Framework for Joint Recognition of Pronounced and Spelled Proper Names

by

Atiwong Suchato

B.S. Electrical Engineering, (1998)
Chulalongkorn University

Submitted to the Department of Electrical Engineering and Computer Science
in Partial Fulfillment of the Requirements for the Degree of

Master of Science in Electrical Engineering and Computer Science

at the

Massachusetts Institute of Technology

September 2000

Author ……...
Department of Electrical Engineering and Computer Science
August 29, 2000

Certified by ……..
Dr. Stephanie Seneff
Principal Research Scientist
Thesis Supervisor

Accepted by ……..
Arthur C. Smith
Chairman, Departmental Committee on Graduate Theses

Framework for Joint Recognition of Pronounced and Spelled Proper Names

by

Atiwong Suchato

submitted to the
Department of Electrical Engineering and Computer Science
August 29, 2000
In Partial Fulfillment of the Requirements for the Degree of
Master of Science in Electrical Engineering and Computer Science

Abstract

In this thesis, the framework for a proper name recognition system is studied. We attempt to combine the information in both the spelling and the pronunciation of a proper name and find ways to let both sources of information improve the recognition accuracies of that name from the accuracies obtained by performing the spelling recognition and pronunciation recognition tasks separately.

A set of "morph" sub-units is introduced. These syllable-sized units capture both orthographic and phonemic information of the names they represent. Our morphs are arbitrarily categorized into six groups: prefix, onset, rhyme, uroot, dsuf and isuf. Certain combinations of these morph categories, defined by a specific word decomposition rule, are used to represent proper names.

For each proper name, the name-spelling utterance is recognized by a letter recognizer. Then, the proposed letter sequences are parsed by the TINA parser into possible sequences of morphs. These sequences of morphs are used to provide additional constraints for the morph recognizer, which is responsible for proposing morph sequences from the corresponding pronounced utterance. Since both orthographic and phonemic information is encoded in morphs, the letter and phone recognition accuracies can be calculated from the proposed morph sequence. Various methods of providing the morph recognizer with the spelling knowledge using two types of morph representation are explored.

The results are promising. In most of the cases, the spelling information helps improve the phone recognition accuracy of the overall system. There is significant reduction in phone recognition error rate when the system is provided with the correct letter sequences. However, there is no promising sign that the pronunciation knowledge could help the letter recognition task in most cases.

Thesis Supervisor: Dr. Stephanie Seneff

Title: Principal Research Scientist

Acknowledgements

It has been a long and challenging task to complete this thesis. Along the way, supports from many people help me get through all the obstacles. Stephanie Seneff is the first person to be praised for her tireless effort in this work. She not only guided me into this field of research but also was the one who understood me very well. I would like to thank T.J. Hazen, Lee Hetherington, and Jim Glass for their patient explanations through out this research. Many thanks are due to Victor Zue for being a group leader who is the center of such a friendly atmosphere in the SLS group. I thank all of the students and staff at the SLS group for sharing their visions and knowledge.

I would like to thank dearly to my family for all the love and support from far away. I would like them to know that their messages in my mailbox re-energized me times after times. During my deepest loneliness, they were my mom's messages that help me get through.

Finally, I would like to thank the Ananda Mahidol Foundation for the financial support.

This research was supported in part by DARPA under grant number N66001-99-1-8904, monitored through Naval Control, Command and Ocean Surveillance Center.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Problem Definition

The universe of proper names is a large set of words, many of which have very rare usage. Computer conversational system will benefit greatly if machines can understand proper names. For example, an automatic flight reservation system would be more practical if the machine can understand all proper names that are absent from its name lexicon, including the names of place of origin and destination, as well as the person it is conversing with, correctly. Two important capabilities that would help a computer to achieve a goal of universal name recognition are (1) given a sequence of letters in a proper name, the machine should be able to know how that letter sequence is pronounced, and (2) given the pronunciation of a proper name, it should be able to propose plausible spellings for that name.

The problem of automatic speech recognition of proper nouns is different from the one of common words in several ways. Proper nouns have a different structure from common words. Specifically, common words can be decomposed into substructure such as prefix, stressed-root, unstressed-root, suffix, etc. In proper nouns, it is possible to do such subword decomposition, but the rules and the units used should be different. In other words, the decomposition rules that work well with most of the common words are not appropriate for proper nouns. Furthermore, recognition of foreign proper names, which can be spelled using the English alphabet, can be hard. For example, a Japanese name like "Takayama" does not have an obvious stressed root. Some letter sequences rarely occur, if at all, in English common words but appear quite prevalently in popular names. For example, "-ohn", in the name "John", is rarely, if ever, used in common words. This problem is even more pronounced in foreign names, such the "vl" in "Vladimir".

When a human hears the pronunciation of proper names that he/she has never encountered before, he/she might not be able to recognize those names perfectly and is likely to try to get more information about them. One of the additional pieces of information one often seeks to have is the spelling of those names. On the other hand, when one comes across a sequence of letters that one is unfamiliar with, the pronunciation of that letter sequence often yields understanding. So, it should be reasonable to apply the situations above to computer recognition of proper nouns. Specifically, the machine should be able to utilize mutually supportive information between the pronunciation and spelling of a proper noun in the general proper noun recognition task.

1.2 Subword modeling

It is very inefficient, if possible, to let a machine know all of the proper names by making it memorize each proper name as a whole word, due to the large number of possible names. In order to handle this problem, the idea of teaching the machine to understand unknown words by learning from some relevant information obtained from known words needs to be deployed. One way of doing this is to break known words into sequences of smaller units and then reassemble these units to represent the desired unknown words. Sequences of phonemes are one of the possible alternatives for representing words. Given that we have all possible phonemes in our inventory of subword units, we can represent any words by sequences of our phonemes. However, without any intermediate layers between word and phonemes, the order of phonemes in a word can be quite arbitrary. Learning the sequences from a given set of words does not provide strong enough constraints for a machine to propose the phoneme sequences for unknown words. Replacing phoneme sequences with syllable sequences yields stronger constraints with a larger inventory size. Yet, in this representation, there is still the lack of a positional property. More specifically, given a specific syllable, we do not know where it would most possibly occur in the sequence of syllables for a word. To gain still more constraints, one can explore the idea of marked syllables, in which each syllable is attached with positional marker. This method makes the order of subword units for representing words less arbitrary. For example, if a syllable is marked “prefix”, this syllable can be used to represent a word only when it can be put at the very front of the sequence. One can use statistical methods to distinguish the properties of a syllable in this prefix position from those of other syllables.

Lau [2] and Parmar [1] used a set of subword units called "morphs" to represent *common* words. In this thesis we will adopt the idea of morphs as well. We expect this morph notation will work with the proper name decomposition task as well as it did for the common word case in Lau's and Parmar's research.

Parmar used the probabilistic framework called "ANGIE" [5] to decompose common words into their corresponding morph sequences. For each common word, a parse tree, with word as its root, letters as its leaves and morphs as an intermediate layer helping constrain the parse tree, was constructed. In this thesis, we will explore the possibility of utilizing another probabilistic framework called TINA [4], instead of ANGIE, to provide the decompositions of proper names. Although TINA was originally designed to parse a sentence into its corresponding word sequences, we believe that TINA can be adapted to accomplish the proper name decomposition task as well, by looking at the proper names as if it were a sentence with the individual letters substituting for terminal nodes.

The details of TINA and our morph representation will be discussed in Chapter 2.

1.3 Research Goals

The primary goal of this thesis is to propose a framework for recognition of proper names, especially people's names. The class of names we are interested in is not restricted to only English or American names but also includes any names that can be spelled using the English alphabet. The motivation behind this thesis is that, when someone would like to tell his name to other people, if the name is not a very common proper name, such as "John" or "Bill" for example, it usually helps others to recognize the name more easily if the speaker spells his name for them. This shows that people use the information residing in the spelling to help understand more about the name they hear. Conversely, even though a spoken spelling of a word might not sound clear enough to a listener, they usually figure out what that spelling is once the word is pronounced. In this thesis, we study ways to use such mutual information in the spelling and pronunciation to improve the recognition performance of both the recognition of the spelling alone and the recognition of the pronunciation alone.

We will use "morphs", defined as a particular spelling convention representing the syllables of words, as the medium to convey information between the pronounced utterances and the name-spelling utterances. The spelling knowledge from the name-spelling utterances is encoded in the morph representation. Also the information about pronunciation is encoded in the same representation. Then, the recognition results for both phonemics and spelling can be derived directly from morphs. The details of the morph representation will be discussed in later chapters.

One of the tasks embedded in our primary goal is to study how different sets of morphs and word decomposition rules affect the overall recognition results. To be more specific, we would like to see the difference between the performance of the system using different sets of morphs and ways to decompose words. We would also like to find out whether the sets of morphs derived from a limited corpus of data can be reasonably used to represent a different body of proper nouns and yield plausible performance for the overall system.

In the rest of this thesis, we will use the word "name-spelling utterance" to refer to the utterance that contains the spelling of the name, such as " d a v i d ". And we will refer to the utterance that contains the spoken name as "pronounced utterance".

Conducting experiments on the framework of joint recognition between spelled and pronounced names, we can extend our study to cover the studying of the letter-to-sound and the sound-to-letter problem. Since we have name-spelling utterances and pronounced utterances as inputs to our integrated system, and can process the recognition results in order to get resulting phone and letter sequences, this integrated system can be viewed as another alternative method for letter-to-sound and sound-to-letter generation.

1.4 Research Plan

The primary goal of this research is to develop a framework for recognition of proper names by using mutual information in both pronounced and spelled utterances. In order to evaluate the recognition performance of the developed framework, we need to have the baseline performances. These numbers are the performances of the letter recognizer and morph recognizer, which are also the recognizers to be used in the block called integrated system in Figure 1-3.

According to Figure 1-3, instead of performing the recognition on each piece of information separately, both name-spelling utterances and pronounced utterances are fed into the integrated system, and the system proposes the final recognition result. The letter recognition accuracy obtained in this case is compared with the letter recognition accuracy obtained from running the letter recognizer in Figure 1-1 alone, while the phone recognition accuracy is compared with the phone recognition accuracy obtained from the recognizer in Figure 1-2.

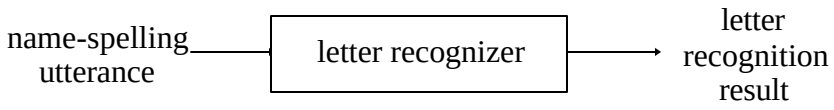


Figure 1-1: Block diagram of the letter recognition task

Figure 1-2: Block diagram of the morph recognition task

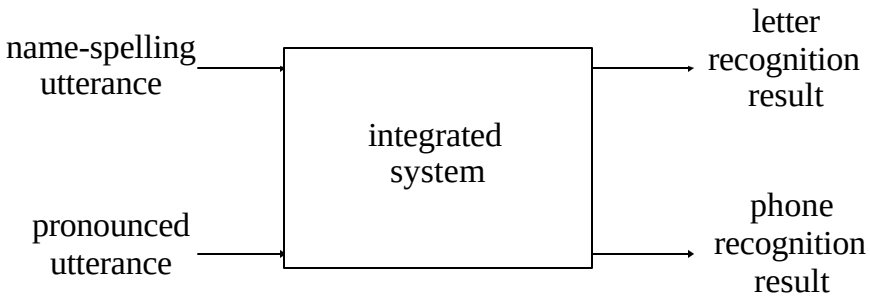


Figure 1-3: Block diagram of the integrated system performing both the spelling and pronunciation recognition

In the block called integrated system in Figure 1-3, name-spelling utterances are fed to the same letter recognizer in Figure 1-1, while pronounced utterances are fed to the morph recognizer in Figure 1-2. Then the proposed letter sequences from the letter recognizer are parsed by TINA, a natural language system that is capable of proposing candidate morph representations of the input letter sequences. Then these sets of morph representations are used as additional constraints for the morph recognizer for recognizing the corresponding pronounced utterance. To gain more knowledge about how suitable different sets of morphs and word decomposition rules are for representing the proper nouns, the system in Figure 1-3 is elaborated into the block diagram in Figure 1-4 below. Each block is the same as the block called integrated system in Figure 1-3 but with different inventories of morphs and different rules for decomposing proper nouns into sequences of morphs. In this thesis, two types of morph representations are used. These two sets of morphs were developed along the way as we observed the training data and built the two recognizers.

- Experiments done on "morph representation type I" are discussed in Chapter 7.
- Experiments done on "morph representation type II" are discussed in Chapter 8.

Furthermore, in addition to the information on spelling obtained from the proposed letter sequences from the letter recognizer, experiments are also conducted by giving the morph recognizer perfect knowledge of each name's spelling. This perfect knowledge of the spelling is obtained by providing TINA with the correct letter sequences for each name instead of the noisy letter sequences from the letter recognizer. The performance of the system in this case should provide insight into how well the system performs on the letter-to-sound task augmented with a spoken pronunciation.

In summary, the series of experiments conducted are grouped according to the morph inventories used. In each group, the varied parameters are the language model for the morph recognizer. The baseline morph recognizer has language models which are the bigram and trigram constructed from the lexicon of training data. Then, by composing this baseline language model with various sources of spelling information, we can generate different language models. The sources of spelling knowledge are the following:

- Spelling knowledge obtained from the ten letter sequences most preferred by the letter recognizers.
- Spelling knowledge obtained from the top choice of the proposed letter sequences from the letter recognizers.
- Spelling knowledge obtained from the correct letter sequence.

After we have the results from various experiments, we can compare the letter and phone accuracy with the ones from the baseline recognizer. Then the analysis and comparisons are done to identify the most suitable scenarios for the proper name recognition task.

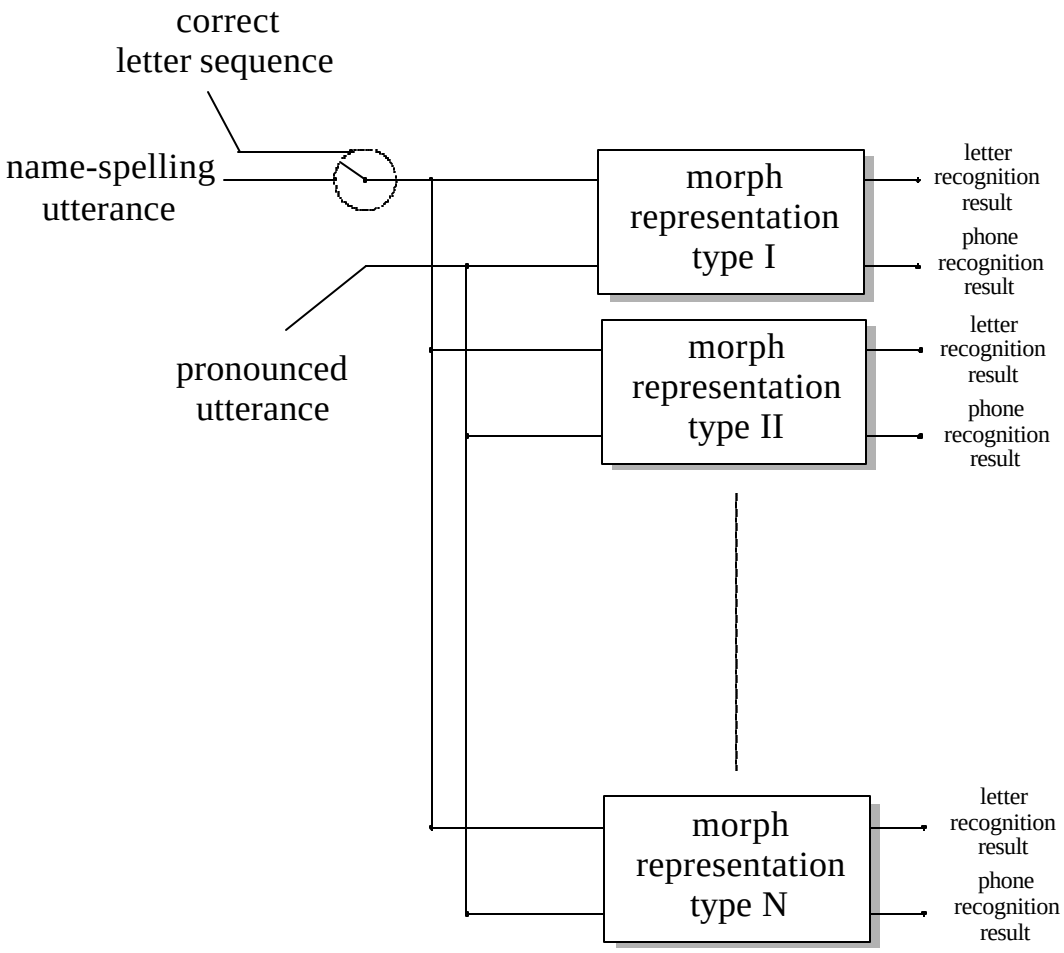


Figure 1-4: Block diagram of the experiments conducted in this thesis

1.5 Chapter Summary

Enabling computers to understand proper names can be very beneficial to conversational systems. Proper names are distinctly different from common words in terms of their pronunciations and spellings. In this thesis we are trying to study how we could combine the information in both the spellings and pronunciations of proper names and use the combined information to fully recognize those proper names, assuming they are absent from the lexicon. One of the key ideas in this system is the proper name decomposition, in which a proper name was parsed into its sub-word unit representation. "Morphs", syllable-sized sub-word units, were used to represent proper names. This work is to some degree an extension of a previous study of the procedure for representing words using these morphs as basic units. It was shown, in the common nouns case, that a morph representation can provided substantial coverage for common English words, along with reasonable accuracy in terms of pronunciations [1].

The primary goal of this thesis was to propose a framework for recognition of proper names. This framework incorporates knowledge about the spelling of a proper name with knowledge of its pronunciation in order to improve the recognition performance. Our experiments varied along several dimensions, including the type of morphs and word decomposition rules used.

Baseline recognizers, both for letters and morphs, were built. Their recognition results were kept in order to be compared with the corresponding recognition accuracies of the integrated system. An overview of the experiments conducted is described in Section 1.4. Various language models for the pronunciation recognizer in the integrated system were used in the experiments, and their resulting accuracies were compared with the corresponding accuracies in the baseline case. Each language model to be used was composed of two parts. The first part was common for all names, while the other part was derived specifically for each name from its corresponding spelling information obtained in various ways, including the letter recognition results from the letter recognizer and the correct letter sequences. Furthermore, two types of morph representation and their corresponding word decomposition rules were used in the experiments.

1.6 Thesis Outline

In this chapter, we talked about the motivation of this thesis. Some of the background about the components used in building the system was discussed. Other background material, not discussed in this chapter, will be covered in later chapters.

In Chapter 2, we will look at the TINA natural language system [4], which allows us to perform the word decomposition task, along with the details concerning the morph representation and the word decomposition procedure.

The background used in building speech recognizers, another important component of the framework, will be discussed in Chapter 3. The system that is the basis of the recognizers built in this thesis is called the SUMMIT recognition system [9], [10]. The details about using this system in building two types of specific recognizers will be discussed later in Chapter 6.

Another important part of these experiments is data. In Chapter 4, we will provide the information about the data we used throughout this research. The information includes the types of data we need, their sources, how we divide them into groups according to the purpose we want to use each group of data for, and some details about their contents.

After describing the basis of the required components, we will introduce the integration of various components together. The broad picture of the overall system will be provided, and some issues about their interconnection will be discussed.

In Chapter 6, we will describe how we build the recognizers we used to obtain the baseline performances. There are two recognition systems, the letter recognizer and the morph recognizer. Two of each type of recognizer were used in this research. The baseline accuracies of these recognizers will be shown.

Chapter 7 and Chapter 8 are devoted to the actual experiment conducted in this research. The experiments in the two chapters differ by the morph representations and word decomposition

rules used. The goal of each experiment will be described along with the corresponding procedures. The results will be shown and analyzed.

Finally, in Chapter 9, we will conclude our research and propose some possible work that can be done on the results we obtained. Also we will suggest some possible future experiments that cannot be done in this thesis due to time constraints.

Chapter 2

Morph Representations and TINA

2.1 Motivation

In order to let the components involved in our framework share information, we use a syllable-sized subword unit that we call "morphs" to represent proper names. In this chapter, we will introduce these morphs and discuss the method we used to decompose a proper name into the corresponding sequence of morphs. These morphs were used as the common unit for both the phone recognizers and the letter recognizers. Different sets of morphs and their corresponding word decomposition rules were used in the system in order to study the differences that various methods provide in term of recognition performance.

One of the questions we are addressing in this thesis is whether TINA, a probabilistic natural language framework intended to parse sentences into linguistic substructure, can be adapted to the task of parsing words into morphological substructure. The earlier part of the TINA discussion will provide the overview of the system and the basic operations TINA performs on understanding sentences, namely parsing sentences into the syntactic and semantic constituents defined in its grammars. Also, TINA can take an N-best list and turn it into a cross-pollinated graph to be parsed in a single search to yield M-best parse trees. The later part of the TINA discussion concerns how we can create TINA grammars to be used in the task of parsing a proper noun into a sequence of morphs. In other words, we look at the proper nouns in this case as the whole sentences and treat letters as words, which are terminal elements of the parse tree.

