Corpus-Based Unit Selection for Natural-Sounding Speech Synthesis

by

Jon Rong-Wei Yi

S.B., Massachusetts Institute of Technology, 1997
M.Eng., Massachusetts Institute of Technology, 1998

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

Doctor of Philosophy in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 2003

Author .....
Department of Electrical Engineering
and Computer Science
May 27, 2003

Certified by .....
James R. Glass
Principal Research Scientist
Thesis Supervisor

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

Corpus-Based Unit Selection for Natural-Sounding Speech Synthesis

by

Jon Rong-Wei Yi

Submitted to the Department of Electrical Engineering
and Computer Science
on May 27, 2003, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Electrical Engineering and Computer Science

Abstract

Speech synthesis is an automatic encoding process carried out by machine through which symbols conveying linguistic information are converted into an acoustic waveform. In the past decade or so, a recent trend toward a non-parametric, corpus-based approach has focused on using real human speech as source material for producing novel natural-sounding speech. This work proposes a communication-theoretic formulation in which unit selection is a noisy channel through which an input sequence of symbols passes and an output sequence, possibly corrupted due to the coverage limits of the corpus, emerges. The penalty of approximation is quantified by substitution and concatenation costs which grade what unit contexts are interchangeable and where concatenations are not perceivable. These costs are semi-automatically derived from data and are found to agree with acoustic-phonetic knowledge.

The implementation is based on a finite-state transducer (FST) representation that has been successfully used in speech and language processing applications including speech recognition. A proposed constraint kernel topology connects all units in the corpus with associated substitution and concatenation costs and enables an efficient Viterbi search that operates with low latency and scales to large corpora. An A^* search can be applied in a second, rescoring pass to incorporate finer acoustic modelling. Extensions to this FST-based search include hierarchical and paralinguistic modelling. The search can also be used in an iterative feedback loop to record new utterances to enhance corpus coverage.

This speech synthesis framework has been deployed across various domains and languages in many voices, a testament to its flexibility and rapid prototyping capability. Experimental subjects completing tasks in a given air travel planning scenario by interacting in real time with a spoken dialogue system over the telephone have found the system "easiest to understand" out of eight competing systems. In more detailed listening evaluations, subjective opinions garnered from human participants are found to be correlated with objective measures calculable by machine.

Thesis Supervisor: James R. Glass
Title: Principal Research Scientist

Acknowledgements

After defending my thesis last Wednesday, I began to reflect upon all the interactions I have had with people I met in the past decade or so and upon how I entered this area of research. Although my fascination with audio, particularly digital, began in primary school, it was not until high school that I picked up a textbook on Fourier and spectral analysis. That along with my interest in neural networks led me to the field of speech processing to which I was introduced by *Readings in Speech Recognition* by Alex Waibel and Kai-Fu Lee. With the support of Bob Clark and Rick MacDonald while volunteering at the United States Geological Survey, I assembled a time-delay neural network for phonetic classification and submitted my results to the Westinghouse Science Talent Search. The summer before I came to MIT in 1993, I worked on human-computer interaction for Helen Gigley at the Naval Research Laboratory. She suggested that I find Victor Zue when I arrive at MIT. At the time when Victor was still head of the Spoken Language Systems (SLS) Group at the Laboratory for Computer Science (LCS), I started building an HTML interface for the then-nascent GALAXY system working with Dave Goddeau under the auspices of the Undergraduate Research Opportunities Program (UROP). By the end of summer 1994, I had already constructed a simple Mandarin concatenative speech synthesizer. Later I began working with Jim Glass on other speech synthesis related UROP projects which naturally led into my bachelor's, master's, and, now, doctorate degrees. Jim is a marvelous teacher from whom I have learned how to be a researcher. Although it was a change when Victor became the director of the LCS, he has always been a cheerleader for my work and I wish him luck on his co-directorship of the to-be-merged LCS and Artificial Intelligence laboratories.

At SLS many graduate students well before me have set good examples for how I should pursue my PhD: Helen Meng, Jane Chang, Raymond Lau, Mike McCandless, Giovanni Flammia, Drew Halberstadt, T. J. Hazen, and Kenney Ng. Other members have the group had made invaluable contributions. Christine Pao maintained my

