zhao, "The 1994 BBN/BYBLOS speech recognition system," in *Proc. ARPA Spoken Language Systems Technology Workshop '95*, Austin, TX, pp. 77–81, Jan 1995.

[55] D. B. Paul, "Algorithms for an optimal A^* search and linearizing the search in the stack decoder," in *Proc. ICASSP '91*, Toronto, Canada, pp. 693–696, May 1991.

[56] D. B. Paul, "Efficient A^* stack decoder algorithm for continuous speech recognition with a stochastic language model," Tech. Rep. TR 930, MIT Lincoln Laboratory, July 1991.

[57] D. B. Paul, "An efficient A^* stack decoder algorithm for continuous speech recognition with a stochastic language model," in *Proc. ICASSP '92*, San Francisco, CA, pp. 25–28, Mar. 1992.

[58] P. Price, W. Fisher, J. Bernstein, and D. Pallett, "The DARPA 1000-word resource management database for continuous speech recognition," in *Proc. ICASSP '88*, New York, NY, pp. 651–654, 1988.

[59] L. Rabiner and B. Juang, *Fundamentals of Speech Recognition.* Englewood Cliffs, NJ: Prentice-Hall, 1993.

[60] M. A. Randolph, *Syllable-based Constraints on Properties of English Sounds.* Ph.D. thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, Sept. 1989.

[61] M. D. Riley, "A statistical model for generating pronunciation networks," in *Proc. ICASSP '91*, Toronto, Canada, pp. 737–740, May 1991.

[62] J. R. Rohlicek, W. Russell, S. Roukos, and H. Gish, "Continuous hidden Markov modeling for speaker-independent word spotting," in *Proc. ICASSP '89*, Glasgow, Scotland, pp. 627–630, May 1989.

[63] J. Rohlicek, P. Jeanrenaud, K. Ng, H. Gish, B. Musicus, and M. Shiu, "Phonetic training and language modeling for word spotting," in *Proc. ICASSP '93*, Minneapolis, MN, pp. II–459–II–462, Apr. 1993.

[64] R. C. Rose, "Definition of subword acoustic units for wordspotting," in *Proc. Eurospeech '93*, Berlin, Germany, pp. 1049–1052, Sept. 1993.

[65] R. C. Rose and D. B. Paul, "A hidden Markov model based keyword recongition system," in *Proc. ICASSP '90*, Alburquerque, NM, pp. 129–132, Apr. 1990.

[66] E. Sanchis, F. Casacuberta, I. Galiano, and E. Segarra, "Learning structural models of subword units through grammatical inference techniques," in *Proc. ICASSP '91*, Toronto, Canada, May 1991.

[67] R. Schwartz, Y. Chow, O. Kimball, S. Roucos, M. Krasner, and J. Makhou, "Context-dependent modeling for acoustic-phonetic recognition of continuous speech," in *Proc. ICASSP '85*, Tampa, FL, pp. 1205–1208, Mar. 1985.

[68] E. Selkirk, "The syllable," in *The Structure of Phonological Representations (Part II)*, pp. 337–385, Dordrecht, Holland: Foris, 1982.

[69] S. Seneff, "TINA: A natural language system for spoken language applications," *Computational Linguistics*, vol. 18, no. 1, pp. 61–86, Mar. 1992.

[70] S. Seneff, R. Lau, and H. Meng, "ANGIE: A new framework for speech analysis based on morpho-phonological modelling," in *Proc. ICSLP '96*, Philadelphia, PA, vol. 1, pp. 110–113, Oct. 1996. URL http://www.sls.lcs.mit.edu/raylau/icslp96_angie.pdf.

[71] S. Seneff, M. McCandless, and V. Zue, "Integrating natural language into the word graph search for simultaneous speech recognition and understanding," in *Proc. Eurospeech '95*, Madrid, Spain, pp. 1781–1784, Sept. 1995.

[72] E. C. Shukat-Talamazzini, H. Niemann, W. Eckert, T. Kuhn, and S. Rieck, "Acoustic modeling of sub-word units in the ISADORA speech recognizer," in *Proc. ICASSP '92*, San Francisco, CA, pp. 577–580, Mar. 1992.

[73] "The road rally word-spotting corpora (RDRALLY1)," Sept. 1991. NIST Speech Disc 6-1.1.

[74] A. Viterbi, "Error bounds for convolutional codes and an asymptotically optimal decoding algorithm," *IEEE Trans. Information Theory*, vol. IT-13, pp. 260–269, Apr. 1967.

[75] M. Weintraub, "Keyword-spotting using SRI's DECIPHER large-vocabulary speech recognition system," in *Proc. ICASSP '93*, Minneapolis, MN, pp. II–463 – II–466, Apr. 1993.

[76] P. Winston, *Artificial Intelligence*. Reading, MA: Addison-Wesley, third ed., 1992.

[77] V. Zue, J. Glass, D. Goodine, H. Leung, M. Phillips, J. Polifroni, and S. Seneff, "Integration of speech recognition and natural language processing in the MIT VOYAGER system," in *Proc. ICASSP '91*, Toronto, Canada, pp. 713–716, May 1991.

[78] V. Zue, J. Glass, D. Goodine, M. Phillips, and S. Seneff, "The SUMMIT speech recognition system: Phonological modelling and lexical access," in *Proc. ICASSP '90*, Alburquerque, NM, pp. 49–52, Apr. 1990.

[79] V. Zue, J. Glass, M. Phillips, and S. Seneff, "The MIT SUMMIT speech recognition system: A progress report," in *Proc. DARPA Speech and Natural Language Workshop Feb '89*, Philadelphia, PA, pp. 179–189, Feb. 1989.

[80] V. Zue, S. Seneff, J. Polifroni, M. Phillips, C. Pao, D. Goddeau, J. Glass, and E. Brill, "The MIT ATIS system: December 1993 progress report," in *Proc. ARPA Spoken Language Technology Workshop '94*, Plainsboro, NJ, pp. 66–71, Mar. 1994.