2.2 Word formation of proper names

Since the universe of proper names is a large set of words, it is impractical to provide all of them to the vocabulary of a recognition system in order to understand all of the proper names. One approach, which is also adopted in the common word case, is to find methods to represent this whole universe of words with sequences of smaller units, which, hopefully, are far fewer in number. In English, one might use syllabification for dividing a word into its corresponding sequence of smaller units, which are then syllables. We feel that syllable-sized units provide a reasonable sized inventory with coverage over the majority of unseen names. However, due to the lack of a positional property of the syllable, we believe that it overgeneralizes the characteristics of its pattern. Thus, it sacrifices some potential constraints. For example, we believe that the occurrence of a syllable that appears at the front of a proper noun has a dramatically different probability distribution from the case when that syllable appears at the end, or any other location. Thus, we adopt the idea of syllable-sized units but incorporate some positional constraints into our subword unit, which is expected to provide us considerably stronger constraints for the proper noun recognition task. These units are called "morphs[1]". In order to discriminate the probability spaces among the morphs in different positions, our morphs are somewhat arbitrarily divided into five categories. The five categories and their definitions are shown in Table 2-1. In the probabilistic training of our parser, the counting of the occurrences of letter sequences takes the morph categories into account. Thus, for example, we can distinguish between the probability of the letter "o" followed by the letter "n" in the prefix "on-" from the probability of the same letter sequence in the dsuf "-on".

Morph categories	Definition
Prefix	an unstressed syllable preceding the first stressed syllable
Sroot	a stressed syllable
Uroot	an unstressed syllable immediately following a stressed syllable
Dsuf	a default unstressed syllable not falling in any of the other categories
Isuf	a terminal unstressed syllable characteristic of proper nouns

Table 2-1:Our morph categories and their definitions

[1] "Morphs" are arbitary subword units defined to represent words in our research. They are unrelated to "morphemes" in the classical linguistic theory.

In order to make these sub-word units more general, *sroot* is further divided into smaller units called *onset* expressing all initial consonants, and *rhyme*, containing the stressed vowel and all subsequent consonants.

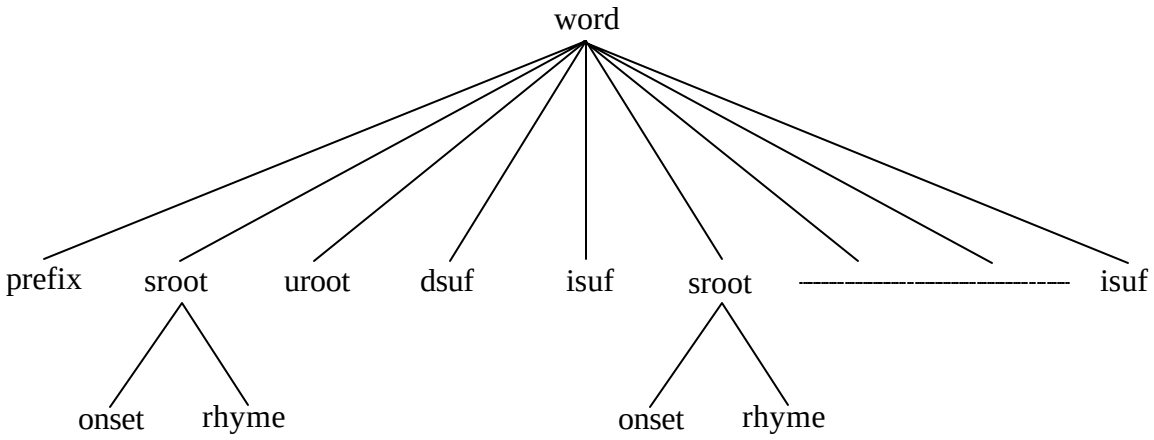


Figure 2-1: Our decomposition of a proper name's structure[2]

There are some heuristic context-free rules governing the decomposition of a proper noun. Such rules were deployed to keep the decomposition meaningful and consistent. The rules were developed according to the definition of each morph category. The entire space of proper noun decomposition according to the rules we used in this thesis is shown in the network in Figures 2-2 and 2-3.

[2] The pattern repeats for words with multiple stressed syllables.

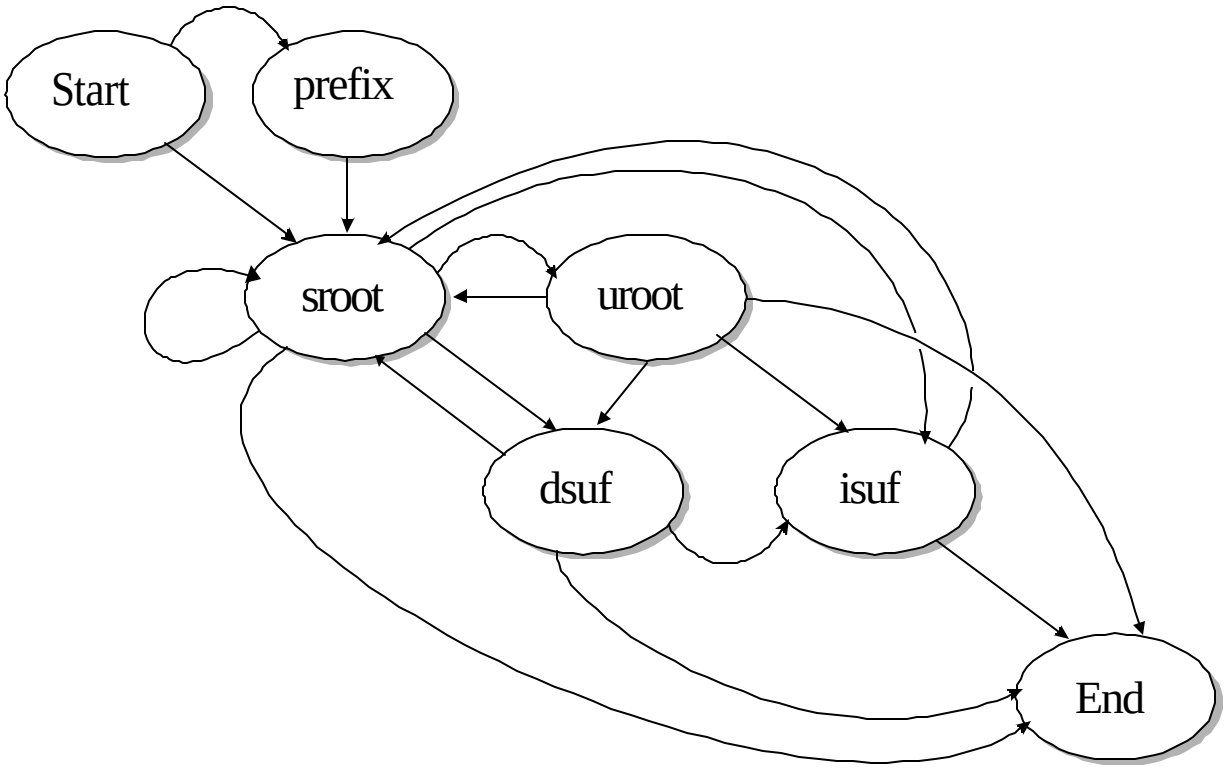


Figure 2-2: Network structure of a proper name

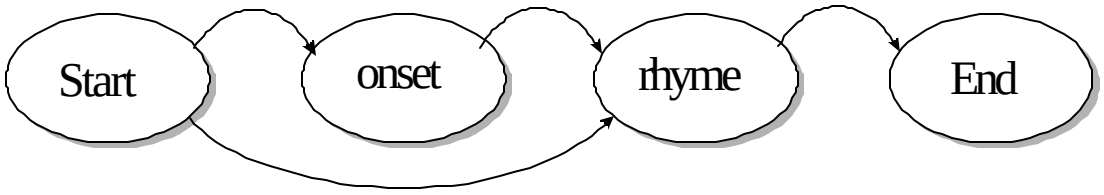


Figure 2-3: Network structure of an *sroot*

2.3 Morph notations

As mentioned above, we used "morphs" as the common units for conveying information between a spelled utterance and a pronounced utterance. Morphs are defined as syllable-sized units of a word, with both a phonemic and orthographic representation. Each morph belongs to one of the six sub-word unit categories, *prefix, onset, rhyme, uroot, dsuf* and *isuf*. Also, each morph consists of letters, upper and/or lower case, and a sub-word category marker. Sub-word category markers are the symbols, including "-", "=", "+", used to denote different sub-word categories. How these marked morphs relate to the six categories is shown in Table 2-2 below.

Category	Marker	Example morphs
prefix	"morph-"	mc-, pro-, for-
onset	"morph="	j=, J=, bh=, str=
rhyme	"=morph+"	=ince+, =at+, =Ar+
uroot	"morph"	a, al , i
dsuf	"-morph"	-y, -ie, -ger, -gia
isuf	"=morph"	=s, =son, =sen

Table 2-2: Morph categories

Word decomposition into the sequence of morphs was designed in such a way that, if we discard all the markers appearing in each morph and convert all of the upper-case letters to lower-case, we can restore the way the original word was spelled. Apart from the spelling information that is encoded in the morphs, morphs also contain pronunciation information. Each morph has one or more pronunciations depending on how we design the morph system.

In this thesis, we used two systems of morph representations in separate experiments, in order to study how we can design a suitable morph representation for proper names. The main distinction between the two is the issue of how to deal with spelling sequences that produce exactly the same morph except with different pronunciations. In the first system, each morph has a unique pronunciation. In this system we exploit upper-case letters in order to construct morphs with

different pronunciations but the same orthographies. An example of such a case is shown in Table 2-3 below.

decompositions	morphs' pronunciation
diane : d= =i+ =ane+ diaz : d= =I+ =az+	=i+ : ay =I+ : iy
jon : j= =on+ jose : J= =o+ s= =E+	j= : jh J= : hh

Table 2-3: Examples of proper name decomposition using morph representation Type I

The other system we used in this thesis does not use upper-case letters, but, instead, each morph is allowed to have multiple pronunciations. So, the number of morphs in this system is less than in the first system but, in return, we face more confusion about the pronunciation of each morph. The names "diane" and "diaz" in Table 2-4 can be decomposed using the same "=i+", while the pronunciations of "=i+" can be either "ay" or "iy". And the names "jon" and "jose" can be decomposed using the same "j=", where the pronunciations of "j=" can be either "jh" or "hh"

2.4 TINA: A Natural language system for spoken language applications

2.4.1 Overview

TINA is a natural language system developed by the Spoken Language Systems group, Laboratory for Computer Science, at MIT, for applications involving spoken language tasks. TINA produces a highly constraining hierarchical probabilistic language model to improve recognition performance. TINA has the ability to convert the input sentence into a parse tree corresponding to the sentence according to the given rules. TINA's grammar rules are written such that they describe syntactic structures at high levels of a parse tree and semantic structures at the low levels. All of the meaning-carrying content of the sentence is completely encoded in the names of the categories of the parse tree. The context free rules are automatically converted to a

shared network structure, and probability assignments are derived automatically from a set of parsed sentences. In the next section, we will discuss the detailed description of TINA's grammars and the probabilistic models.

2.4.2 TINA's grammars

TINA's context free grammars are the rules that let TINA know all the possible structures of the sentences of interest. For example, if we wish to parse a noun phrases (NP), one could provide the rules as in Figure 2-4 below to TINA.

```
Sentence =    NP
NP       =    Article [Adjective] [Adjective] Noun
```

Figure 2-4: An example of TINA's rules

Each word is the name of a category or node in the parse tree, i.e. Sentence, NP, Article, Adjective and Noun, where the brackets signify optional nodes in the structure. This grammar can be used to parse the set of phrases shown on the left of Figure 2-5, each of which corresponds to the parsing result shown on the right.

```
"a boy"                  NP = Article Noun
"the bottle"             NP = Article Noun
"a beautiful town"       NP = Article Adjective Noun
"the blue box"           NP = Article Adjective Noun
"a cute young girl"      NP = Article Adjective Adjective Noun
"the white tall building"NP = Article Adjective Adjective Noun
```

Figure 2-5: Parsing of noun phrases according to the rules given in Figure 2-4

The grammar is converted to a network structure by merging common elements on the right-hand side of all rules sharing the same left-hand side category. Each left-hand side category becomes

associated with a parent node whose children are the collections of unique categories appearing in the right-hand sides of all the rules in the common set. Each parent node establishes a two dimensional array of permissible links among its children, based on the rules. Each child can link forward to all of the children that appear adjacent to that child in any of the shared rule set. Probabilities are determined for pairs of siblings though frequency counts on rules generated by parsing the training sentences. The parsing process achieves efficiency through structure-sharing among rules, resembling in this respect a top-down chart processor.

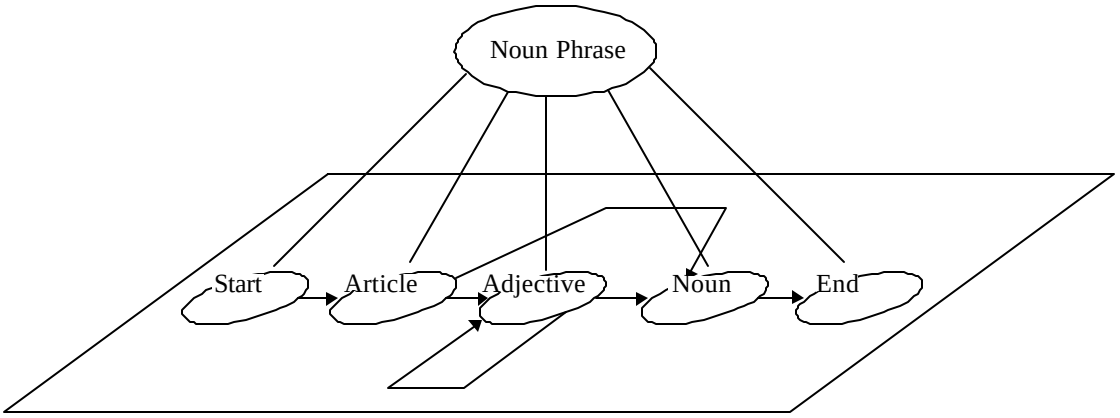


Figure 2-6: Illustration of Noun Phrase parse tree

2.4.3 Training of probabilities

The grammar is built from a set of training sentences. TINA is exposed to this set of sentences and the probabilities of every arc between siblings is computed. More specifically, a record is kept of the relative counts of each subsequent sibling, with respect to each permissible child of the parent node as they occurred in an entire set of parsed training sentence. For example, in the noun phrase case above, suppose we end up with the probabilities as trained from the "corpus" in Figure 2-5 as in Figure 2-7. We can see that [Adjective] is followed four time by [Noun] and twice by [Adjective], so the network shows a probability of 1/3 for the self loop and 2/3 for the advance to [Noun].

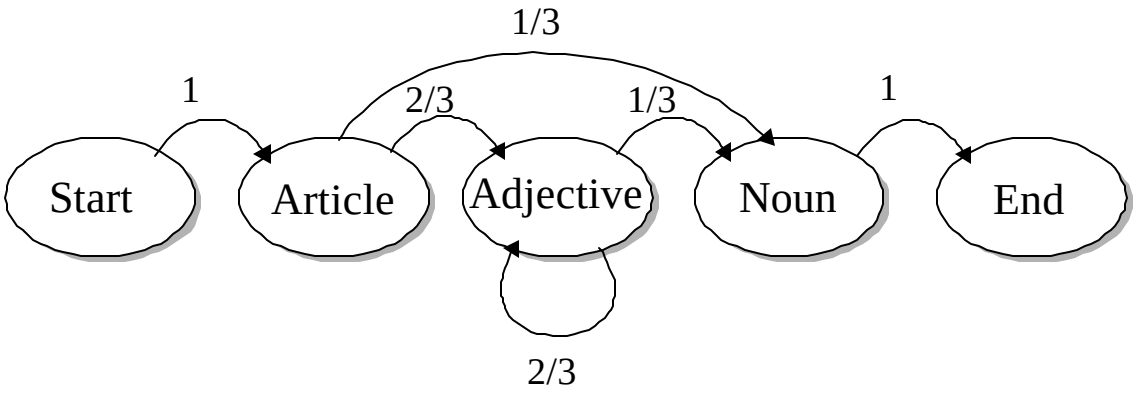


Figure 2-7: An example of a probabilistic network for Noun Phrase in the example

In Figure 2-7, we can see that there is one self-loop in the network. The network with a self-loop is generalized to include any number of elements whose nodes contain the loop to themselves in a row. So, in our example, the rule is now capable of including any number of adjectives in a row. A phrase like "a thin tall Chinese guy" can be handled by the rule "NP = Article Adjective Adjective Adjective Noun". Furthermore, there is a significant amount of sharing of individual sibling pairs among different rules. This sharing causes the so-called "cross-pollination effect". An example of this effect is shown below.

Suppose we have the following rules:

Parent = Child_1 Child_2 Child_2
Parent = Child_1 Child_2 Child_3
Parent = Child_2 Child_4
Parent = Child_1 Child_3 Child_4

We can obtain the probabilistic network as in Figure 2-8 below.

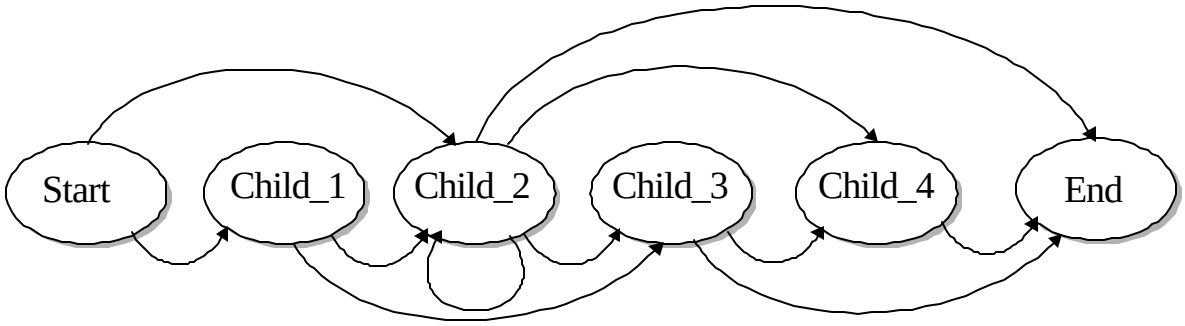


Figure 2-8: Probabilistic network derived from given rules

From the network in Figure 2-8, we can see that the [Parent] that has structure like:

Parent = Child_1 Child_2 Child_3 Child_4
Parent = Child_1 Child_3

can be parsed perfectly even though we did not specify this pattern explicitly in the rules.

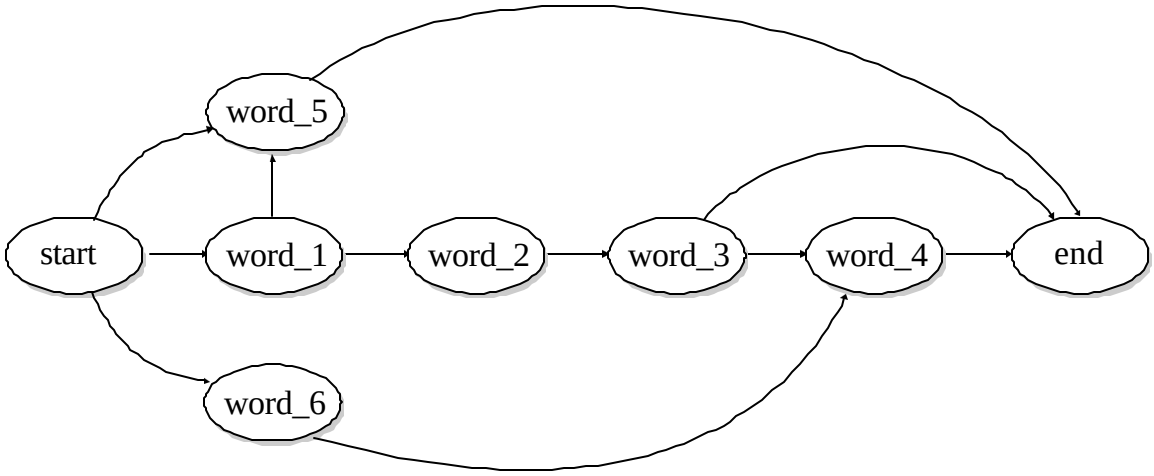
2.4.4 Parsing N-best lists

The input passed to TINA is usually the recognition result from a speech recognizer. Thus, apart from taking one sentence and parsing it into a series of parse trees, TINA is also capable of taking in an N-best list of sentences, which can come from a speech recognizer's output, as an input for building the resulting parse trees. By first turning the input N-best list into a cross-pollinated graph, in which the same words appearing in any of the N-best list items are treated as the same node in the graph, TINA searches for M best resulting parse trees according to the graph and its trained grammars. This mechanism of constructing an interconnected graph allows an expansion of the solution space, which might permit the recovery of the correct answer omitted in the original list due to the recognition error. An example of an N-best list and its corresponding interconnected graph are illustrated in Figure 2-9 below.

```
NBEST

Word_1 Word_2 Word_3 Word_4
Word_5 Word_3 Word_4
Word_1 Word_5
Word_1 Word_2 Word_3
Word_6 Word_4
```

(a)



(b)

Figure 2-9: (a) An example of an N-best list of sentences (b) its corresponding cross-pollinated graph

In the next section we will look at how we can decompose words like proper names into sub-word units and then discuss how we define the word decomposition rules for TINA to construct a parse tree of a proper name.