initial speech synthesis GALAXY servers for English and Mandarin for which Ed Hurley later assumed responsibility. Mike McCandless and Lee Hetherington developed SAPPHIRE, the basis of how we do speech research at SLS. The finite-state transducer (FST) library that Lee later developed has been immensely useful for representing the unit selection search and I thank him for his insights in and co-authorship (LaTeXwizard!) of the ICSLP 2000 paper. He has recently introduced dynamic autoreloading FST's for dynamic vocabulary. Michelle Spina has spent countless hours in the studio recording speech for my work and I thank her for her patience. I have empirically amassed speech knowledge through her recordings. Nikko Ström provided direction on how to separate the search and waveform GALAXY servers. Stephanie Seneff and Joe Polifroni have always pushed the outer limits of the synthesizer capabilities in the dialogue systems they build. They along with Scott Cyphers assembled the systems for the DARPA/NIST Communicator 2000 and 2001 evaluations. T. J. Hazen introduced sub-phonetic units into the speech recognizer which gave good segmentations for concatenative synthesis. I was taught to use his decision tree clustering tool by Chao Wang and she guided the initial creation of training processes for automatic determination of phonological classes. Matthew Brand at the Mitsubishi Electric Research Laboratories provided me with his unpublished derivation of the Kullback-Leibler distance formula for multivariate Gaussian random variables. The adoption of ENVOICE for Mandarin and Japanese synthesis has been facilitated by Chao Wang and Shinsuke Sakai. Jonathan Lau and Chian Chuu were brave enough to use Mandarin synthesis for their thesis work. Doreteo Torre Toledano assisted me in adding multi-modal support by transmitting word boundary information. Jonathan helped debug this functionality in his system. Discussions with Han Shu have kept me abreast of developments in speech recognition and his suggestions for synthesis were helpful. My knowledge of probabilistic models has been bolstered by discussions with Karen Livescu and she has continued some of my work with CRYSTAL and inverse Fourier waveform reconstruction methods. Ernie Pusateri used an earlier version of ENVOICE for a class project and is now examining re-scoring methods. Tony Ezzat has been a great supporter of my work and I wish him luck on audio-visual integration.

Paul St. John Brittan, a visitor from HP Research Labs in Bristol, encouraged me to finish towards the end of my thesis. Rita Singh, a visitor from Carnegie Mellon University, expanded my knowledge of dynamical systems. Philipp Schmid performed initial research on feature quantization and how it affected recognition word error rates. After I re-implemented it, Laura Miyakawa extended my work in her thesis. While an officemate of mine, Eugene Weinstein worked on SPEECHBUILDER and I was constantly reminded of how important it is to define processes for rapid prototyping. Other officemates of NE43-607 who have come and gone include Ray Chun, Joshua Koppelman, Jane Chang, Chao Wang, Justin Kuo, and Jay Hancock. Alex Park, a current officemate, has endowed me with knowledge of auditory models. My other current officemates whose late-night hacking company and discussions I treasure are Mitch Kapor and John Lee. I wish the best to my officemates and to the other graduate students: Alicia Boozer, Brooke Cowan, Ed Filisko, Ekaterina Saenko, Min Tang, and Sybor Wang. Vicky Palay, Sally Lee, and Marcia Davidson have made my stay at SLS wonderfully pleasant. I would like to acknowledge Marcia, Frank Tilley (of Computer Resource Services), Mary Ann Ladd (CRS), and Ty Sealy (CRS) for their preparations for my thesis defense and the videotaping of said defense. Thanks to Ty for his work on the cameras and microphones.

After finishing my master's in 1998, I spent a summer examining research in, primarily, array processing for Allen Sears at DARPA in Arlington, Virginia. I thank him for his guidance on how to pursue a PhD. Nadine Hadge of the Information Sciences Institute (ISI) and Bob Kahn of the Corporation for National Research Initiatives supported me during this time too. Alec Aakesson, Forrest Houston, and Steve Ray of ISI were great colleagues as well.

I was fortunate to have had the chance to interact with many professors at MIT. Greg Wornell, Amos Lapidoth, and Ken Stevens were on my first Oral Qualifying Examinations (OQE) committee. Upon the insistence of my second OQE committee, Louis Braida and John Tsitsiklis to which I am indebted, I was able to hone my teaching skills through Teaching Assistantships (TA). I was a 6.003 TA for Jacob

White and a 6.041 TA for Dimitri Bertsekas. While TA'ing 6.003, I met Jesus del Alamo and Steve Massaquoi through recitations, staff meetings, and enjoyable grading sessions. He served on my thesis committee and offered insights on biologically informed modelling. I became fully qualified as a doctoral candidate after my third try for the OQE thanks to George Verghese, Munther Dahleh, and Denny Freeman. Fortunately my area exam went much more smoothly with Jim Glass, Deb Roy, and Bill Freeman. My penultimate year of doctoral studies was funded by the Intel PhD Fellowship Program. Apart from recommendations given by Jim and Victor, Jesus supplied a last-minute recommendation for my Intel application which undoubtedly explains my receiving the fellowship. I learned how to teach clearly and concisely by watching Jesus dazzle students in 6.003 recitations. Of course, I would have never entered the PhD program without recommendations from Jim, Gilbert Strang, and David Deveau. I enjoyed Gilbert's wavelet class. David, other members of the music faculty, and musicians from around Boston including Marcus Thompson, Bill Cutter, Pamela Wood, Charles Shadle, Margaret O'Keefe, Lynn Chang, and John Harbison have added a soothing balance to the technical nature of my life. Kevin McGinty, my piano teacher, has encouraged throughout the years.