2.5 Using TINA at the sub-word level

We have mentioned how TINA can parse sentences into a series of words. Here, we are going to look at the sentence level as only one word, more specifically, one proper name. TINA's task is to decompose the proper name into a series of appropriate morphs according to the given rules. In order to obtain the probabilistic network as in Figures 2-2 and 2-3, the following rules are provided to TINA.

sentence = [prefix] sroot [dsuf] sroot uroot [dsuf] [isuf]
sentence = [prefix] sroot uroot [dsuf] sroot [dsuf] [isuf]
sentence = [prefix] sroot uroot [dsuf] sroot [dsuf] isuf
sentence = sroot isuf sroot [uroot] [dsuf] [isuf]
sentence = sroot isuf sroot sroot [uroot] [dsuf] [isuf]
sentence = sroot [dsuf] [isuf]
sentence = prefix sroot
sroot = [onset] rhyme

According to this set of rules, the network looks like the one in Figures 2.2 and 2.3. The probabilities of the arcs are trained by providing TINA a name training lexicon. In this lexicon, the names in the training set are listed and provided with the corresponding morph representations. By counting the frequency of each pair of sub-word categories and normalizing, the probabilities are added to the appropriate arcs. An example showing the results of word decomposition according to this set of rules is shown in Figure 2-10.

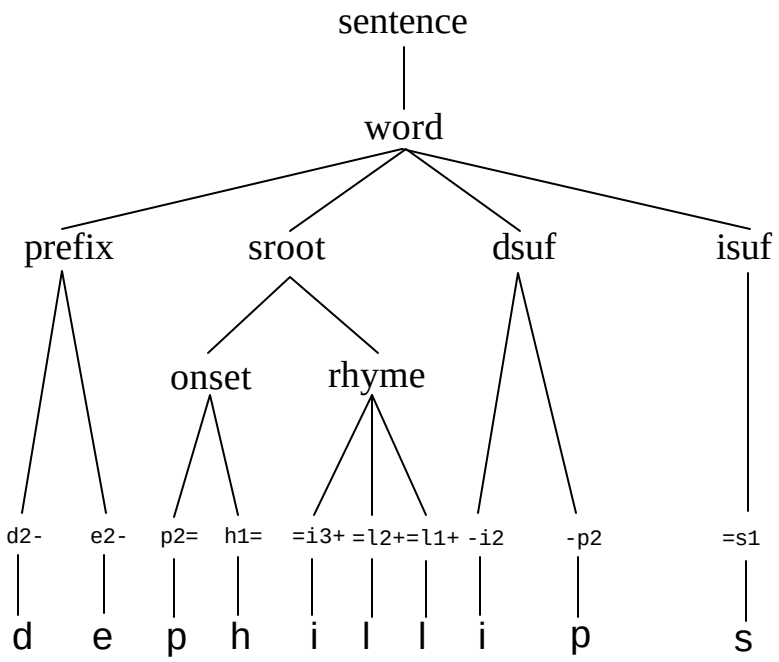


Figure 2-10: TINA's parse tree of the name "dephillips" (de- ph= =ill+ -ip =s)

The root of the parse tree is the sentence layer, which acts as the parent of the word layer. Directly below the word layer is the morph category layer, in which the names of morph categories appear. The word layer sprouts nodes on the morph layer according to the word decomposition rules given to TINA. The pre-terminal layer is the morph layer. Morphs presented in this layer of a TINA parse tree are not encoded as whole morphs. Here each morph is represented by a sequence of pre-terminal units, which are derived from the corresponding morph by separating each letter in that morph into a separate pre-terminal unit. Then, each pre-terminal unit is attached with the order of the letter in the original morph, counted *from right to left*. For example, the first letter in a three-letter morph is marked with number three following that letter. The second is marked with number two and the last letter is marked with number one. In order to identify the morph category, the morph category markers are preserved in each pre-terminal unit. For example, the pre-terminal unit representation of the morph "=Ind+" is "=I3+ =n2+ =d1+". Each pre-terminal node has one child which is the letter used in that node. Thus, the terminal layer of the parse tree in the sub-word case contains the letters used to spell the original proper names. The reason why we marked the pre-terminal nodes with their order is to prevent some cross-pollination effects that will bring an over-generalization problem in the morph level. If we

omit the numbers in the nodes in the pre-terminal layer, TINA is allowed to create undesired morphs by cross-pollination, because those nodes are considered the same node in the probabilistic network. An example of such a case is when we have the morph "=ine+" and "=er+", without the order notation in the pre-terminal layer, TINA might come up with a morph like "=iner+", which is not allowed, at least, for its multi-syllable property. And, since the pre-terminal nodes, in the same morph category, labeled with the same letter but different number are considered different nodes in the probabilistic network, the creation of undesired morphs is reduced, if not eliminated. According to the previous example, the pre-terminal unit "=e1+" in "=ine+" is different from the pre-terminal layer node "=e2+" in the other morph. This also yields a significantly more specific probability model, encoding sequence information at both the morph and the letter level.

Another interesting issue about TINA's rules we implemented in this thesis is the division of morph categories. We should note that no matter how we divide any morph categories in Figure 2-2, the probabilistic network will still hold. In this thesis, we suspect that the probability distribution of uroots starting with vowels is different from the probability distribution of uroots starting with consonants. Thus, we categorize all uroots according to whether they start with a vowel or consonant. Similarly, we separate dsufs starting with vowels and dsufs starting with consonants into different groups.

2.6 Chapter Summary

In this chapter, we introduced the method we used to decompose proper nouns into subword units we call "morphs". The idea of this method is the blending of syllabification and morphology, which are two of many possible technique people might exploit in decomposing English words. Each morph is syllable-sized and does not necessarily have meaning, while it is categorized as one of the six morph categories, which are *prefix*, *onset*, *rhyme*, *uroot*, *dsuf* and *isuf*. Proper name decomposition into morph sequences requires specific rules, which define allowed transitions between morph categories in the level directly below the word level. Each morph is marked with a special marker according to its category. There are two types of morph representation used in this thesis. In one type, each morph has a unique pronunciation, while in the other type, the orthography, excluding case information, of a morph is unique.

TINA is a system originally designed for parsing a sentence into the corresponding parse tree according to a set of grammar rules and trained probabilities. In this thesis, we tried to adapt TINA to a subword grammar in order for TINA to be able to work at the subword level. By utilizing a subword grammar, TINA can be used in the proper name decomposition task. Specifically, TINA can propose parse trees from the input letter sequence of a proper name. The parse trees are constructed by a top-down algorithm. They are constrained by word decomposition rules, the inventories of morphs used, and the input letter sequences at the parse tree terminal.

Chapter 3

The SUMMIT Speech Recognizer

3.1 Motivation

In this chapter, we will describe the recognition system that we used to build the recognizers used in this thesis. This system is called the "SUMMIT" recognition system [9], [10]. The purpose of this chapter is to provide the basic idea about this recognition system. The background in this chapter should be useful in understanding the building of recognizers, to be described in Chapter 6.

3.2 Speech recognition in SUMMIT

The SUMMIT recognition system is a segment-based speech recognition engine developed in the Spoken Language System group at MIT. This system is used in the group to build speech recognizers, based on a probabilistic framework. The function of this system is to find the word sequence, which is probabilistically closest to the input acoustic information contained in the audio waveform.

Speech recognizers typically represent the acoustic information as a frame-based sequence of observations. They convert the input acoustic information into a series of vectors. Each vector captures information about the speech signal during a short window in the input speech waveform. Such windows are usually called "frames". The observation frames are usually equally spaced and slightly overlapping in time. The full set of frame-based observations is represented as:

$$O = \{ o_1, o_2, \ldots, o_{Nf} \}$$

where O : set of observations

o_n : a frame-based observation vector.

N_f : total number of frames throughout the waveform.

Usually, the observation vectors are some representations of the spectral information of the waveform presented in the corresponding frames. The spectral representation that is widely used is Mel frequency scale cepstral coefficients (MFCC's). In frame-based approaches, such as Hidden Markov Models (HMM's), recognition uses the frame-based observations directly in the probabilistic framework used for scoring and search. The problem of speech recognition in these approaches is basically the problem of finding a hypothesized string of words, *W'*, which is most likely given *O*, to maximize the probability that that string really occurs given the observations. This is represented as:

$$W' = \arg\max_{W} p(W \mid O)$$

In SUMMIT, which is a segment-based system, the set of observation vectors, *O*, is first transformed into a network of segment-based feature vectors, *V*. And the recognition is performed using the segment network instead of the sequence of frames. Thus, the generic probabilistic expression used to describe the recognition process by SUMMIT is represented as:

$$W' = \arg\max_{W} p(W \mid V)$$

In SUMMIT, it is presumed that each segmental unit contains one phonetic unit. Thus, SUMMIT models a word as a sequence of phones, each of which occupies one segment in the segment network. Segment networks created by SUMMIT are shown in Figure 3-1. On the top is the time waveform of the utterance "mark". Directly below the time waveform is the corresponding spectrogram. Shown below the spectrogram is the segment network, where the blocks with darker shade are corresponding to the chosen phonetic string, which is shown below the segment network. In this case, the recognized phonetic string is "m aa –r kcl k". And the word (morph in

this case) string chosen is “m= =arc+”. The correct phonetic string, correct word string and the original transcription for this utterance are shown at the bottom of the figure.

During the search for the best word sequence, *W’*, SUMMIT chooses the single best path through the segment network. Also, the search process finds the single best path of segments *S’* through the segment network and the single best sequence of phonetic units *P’*. With these additions, the probabilistic expression describing the recognition problem becomes:

$$\{W', P', S'\} = \arg\max_{W,P,S} p(W, P, S \mid V)$$

By Bayes’ Rules, we have:

$$p(W, P, S \mid V) = \frac{p(V \mid S, W, P)\, p(S, W, P)}{p(V)}$$

$$p(W, P, S \mid V) = \frac{p(V \mid S, W, P)\, p(S \mid P, W)\, p(P \mid W)\, p(W)}{p(V)}$$

And, since p(*V*) is constant over all *W*, *P* and *S*, the recognizer’s decoding expression can be written as:

$$\{W', P', S'\} = \arg\max_{W,P,S} p(V \mid S, W, P)\, p(S \mid P, W)\, p(P \mid W)\, p(W)$$

We can see that the argument we want to maximize is the product of four probabilities, which are p(*V*|*S*,*W*,*P*), p(*S*|*W*,*P*), p(*P*|*W*) and p(*W*). According to their meaning, these four probabilities are referred to as “the acoustic model”, “the duration model”, “the pronunciation model” and “the language model” respectively. Each model will be discussed in the following separate sections except the duration model. Although the SUMMIT recognition system has the capability to incorporate a duration model, we will not use one in our work since features related to duration are already included in the segment feature vectors used in *V*.

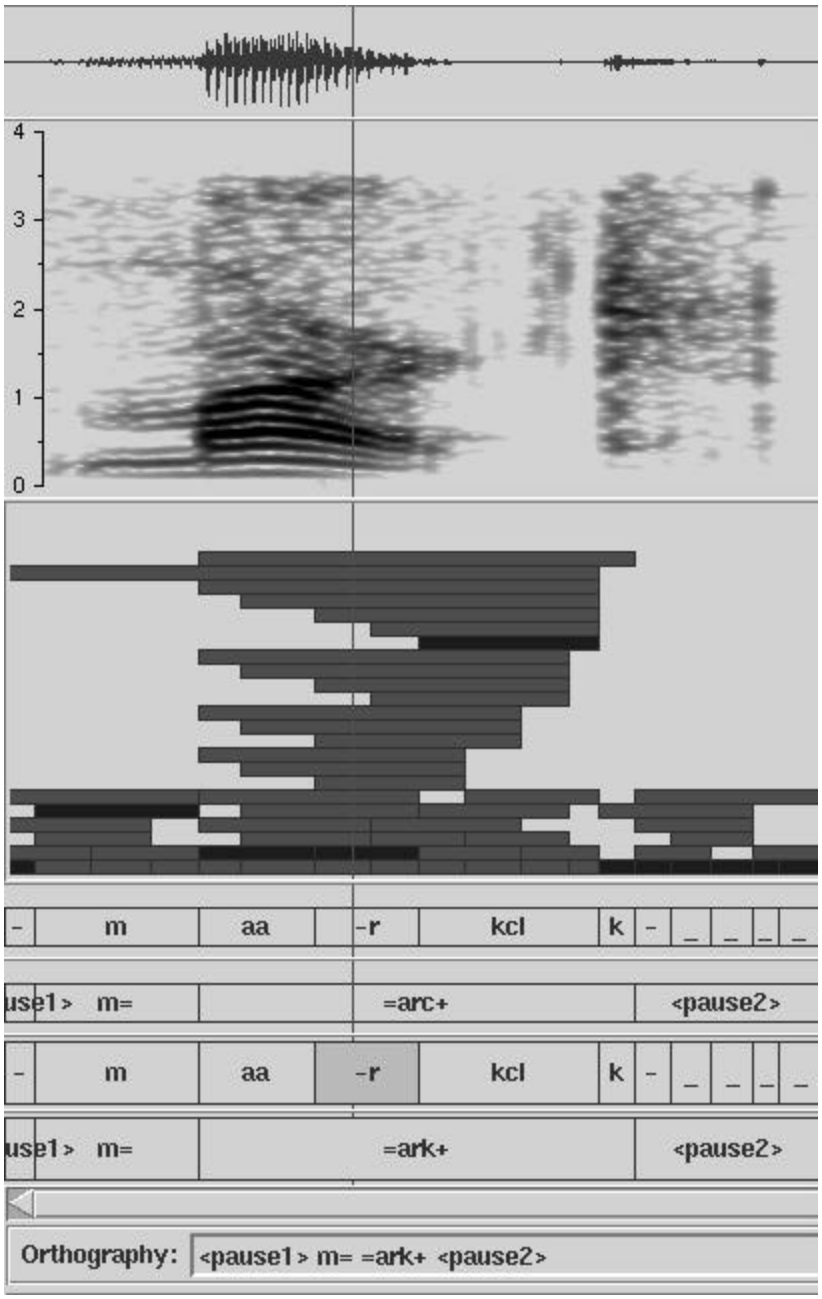


Figure 3-1: An example of word recognition results for the name “Mark”.

Note: -r is a special symbol reserved for postvocalic ‘r’
and kcl stands for the closure interval of the ‘k’ consonant

In addition to p($V|S,P,W$), we can then add context dependency into our acoustic model by augmenting the series of landmarks, Z, into the calculation of probability. The series of acoustic landmarks represents the potential segment boundaries separating the segment elements in V. Acoustic measurements can be made around these landmarks in both the forward and backward directions. The acoustic information at these landmarks is called "diphone" models since it captures acoustic information of both the phones preceding and succeeding the phone boundaries. The general expression for the acoustic model when both segment and diphone models are used is p($V,Z|S,P,W$). We assume independence between V and Z. Then p($V,Z|S,P,W$) is the product of p($V|S,P,W$) and p($Z|S,P,W$). To calculate the acoustic model, we estimate the probability density function to be mixtures of diagonal Gaussians. These models are trained using the standard K-means and EM algorithms. The number of mixture components in each model is allowed to vary depending on the size of the measurement vector and the number of available training vectors.

The pronunciation model is represented with the expression p($P|W$), the probability that the given word W has the phonetic string P. In order to calculate this probability, first, each word in the vocabulary of the recognizer must be given a phonetic baseform pronunciation as well as its possible pronunciations. Then, the baseform pronunciations are expanded to cover all the possible alternatives, which might occur due to general phonological variations. Some phonological rules, which define potential variations, are provided. All of the possible pronunciations for each word are presented in a phonetic network, in which each arc has its corresponding probability of traversing that arc. These probabilities are then used to calculate the pronunciation score.

The language model is represented by p(W), the probability that a word W occurs. Generally, a form of language models called "N-gram" is used as SUMMIT's language model, where N is a number. In an N-gram language model, occurrences of each particular group of N words are counted and then normalized in order to obtain the probabilities. The most common N-grams used with the SUMMIT recognizer are bigram and trigram, in which probabilities of the occurrences of particular sequences of two and three words are calculated.

In this thesis, this language model is the part leading to most of the experiments for studying the framework of our integrated recognizers. The details of how we vary the types of language models used will be discussed in Chapter 7 and Chapter 8.

In SUMMIT, the representations of the acoustic model, the pronunciation model and the language model are in the form of a Finite State Transducer (FST), whose structure will be discussed in Section 3.3. The acoustic model is represented by the FST mapping acoustic features to phonetic units. The pronunciation model is represented by the FST mapping phonetic units to word sequences, and the language model is represented by the FST describing the probability of occurrence of each word sequence.

3.3 Finite State Transducers (FSTs)

Most of the components and constraints in a speech recognition system, for example, language models, phonological rules and recognition paths, are finite state. Thus, the same representation should be utilized for consistency and flexibility. In SUMMIT, the Finite State Transducers (FSTs) are used for the representation of these components. In this section, we will introduce the Finite State Acceptor (FSA), which is the simpler version of the FST and then move forward to the FST and the composition operation of the FSTs that we will use as our key operation to construct the language models used in our experiments.

3.3.1 FSA, FST and weighted FST

A Finite State Acceptor (FSA) is a graph consisting of a finite number of states and transitions among the states. The graph begins in a particular state and change states when specific conditions occur. At any point in its operation, the next state can be determined by knowing the current state and the conditions which can cause the transitions. The graph ends at one or more final states. In our notation, each circle represents a node in the graph and each arc shows the transition between two nodes. Attached with each arc is the condition in which the transition corresponding to that arc occurs. A double circle represents an end node. An example of an FSA is shown in Figure 3-2.

A Finite State Transducer (FST) is defined in the same way as an FSA except that, each transition in an FST is allowed to have output. Each arc in an FST is labeled by the form "input:output". With the conditions and outputs attached to FST arcs, it can be used to relate the input sequences to the output sequences. Furthermore, a number can also be attached to each arc in order to determine the weight of the corresponding transition. An example of a weighted FST is shown in Figure 3-3. This way of mapping the input sequence to the output sequence is used to represent various components, including the acoustic model, the pronunciation model and the language model, in a SUMMIT recognizer. The input sequence of the FST representing the acoustic model is the sequence of segment-based feature vectors, and the output sequence is the sequence of phones. For the pronunciation model, the input sequence of the FST is the phone sequence and the output sequence is the word sequence. Both the input and the output sequences of the FST representing the language model are typically word sequences, but in our case are morph sequences. The weights along the path in each FST represent the probability of the occurrence of the corresponding transitions. As the graph is traversed, the weights along the path passed are accumulated. The accumulated weight when an end node is reached is the score for the mapping between the corresponding input and output sequences.

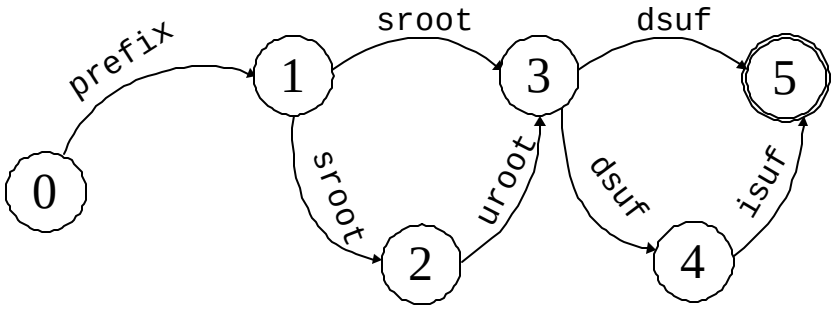


Figure 3-2: An example of an FSA

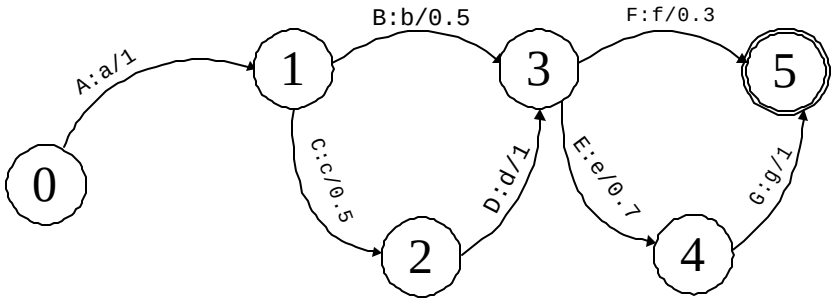


Figure 3-3: An example of a weighted FST

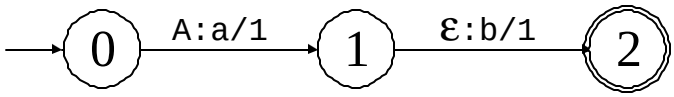
3.3.2 Composite FST

In building a series of cascaded FSTs, the composition approach is utilized. The output sequence of an FST is fed to the input of the next FST. This approach is used to cascade the acoustic model FST, the pronunciation model FST and the language model FST together in order to obtain the FST that maps the input segment-based feature vectors to the output word sequence. For example, if we have the mapping FSTs called FST_a and FST_b shown in Figure 3-4(a) and 3-4(b), the composite FST (FST_a o FST_b) will be as in Figure 3-4(c). This example is adapted from Dr. Hetherington's example of the FST composition in his talk given in a discussion group in the Spoken Language System group. State (a,b) in the composite FST is associated with state a from FST_a and state b from FST_b. The final state occurs only if both associated states from FST_a and FST_b are final. The transition with label x:y occurs only if FST_a has x:i and FST_b has i:y transition. "ε" means the transition takes no input. The weights are calculated by multiplication of the weights of the associated transitions. Suppose the input sequence is "A", the mapping works as follows:

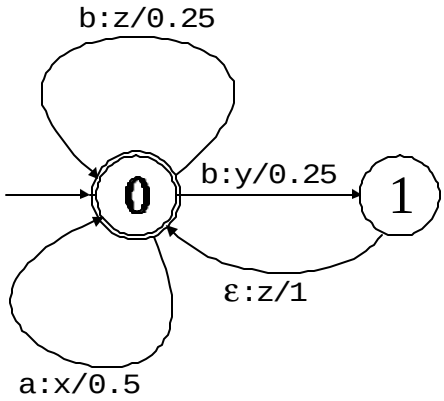
$(FST_a \text{ o } FST_b)(A) = FST_b\ (\ FST_a(A))$

$= FST_b\ (ab)$

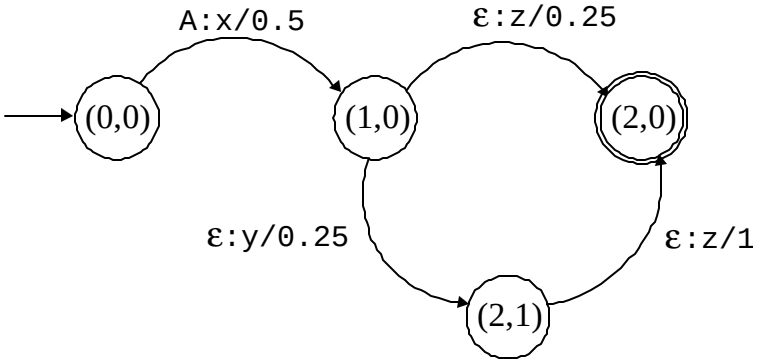
= xz with score 0.125 or xyz with score 0.125



(a)



(b)



(c)

Figure 3-4: (a) FST_a in the example (b) FST_b in the example (c) (FST_a o FST_b)

(adapted from Dr. Hetherington's example given in a discussion group session in the SLS group)

3.4 Composite language model of our pronunciation recognizers

As suggested in the preceding section, the basic units the pronunciation recognizer used in our integrated system are morphs instead of words. Thus the mapping becomes the mapping from segment-based feature vectors to phonetic sequences to morph sequences. And then the language model specifies the probability of the occurrence of each morph sequence. In different experiments, we use different language models. One part of the language model is the baseline FST obtained from the training name lexicon. These FSTs are applied to every pronunciation utterance. The other parts of the language model are the FSTs derived from the spelling knowledge, treated in different methods. All of the language model FSTs can be considered the mapping from input morph sequences to themselves, but with different scores in the different FSTs. Thus, composing the baseline FST with each FST, obtained from the spelling knowledge, creates different language model to be experimented with. More details about where each FST obtained from the spelling knowledge comes from will be shown in Chapters 7 and 8.