Finally, I would like to acknowledge my friends and family. I will miss my dearly departed friends, Meng-meng Zhao and Jaemin Rhee. Meng was always there for me since my freshman days. Jaemin enriched my life through music and encouraged me to finish. My college buddies, Philip Kim and Eric Choi, have helped me through these recent difficult times and we have become closer as a result. My younger brothers spur me on and my parents' love makes me strong. Indeed, I was proud to present my thesis work to my mother who attended my defense. I am looking forward to spending more time with my family. This part of the journey is over and I am grateful to Weiwei who was a shining star as I navigated my sea of apparent torrential darkness.

This research was supported by DARPA under Contract N66001-99-1-8904 monitored through Naval Command, Control and Ocean Surveillance Center, and contract DAAN02-98-K-003, monitored through U.S. Army Natick Research Development and Engineering Center.

Contents

1 Introduction **21**

1.1 Background 23

1.2 Approach 26

1.3 Outline 29

2 Unit selection **33**

2.1 Introduction 34

2.2 Background 37

2.3 Communication-theoretic formulation 39

2.4 Parameter reduction: linearization 42

2.5 Summary 47

3 Analysis of sonorant sequences **49**

3.1 Introduction 49

3.2 Distributional properties 50

3.3 Crystal tool 54

3.4 Sub-phonetic: manual experiments 56

3.5 Sub-phonetic: automatic experiments 61

3.6 Frame-based units 68
3.7 Source-filter separation 70
3.8 Nasals 76
3.9 Discussion 80

4 Acoustic modelling **81**

4.1 Introduction 81
4.2 Observation space 83
4.2.1 Measurements 84
4.3 Categorization 85
4.3.1 Substitution classes 86
4.3.2 Concatenation classes 87
4.4 Characterization 90
4.5 Comparison 91
4.5.1 Information-theoretic distance metrics 91
4.6 Analysis 96
4.6.1 Automatically determined costs 96
4.6.2 Automatically determined classes 98
4.7 Related work 104
4.8 Summary 104

5 Finite-State Transducers **107**

5.1 Introduction 107
5.2 Background 108

5.3 Unit selection cascade 108
5.4 Input specification 110
5.5 Corpus utterances 112
5.6 Constraint kernel 114
5.7 Design considerations for search 120
5.8 Search 122
5.9 Pruning 124
5.9.1 Pushed weights 126
5.10 Pipelined search 130
5.11 Rescoring 132
5.12 Extensions 135
5.12.1 Hierarchical unit selection 135
5.12.2 Segmental diacritics 138
5.12.3 Dynamic determination of location of concatenation 140
5.13 Summary 140

6 Design and evaluation **143**

6.1 Related work 144
6.2 Processes 145
6.3 Empirical evaluation 151
6.4 Automatically determined costs 152
6.5 Intonation modelling 158
6.6 Evaluation by recognition 166

6.7 Summary 171

7 Conclusions and future work **173**

Bibliography **178**

A Finite-state transducer (FST) composition **197**

A.1 Example of composition 199

A.2 Two-sided composition 200

B Clustering questions **203**

C Gaussian random variables **209**

D Information theory **211**

D.1 Introduction 211

D.2 Information-theoretic metrics 212

D.2.1 Expressions for Gaussian condition 213

D.2.2 Derivations 214

D.3 Automatic determination of synthesis costs 214

List of Figures

1-1 Architecture of rule-based formant speech synthesizer. 24

1-2 Architecture of corpus-based unit selection concatenative speech synthesizer. 26

2-1 Communication-theoretic formulation of unit selection. 40

2-2 Cluster modelling framework for parameter reduction. 44

3-1 Distribution of VV finer structure. 52

3-2 Screenshot of the CRYSTAL tool. 55

3-3 Screenshot of the word 'alloy' taken from *cv* tool. 59

3-4 Screenshot of the word 'airy' taken from *cv* tool. 60

3-5 Screenshot of the word 'aerial' taken from *cv* tool. 60

3-6 Screenshot of the synthetic word 'aerial' taken from *cv* tool. 60

3-7 LPC analysis applied to a pitch period from [u^{w}]. 63

3-8 Automatic determination of pitch periods in 'lawyer'. 64

3-9 Jackknived synthesis of 'lawyer' from 'loyal' and 'layer.' 67

3-10 Plot of voiced speech waveform and Hanning-windowed pitch period. 69

3-11 Comparison of pitch continuity without and with smoothing. 71

3-12 Smoothed magnitude from 'IMport' combined with excitation from 'imPORT'. 73

3-13 Smoothed magnitude from 'IMport' combined with excitation from 'comPACT'. 75