3.5 Chapter summary

In this chapter, we discussed the basis of the SUMMIT recognition system, in which various probabilities are calculated in order to perform the speech recognition task. The input waveform is represented in the form of a segment network which is walked, and scores are calculated based on this network. There are three models or scores, apart from the duration model that is not currently used, involved in recognizing the input waveform. They are the acoustic model, which also includes context-dependency through the boundary model, the pronunciation model and the language model. The acoustic model is the probability of occurrence of the segment-based feature vectors, derived from the input waveform, given sequences of words, segments and phonetic units. The pronunciation model is the probability of occurrence of sequences of phonetic units given specific words. And the language model is the probability of occurrence of each word given its linguistic context. Also, the FST representation, which is used to represent the models in the SUMMIT recognizer, is discussed along with examples of composition among various FSTs.

Chapter 4

Data

4.1 Motivation

Since data play a crucial part in our research, we dedicate this chapter to describing the data and how we prepare and use them. Apart from the data in the form of audio files, we also made use of a list of proper names gathered from various places as the source used to obtain statistical models of letter sequences for names. The separation of the data into sets and the broad characteristics of each set will also be discussed.

4.2 Type of data needed

The data we used throughout this research can be divided into two main categories. First are the audio waveforms. These data required somebody to say or spell the names while being recorded. These audio files are then categorized into two groups according to their contents. The audio files whose contents contain spelled proper names are called "name-spelling utterances". In the other group, the contents of the audio files contain pronunciations of proper names. These audio files are called "pronouncing utterances".

The other type of data needed is a lot simpler but as important. These data are the list of proper names that provide us with examples of letter patterns in proper names. These data are collected in the form of plain text. We would like to collect as many proper names as possible in order for them to represent the universe of the spelling of proper names. We should note that, by using the

word proper names, we really are interested in people's names, as opposed to, for example, place names.

4.3 Data Sources

The sources of data can be divided according to the type of data, i.e. name lists and audio files.

4.3.1 Name lists

This type of data was easier to collect and required less work to prepare since they are in plain text form. We gathered the names from various sources, especially from the World Wide Web. Such websites where we gathered names were, for example, websites providing telephone directories, websites dedicated to baby naming and websites containing census information. Another portion of our name list came from publications containing people's names, such as, names of US senators and names of MIT's personnel. Apart from those sources mentioned, we also obtained some names from transcribing the audio files. The sources of these audio files will be discussed in the next section.

The total number of names gathered was 104,356 names. The spellings of the names in this name list were used to train the language model for the "general" letter recognizer, to be discussed in Chapter 6.

4.3.2 Audio files

Throughout the period of this research, we obtained the audio files we need from two sources. The first source is the OGI corpus [12], [13], which is the spelled and spoken word telephone corpus created by the Oregon Graduate Institute (OGI). This corpus consists of speech recordings from over 3650 telephone calls, each made by a different speaker, to an automated prompting/recording system. Speakers were asked to say their name, where they were calling from, and where they grew up. Also, they were asked to spell their first and last names. However,

the data we used were the waveform files whose contents contained the saying and the spelling of the first and last names only. Each response to a prompt was stored as a separate waveform file, tp make it easier to use the data in our experiments. Also the text transcription of each utterance was already provided.

The other source of audio files is the Jupiter [11] system built by the Spoken Language Systems group, Laboratory for Computer Science at Massachusetts Institute of Technology. The Jupiter system is the system where people can call via a toll-free number and ask about the weather forecast and some other weather related questions. Our data were collected over a period of several months in 1999. Each caller was asked to say and spell a name of a person he or she knew, at the very beginning of their dialogue. The response to this prompt was recorded as one waveform file. Thus, in order to use the audio files recorded from the Jupiter system, the files needed to be properly cut into pieces. Each piece contained either the spelling or the pronouncing information only.

Though both sources provide telephone-quality audio files, we prefer to have more of the data from the Jupiter domain since these data better represent the real data in the system we are going to deal with in the future at the Spoken Language System group. However, due to time constraints and the amount of data we need, we found it necessary to fully utilize the OGI corpus together with the data form the Jupiter system.

4.4 Jupiter data preparation

Since the naturalness of the speech is concerned, we have asked each caller to say and spell a person's name in one turn and we did not restrict how the callers formulate their expressions. Thus, we cannot expect consistent structure for each utterance. Also, the quality of utterances varied from call to call due to the fact that the calls were made through the phone line. Due to these variations, collecting the audio data to be used in the research was selective, and the extraction of the name-spelling utterances and the pronunciation utterances from the original waveforms was done manually, with the help of a forced recognition technique.

Some examples of the utterances from which we can extract the name-spelling utterances and/or the pronounced utterance are in the following forms.

“Bob” ‘b o b’
‘b r i t t a n y’ “Brittany”
The name of someone I know is “Michelle”
“Jim” “Glass” ‘j i m’ “Jim” ‘g l a s s’ “Glass”
Uh “Mary” ‘m a r y’
‘j a s o n’ jas- [incomplete]
“John” j o [incomplete]
[incomplete] –ohn ‘j o h n’
[incomplete] u s a n “Susan”
“Stephen”
‘j o e’

Double quotes indicate that the word enclosed by them can then be extracted as the usable pronounced utterance and single quotes indicate that the word enclosed can be the name-spelling utterance.

There are some utterances that we discard since they contain no usable information. The types of utterances we did not use are, for example, the utterances in which both saying and spelling of the names are incomplete, the utterances which are too noisy or unintelligible, and the utterances that contain neither the saying nor the spelling of proper names.

The first step in preparing the audio files was to transcribe all of the utterances collected. Then we listed all the vocabularies used along with their pronunciation. All of the utterances were forced recognized in order to create the time-aligned phonetic path for each utterance. By looking at the time-aligned phonetic paths, we could then tell the location of the desired part of the waveform with respect to time. Consequently, we extracted the waveform in the time interval we want and suitably stored it as the name-spelling utterance or the pronounced utterance.

Apart from the preparation of the audio files, we also need a name lexicon which provides a pronunciation in the form of morphs in both Type I and Type II formats. The names were given the pronunciation by manually decomposing each name. During the process of building the name

lexicon, the morph lexicon in which the pronunciation of each morph was defined was created from the list of morphs used in the name lexicon.

4.5 The training set and the test set

We divided our data into two sets, namely, a training set and a test set. The data in the training set were used in various tasks throughout the research. The tasks we used the training set for were as follows:

- The names in the training set were studied in order to define the inventories of morphs for both Type I and Type II. In other words, both morphs Type I and Type II were defined to cover all the names in the training set.
- The spellings of the names in the training set were used to build the bigram and trigram language models for the "training-set oriented" letter recognizer, to be discussed in Chapter 6.
- The morph decompositions of the names in the training set were used to train the probabilities of TINA's grammars for both Type I and Type II.
- The morph decompositions of the names in the training set were used to build the bigram and trigram language models for the baseline morph recognizer
- In the experiments, to be discussed in Chapter 7 and Chapter 8, we adjusted the parameters of the integrated system to optimize the performance of each case based on the training set.

The data in the test set were used for testing the performances of the pronunciation and letter recognizers. Also they were used to test various frameworks of the integrated system defined in Chapter 7 and Chapter 8.

Some of the names in the training set came from the OGI corpus and the others come from the data collected earlier in the Jupiter system. The names in the test set came form the data collected from the Jupiter system after the training set was defined. The details about the two sets are shown in Tables 4-1 and 4-2.

The training set		
Sources		OGI corpus / Jupiter system
Number of pronounced utterances	OGI	1299
	Jupiter	3126
Number of name-spelling utterances	OGI	1169
	Jupiter	3572
Number of unique names	OGI	574
	Jupiter	1887
Total number of pronounced utterances		4425
Total number of name-spelling utterances		4741
Total number of utterances		9266
Total number of unique names		2248

Table 4-1: Details of the training set

The test set	
Sources	Jupiter system
Number of pronounced utterances	1383
Number of name-spelling utterances	1275
Number of unique names	475
Total number of utterances	2658

Table 4-2: Details of the test set

We provided the name lexicon and morph lexicon of both types for each of the sets. There were 1266 Type I morphs and 1087 Type II morphs used in the name lexicon of the training set and 451 Type I morphs and 412 Type II morphs used in the name lexicon of the test set. The sizes of the overlap were 390 morphs for Type I and 377 for Type II. Note that, in building the morph recognizer and defining TINA's rules, only the morphs appearing in the training set were used. And, since 91.5% of Type II morphs used in the test set are also used in the training set, while it is 86.4% for Type I morphs, Type II morphs should have better generalizing properties than Type I morphs. Among the 475 names in the name lexicon of the test set, there are 275 names that are also present in the training set.

Figures 4-1 and 4-2 below illustrate the sizes of the various name and morph lexicons along with their overlaps.

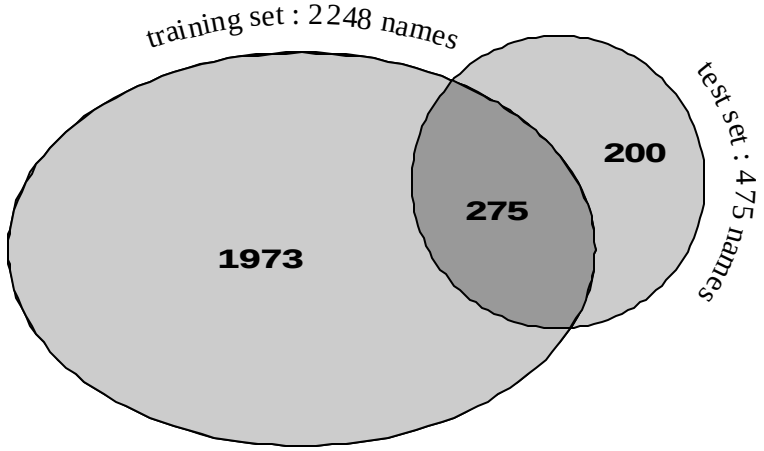


Figure 4-1: Sizes of the name lexicons of the training set and the test set

(a)

(b)

Figure 4-2: Sizes of the morph lexicons of the training set and the test set

(a) using morph representation Type I

(b) using morph representation Type II

4.6 Chapter summary

There were two types of data used throughout this research. They were lists of names and audio files containing either pronunciations or spellings of proper names. The list of names used was gathered from various sources including transcription of the audio files. Some audio files were obtained by recording the utterances from phone calls to the Jupiter weather information system. Other audio files were taken from the OGI corpus, which contained telephone-quality utterances of people spelling or saying names. Data were grouped into two sets, the training set and the test set. The details of each set as well as their corresponding morphs, both Type I and Type II, as well as their sizes and overlapped portions were discussed in the chapter.

Chapter 5

The Integration of Components

5.1 Motivation

In Chapter 2, we talked about TINA, which is the natural language component used to propose pronunciations from the given letter sequences. In this chapter we will describe how we combined TINA and other components together in order to perform the letter and pronunciation task we desired. The other two components apart from TINA are the two recognizers performing two different tasks, which are (1) recognizing the letter sequences from the name-spelling utterances, and (2) recognizing the morph sequences from the pronounced utterances. How the functions of the two recognizers are related in the system with the assistance of TINA will be discussed. The details in building the two recognizers will be discussed later in Chapters 6 and 7.

5.2 Overview

The goal of our system is to gather the information contained in both the name-spelling utterances and the pronounced utterances and use the information from the two sources together to propose the recognition result of both types of utterances. The hope is that the recognition results of this integrated system will provide better accuracy in terms of both the letter recognition accuracy and the phone recognition accuracy than running the spelling recognition task and the pronunciation recognition task individually.

In order to accomplish this, we need at least three basic functions in this system. First, we need a component that is able to deal with the name-spelling utterances and propose the possible letter

sequences for those utterances. Next, we need a component that is able to recognize the input pronounced utterances. The morph sequences corresponding to the input pronunciation utterances are expected from this component. Finally, we need a way for the two components to share information where appropriate. And the morph unit is chosen as a means for the two recognizers to share information.

Clearly, the first two components are the letter recognizer and the morph recognizer. Our expectation is that the letter recognition task should be easier and yield more accurate recognition results than the morph recognition task. Therefore, we designed the system so that the preliminary letter recognition task does not use any information contained in the pronounced utterances but pays attention to the name-spelling utterance alone. Then, the morph recognition task exploits the information in the proposed letter sequences from the letter recognizer along with the information in the input pronounced utterances. Thus, we need a component that allows parsing from the letter sequences into the form that the morph recognizer can use. This is where TINA comes into play. TINA, operating at a sub-word level, is responsible for turning the input letter sequences into the appropriate morph sequences for each name. These morph sequences are used as additional constraints for the morph recognition task.

The proposed morph sequences, derived from the name-spelling utterances via the letter recognizer and TINA, are the recognition results of the integrated system. From these proposed morph sequences, we can determine the pronunciation and the spelling recognition performance of the system, due to the fact that both the pronunciation and the spelling information are encoded in each morph already.

5.3 Details of integration

A block diagram of the interconnection between the components in our system is shown in Figure 5-1 below.

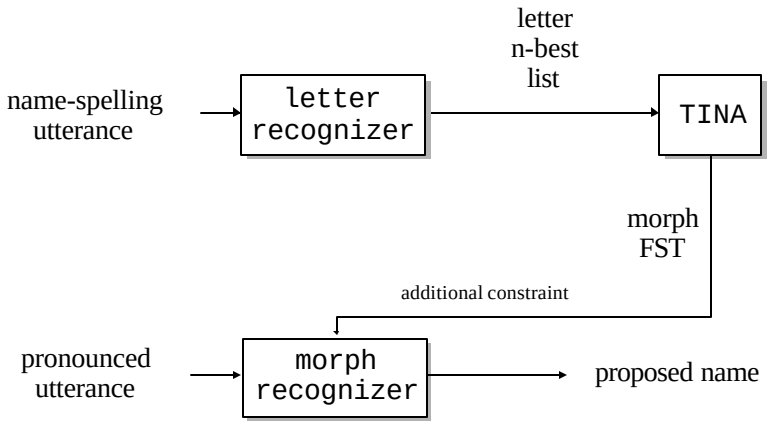


Figure 5-1: The integration of the components

At the output of the letter recognizer, the 10-best list of the proposed letter sequences is presented. This list contains the ten letter sequences with the highest total score from the letter recognizer. The letter sequences were listed in a certain format, in which the 10-best list letter sequences corresponding to each name-spelling utterance is grouped together. Some examples of letter sequences sent to TINA are shown in Figure 5-2.

TINA has a mechanism to construct an interconnected graph from an n-best list, which provides an expanded solution space due to cross-pollination, as mentioned earlier in Chapter 2. We felt that this mechanism might permit the recovery of a correct letter sequence that was absent from the original list. For example, if we look at the last 10-best list in Figure 5-2, we can see that if we do the cross-pollination between the letter sequences in the third choice and the ninth choice, we can recover the letter sequence " c h a d ", which is the correct letter sequence. TINA needs a trained grammar in order to be able to parse the graphs of the letter sequences into the possible morph decompositions. TINA was trained on the name lexicon of the training set in the experiment conducted with the morph representation Type I in Chapter 8, while, in the experiment conducted with the morph representation Type II in Chapter 9, TINA was trained on the same set of names but with different (Type II based) transcribed morph representations.

```
NBEST

a p p o l l a h
a b d u l l a h
d o l l a h
a p t o l l a h
a b d o l l a h
d u l l a h
a d o l l a h
p o l l a h
b o l l a h
a n d o l l a h

h u
a p u
a p a u
a u
a b a u
a b u
a t u
h e u
h a u
s p a u

c a m
c a n
j a m
g a n
c a s
c a r
j a n
k a m
g a m
g a r

h a
h a n
h a d
h a e
s h a
h a n d
j a
h a t
c h a
s h a n
```

Figure 5-2: Examples of the 10-best output letter sequences

for the inputs “abdullah”, “abu”, “adam” and “chad”

TINA presented the proposed morph sequences with their corresponding scores in the form of Finite State Transducers (FSTs). An example of such a network is shown in Figures 5-3 and 5-4. The concatenation of the morph specified along each path between the start node and the end node is one of the morph decomposition hypotheses according to the input letter sequences in the n-best file. The number shown along with the morphs is the score along that arc. TINA was set, as a default in this thesis, to propose up to ten morph decomposition hypotheses for each name-spelling utterance. In general, TINA would create the FSTs with ten paths between the start node and the end node. However, if TINA could not come up with ten hypotheses for a given search depth, it was perfectly legal for the FSTs to have fewer than ten paths.

On the pronunciation front, the language model of the morph recognizer was pre-trained with the name lexicon of the training set. This trained language model was then converted into the FST form and combined with the FST of the morph representation from TINA in order to recognize the pronounced utterance whose content was the same proper name as the one spelled in the name-spelling utterance. So, for each pronounced utterance, the language model of the morph recognizer was the combination of the baseline FST, common to all of the names, and the FST from the corresponding name-spelling utterance.

The FSTs created from TINA were associated with adjustable weights to optimally compose the two independent language models. In the composing process, the score on each arc is computed by multiplying the scores on the corresponding parallel paths and summing up the scores on the corresponding serial paths. The paths that do not have corresponding paths in the other FST are pruned away in the resulting FST.

With the language model being dynamically updated according to the names contained in the content of the pronounced utterances, the morph recognizer then proposed the recognized morph sequences from the information on both the name-spelling utterances and the pronounced utterances. The morph sequences proposed at the last state can be converted to phone sequences by looking up the morph pronunciations in the morph lexicon. Also, the letter sequences can be obtained from the morph sequences by discarding all of the markers tagged with the morphs. Then we can compare the pronunciations and the spellings of the proper names with the baseline systems running separately.