3-14 Rob Kassel's Happy Consonant chart. 77

3-15 Rob Kassel's Happy Little Vowel chart. 77

4-1 Categorization and characterization in modelling. 82

4-2 Graphical representation of observation space and boundary measurements. 83

4-3 Prototype measurements are used to form boundary and segment measurements. 85

4-4 Sample clustering run for left context of [ʌ]. 87

4-5 Sample clustering run for stop-vowel and stop-glide transitions. 89

4-6 Bubble plots of concatenation and substitution costs. 97

4-7 Substitution costs for [iʸ]. 99

4-8 Distributional analysis of automatically determined concatenation costs. 100

4-9 Analysis of number of observations in concatenation class versus concatenation costs. 101

4-10 Distributional analysis of automatically determined substitution costs. 103

5-1 FST representation of unit selection. 109

5-2 FST of the word, 'Boston.' 110

5-3 FST of the pronunciation of the word, 'Boston.' 111

5-4 FST of the pronunciation and transitions for 'Boston.' 111

5-5 FST for corpus representation of utterance, 'Boston.' 113

5-6 FST for corpus representation of utterance, 'Boston', with skip arcs. . 113

5-7 FST for corpus representation of utterance, 'Boston', with skip arcs for non-contiguous and anti-chronological selection. 113

5-8 FST, S, contains corpus utterances and constraint kernel for efficient unit selection. 116

5-9 Tradeoff between search speed and accuracy as pruning beam varies. . 126

5-10 Tradeoff between search speed and accuracy as number of nodes varies. 127

5-11 Example of pushing weights from S to T. 128

5-12 Tradeoff between search speed and accuracy as score-based and count-based thresholds are varied for pushed weights configuration. 128

5-13 Overlay of tradeoff between speed and accuracy when pruning beam is varied for original and pushed weights configurations. 129

5-14 Overlay of tradeoff between speed and accuracy when number of nodes is varied for original and pushed weights configurations. 130

5-15 Growth of Viterbi trellis as pruning beam varies 132

5-16 Growth of Viterbi trellis as pruning beam varies (zoomed view). . . . 133

5-17 Process flow for generation, rescoring and resorting of N-best lists. . . 134

5-18 Model for rescoring of concatenations. 135

5-19 Hierarchical configuration for synthesis FST, S. 137

5-20 FST of corpus utterance in S incorporating diacritics. 139

5-21 FST of input specification incorporating diacritics. 139

6-1 Mapping of objective distances to subjective distances. 144

6-2 Process for adapting synthesizer to new language. 146

6-3 Process for adapting synthesizer to new domain. 147

6-4 Process for selecting generating prompts. 148

6-5 Process for meaning-to-speech conversion. 148

6-6 Process for tuning synthesis parameters. 150

6-7 Sample system response for air travel domain. 152

6-8 Results of listening evaluation of automatically determined costs. . . 155

6-9 Detailed analysis of evaluation of automatically determined costs. . . 156

6-10 Breakup of intonation classes by pitch contour and sonorant composition. 160

6-11 Results of listening evaluation of automatically determined costs. . . 161

6-12 Results of listening evaluation of automatically acquired intonation model. 162

6-13 Detailed analysis of evaluation of automatically acquired intonation model. 164

6-14 More detailed analysis of evaluation of automatically acquired intonation model. 165

6-15 Regression of results of listening evaluation and likelihood. 167

6-16 Regression of average utterance likelihood and word error rate. 169

7-1 Sinewave and mumble resynthesis: There are 49 Honda Civics. 177

A-1 FST for example utterance in toy language. 199

A-2 FST for diphone rewriting in toy language. 200

A-3 FST for previous example utterance as rewritten with diphones. . . . 200

A-4 FST for $D \circ D^{-1}$ before and after optimization. 201

B-1 Left-sided triphone questions. 204

B-2 Right-sided triphone questions. 205

B-3 Left-sided diphone questions. 206

B-4 Right-sided diphone questions. 207

B-5 Initial diphone questions. 208

List of Tables

3.1 Analysis of vowel and semivowel sequences in PRONLEX. 50

3.2 Analysis of vowel and semivowel sequences of length 2 in PRONLEX. . 51

3.3 Analysis of vowel and semivowel sequences of length 3 in PRONLEX. . 53

3.4 Analysis of multi-vowel sequences in PRONLEX. 54

3.5 Multi-sonorant sequences. 57

3.6 Word pairs with transposed lexical stress patterns [2]. 74

3.7 Nasal experiments with NVN set. 79

3.8 Nasal experiments with NVS and SVN set. 79

4.1 Substitution classes for [i^{y}]. 99

5.1 Number of states in each layer of constraint kernel. 118

5.2 Number of arcs in between layers of constraint kernel. 119

5.3 Expansionary factors between layers that impact search. 121

6.1 Twenty-two utterances form the test set for synthesis evaluations. . . 153

6.2 Makeup of automatically acquired intonation classes. 159

6.3 Recognition error analysis for different synthesis configurations. . . . 168

6.4 Word error rates for speech synthesized with rescoring. 170

Chapter 1

Introduction

Speech synthesis [71] is an automatic encoding process carried out by machine through which symbols conveying linguistic information are converted into an acoustic pressure waveform. The speech message may originate from a representation of the meaning to be communicated. The process of language generation produces a set of lexical unit sequences, or sentences, that best convey the message. Higher-level constraints such as discourse or dialogue may influence the structure, or syntax, of the sentences in the interest of unambiguous, efficient communication. In humans, language generation may also be influenced by paralinguistic factors (vocal effects which may convey meaning external to linguistic information) such as emotion, intent, diction, and socioeconomic background. The phonemic baseforms of the lexical units, or words, are accessed from a lexicon, and variational, allophonic phenomena may occur within and between words. Pronunciation variations may depend on phonotactics (permitted sound sequences in a language), regional dialects, speaking rate, effort of enunciation, or, in humans, paralinguistic factors such as formality, and familiarity. In parallel to the phonetic sequence comes a description of the prosodic intonation contour which includes pitch, duration, and energy. Prosody is affected by all of the above linguistic and paralinguistic factors. For example, phrases may be separated into pitch groups to lessen cognitive load and accents may be applied onto words carrying information

new to the running dialogue. The final pressure waveform is synthesized from the acoustic-phonetic and prosodic descriptions through simulation of the human speech production mechanism. Again, in humans, paralinguistic factors such as background noise, duress, or health may apply.

The process described above is multi-staged and involves many decisions made at the interface of each stage. As it progresses from symbols to signals, it is inherently a one-to-many mapping and many possible output waveforms (as interpreted by the human listener) may actually convey the same, original message. Information about the encoding process is typically captured through some combination of knowledge-engineered rules and data-driven models. This dichotomy reflects the classic trade-off between top-down and bottom-up approaches. In typical rule-based systems, experts infer a set of hard rules from mostly knowledge and experience. The rules may be stored in structures such as decision trees, which have a top-down view of the world. In typical data-driven systems, experts infer a set of features which are measured over a corpus of data and are used to estimate statistical models. As these models are crafted, a compromise must be struck between parameter parsimony and data underfitting. Top-down and bottom-up learning can also be combined into a two-stage learning process. For example, an acoustical model that incorporates context may use a decision tree to split contexts into more homogeneous categories. Bottom-up estimation proceeds to estimate statistical models for each of the categories.

One particularly enlivening application of speech synthesis is in spoken dialogue systems which allow humans to interact via voice with a machine to access information and process transactions. In short, their sole function should be to offload burdensome tasks and increase human productivity. In this context, speech synthesis is used by machine on the output side to provide information and aural feedback to the human user. This auralization defines a persona or voice that the user associates with the system, which, more often than not, leaves a lasting impression of the overall system. In order to reduce the cognitive load [110, 85] of understanding on the part of the user, it is then imperative to design a speech synthesizer for outputting system

responses which sound natural, like a human. Moreover, in displayless systems where audio is the sole channel of feedback, any shortcomings in the naturalness and intelligibility of the speech synthesis will be irreparable. Attention and focus diverted to understanding synthetic speech can only negatively impact the ability to recall, for example, information from a list [80].

In spoken dialogue systems, the interface medium is speech, and dialogue interactions are speech-to-speech, which means that the user speaks to the system and the system speaks back to the user. However, within the computer, the representation is necessarily symbolic. Hence, the user's query is translated to a lexical representation by means of speech recognition. From that hypothesized word sequence the meaning and intent can be derived by means of language understanding. Now, the machine can then retrieve information based on the query and formulate a reply via language generation. The symbolic response is then transformed back to the speech domain by the process of speech synthesis described above.

1.1 Background

Although speech synthesis has had a long history [71, 87], progress is still being made [43, 152] and recent attention in the field has been primarily focused on concatenating real human speech selected from a large corpus or inventory. The method of concatenation [107] is not a new idea and has been applied to other parametric speech representations including, for example, linear prediction coefficients [102], sinusoidal modelling [88], multiband excitation modelling [44], and harmonic-noise modelling [138]. Whereas the units of concatenation were selected by rule before, nowadays unit selection techniques [61] are typically employed to select variable-length or non-uniform units [118] from a large corpus [61, 8]. Because of reduced storage costs over the years, unit representation has tended away from parametric and towards non-parametric models (i.e., using the speech waveform itself). Before the advent of using

real human speech, rule-based formant methods as shown in Figure 1-1 were popular. Unit selection was trivial with a database of very few (one) examples, while much manual tuning was required in the modification and generation stages. Hard decisions made by decision trees [13, 59, 148, 29] or omission of units from the corpus [9, 38] are typically made before the unit selection search to prune the search space. Intonation contours can be applied by waveform modification as part of a post-processing stage [12] or it can be modelled in the unit selection search [29, 19, 21]. Speech synthesis in limited domains [10] can sound natural and is often a good starting point for synthesis research. The above techniques are mostly independent of the target language and can be applied in a multilingual setting [133].