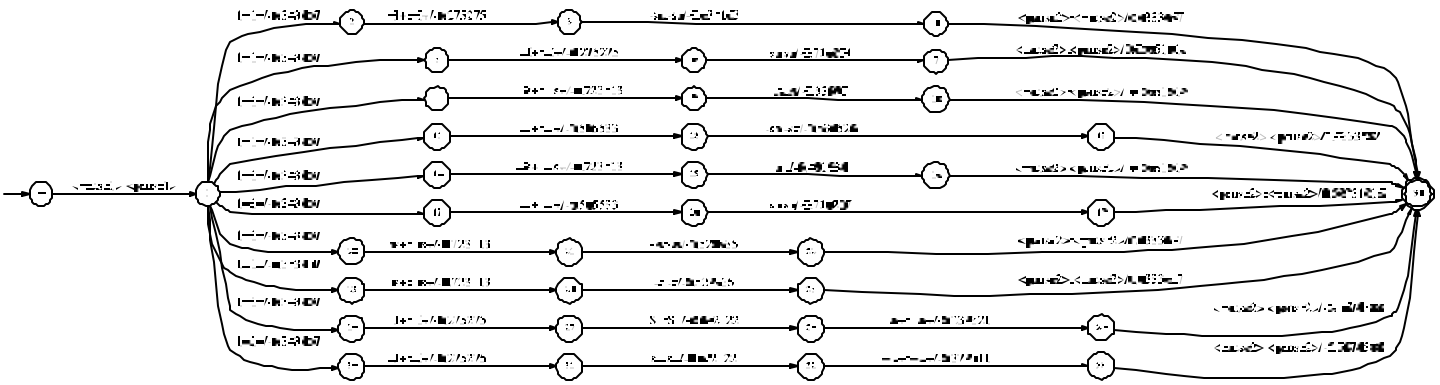


Figure 5-3: An example of the FSTs of the morph sequences from TINA, for the spelled word "Lisa"

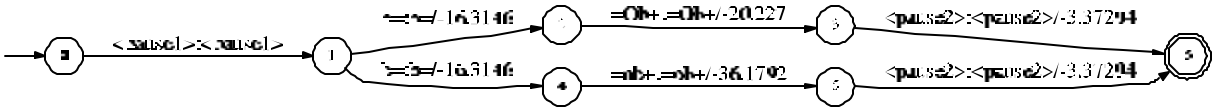


Figure 5-4: Another example of the FSTs of the morph sequences from TINA, for the spelled word "Bob"

5.4 Chapter summary

In this chapter, we talked about how the three components, the letter recognizer, the morph recognizer and TINA, share information and provide assistance to the common goal. The letter recognition process, considered to be the task given the more reliable result, was operated on the name-spelling utterances without the help of the other components. From the 10-best letter sequences, TINA proposed the possible morph representations in the form of a Finite State Transducer (FST). The FST of the morph sequences was composed with the baseline language model, common to all of the utterances, of the morph recognizer in order to recognize the corresponding pronounced utterances. The morph recognizer then proposed the possible morph recognition results of the pronounced utterances with the assistance of the spelling knowledge conveyed through the letter recognizer and TINA. Both the pronunciation and the spelling information have already been encoded in the morph notations. Thus, from the proposed morph sequences of the morph recognizer, we could examine the pronunciation and the spelling recognition results.

Chapter 6

The recognizers

6.1 Motivation

In the last chapter, we described the interconnection of the components required in the framework designed to use both the spelling and pronunciation knowledge in recognizing proper names. The natural language component, which is used to propose the possible pronunciations from the letter sequences, and the morph notations used in the system are described in Chapter 2. In this chapter we will focus on the letter recognizers and the morph recognizers used in the integrated system mentioned in Chapter 5. The first letter recognizer to be mentioned is the letter recognizer whose language models were trained on a large name list. The other letter recognizer is the letter recognizer whose language models were trained on the letter sequences in the training set defined in Chapter 4. There are also two morph recognizers involved in this thesis. The first one uses the morph inventory of Type I while the other uses the morph inventory of Type II. The recognition accuracies of both the letter recognizers and the morph recognizers are reported in this chapter.

6.2 The "general" letter recognizer

6.2.1 General information

The general letter recognizer was built from MIT's SUMMIT recognition system mentioned in Chapter 3. The purpose of this letter recognizer was to perform the spelling recognition task on the name-spelling utterances. The proposed letter sequences were expected from the recognizer

when operating on the name-spelling utterances. The letter recognition accuracy of this letter recognizer would then be used to benchmark the integrated system in which this recognizer was also used to provide the letter sequences to TINA. The ten hypotheses of the letter sequences which had the highest recognition scores were listed. The hypothesis with the best scores was at the top while the others were listed below, ranked according to their recognition scores.

Since the basic units of the letter recognizer are the English alphabet, specifically lower-case a to z, the entire vocabulary required to perform this letter recognition task is just these 26 words. For this thesis, in order to simplify the problem, we concentrated on just recognizing these letters, and thus we set aside any utterances containing other artifacts, such as special symbols like "hyphen", the word "capital", the word "double" and other fill words like "um".

The language models of the "general" letter recognizer were n-gram language models, where n equal to 2 and 3 were used. These letter bigrams and trigrams were trained on the letter sequences appearing in the large name list mentioned in Section 4.3.1. Although, to come up with the 10-best letter sequences as the recognition results of the letter recognizer, the trigram language model was used, we also did the recognition using the bigram language model, which was normally used when only the one-best hypothesis was concerned. The recognition accuracies in the bigram case will be used as another baseline performance to be compared with the performance of the integrated system.

The system used pre-existing acoustic models trained from a collection of utterances obtained from the Jupiter and Pegasus domains. One could expect that these models would be suboptimal for this letter recognition task, since the models were not trained on the pronunciation of letters directly. However, as long as our research's major interest was in providing a framework for the whole system and these same models are used for benchmark systems as well, these acoustic models should be acceptable.

6.2.2 Performance

The results of letter recognition on the training set and the test set are shown in Table 6-1 below.

The "general" letter recognizer		
Language model: bigram		
set	Letter sentence recognition accuracy (%)	Letter recognition accuracy (%)
training set	33.6	76.1
test set	34.7	73.5
Language model: trigram		
training set	48.1	81.4
test set	49.7	79.2

Table 6-1: The performance of the "general" letter recognizer

6.3 The "training-set oriented" letter recognizer

6.3.1 General information

The "training-set oriented" letter recognizer was built in the same fashion as the "general" letter recognizer except for the difference in training the language models. The language models used in both letter recognizers are the letter bigram and the letter trigram. However, the language models of the two letter recognizers were trained on different sets of letter sequences. As mentioned earlier, the language models of the "general" letter recognizer were trained on the letter sequences of all of the names we had, of which the size is 104,356 names. On the other hand, the language models of the "training-set oriented" letter recognizer were specifically trained on the letter sequences of the names contained in the training set. Thus, this letter recognizer was biased to the data in the training set more than the "general" letter recognizer.

Other factors related to building the "training-set oriented" letter recognizer, such as the vocabularies and the acoustic models, were the same as the ones used in building the "general" letter recognizer.

6.3.2 Performance

The results of letter recognition on the training set and the test set are shown in Table 6-2 below.

The "training-set oriented" letter recognizer		
Language model: bigram		
Set	Letter sentence recognition accuracy	Letter recognition accuracy
training set	39.9	79.4
test set	30.4	69.6
Language model: trigram		
training set	54.1	84.2
test set	41.4	74.0

Table 6-2: The performance of the "training-set oriented" letter recognizer

6.4 Letter recognition result analysis and comparison

If we compare the letter recognition results of the two letter recognizers in Tables 6-1 and 6-2, we can see that both the sentence recognition accuracy and the letter recognition accuracy of the "general" letter recognizer operating on the training set are lower than the ones of the "training-set oriented" letter recognizer. In the bigram case, the letter recognition error rate is 23.9% for the "general" letter recognizer, while it is 20.6% for the "training-set oriented" letter recognizer. In the trigram case, the letter recognition error rate is 18.6% for the "general" letter recognizer, while it is 15.8% for the "training-set oriented" letter recognizer. We can see that, by training the language models directly on the training set, the error rates for the training set recognition are

reduced by 13.8% and 15.1% in the bigram and trigram cases respectively. This result turned out as we expected since the language models of the "general" letter recognizer trained on the much larger set of letter sequences, were more generalized. However, being very specific to the names in the training set has a drawback on the recognition accuracies of names that are not in the training set. In the bigram case, the letter recognition error rate on the test set recognition is 26.5% for the "general" letter recognizer, while it is 30.4% for the "training-set oriented" letter recognizer. In the trigram case, the letter recognition error rate is 20.8% for the "general" letter recognizer, while it is 26.0% for the "training-set oriented" letter recognizer. We can see that being more generalized improves the test set recognition error rates by 12.8% and 20.0% in bigram and trigram cases respectively.

In the experiments conducted in this thesis, we took advantage of the different letter recognition accuracies of the two letter recognizers in studying how the performance of the letter recognizer affects the integrated system.

6.5 The morph recognizer using the morph representation Type I

6.5.1 General information

The morph recognizer was also built from MIT's SUMMIT recognition system, as for the letter recognizer. The function of the morph recognizer was to propose the possible morph representations of the input pronounced utterances. From these proposed morph sequences, the phone sequences can be obtained through morph-to-phone mapping. The morph recognition accuracy and the mapped phone accuracy would be used as the baseline accuracy to be compared with the morph recognition accuracy and the phone recognition accuracy obtained from the integrated system in which this morph recognizer was used. The 10-best lists of the morph sequences were expected from the morph recognizer.

The basic units of this morph recognizer were the 1266 morphs defined from the names in the training data using Type I notation. Note that the morphs used to represent the names in the test set were not restricted to be the morphs appearing in the training set. At the time of the

recognition of the test set, we were interested to see how well this inventory of the training set morphs could generalize to represent the names that never occurred in the training set. The language models of this morph recognizer were the morph bigram and the morph trigram trained on the name lexicon of the training set, where the morph representation Type I was used. As in the letter recognizers, to come up with the 10-best letter sequences as the recognition results of the letter recognizer, the trigram language model was used. However, we also did the recognition using the morph bigram language model.

The system used the same pre-existing acoustic models trained from a collection of utterances collected from the Jupiter and Pegasus domains as in the letter recognizers. Again, these models would not be the optimal acoustic models for the morph recognizers since we did not train them according to the pronunciation of the proper names.

6.5.2 Performance

The results of morph recognition on the training set and the test set are shown in Tables 6-3 and 6-4 below.

The morph recognizer using the morph representation Type I		
Language model: bigram		
Set	Morph sentence recognition accuracy	Morph recognition accuracy
training set	27.2	41.3
test set	22.4	36.0
Language model: trigram		
training set	37.3	47.9
test set	29.2	40.0

Table 6-3: The morph recognition accuracy of the morph recognizer using morph representation Type I

The morph recognizer using the morph representation Type I		
Language model: bigram		
Set	Phone sentence recognition accuracy	Phone recognition accuracy
training set	33.5	51.1
test set	31.0	47.9
Language model: trigram		
training set	42.4	56.2
test set	35.9	50.8

Table 6-4: The phone recognition accuracy of the morph recognizer using morph representation Type I

From the proposed morph sequences, we were also interested to see how our morphs worked in the letter-to-sound aspect. Thus we mapped the morph sequences proposed from the morph recognizer into the letter sequences and calculated the performance. The letter accuracy of the mapped letter sequences is shown in Table 6-5 below.

The morph recognizer using the morph representation Type I		
Language model: bigram		
Set	Letter sentence recognition accuracy	Letter recognition accuracy
training set	27.7	57.9
test set	22.8	50.4
Language model: trigram		
training set	37.6	62.2
test set	29.4	52.8

Table 6-5: The letter recognition accuracy of the morph recognizer using morph representation Type I

This can be viewed as a sound-to-letter task.

6.6 The morph recognizer using the morph representation Type II

6.6.1 General information

This morph recognizer was created in the same fashion as when the morph representation Type I was utilized. But the vocabulary used contained the 1087 morphs appearing in a different name lexicon of the training set from the one used in Section 6.4. This name lexicon provided the morph representations of all of the names in the training set according to the Type II notation. The morph bigram and the morph trigram language models were trained as well on this Type II name lexicon of the training set.

6.6.2 Performance

The results of morph recognition on the training set and the test set are shown in Tables 6-6 and 6-7 below.

The morph recognizer using the morph representation Type II		
Language model: bigram		
Set	Morph sentence recognition accuracy	Morph recognition accuracy
training set	26.9	41.5
test set	24.8	38.6
Language model: trigram		
training set	36.3	48.5
test set	29.6	43.3

Table 6-6: The morph recognition accuracy of the morph recognizer using morph representation Type II

The morph recognizer using the morph representation Type II		
Language model: bigram		
Set	Phone sentence recognition accuracy	Phone recognition accuracy
training set	31.7	60.8
test set	33.0	57.4
Language model: trigram		
training set	41.4	64.9
test set	38.8	60.5

Table 6-7: The phone recognition accuracy of the morph recognizer using morph representation Type II

Also the letter accuracy of the mapped letter sequences is calculated.

The morph recognizer using the morph representation Type II		
Language model: bigram		
Set	Letter sentence recognition accuracy	Letter recognition accuracy
training set	27.3	57.4
test set	24.9	51.3
Language model: trigram		
training set	36.5	62.0
test set	29.8	54.2

Table 6-8: The letter recognition accuracy of the morph recognizer using morph representation Type II

This can be viewed as a sound-to-letter task.

6.7 Morph recognition results' analysis and comparison

Before discussing the performances of the two morph recognizers, we should note some issues about the differences between them. The two morph recognizers were built by the same procedures. The differences between them were the inventories of morphs used as basic units and the name lexicon used to train their language models. The first recognizer was built using morphs of Type I. Each Type I morph has a unique pronunciation. Thus, in the calculation of phone recognition accuracies, we can utilize a one-to-one mapping from each morph in the proposed morph sequence to its corresponding phone sequence. However, each Type II morph, which was used in the other morph recognizer, was allowed to have more than one possible pronunciations. Thus, we could not apply one-to-one mapping in this case. To calculate the phone recognition accuracy in this case, we defined what it means to be a correct "phone". Each phone was said to be a correct phone when that phone was in the set of allowed phones for the corresponding morph. For example, if the reference morph was "=a+", which was allowed to be pronounced either "ey" or "aa", and whichever phones between "ey" and "aa" was present, it was said to be a correct phone. Because of this notation of correct "phones" in Type II morphs, the morph recognizer with Type II basic units had some advantages in phone recognition accuracies compared to the other morph recognizer. Still, this definition of a correct phone should be acceptable, since, as we have discussed in Chapter 2, this multiple pronunciation case can also happen in actual English words (e.g. "either"). Another factor that might contribute to the performance of the two morph recognizers, apart from the morphs themselves, was the consistency of the transcribed morph representation for each name in the lexicons used to train their language models. However, it was hard to compare the qualities of the transcriptions.

The comparisons between various accuracies of the two morph recognizers are summarized in Table 6-9 below.

	language models	set	recognition error (%)		reduction in error rate (%)
			Type I	Type II	
morph	bigram	training set	58.7	58.5	0.34
		test set	64.0	61.4	4.06
	trigram	training set	52.1	51.5	1.94
		test set	60.0	56.7	5.50
phone	bigram	training set	48.9	39.2	19.83
		test set	52.1	42.6	18.23
	trigram	training set	43.8	35.1	19.86
		test set	49.2	39.5	19.71
letter	bigram	training set	42.1	42.6	-1.19
		test set	49.6	48.7	1.81
	trigram	training set	37.8	38.0	-0.53
		test set	47.2	45.8	2.97

Table 6-9: Comparisons between various accuracies of morph recognizers using morph representation Type I and Type II

% reduction in error rate = (recognition error in Type I – recognition error in Type II) / recognition error in Type I

From the comparisons in Table 6-9, we can see that, for morph and phone recognition accuracies, Type II morphs yielded better accuracies than Type I morphs. Using Type II morphs reduced morph error rates more in the test set than in the training set. This might be because of the size of the overlap between morphs present in the training set and in the test set. For Type II, the overlap size was 91.5%, while it was 86.5% for Type I. The bigger, the size of the overlapping morphs, the better the chance that the morph recognizer can provide correct answers. The phone recognition accuracies improved more than the morph recognition accuracies because of our definition of a correct phone, as mentioned earlier. However, from the letter recognition accuracies point of view, it was not consistent whether Type I or Type II morphs would provide

better letter recognition accuracies. As we can see from the comparisons, the letter recognition accuracies of the training set dropped a little when switching to Type II morphs, while it acted in the opposite way with the test set. Still, in terms of how Type II morphs affected each set of data, the letter recognition accuracies tended to perform the same way as the morph recognition accuracies, i.e. they worked better on the test set than on the training set.

6.8 Chapter summary

There were two types of recognizers involved in this thesis. They were the letter recognizers and the morph recognizers. The letter recognizers were used to handle the spelling input and propose the 10-best lists of the letter sequences, to be passed to TINA. Two letter recognizers were used in the experiments. Both letter recognizers were created with the same acoustic models, which were obtained from the Jupiter and Pegasus domains. Although, both recognizers utilized the letter bigram and the letter trigram language models, these language models were trained on different sets of proper names. The first letter recognizer's language models were trained on a more general and bigger set of proper names. The other letter recognizer's language models were trained on only the proper names appearing in the training set. Thus, the latter was more custom-made to the data in the training set. The morph recognizers were used to propose the morph sequences from the pronunciations of the proper names. There were two morph recognizers used in this thesis. Both of them were created with the same procedure, but with different basic units. Type I morphs were used as the basic units of one morph recognizer while the other used Type II morphs as its units, as discussed in Chapter 2. The performances of the four recognizers in term of the morph recognition accuracy, the phone recognition accuracy and the letter recognition accuracy, whichever applied, were reported and discussed in the chapter.

Chapter 7

Experiments using the morph representation Type I

7.1 Motivation

In earlier chapters, we described how we build various components, including the letter recognizer, the morph recognizer and TINA, the natural language component, and combine them together to form the system where we expect that information in the spelling and pronunciation will help each other. In this chapter we will describe the experiments conducted on the overall system using the morph representation Type I, in which each morph has a unique pronunciation and case sensitivity is utilized. The goal and the procedures are described in the next two sections, and then the results will be presented and analyzed.

7.2 Goals

There are four main objectives in the experiments in this chapter.

1) To see whether and how much the spelling information helps the morph/phone recognition accuracy compared to the recognition accuracy when the morph recognizer is running alone.
2) To see whether and how much the pronunciation information helps the letter recognition accuracy compared to the recognition accuracy when the letter recognizer is running alone.
3) To determine a practical way to utilize the spelling information as an additional constraint to the morph recognizer.
4) To see how crucial the letter recognition accuracy of the letter recognizer is to the overall performance of the whole system.

In order to fulfill these objectives, there are three main performance computations involved. They are:

- morph recognition accuracy : the percentage of the correctly recognized morphs compared to all of the morphs proposed.
- phone recognition accuracy : the percentage of the correctly recognized phones compared to all of the phones proposed.
- letter recognition accuracy : the percentage of the correctly recognized letters compared to all of the letters proposed.

7.3 Procedure

7.3.1 Overview

All of the experiments conducted are based on the integrated system discussed in Chapter 5. Basically, at one end of the system, the name-spelling utterances are fed through the letter recognizer. The letter recognizer proposes a 10-best list of letter sequences, the ten letter sequences most preferred by the letter recognizer. These letter sequences are used as the input to TINA, which is used to probabilistically propose the morph sequence that is best matched to the input letter sequences. The morph inventories and TINA's grammars used in this chapter are Type I. The ways TINA utilizes these letter sequences are different among the different experiments. In the experiments in Section 7.3.3, TINA uses all the information in the ten letter sequences to come up with the corresponding sequences of morphs according to TINA's trained grammars. In the experiment in Section 7.3.4, TINA uses only the spelling information from the letter sequence with the best score. The purpose of using these two different ways to obtain letter information is that we would like to see whether the additional nine letter sequences with lower scores help or degrade the spelling knowledge.

In this system, the number of morph sequences proposed by TINA is ten. And these morph sequences are presented in the form of a finite state transducer (FST). Being in the FST form, these morph sequences can be combined with the language model of the baseline morph recognizer. The language model of the baseline recognizer is also in the form of a finite state transducer, but is created from the lexicon of the training data. This combined FST is then used as

the language model for the morph recognizer to recognize the pronunciation utterance corresponding to the name-spelling utterance from which TINA's FST is created. So, for each utterance, the language model is the combination of the common FST from the lexicon of the training data and the FST specific to the name spelled in the name-spelling utterance.

From the output of the morph recognizer in the integrated system, the morph recognition accuracy is reported. By mapping each morph to its corresponding pronunciation in term of phones, we then have sequences of phones and are able to calculate the phone recognition accuracy. As mentioned in an earlier chapter, the letter information is already encoded in the morphs, so we can extract the letters from the resulting morph sequences. More specifically, we can discard all of the markers, including "=", "+", and "-", and convert all the upper case letters into lower-case to obtain the letter sequences. Consequently, we can also calculate the letter recognition accuracy.

In the experiment in Section 7.3.2, instead of feeding the 10-best list of the proposed letter sequences from the letter recognizer into TINA, we use the correct letter sequence for each name. This lets TINA choose to propose morph decompositions for each name uncorrupted by the letter recognition errors from the letter recognizer. This scenario emulates the situation where we have an ideal letter recognizer, which always has perfect letter recognition accuracy. This experiment partly fulfills the fourth objective in Section 7.2.

Apart from emulating the ideal letter recognizer scenario, in the experiments in Section 7.3.3 and Section 7.3.4, we conduct the experiments with the two letter recognizers described in Chapter 6. The two recognizers have different performances. The first one yields inferior letter recognition accuracy on the training set but superior accuracy on the test set. We conduct the experiments with two letter recognizers in order to see how the letter recognition accuracy influences the overall performance of the integrated system.

In each experiment for the morph recognizer, we tried to conduct experiments with different language models but using the same acoustic models in every experiment. Due to the differences in the qualities of different language models, we had to decide how much weight to give to the acoustic and language model in each case. The method we have used is to vary the weight of the acoustic model compared to the language model, and to choose the weight that maximizes the

performance of the whole system for each case. Our criterion is that the system with the best phone recognition accuracy has the best performance.