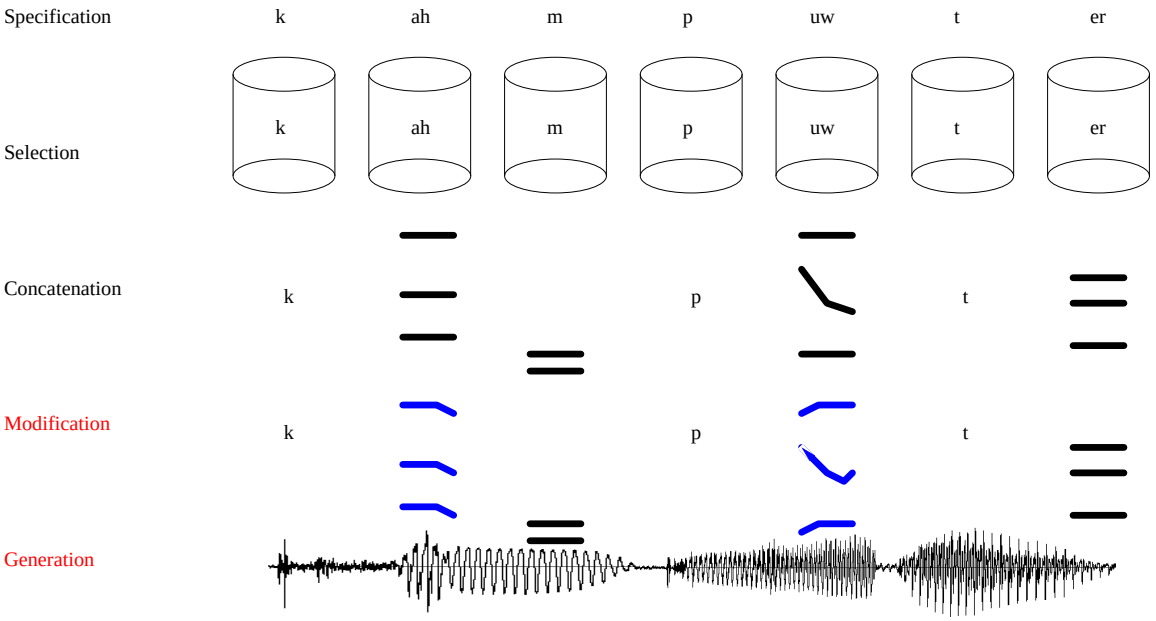


Figure 1-1: Architecture of rule-based formant speech synthesizer.

Initial work by the author on concatenative synthesis in an automobile classified advertisement system [90] explored using synthesis units at the lexical level and constructed entire sentences from pre-recorded words and phrases. Because the units were relatively long in duration, concatenations (and any associated concatenation artifacts) occurred less frequently in time. Anecdotally speaking, prosody, not co-articulation, matters at the supra-lexical level. By recording words and phrases in prosodic contexts similar to where they would be used, a very high naturalness was

achievable with a compact corpus. However, the type of responses was limited to the type of carrier phrases used in recording. Although this approach was well suited to constrained domain applications, recording entire words became unwieldy for large vocabulary sizes.

Later work by the author [162] on a precursor to the flight arrival system described in [126] lifted the restriction on unit size and experimented with using sub-lexical, or sub-word units. As described previously, costs were devised at the phonological level to favor certain substitutions and concatenations over others. Informal, perceptual studies were used to determine what the phonological classes and synthesis costs should be. It was found that manner of production and place of articulation could be linked to co-articulatory effects. Accordingly the classes and costs were manually set to consider these co-articulatory factors. In experiments with English words, longer units were found to sound more natural. Prosody seemed to suffer when concatenating smaller units, perhaps, because prosodic effects of lexical stress were not considered, for example. Nonetheless, it gave a glimpse of what naturalness was possible using phonological units.