The details of each experiment are elaborated in the following sections.

7.3.2 Providing the correct spelling knowledge

In this experiment, we set aside the usage of the letter recognizer but provided the correct letter sequence for each name-spelling utterance. The procedure is illustrated in Figure 7-1 below.

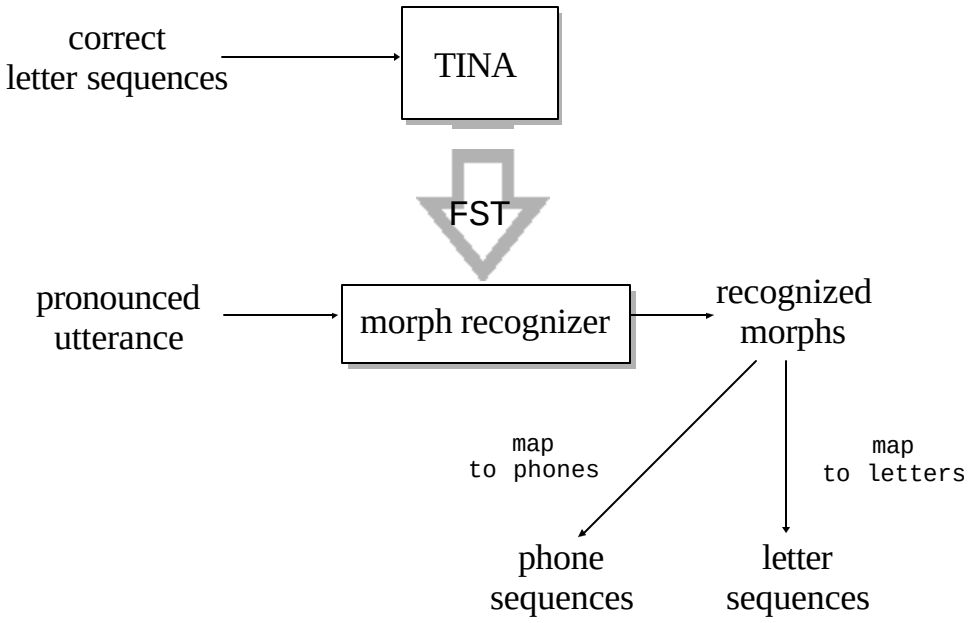


Figure 7-1: Block diagram of the experiment in which the correct letter sequences are provided

The details are as follows:

1) Prepare the list of names in the training set and feed this list of correct answers to TINA. For each name, TINA proposes ten possible morph decompositions. The finite state transducer for each name's proposed morph decompositions is created and stored in order to be

combined with the finite state transducer of the baseline language model of the morph recognizer.

2) For each pronunciation utterance, combine the finite state transducer of the baseline language model of the morph recognizer with the finite state transducer of the corresponding morph sequences proposed by TINA in step (1). Perform the morph recognition task using the combined language model.
3) Calculate the morph recognition accuracy form the results of the morph recognizer. Map each proposed morph to its phone representation and calculate the phone recognition accuracy. Finally, calculate the letter recognition accuracy of the whole system by converting the proposed morph sequences to the sequences of letters.
4) Repeat steps (1) to (3) with the test set.

The results of the experiments in this section are shown in Table 7-9 and Table 7-10 in Section 7.4.

In the experiments in Section 7.3.3 and Section 7.3.4, we extracted the spelling knowledge by using the letter recognizer, but with a different method for constructing the morph decompositions. In both sections, we used the two recognizers introduced in Chapter 6, as mentioned above. For ease of referring to each letter recognizer, we will call the letter recognizer in Section 6.2 the "general" letter recognizer. And we will call the other one, in Section 6.3, the "training-set oriented" letter recognizer.

7.3.3 Utilizing the spelling information from the proposed n-best list from the letter recognizer

The procedure of the experiments in this section is illustrated in Figure 7-2 below.

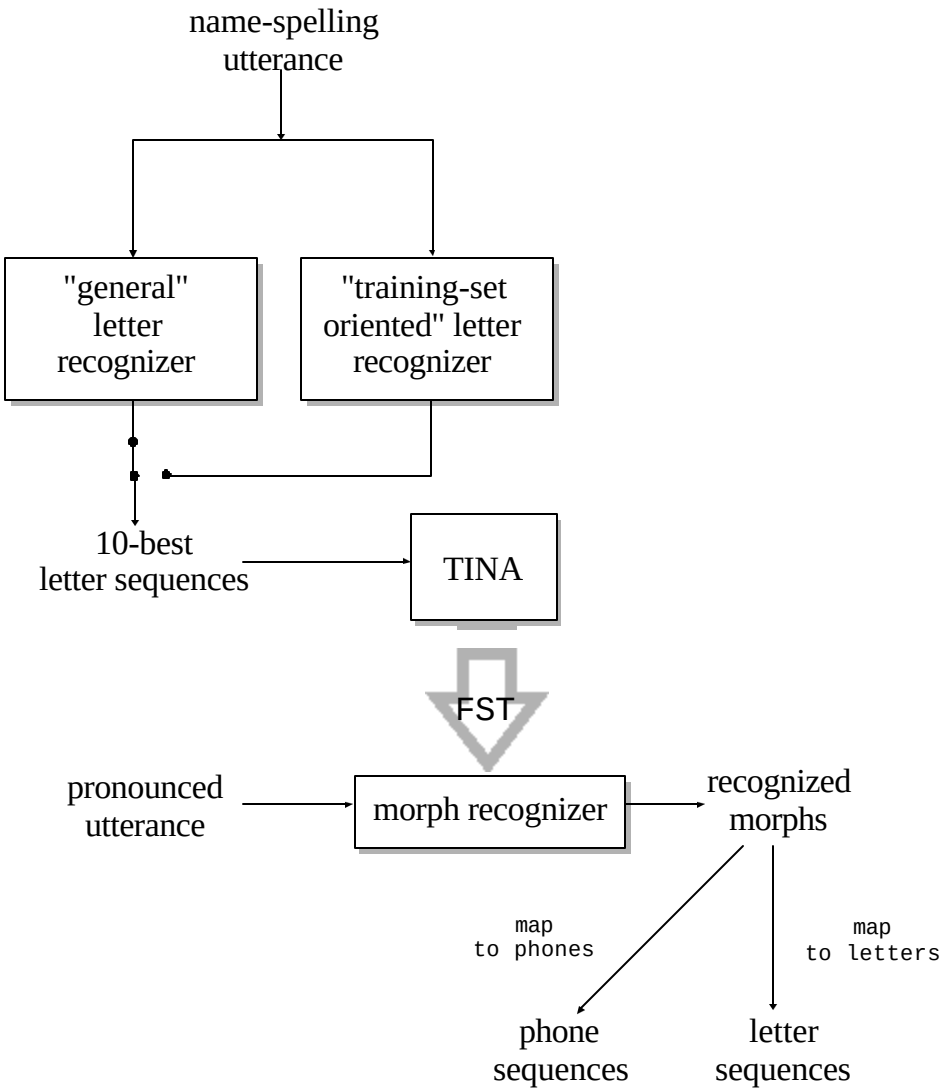


Figure 7-2: Block diagram of the experiment in which the combined FSTs built from the proposed 10-best letter sequences form the letter recognizer

The details are as follows:

1) For each pronunciation utterance in the training set, randomly pick a name-spelling utterance of the same name in the pronounced one.
2) Use one of the letter recognizers to recognize the selected name-spelling utterance. The results of the letter recognition in this step are the 10-best list of the proposed letter sequences.
3) Use TINA to create the finite state representation of this name's morph representation based on the 10-best list of the letter sequences proposed in step (2).
4) Combine the finite state transducer of the morph sequences from TINA with the finite state transducer of the baseline language model of the morph recognizer and use this combined finite state transducer as the language model for recognizing the corresponding pronunciation utterance.
5) Calculate the morph recognition accuracy form the results of the morph recognizer. Map each proposed morph to its phone representation and calculate the phone recognition accuracy. Finally, calculate the letter recognition accuracy of the whole system by converting the proposed morph sequences to the corresponding sequences of letters.
6) Repeat steps (1) to (5) but use the 10-best list of letter sequences from the other letter recognizer.
7) Repeat steps (1) to (7) with the test set.

The result of the experiments in this section are shown in Tables 7-11 to 7-16 in Section 7.4.

7.3.4 Utilizing the spelling information from the most preferred letter sequence from the letter recognizer

The procedure of the experiments in this section is illustrated in Figure 7-3.

The details are as follows:

1) For each pronunciation utterance in the training set, randomly pick a name-spelling utterance of the same name in the pronounced one.
2) Use one of the letter recognizers to recognize the selected name-spelling utterance. The results of the letter recognition in this step are the 10-best list of the proposed letter sequences. *Feed only the top choice*, the letter sequence with the best score, to TINA.
3) Use TINA to create the finite state representation of this name's morph representation based on the top choice of the 10-best list of the letter sequences proposed in step (2).
4) Combine the finite state transducer of the morph sequences from TINA with the finite state transducer of the baseline language model of the morph recognizer, and use this combined finite state transducer as the language model for recognizing the corresponding pronunciation utterance.
5) Calculate the morph recognition accuracy form the results of the morph recognizer. Map each proposed morph to its phone representation and calculate the phone recognition accuracy. Finally, calculate the letter recognition accuracy of the whole system by converting the proposed morph sequences to the sequences of letters.
6) Repeat steps (1) to (5), but use the 10-best list of letter sequences from the *other* letter recognizer.
7) Repeat steps (1) to (7) with the test set.

The results of the experiments in this section are shown in Tables 7-17 to 7-22 in Section 7.4.

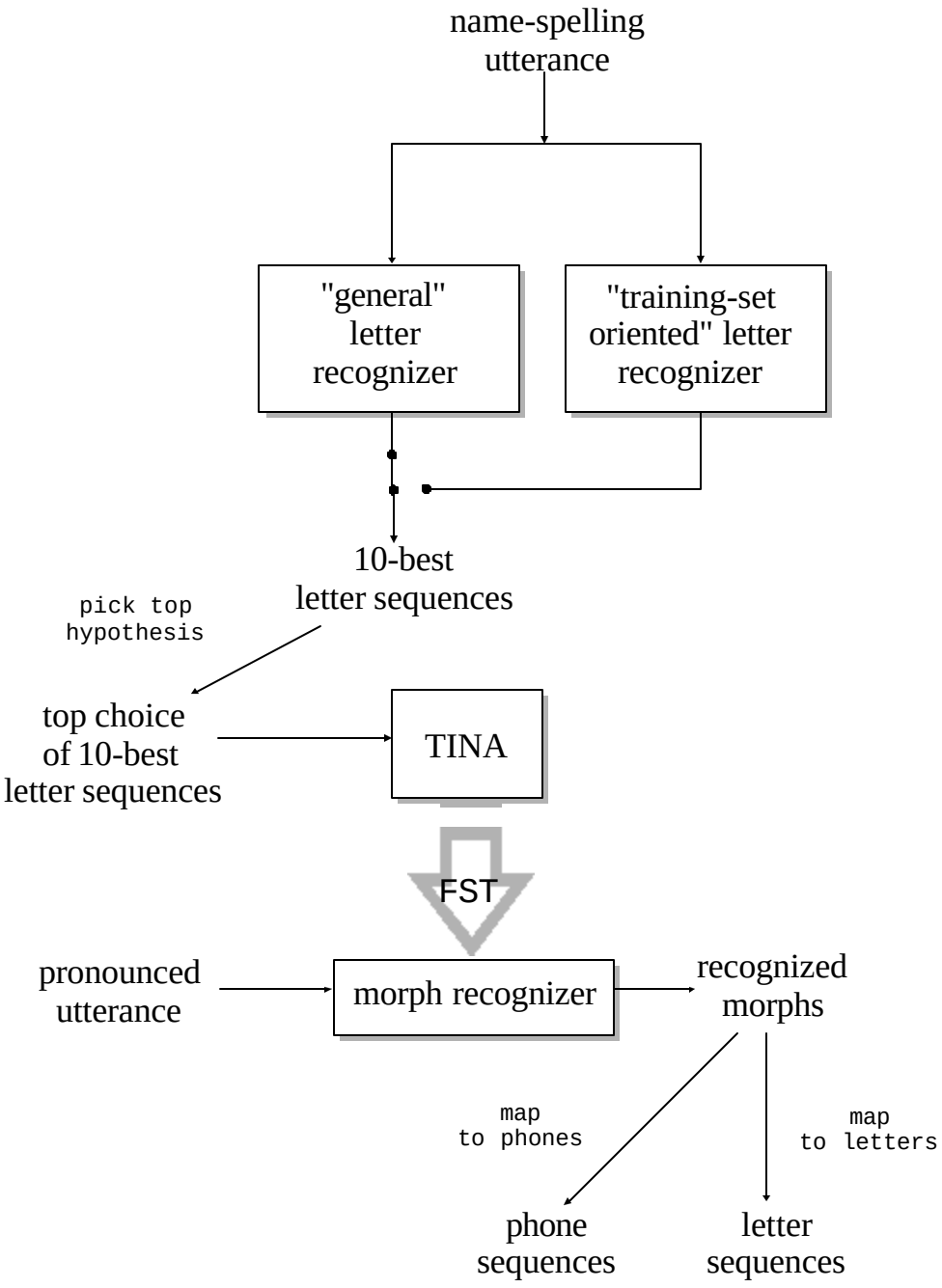


Figure 7-3: Block diagram of the experiment in which the combined FSTs built from the top choice of the proposed 10-best letter sequences form the letter recognizer

7.4 Results and analysis

In this section we will show the results and compare the performances among the different language models for the morph recognizer, including bigram and trigram language models of the baseline version of the morph recognizer in Chapter 5. Tables 7-1 to 7-6 show the performance of the baseline morph recognizers with bigram and trigram language models. And Tables 7-7 and 7-8 show the performance of the two letter recognizers.

Language model: bigram		morph
Set	morph sentence recognition accuracy (%)	Morph recognition Accuracy (%)
Training set	27.2	41.3
Test set	22.4	36.0

Table 7-1: Morph recognition accuracy of the morph recognizer with bigram language model (Type I)

Language model: bigram		phone
Set	phone sentence recognition accuracy (%)	Phone recognition Accuracy (%)
Training set	33.5	51.1
Test set	31.0	47.9

Table 7-2: Phone recognition accuracy of the morph recognizer with bigram language model (Type I)

Language model: bigram		Letter
Set	letter sentence recognition accuracy (%)	Letter recognition accuracy (%)
Training set	27.7	57.9
Test set	22.8	50.4

Table 7-3: Letter recognition accuracy of the morph recognizer with bigram language model (Type I)

Language model: trigram		morph
Set	morph sentence recognition accuracy (%)	morph recognition accuracy (%)
Training set	37.3	47.9
Test set	29.2	40.0

Table 7-4: Morph recognition accuracy of the morph recognizer with trigram language model (Type I)

Language model: trigram		phone
Set	phone sentence recognition accuracy (%)	Phone recognition Accuracy (%)
Training set	42.4	56.2
Test set	35.9	50.8

Table 7-5: Phone recognition accuracy of the morph recognizer with trigram language model (Type I)

Language model: trigram		Letter
Set	letter sentence recognition accuracy (%)	Letter recognition Accuracy (%)
Training set	37.6	62.2
Test set	29.4	52.8

Table 7-6: Letter recognition accuracy of the morph recognizer with trigram language model (Type I)

"general" letter recognizer **language model:** trigram		Letter
Set	Letter sentence recognition accuracy (%)	letter recognition accuracy (%)
Training set	48.1	81.4
Test set	49.7	79.2

Table 7-7: Letter recognition accuracy of the "general" letter recognizer with trigram language model (Type I)

"training-set oriented" letter recognizer **language model:** trigram		Letter
Set	letter sentence recognition accuracy (%)	letter recognition accuracy (%)
Training set	54.1	84.2
Test set	41.4	74.0

Table 7-8: Letter recognition accuracy of the "training-set oriented" letter recognizer with trigram language model (Type I)

From the results of the baseline recognizer, in every case the phone recognition accuracy is higher than the morph recognition accuracy due to the redundancy in morph pronunciation, as mentioned earlier. The trigram language model gives better accuracy than the bigram language model. So, we will use the accuracy obtained from the morph recognizer with trigram language model as the baseline accuracy, which will be compared with the accuracy of the morph recognizers with other language models. When we combine the morph recognizer with perfect knowledge of the spellings, we get huge improvements in the morph and phone recognition accuracy as anticipated. The details of the results are shown in Tables 7-9 and 7-10 below.

Language model: baseline FST + FST from correct answer		morph
Set	morph sentence recognition accuracy (%)	morph recognition accuracy (%)
Training set	65.2	73.1
Test set	69.0	72.8

Table 7-9: Morph recognition accuracy of the system when the combined FSTs between the baseline FST and the FSTs built from the correct letter sequences are used as the language model (Type I)

Language model: baseline FST + FST from correct answer		phone
Set	phone sentence recognition accuracy (%)	phone recognition accuracy (%)
Training set	67.5	81.9
Test set	70.3	82.1

Table 7-10: Phone recognition accuracy of the system when the combined FSTs between the baseline FST and the FSTs built from the correct letter sequences are used as the language model (Type I)

After being provided with perfect letter knowledge, the morph recognizer yields better accuracy in terms of both morph and phone recognition accuracy. On the training set, the morph recognition accuracy in this case is 25.2% (73.1%-47.9%) better than the one in the trigram case. On the test set, it is 32.8% (72.8%-40.0%) better than the one in the trigram case. Speaking in terms of reduction in morph recognition error rate, we have 48.4% reduction in morph recognition error rate on the training set and 54.2% on the test set. Looking at the phone recognition accuracy, we have found that the absolute improvements are 25.7% (81.9%-56.2%) on the training set and 31.3% (82.1%-50.8%) on the test set. The corresponding rates of reduction in phone recognition error are 58.7% and 63.6% on the training set and the test set respectively. The results turned out as we expected. One of the interesting points is that, if we look at the improvements on the training set and the test set, the incorporation of perfect letter knowledge did improve both the morph and phone recognition accuracy more on the test set than on the training set. The trigram was built upon the names in the training set, so it is expected that the recognition accuracy is higher on the training set. However, when we change the language model from trigram to FST, composed from the FST from the lexicon of the training set and the FST constructed from the correct letter sequence by TINA, the system seems to perform equally well on both data sets. And the phone recognition accuracy is even better on the test set. Even though the baseline FST was constructed from the data in the test set, as we can see from the recognition results, the FSTs of the morph representation proposed by TINA based on the correct letter sequences seems to have an impact on the baseline FST at some amount. So the whole system does not favor the data in the training set as much as the morph recognizer running alone. The differences of the morph and phone recognition accuracy between the two sets are only 0.3% in both cases.

Next, we used TINA to propose the FSTs of the morph representation of the 10-best list of the letter sequences from two letter recognizers and combined the proposed FSTs with the baseline FST. The results are shown in Tables 7-11 to 7-16 below.

Language model: baseline FST + FST from 10-best list of letter seq. **Letter recognizer:** "general"		Morph
Set	Morph sentence recognition accuracy (%)	morph recognition accuracy (%)
Training set	24.8	38.5
Test set	20.0	33.7

Table 7-11: Morph recognition accuracy of the system when the combined FSTs between the baseline FSTs and the FSTs built from the 10-best letter sequences, proposed by the "general" letter recognizer, are used as the language model (Type I)

Language model: baseline FST + FST from 10-best list of letter seq. **Letter recognizer:** "general"		phone
Set	phone sentence recognition accuracy (%)	phone recognition accuracy (%)
Training set	28.0	55.0
Test set	23.4	48.8

Table 7-12: Phone recognition accuracy of the system when the combined FSTs between the baseline FSTs and the FSTs built from the 10-best letter sequences, proposed by the "general" letter recognizer, are used as the language model (Type I)

Language model: baseline FST + FST from 10-best list of letter seq. **Letter recognizer:** "general"		letter
Set	letter sentence recognition accuracy (%)	letter recognition accuracy (%)
Training set	39.3	80.0
Test set	33.8	74.8

Table 7-13: Letter recognition accuracy of the system when the combined FSTs between the baseline FSTs and the FSTs built from the 10-best letter sequences, proposed by the "general" letter recognizer, are used as the language model (Type I)

Language model: baseline FST + FST from 10-best list of letter seq. **Letter recognizer:** "training-set oriented"		morph
Set	morph sentence recognition accuracy (%)	morph recognition accuracy (%)
Training set	27.2	44.7
Test set	19.0	35.5

Table 7-14: Morph recognition accuracy of the system when the combined FSTs between the baseline FSTs and the FSTs built from the 10-best letter sequences, proposed by the "training-set oriented" letter recognizer, are used as the language model (Type I)

Language model: baseline FST + FST from 10-best list of letter seq. **Letter recognizer:** "training-set oriented"		phone
Set	phone sentence recognition accuracy (%)	phone recognition accuracy (%)
Training set	31.4	58.9
Test set	23.2	48.6

Table 7-15: Phone recognition accuracy of the system when the combined FSTs between the baseline FSTs and the FSTs built from the 10-best letter sequences, proposed by the "training-set oriented" letter recognizer, are used as the language model (Type I)

Language model: baseline FST + FST from 10-best list of letter seq. **Letter recognizer:** "training-set oriented"		letter
Set	letter sentence recognition accuracy (%)	Letter recognition accuracy (%)
Training set	46.0	83.2
Test set	33.0	71.6

Table 7-16: Letter recognition accuracy of the system when the combined FSTs between the baseline FSTs and the FSTs built from the 10-best letter sequences, proposed by the "training-set oriented" letter recognizer, are used as the language model (Type I)

From Tables 7-11 and 7-12, we see that both the morph recognition accuracy and the phone recognition accuracy are worse in these cases than the recognition accuracy of the baseline morph recognizer with a trigram language model. Both cases made use of the "general" letter recognizer, which has letter sentence recognition accuracy and letter recognition accuracy of 48.1% and 81.4% respectively on the training set. On the test set, the letter sentence recognition accuracy is 41.4% and the letter recognition accuracy is 79.2%. From the degradation of the accuracy after utilizing the spelling knowledge through the "general" letter recognizer, we can infer that, with these levels of accuracy, the spelling knowledge hurts the performance of the overall system. In other words, we can say that this letter information is too noisy to be useful for the morph recognizer.

By using the "training-set oriented" letter recognizer, which has better letter recognition accuracy on the training set than the one of the "general" letter recognizer, the phone recognition accuracy of the training set increases. Although the morph recognition accuracy is still worse, the phone recognition accuracy becomes 2.7% (58.9%-56.2%) higher than the one of the baseline morph recognizer. However, the morph and phone recognition accuracy of the test set are still worse, since the recognized letter sequences from the letter recognizer performed on the test set are still too noisy to be useful to the morph recognizer. The detailed numbers mentioned in this paragraph are shown in Tables 7-14 and 7-15.

Language model: baseline FST + FST from top choice letter seq. **Letter recognizer:** "general"		morph
Set	morph sentence recognition accuracy (%)	Morph recognition Accuracy (%)
Training set	35.3	45.3
Test set	35.6	43.8

Table 7-17: Morph recognition accuracy of the system when the combined FSTs between the baseline FSTs and the FSTs built from the top choice of the 10-best letter sequences, proposed by the "general" letter recognizer, are used as the language model (Type I)

Language model: baseline FST + FST from top choice letter seq. **Letter recognizer:** "general"		phone
Set	phone sentence recognition accuracy (%)	Phone recognition Accuracy (%)
Training set	38.3	59.7
Test set	37.1	55.8

Table 7-18: Phone recognition accuracy of the system when the combined FSTs between the baseline FSTs and the FSTs built from the top choice of the 10-best letter sequences, proposed by the "general" letter recognizer, are used as the language model (Type I)

Language model: baseline FST + FST from top choice letter seq. **Letter recognizer:** "general"		Letter
Set	letter sentence recognition accuracy (%)	letter recognition accuracy (%)
Training set	49.3	82.0
Test set	49.0	79.3

Table 7-19: Letter recognition accuracy of the system when the combined FSTs between the baseline FSTs and the FSTs built from the top choice of the 10-best letter sequences, proposed by the "general" letter recognizer, are used as the language model (Type I)

Language model: baseline FST + FST from top choice letter seq. **Letter recognizer:** "training-set oriented"		morph
Set	morph sentence recognition accuracy (%)	morph recognition accuracy (%)
Training set	32.3	44.8
Test set	30.6	40.0

Table 7-20: Morph recognition accuracy of the system when the combined FSTs between the baseline FSTs and the FSTs built from the top choice of the 10-best letter sequences, proposed by the "training-set oriented" letter recognizer, are used as the language model (Type I)

Language model: baseline FST + FST from top choice letter seq. **Letter recognizer:** "training-set oriented"		phone
Set	phone sentence recognition accuracy (%)	phone recognition accuracy (%)
Training set	35.9	60.4
Test set	33.0	53.7

Table 7-21: Phone recognition accuracy of the system when the combined FSTs between the baseline FSTs and the FSTs built from the top choice of the 10-best letter sequences, proposed by the "training-set oriented" letter recognizer, are used as the language model (Type I)

Language model: baseline FST + FST from top choice letter seq. **Letter recognizer:** "training-set oriented"		Letter
Set	letter sentence recognition accuracy (%)	letter recognition accuracy (%)
Training set	55.0	83.4
Test set	44.8	74.6

Table 7-22: Letter recognition accuracy of the system when the combined FSTs between the baseline FSTs and the FSTs built from the top choice of the 10-best letter sequences, proposed by the "training-set oriented" letter recognizer, are used as the language model (Type I)

Tables 7-17, 7-18, 7-20 and 7-21 show the results when we took only the letter sequence with the best score to be used as TINA's input instead of using all of the 10-best choices. The results are promising in terms of phone recognition accuracy. By utilizing the "general" letter recognizer and this way of creating the FSTs, the phone recognition accuracy increases by 3.5% (59.7%-56.2%) on the training set and 5.0% (55.8%-50.8%) on the test set. The corresponding rates of reduction in phone recognition accuracy are 8.0% and 10.2% on the training set and the test set respectively. And by using the "training-set oriented" letter recognizer, the phone recognition accuracy increases by 4.2% (60.4%-56.2%) on the training set and 2.9% (53.7%-50.8%) on the

test set. The corresponding rates of reduction in phone recognition accuracy are 9.6% and 5.9% on the training set and the test set respectively.

From the results we have, we can see, as expected, that the higher the letter recognition accuracy, the more the spelling knowledge can help the performance of the phone recognition task. With low quality letter information, the spelling knowledge even degrades the performance of the whole system. However, at acceptable letter recognition accuracy, the information about the spelling yields better phone recognition accuracy.

Comparing the letter recognition accuracy in various experiments, we have found that in most cases the knowledge of the pronunciation does not help the letter recognition accuracy, except for one case where the incorporation of the pronunciation knowledge improves the letter recognition accuracy a little on both the training set and the test set. It is the case when the FST, combined with the baseline language model, comes from the top choice of the 10-best letter sequences from the "general" letter recognizer. The results for this case are shown in Table 7-19. On the training set the letter recognition accuracy is 0.6% (82.0-81.4%) better than the accuracy of the "general" letter recognizer and 0.1% (79.3%-79.2%) on the test set. The corresponding rates of reduction in letter recognition accuracy are 3.2% and 0.5% on the training set and the test set respectively.

7.5 Chapter Summary

This chapter describes the goal, procedure and results of the experiments conducted using the morph representation Type I. The goals of this set of experiments were to find out how the spelling and pronunciation knowledge can help each other in recognizing both the name-spelling and the pronounced utterances, to study the ways to combine them together and to see how crucial the quality of the letter recognizer is to the overall recognition accuracy. The experiments were done by varying the language models of the morph recognizer. The language models used were bigram, trigram and the combined FSTs. The combined FSTs were created by combining the baseline FST, which was created from the lexicon of the training data, with three types of FSTs. The first one was the FST proposed from TINA when provided with the correct letter sequence for each name. The next was the FST proposed from TINA when provided with the 10-best letter sequences from the letter recognizer. And the last one was the FST proposed form

TINA when provided with the top choice of the 10-best letter sequences form the letter recognizer. Two letter recognizers were used. One gave higher accuracy on the training set than the other, while, on the other hand, the other gave higher accuracy on the test set. The result shows that the letter knowledge can help the phone recognition task, especially when the correct letter sequences were provided. With both letter recognizers, using the top choice of the 10-best letter sequences was found to be more useful to the phone recognition task than using all of the 10-best letter sequences. However, the pronunciation knowledge did not help the letter recognition task in most cases.

Chapter 8

Experiments using the morph representation Type II

8.1 Motivation

In Chapter 7, we described the experiments done based on the morph representation Type I, in which each morph has a unique pronunciation. In this chapter, we will describe the context whose outline is similar to the one in Chapter 7, but the morph representation Type II has been exploited.[3]

8.2 Goals

The objective of this set of experiments is to study how alternative sets of morphs affect the performance of our integrated system. We would like to see whether using the morph representation Type II will improve or degrade the recognition accuracy. The criteria used in the experiments in this chapter are similar to the ones in the experiments in Chapter 7, which are morph recognition accuracy, phone recognition accuracy and letter recognition accuracy.

8.3 Procedure

The procedures of the experiments in this chapter are similar to those in Chapter 7. All of the experiments conducted are based on the same integrated system. The name-spelling utterances are fed through the letter recognizer. The letter recognizer proposes 10-best lists of letter

[3] See Chapter 2 for definitions of Type I and Type II morph representations

sequences, the ten letter sequences most preferred by the letter recognizer. These letter sequences are used as the input of TINA, which is used to probabilistically propose the morph sequence that is best matched to the input letter sequences. The difference between the experiments in this chapter and the ones in Chapter 7 is that, in Chapter 7 the morph inventories and TINA's grammars are Type I, but the morph inventories and TINA's grammars of Type II were used in the experiments in this chapter.

As in Chapter 7, there are two ways for TINA to use the letter sequences proposed by the letter recognizer. In the first one, TINA uses all of the information provided in all of the 10-best letter sequences proposed. In the other, TINA discards the letter information in the nine proposed letter sequences with inferior scores and only makes use of the top choice of the 10-best letter sequences. Also, instead of extracting the letter knowledge from the name-spelling utterances, we provide the correct letter sequences for all of the pronunciation utterances and decompose them into the sequences of morphs according to the morph inventories and TINA's grammars for Type II.

The morph recognition accuracy, the phone recognition accuracy and the letter recognition accuracy were calculated and compared with those of the baseline recognizers, which used a bigram and trigram as their language models. However, the morph recognizer used in this chapter is different from the morph recognizer used in Chapter 7, since we have changed the morph inventories. The morph recognizer in this chapter was trained on the data in the same training set. The lexicon of the training set is different in the aspect of decomposing the names into their corresponding morph representation, due to the different morph inventories.

The details of the results and also the new baseline performance are shown in the next section.

8.4 Results and analysis

The performances of the baseline morph recognizer based on morph representation Type II are shown in Tables 8-1 to 8-6 below. And, for ease of comparing, the letter recognition accuracy of the two letter recognizers, the "general" letter recognizer and the "training-set oriented" letter recognizer, are repeated again in Tables 8-7 and 8-8.

Language model: bigram		morph
Set	morph sentence recognition accuracy (%)	Morph recognition Accuracy (%)
Training set	26.9	41.5
Test set	24.8	38.6

Table 8-1: Morph recognition accuracy of the morph recognizer with bigram language model (Type II)

Language model: bigram		phone
Set	phone sentence recognition accuracy (%)	Phone recognition Accuracy (%)
Training set	31.7	60.8
Test set	33.0	57.4

Table 8-2: Phone recognition accuracy of the morph recognizer with bigram language model (Type II)

Language model: bigram		letter
Set	Letter sentence recognition accuracy (%)	letter recognition accuracy (%)
Training set	27.3	57.4
Test set	24.9	51.3

Table 8-3: Letter recognition accuracy of the morph recognizer with bigram language model (Type II)

Language model: trigram		morph
Set	Morph sentence recognition accuracy (%)	Morph recognition Accuracy (%)
Training set	36.3	48.5
Test set	29.6	43.3

Table 8-4: Morph recognition accuracy of the morph recognizer with trigram language model (Type II)

Language model: trigram		phone
Set	Phone sentence recognition accuracy (%)	Phone recognition Accuracy (%)
Training set	41.4	64.9
Test set	38.8	60.5

Table 8-5: Phone recognition accuracy of the morph recognizer with trigram language model (Type II)

Language model: trigram		letter
Set	letter sentence recognition accuracy (%)	letter recognition accuracy (%)
Training set	36.5	62.0
Test set	29.8	54.2

Table 8-6: Letter recognition accuracy of the morph recognizer with trigram language model (Type II)

"general" letter recognizer **language model:** trigram		Letter
Set	letter sentence recognition accuracy (%)	letter recognition accuracy (%)
Training set	48.1	81.4
Test set	49.7	79.2

Table 8-7: Letter recognition accuracy of the "general" letter recognizer with trigram language model

"training-set oriented" letter recognizer **language model:** trigram		Letter
Set	Letter sentence recognition accuracy (%)	letter recognition accuracy (%)
Training set	54.1	84.2
Test set	41.4	74.0

Table 8-8: Letter recognition accuracy of the "training-set oriented" letter recognizer with trigram language model

As before, we will use the accuracy obtained from the morph recognizer and the letter recognizer with trigram language model as the baseline accuracy, which will be compared with the accuracy of the integrated system with other language models. When we combined the morph recognizer with perfect knowledge of the spellings, we got huge improvements in the morph and phone recognition accuracy. The details of the results are shown in Tables 8-9 and 8-10 below.

Language model: baseline FST + FST from correct answer		morph
Set	Morph sentence recognition Accuracy (%)	Morph recognition Accuracy (%)
Training set	92.0	93.7
Test set	85.5	87.6

Table 8-9: Morph recognition accuracy of the system when the combined FSTs between the baseline FST and the FSTs built from the correct letter sequences are used as the language model (Type II)

Language model: baseline FST + FST from correct answer		Phone
Set	Phone sentence recognition accuracy (%)	Phone recognition Accuracy (%)
Training set	92.8	97.3
Test set	89.5	94.8

Table 8-10: Phone recognition accuracy of the system when the combined FSTs between the baseline FST and the FSTs built from the correct letter sequences are used as the language model (Type II)

After being provided with perfect letter knowledge, the morph recognizer yields plausible accuracy in term of both morph and phone recognition accuracy, especially the latter which is over ninety percent. On the training set, the morph recognition accuracy in this case is 45.2% (93.7%-48.5%) better than the one in the trigram case. On the test set, it is 44.3% (87.6%-43.3%) better than the one in the trigram case. The corresponding rates of reduction in morph recognition accuracy are 87.8% on the training set and 78.1% on the test set. Looking at the phone recognition accuracy, we have found that the improvements are 32.4% (97.3%-64.9%) on the training set and 34.3% (94.8%-60.5%) on the test set. The corresponding rates of reduction in phone recognition accuracy are 92.3% and 86.8% on the training and the test set respectively. The results turned out as we expected and the morph recognition accuracy and the phone recognition accuracy are better than the ones using morph representation Type I on both the training set and the test set. On the training set, the morph recognition accuracy is 20.6% (93.7%-73.1%) higher and the phone recognition accuracy is 15.4% (97.3%-81.9%) higher than the ones using the other

type of morph representation. And, on the test set, the morph recognition accuracy is 14.8% (87.6%-72.8%) higher and the phone recognition accuracy is 12.7% (94.8%-82.1%) higher than the ones using the morph representation Type I. In terms of the reduction in error rate, using Type II morphs instead of Type I provides 76.6% and 54.4% reduction in morph recognition errors on the training set and the test set respectively, while it also provides 85.1% and 70.9% reduction in phone recognition errors on the training set and the test set respectively.

Next, we used TINA to propose the FSTs of the morph representation of the 10-best list of the letter sequences from two letter recognizers and combined the proposed FSTs with the baseline FST. The results are shown in Tables 8-11 to 8-16 below.

Language model: baseline FST + FST from 10-best list of letter seq. **Letter recognizer:** "general"		`morph`
Set	Morph sentence recognition accuracy (%)	Morph recognition Accuracy (%)
Training set	46.4	54.1
Test set	37.0	48.0

Table 8-11: Morph recognition accuracy of the system when the combined FSTs between the baseline FSTs and the FSTs built from the 10-best letter sequences, proposed by the "general" letter recognizer, are used as the language model (Type II)

Language model: baseline FST + FST from 10-best list of letter seq. **Letter recognizer:** "general"		`phone`
Set	Phone sentence recognition Accuracy (%)	Phone recognition Accuracy (%)
Training set	49.3	72.5
Test set	41.7	67.1

Table 8-12: Phone recognition accuracy of the system when the combined FSTs between the baseline FSTs and the FSTs built from the 10-best letter sequences, proposed by the "general" letter recognizer, are used as the language model (Type II)

Language model: baseline FST + FST from 10-best list of letter seq. **Letter recognizer:** "general"		letter
Set	letter sentence recognition accuracy (%)	Letter recognition Accuracy (%)
Training set	49.9	81.4
Test set	40.8	75.5

Table 8-13: Letter recognition accuracy of the system when the combined FSTs between the baseline FSTs and the FSTs built from the 10-best letter sequences, proposed by the "general" letter recognizer, are used as the language model (Type II)

Language model: baseline FST + FST from 10-best list of letter seq. **Letter recognizer:** "training-set oriented"		morph
Set	morph sentence recognition accuracy (%)	Morph recognition Accuracy (%)
Training set	44.2	54.7
Test set	30.7	45.5

Table 8-14: Morph recognition accuracy of the system when the combined FSTs between the baseline FSTs and the FSTs built from the 10-best letter sequences, proposed by the "training-set oriented" letter recognizer, are used as the language model (Type II)

Language model: baseline FST + FST from 10-best list of letter seq. **Letter recognizer:** "training-set oriented"		phone
Set	phone sentence recognition accuracy (%)	Phone recognition Accuracy (%)
Training set	47.1	72.8
Test set	37.3	63.8

Table 8-15: Phone recognition accuracy of the system when the combined FSTs between the baseline FSTs and the FSTs built from the 10-best letter sequences, proposed by the "training-set oriented" letter recognizer, are used as the language model (Type II)

Language model: baseline FST + FST from 10-best list of letter seq. **Letter recognizer:** "training-set oriented"		letter
Set	letter sentence recognition accuracy (%)	Letter recognition Accuracy (%)
Training set	48.2	82.1
Test set	32.8	71.3

Table 8-16: Letter recognition accuracy of the system when the combined FSTs between the baseline FSTs and the FSTs built from the 10-best letter sequences, proposed by the "training-set oriented" letter recognizer, are used as the language model (Type II)

From Tables 8-11 and 8-12, we can see that all of the morph recognition and phone recognition accuracies have improved from their corresponding accuracies in the baseline recognition cases. On the training set, the morph recognition accuracy is 5.6% (54.1%-48.5%) higher and the phone recognition accuracy is 7.6% (72.5%-64.9%) higher than the morph and phone recognition accuracy of the baseline morph recognizer with trigram language model. The corresponding rates of reduction in morph recognition errors and phone recognition errors are 10.9% and 21.7% respectively. Also, on the test set, the phone recognition accuracy and the morph recognition accuracy are 4.7% (48.0%-43.3%) and 6.6% (67.1%-60.5%) higher than the ones of the baseline morph recognizer respectively. The corresponding rates of reduction in morph recognition errors and phone recognition errors are 8.3% and 16.7% respectively.

By using the "training-set oriented" letter recognizer instead of the "general" letter recognizer, the phone recognition accuracy of the training set increases. In Tables 8-14 and 8-15, all of the morph recognition accuracies and the phone recognition accuracies shown are better than the ones of the baseline recognizer in their corresponding cases. On the training set, the morph recognition accuracy is 6.2% (54.7%-48.5%) higher and the phone recognition accuracy is 7.9% (72.8%-64.9%) higher than the morph and phone recognition accuracy of the baseline morph recognizer with the trigram language model. The corresponding rates of reduction in morph recognition errors and phone recognition errors are 12.0% and 22.5% respectively. Also, on the test set, the phone recognition accuracy and the morph recognition accuracy are 2.2% (45.5%-43.3%) and 3.3% (63.8%-60.5%) higher than the ones of the baseline morph recognizer respectively. The

corresponding rates of reduction in morph recognition errors and phone recognition errors are 3.9% and 8.4% respectively.

Up to this point, we can compare the improvement in the morph recognition accuracy and the phone recognition accuracy gained by switching the type of morph representation and TINA's grammars from Type I to Type II. However, it is not perfectly right to say that the morph representation Type II is a better means to convey the information from the spelling knowledge to help the pronunciation recognition task. There are other variables apart from morph inventories that are different between the experiments in Chapter 7 and Chapter 8. These variables are TINA's grammars and how we train them. However, it is reasonable to say that the overall method we used in the experiments in this chapter appears to be more usable than the one in Chapter 7. Supporting this statement are the data shown in Tables 8-11 to 8-12 and Tables 8-14 to 8-15. In all of these cases, the letter knowledge helps improving the morph and phone recognition accuracy when using morph representation Type II. While using morph representation Type I, we can see the improvement in only one case, as mentioned in Chapter 7. Furthermore, the percentage of improvement of the recognition accuracy in this one case is lower than the one when morph representation Type II is used in the similar case.

Next, we stepped from utilizing all of the spelling knowledge from the 10-best letter sequences to utilizing only the spelling knowledge from the top choice of the 10-best letter sequences. The results are shown in Tables 8-17 to 8-22 below.