The result of the prior work performed by the author described above is a corpus-based, concatenative speech synthesizer which performs unit selection at the phonological level on a domain-dependent corpus of speech. Figure 1-2 shows a diagram of the stages involved in going from an input specification to a synthetic waveform. Unit selection is the most challenging step: with 10 examples for each of the 7 sounds in "computer", there are 10^7 combinations from which to select one sequence. Because it is utilized within spoken dialogue systems, many of the above processes are implicitly captured by the very nature of the constrained domains. For example, neither syntactic parsing nor prosody generation is totally essential for natural-sounding synthesis, because the domain-dependent corpus of speech mirrors the structure of typical system responses. As a result, the speech synthesizer is somewhat custom tailored to the task and does not necessarily generalize well to other domains. Understanding and compensating for this and other aspects is the focus of this thesis.

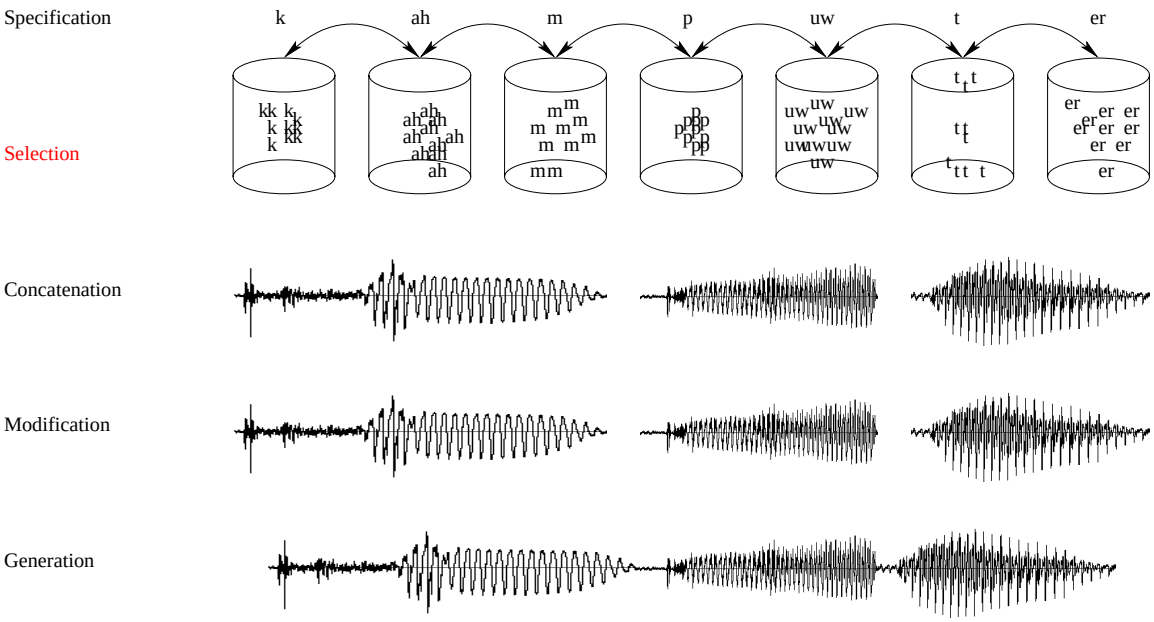


Figure 1-2: Architecture of corpus-based unit selection concatenative speech synthesizer.

1.2 Approach

The problem can be stated as follows: develop a flexible, expressive concatenative speech synthesis framework for scalable, efficient unit selection from large corpora that can be applied to and evaluated across multiple domains, languages, and voices. Desirable features include being able to automatically determine parameter settings from example data and to unify past synthesis research ideas under one framework.

This thesis proposes to capture the entire transformation from symbols to signals in speech synthesis with finite-state models. The appropriate stages will be determined and connected by the appropriate interfaces which serve to buffer and to isolate. Each stage transduces an input language to an output language and should be capable of handling a network or graph of possible inputs and of producing a network or graph of possible outputs. When (soft) decisions are to be made, quasi-probabilistic weights or encoding length costs statistically learned from labelled exemplars should prioritize alternatives. As stages of finite-state transducers (FST's) [115] sequentially

cascade, so will the weights, and the best transformation shall arise from all considered combinations.

One theme explored throughout this thesis is the commonality between speech recognition and synthesis. Modelling techniques for context-dependent acoustics and pronunciation variation, for example, can be directly borrowed from recognition work. Their application is just slightly different because in recognition the word sequence is unknown, whereas in synthesis it is known. The FST representation has been applied to speech recognition with great success in where, for example, Viterbi and A^* search algorithms were used for 1-best and N-best lexical decoding. These search algorithms are generically defined for FST's, oblivious to their specific application, and, therefore, reusable. On the other hand, note that, because many processes in speech synthesis have analogues in speech recognition, a potential by-product of this research work may be the beginnings of a new architecture for speech recognition.