Language model: baseline FST + FST from top choice letter seq. **Letter recognizer:** "general"		`morph`
Set	Morph sentence recognition Accuracy (%)	Morph recognition Accuracy (%)
Training set	47.7	56.0
Test set	44.8	54.3

Table 8-17: Morph recognition accuracy of the system when the combined FSTs between the baseline FSTs and the FSTs built from the top choice of the 10-best letter sequences, proposed by the "general" letter recognizer, are used as the language model (Type II)

Language model: baseline FST + FST from top choice letter seq. **Letter recognizer:** "general"		phone
Set	phone sentence recognition accuracy (%)	Phone recognition Accuracy (%)
Training set	49.1	73.4
Test set	40.5	71.6

Table 8-18: Phone recognition accuracy of the system when the combined FSTs between the baseline FSTs and the FSTs built from the top choice of the 10-best letter sequences, proposed by the "general" letter recognizer, are used as the language model (Type II)

Language model: baseline FST + FST from top choice letter seq. **Letter recognizer:** "general"		Letter
Set	letter sentence recognition accuracy (%)	Letter recognition Accuracy (%)
Training set	51.0	82.5
Test set	51.3	80.4

Table 8-19: Letter recognition accuracy of the system when the combined FSTs between the baseline FSTs and the FSTs built from the top choice of the 10-best letter sequences, proposed by the "general" letter recognizer, are used as the language model (Type II)

Language model: baseline FST + FST from top choice letter seq. **Letter recognizer:** "training-set oriented"		morph
Set	Morph sentence recognition accuracy (%)	Morph recognition Accuracy (%)
Training set	54.2	64.0
Test set	42.3	52.8

Table 8-20: Morph recognition accuracy of the system when the combined FSTs between the baseline FSTs and the FSTs built from the top choice of the 10-best letter sequences, proposed by the "training-set oriented" letter recognizer, are used as the language model (Type II)

Language model: baseline FST + FST from top choice letter seq. **Letter recognizer:** "training-set oriented"		phone
Set	phone sentence recognition accuracy (%)	Phone recognition Accuracy (%)
Training set	55.3	77.6
Test set	44.5	68.1

Table 8-21: Phone recognition accuracy of the system when the combined FSTs between the baseline FSTs and the FSTs built from the top choice of the 10-best letter sequences, proposed by the "training-set oriented" letter recognizer, are used as the language model (Type II)

Language model: baseline FST + FST from top choice letter seq. **Letter recognizer:** "training-set oriented"		Letter
Set	letter sentence recognition accuracy (%)	Letter recognition Accuracy (%)
Training set	57.8	85.4
Test set	46.5	76.7

Table 8-22: Letter recognition accuracy of the system when the combined FSTs between the baseline FSTs and the FSTs built from the top choice of the 10-best letter sequences, proposed by the "training-set oriented" letter recognizer, are used as the language model (Type II)

Since using the top choice in creating the FST to be combined with the baseline FST had been found, in the morph representation Type I case, to be more useful to the pronunciation recognition task than using all of the 10-best letter sequences, we expected the same consequence for the morph representation Type II. From Tables 8-17, 8-18, 8-20 and 8-21, we can see that the result turned out to be as we expected. All of the morph recognition accuracies and the phone recognition accuracies are better than the 10-best case. By utilizing the "general" letter recognizer and creating the FSTs from the top choice of the 10-best letter sequences, the morph and phone recognition accuracy increase from the baseline accuracy by 7.5% (56.0%-48.5%) and 8.5%

(73.4%-64.9%) on the training set and 11.0% (54.3%-43.3%) and 11.1% (71.6%-60.5%) on the test set. The corresponding rates of reduction in morph and phone recognition errors are 14.5% and 24.2% on the training set and 19.4% and 28.1% on the test set. By using the "training-set oriented" letter recognizer, the morph and phone recognition accuracy increases from the baseline accuracy by 15.5% (64.0%-48.5%) and 12.7% (77.6%-64.9%) on the training set and 9.5% (52.8%-43.3%) and 7.6% (68.1%-60.5%) on the test set. The corresponding rates of reduction in morph and phone recognition errors are 30.1% and 36.2% on the training set and 16.8% and 19.2% on the test set.

The results we have found here are consistent with the results in Chapter 7 that the higher the letter recognition accuracy, the more the spelling knowledge can help the performance of the phone recognition task. However, in this system, the overall improvement, gained from incorporating the spelling knowledge from the name-spelling utterances, is greater than the improvement gained in the morph representation Type I case. At a letter recognition accuracy of the letter recognizer that is too low for the morph recognizer of the system in Chapter 7 to gain any advantages from the name-spelling utterance, there is improvement in the morph and phone recognition accuracy here. So, this system is more robust to the noise contaminating the quality of the proposed letter sequences.

Converting every resulting morph sequence to the corresponding letter sequences by discarding all of the markers, "-", "+" and "=", we can then calculate the letter recognition accuracy of the integrated system and compare with the letter recognition accuracy of the baseline letter recognizer. From the results in Tables 8-19 and 8-22, we have found that the letter sequences obtained from the morph sequences proposed by the system, using the top choice of 10-best letter sequences to create the combined FSTs, had better accuracy than the letter sequences proposed from the baseline letter recognizer with trigram language models alone. By using the "general" letter recognizer, the improvements are 0.7% (82.1%-81.4%) in the training set and 1.2% (80.4%-79.2%) in the test set. The corresponding rates of reduction in letter recognition errors are 3.8% and 5.8% on the training set and the test set respectively. And by using the "training-set oriented" letter recognizer, the improvements are 1.2% (85.4%-84.2%) in the training set and 2.7% (76.7%-74.0%) in the test set. The corresponding rates of reduction in letter recognition errors are 7.6% and 10.4% on the training set and the test set respectively.

8.5 Chapter Summary

The goal of the experiments done in this chapter was to find out how the spelling and pronunciation knowledge can help each other in recognizing both the name-spelling and the pronounced utterances by using a different inventory of morphs and TINA's grammar from the ones we used in Chapter 7. The experiments were done by varying the language models of the morph recognizer in the same fashion as conducted in Chapter 7 but under the notation of morph representation Type II. Since we made use of a different morph inventory, the baseline morph recognizer was different from the one in Chapter 7. This baseline morph recognizer's language models were created from the lexicon of the training data using the morph representation Type II. The same two letter recognizers as in Chapter 7 were also used in this chapter. The result shows that the letter knowledge can help the morph and phone recognition task, especially when the correct letter sequences were provided. With both letter recognizers, using the top choice of the 10-best letter sequences was found to be more useful to the phone recognition task than using all of the 10-best letter sequences. Still, both methods did improve the baseline morph and phone accuracy more than they did in the morph representation Type I case. Unlike the case of Type I, the letter knowledge from the name-spelling utterances in this case increased the letter recognition accuracy when the combined FSTs were created from the top choice of the 10-best letter sequences proposed from the baseline letter recognizer.

Chapter 9

Summary and Future Work

9.1 Thesis summary

In this thesis, we worked on designing the framework for a proper noun recognition system, in which both the information in the spelling and the information in the pronunciation were used to assist the system to come up with the resulting letter and phone sequence of the input name. The idea of subword modeling was utilized in the framework. The motivation behind our joint recognition framework is that we try to imitate a human's process of understanding proper names, especially unfamiliar ones. Utilizing mutually supportive information between the spelling and pronunciation of an unfamiliar name helps a person to gain more understanding about that name. Thus the goal of our thesis was to determine methods that practically allow both pieces of information to be incorporated in both the spelling and the pronunciation recognition tasks.

We define a set of "morph" sub-units to represent proper names. These syllable-sized units capture both orthographic and phonemic information of the names they represent. Our morphs are categorized into six groups: prefix, onset, rhyme, uroot, dsuf and isuf. Certain combinations of these morph categories are used to represent proper names. Such combinations are defined by a specific word decomposition rule designed for proper names. Each morph consists of a series of characters and special markers, which are attached to the characters in order to determine its category. Two types of morph representation, called Type I and Type II were used in this thesis. In Type I morph representation, each morph has only one possible pronunciation and capital letters are used to distinguish between the morphs that represent the same spellings but are pronounced differently. The other type of morphs does not utilize upper-case letter. Consequently, each Type II morph is allowed to have more than one possible pronunciation.

TINA, a probabilistic natural language framework, was deployed to perform the task of subword modeling of proper names in this thesis. Although, this framework was originally designed to parse a sentence into a sequence of words, we would like to determine how well it could be adapted to perform such a task at the subword level. TINA's grammar rules were written to define the allowed transitions among morph categories. The score, or probability, of each allowed transition was calculated during the training session by countings the occurrences of each transition in the training data's parse trees, constrained by the names appearing in the training name lexicon and their corresponding morph representations. In the parse mode, TINA was expected to propose sequences of morphs according to each name at its input, according to the trained probability and word decomposition rules.

MIT's SUMMIT recognition system was utilized in building all of the speech recognizers involved in this thesis. The SUMMIT recognition system is a segment-base speech recognition engine, in which the input waveform is converted into a network of segment-based feature vectors. In order to find the recognition result, the system walks the segment network. Each path through the network is given three types of scores, namely the acoustic model, the pronunciation model and the language model. The acoustic model is the probability of occurrence of the segment-based feature vectors given sequences of words, segments and phonetic units. The pronunciation model is the probability of occurrence of sequences of phonetic units given specific words. And the language model is the probability of occurrence of each word given its linguistic context. The total score for each path is calculated by summing the three corresponding scores. And the path with best total score is selected as the recognition result.

There were two types of data used throughout this research. They were lists of names and audio files containing either pronunciations or spellings of proper names. The list of names used was gathered from various sources including transcription of the audio files. Some audio files were obtained by recording the utterances from phone calls to the Jupiter weather information system. Other audio files were taken from the OGI corpus, which contained telephone-quality utterances of people spelling or saying names. Data were grouped into two sets, the training set and the test set. The training set is larger than the test set in terms of the number of names. Around 58% of the names appearing in the test set also appear in the training set. The name lexicons, in which the names in each set are transcribed into their morph representations of both types, have been created. The size of the overlapping parts between morphs used in the name lexicons of the

training set and the test set is larger for Type II morphs than Type I morphs. This shows that Type II morphs have better generalizing properties.

There are three main components in our framework. They are the letter recognizer, the morph recognizer and the TINA parser. The letter recognizer is responsible for proposing letter sequences from the input name-spelling utterances. The function of TINA is to take the letter sequences proposed by the morph recognizer and propose morph representations of those letter sequences to be composed with the baseline language model of the morph recognizer. Thus, the morph recognizer is responsible for proposing morph sequences from the input pronounced utterances according to the composed language model. These resulting morph sequences are then mapped to the corresponding letter and phone sequences, which we need in order to fulfill our recognition tasks of the spelling and the pronunciation.

Two letter recognizers were built and utilized. Both recognizers have a similar building process, but their bigram and trigram language models were trained on different name lexicons. The first one is called the "general" letter recognizer, since its language models were trained on the letter sequences of the names appearing in a large lexicon (>100,000 names), while the language models of the other letter recognizer, the "training-set oriented" letter recognizer, were trained specifically on the letter sequences of only the names appearing in the training name lexicon. The letter recognition results of these two recognizers were used as the baseline recognition accuracies for comparing with the letter recognition accuracies of our integrated system. The difference in the letter recognition accuracies of the two letter recognizers allowed us to determine the effect of the quality of the letter recognizer on the whole system. Also, there are two morph recognizers, which were built based on different inventories of morphs, i.e. the Type I and Type II morphs. The performances of these two morph recognizers were also used as baselines for evaluating our integrated system.

Two sets of experiments were conducted. Each was based on using different types of morphs. The parameter varied across various experiments was the language model of the morph recognizers. In the baseline recognition, bigram and trigram language models were used. The combined language models, which result from the composition between the baseline language model of the morph recognizers and another language model obtained form the spelling information through TINA and the letter recognizers, were used in the integrated system. The first combined language model was the composition of the baseline language model and the

language model obtained from the correct letter sequence for the input name. The other two combined language models were the composition of the baseline language model and the spelling knowledge obtained from the recognition results of the letter recognizers. In one case, all of the 10-best letter sequences proposed from the letter recognizers were used to pass the spelling knowledge to the morph recognizer. In the other case, only the letter sequence with best recognition score was used.

When Type I morphs were used, it was found that the letter knowledge could help the pronunciation recognition task, especially when the correct letter sequences were provided. And it was also found that using the top choice of the 10-best letter sequences was more useful to the phone recognition task than using all of the 10-best letter sequences. Furthermore, as expected, the letter recognition accuracies of the baseline letter recognizers were found to have an effect on the performance of the overall system. However, there was no sign that pronunciation knowledge could help the letter recognition task in most cases.

Using Type II morphs was more promising than using Type I morphs. The result showed that the letter knowledge could help the pronunciation recognition even more than using Type I morphs. And, as in the Type I case, using the top choice of the 10-best letter sequences was more useful to the phone recognition task than using all of the 10-best letter sequences. Still, both methods did improve the baseline pronunciation recognition accuracies more than they did in the Type I case. Finally, unlike the Type I case, the letter knowledge from the name-spelling utterances in this case increased the letter recognition accuracy when the combined language models were created from the top choice of the 10-best letter sequences proposed from the baseline letter recognizer. Although, in general, the recognition result was better when Type II morphs were used, it is not perfectly correct to say that they are a more suitable morph representation, since there might be some differences in the quality of the name lexicon prepared in Type I and Type II. However, we feel that Type II morphs should be chosen as the subword units in developing this system in the future.

In the rest of this chapter, we will suggest some work that can be conducted in the future based on the framework developed in this thesis. Some of it could have been done under the scope of this thesis, if time had allowed.

9.2 Training TINA on a larger set of names

As part of the data collection attempt in this thesis, we have built a list of names, which was collected from various sources. The size of this list of names is around 100,000 names, which were not restricted to be English names but any names that can be spelled using the 26 letters of the English alphabets. It is interesting to ask whether this list of names would improve or degrade the quality of proper name decomposition task if we are able to use these 100,000 names to train TINA's word decomposition grammars. Although the size makes it interesting, it is considered a very exhaustive task to transcribe all of the names into their corresponding morph representations. A semi-automatic technique has provided an initial morph decomposition, but it requires extensive manual editing to correct for errors.

One possible way to handle this problem is to try to let TINA itself help in the transcription task. TINA is capable of producing a name-to-morph lexicon based on the trained grammars. Thus, we need a name lexicon for TINA to produce another name lexicon. Our approach is to let TINA parse the 100,000 names and propose an output name lexicon containing those names from the current trained grammar. It is likely that the output name lexicon provides inappropriate morph representation for many of the words, due to sparse data problems. Some names need new morphs in order for them to be represented appropriately. However, an expert can look at this name lexicon and edit the transcription proposed by TINA. Prominent errors should be fixed while too detailed errors can be ignored. Then we can retrain TINA's grammar on the edited lexicon and expect more accurate decompositions from the parsing. This process will be repeated until we have a reasonably high quality lexicon of 100,000 names. By implementing it in this way, it is easier in that an expert does not have to fix the morph representations of every single word. Manual correction on some words might generalize iteratively to other related words.

In order to simplify the name decomposition task, we should reduce the confusion that lies in the morphs used to represent names. In this thesis, Type I morphs have unique pronunciations: while some morphs can be spelled with the same letters, they are distinguished by the usage of capital letters. Type II morphs can have multiple pronunciations, by eliminating the distinction based on capital letters. Thus, using Type II morphs can be less confusing in the name decomposition task.

We propose avoiding the usage of capital letters in order for TINA to create a more consistent name-to-morph lexicon, since the task of identifying the correct "allomorph" in each word in a large lexicon is daunting.

9.3 Experiment with alternative ways of passing spelling knowledge

In the experiments conducted in this thesis, the knowledge of the spellings is incorporated into the system through letter recognizers. Then the proposed letter sequences from the letter recognizers are presented to the morph recognizer in the form of networks of morph sequences proposed by TINA. Examples of networks containing information about the proposed morph sequences for proper names were shown in Figures 5-3 and 5-4 in Chapter 5. In each figure, each path between the start node and the end node makes an allowed morph sequence proposed as a resulting morph sequence form the morph recognizer. Thus, the number of different morph sequences proposed by the system is ten if the network in Figure 5-3 is used as part of the morph recognizer's language model, while it is two if the network in Figure 5-4 is used.

However, if we allow all of the paths between the start node and the end node to cross-pollinate with each other, the number of different possible resulting morph sequences will increase. And the correct morph sequence, which does not necessarily appear in the original network for that name, might be reconstructed by the cross-pollination effect. However, this could result in additional errors due to inappropriate cross-pollination.

9.4 Improvements to speech recognizers

As we can see from the recognition results of our system, the better the letter recognizer, in terms of its letter recognition accuracy, the higher the performance of the overall system. All of the morph recognition accuracy, phone recognition accuracy and letter recognition accuracy, calculated form the resulting morph sequences, increase as the letter recognition accuracy of the preliminary letter recognizer, on the corresponding data set, improves. Furthermore, in the case where we emulate an ideal letter recognizer, in which the preliminary letter recognizer always provides the correct letter sequences, the recognition accuracies of the overall system are quite

good. The morph and phone recognition errors are less than 10% when Type II morphs are utilized. Thus, it is clear to us that one possible way to improve the overall system's recognition accuracy is to raise the letter recognition accuracy of the letter recognizer used in the system. However, we did not try to maximize our letter recognizer performance in this thesis since the main goal is the framework in which components interact with each other.

Although there are only twenty-six lexical entries used in the letter recognizer, the recognition of a spelled proper name is more difficult than the recognition of an English sentence in a limited domain, since there is less constraint on how the vocabulary is organized. Despite training the language model on a lexicon of spelled names, there will always be some rare letter sequences, compared to the letter sequences in the training lexicon, used for spelling proper names.

One possible way to improve the letter recognition accuracy of the letter recognizer used in this thesis is to train new acoustic models. As mentioned in Chapter 6, the acoustic models used for every letter recognizer are the existing acoustic models trained on sentences collected from the Jupiter system. Thus, if we train an acoustic model directly on letters, the new letter recognizer should provide better performance on letter recognition. It is likely, for example, that vowels in letters are more carefully enunciated than in words.

Another possible method for improving the letter recognizer lies in the language models. The language models we used for the letter recognizer in this thesis are letter N-grams. However, we could make use of class N-gram language models, in which the probabilities of letter sequences are trained according to its class. The classes possibly chosen in this case are the morph categories. TINA has the ability to take in the name lexicon and train the probabilities of letter sequences based on the chosen class. Finally, the FSTs of class N-gram created by TINA can be used as the new language models for the letter recognizer. We suspect that training the letter sequences assisted with the class knowledge should provide stronger constraints to the letter recognition task. For example, the letter sequence "s o n" is much more likely to occur in an isuf than in a prefix.

9.5 Automatic extraction of spelled and pronounced waveforms

The amount of data available plays a crucial role in speech research. Although, in this thesis, we feel that we have a sufficient amount of data available for developing and testing the system, we also are confident that more data will raise the accuracies we obtained from the system. As mentioned, there are two types of data needed. They are the name lists and the audio files. In data preparation, we spent a great deal of time transcribing the audio files recorded in the Jupiter system and manually extracting usable information into separate waveforms. Although the former is unavoidable, the latter might be done automatically or at least semi-automatically in the future.

The waveforms from which we extracted the name-spelling utterances and pronounced utterances are speech of the Jupiter system's callers when they were asked to say and spell the name of a person they know. Even though the answers responding to this query are not restricted to be any definite forms, some forms are used by different callers many times. Examples of these forms were shown in Section 4.4 in Chapter 4. By utilizing these repetitive forms as well as trying to find features that distinguish between the spellings and the sayings of names, we should be able to automatically extract the portion we want. As a consequence, the time used for data preparation will be reduced, and we can thus process more data.

Bibliography

[1] A. Parmar, "A Semi-Automatic System for the Syllabification and Stress Assignment of Large Lexicons," *MIT M.Eng. Thesis*, 1997.

[2] R. Lau, "Subword Lexical Modelling for Speech Recognition," *MIT Ph.D. Thesis*, 1998.

[3] R. Lau, and S. Seneff, "A Unified Framework for Sublexical and Linguistic Modeling Supporting Flexible Vocabulary Speech Understanding," *Proceedings of the 5th International Conference on Spoken Language Processing*, Sydney, Australia, November 1997.

[4] S. Seneff, "TINA: A Natural Language System for Spoken Language Applications," *Computational Linguistics*, vol. 18, no. 1,pp. 61-86, 1992.

[5] S. Seneff, R. Lau, H. Meng, "ANGIE: A New Framework for Speech Analysis Based on Morpho-Phonological Modeling" *Proceedings of ICSLP-96, Philadelphia*, PA, October 1996, pp. 110-113.

[6] H. Meng, "Phonological Parsing for Bi-directional Letter-to-Sound/Sound-to-Letter Generation," *MIT Ph.D. Thesis*, 1995.

[7] H. Meng, S. Hunnicutt, S. Seneff and V. Zue, "Reversible Letter-to-Sound/Sound-to-Letter Generation Based on Parsing Word Morphology," *Speech Communication*, Vol. 18, pp. 47-63, 1996.

[8] S. Seneff, "The Use of Linguistic Hierarchies in Speech Understanding," *Keynote paper at the 5th International Conference on Spoken Language Processing*, Sydney, Australia, November 1997.

[9] V. Zue, J. Glass, M. Phillips, S. Seneff, "The SUMMIT Speech Recognition System: Phonological Modeling and Lexical Access," *Proceedings of ICASSP*: 49-52, 1990.

[10] J. Glass, T. J. Hazen and L. Hetherington, "Real-Time Telephone-Based Speech Recognition in the Jupiter Domain," *Proceedings of ICASSP-99*, Phoenix, AZ, March 1999.

[11] V. Zue, et al., "JUPITER: A Telephone-Based Conversational Interface for Weather Information" *IEEE Transactions on Speech and Audio Processing*, Vol. 8, No. 1, January 2000.

[12] R. A. Cole, M. Fanty, and K. Roginski. "Recognizing spelled names with telephone speech," *Proceedings of Voice Systems Worldwide/Speech Tech 1992*, New York, NY, February 1992.

[13] R. A. Cole, K. Roginski, and M. Fanty. "A telephone speech database of spelled and spoken names," *Proceedings of the International Conference on Spoken Language Processing (ICSLP)*, vol. 2, pp. 891-893, October 1992.