This thesis work will be based on a common model of speech synthesis known as text-to-speech (TTS) synthesis which contains three main stages. Because, many state-of-the-art speech synthesizers operate in this modality, they will serve as references for comparison and contrast. TTS conversion contains three main stages:

- Text analysis includes sentence parsing, abbreviation and numeral expansion, and pronunciation generation.
- Prosody modeling predicts intonation and accents from analyzed text and generates acoustic parameters (e.g., duration).
- Waveform generation produces speech from description of units.

Additionally, waveform generation approaches are categorized as follows: terminal-analog rule-based models (e.g., MITalk [1], DECtalk [71]), articulatory/physical models (e.g., Kelly & Lochbaum [68], HLsyn [159]), and waveform concatenation (e.g., ATR Chatr [24], Festival [14], Bell Labs [8]). Waveform concatenation has recently

become popular in the speech synthesis literature primarily because of its low complexity and of decreasing storage costs. Because it uses speech units excised from natural speech spoken by a human, it has demonstrated potential for producing natural-sounding synthesis.

Concatenative approaches are grounded on two basic premises as described by Hunt and Black [61]: 1) speech spoken in one context may be used in another, and 2) splicing of the speech signal can be performed with little perceptual distortion. To achieve this sleight of hand, certain criteria must be prescribed for the realization of these two ideas. The degree to which the two above premises hold can be measured by substitution and concatenation costs, respectively. In the implementation of the first concept, one defines a set of contextual classes that are deemed to have the same co-articulatory effect on the realization of a speech segment. For example, phonological equivalence classes can be determined with acoustic-phonetics knowledge. For the second concept, it is contended that signal splicing may be less noticeable at places where the signal undergoes changes anyways, such as at places of source change or extreme spectral change. With the above in mind, novel speech can be generated from a pre-existing corpus by locating appropriate segments of speech and making judicious splices.

The above description of concatenation can be likened to the operation of cut-and-paste found in many text editors. In concatenative synthesis, what you cut and where you paste are the determining factors of naturalness. A unit selection algorithm is defined to select the units of appropriate makeup for concatenation at appropriate places. It can be as simple as a deterministic mapping of lexicon entries to speech waveform segments also known as the "canned speech" approach. In general, unit selection can be considered as a constrained optimization problem where the goal is to find the least-cost path through a graph of unit candidates. Because unit selection is a graph-based search, algorithms such as Viterbi and A^* are commonly employed to find a unit sequence that minimizes an overall cost metric. If the search is greedy, variable-length units are created by concatenating adjacent speech segments.

Because the cost metrics are measured over observed variables and not the hidden, psychophysical quantities, it is not clear that the best path corresponds to the most natural-sounding sequence of units. Because human users are the ultimate consumers of speech synthesis - they are the ones interacting with the machine through a spoken interface - their opinion of the synthesis provides important subjective feedback. If their subjective opinion can be related to the objective cost measures used by the search, then the validity and usefulness of the entire optimization framework described here is ensured. Then the limiting factors for widespread deployment of unit selection frameworks for concatenative speech synthesis become the design of the corpus and the design of the costs.

1.3 Outline

By discovering some deeper patterns in acoustic-phonetics, phonology, and prosody, it may be possible to increase the flexibility of this concatenative speech synthesizer and to synthesize less constrained textual input. Under the proper formulation, statistical instruments can be used to glean these patterns from data. Analysis-by-synthesis methods can also be useful in driving incremental feedback processes, such as corpus design. Digital signal processing may potentially play a role in creating novel signals. A truly automated, systematic approach will incorporate these techniques to minimize the amount of human intervention needed, save for the actual acquisition of human speech.

To first gain a foothold, Chapter 2 describes the optimization framework used for unit selection. A communication-theoretic formulation casts the process of unit selection as a noisy channel through which an input specification passes and speech segments emerge. The degree to which the input is degraded is measured by description length costs which describe how well the output approximates the input. A mathematical framework for efficiently applying these description length costs is also introduced.

Perceptual experiments are performed in Chapter 3 to better understand what constitutes a unit. A distributional analysis of the English language enumerates sonorant sequences and examines their composition and frequencies of occurrence. Digital signal processing techniques are employed to determine ideal points of concatenation. Novel speech segments are searched for and created with source-filter separation and re-combination at the sub-phonetic level. The results of these manual experiments provide insight on how to design units automatically.

Chapter 4 defines an acoustic model for synthesis units. Phonological classes are automatically acquired by a decision tree and are used to categorize units into clusters for subsequent characterization by statistical methods. An information-theoretic criterion is introduced to compare the statistical models and to derive substitution and concatenation costs in an automatic fashion. These costs appear to agree with acoustic-phonetics knowledge and with experiments on sonorant units conducted earlier.

Next, Chapter 5 delineates an implementation of the unit selection framework based on finite-state transducers. The search space is defined as a cascade of FST's which successively transform the input sequence of words into a graph of possible sequences of units. A key contribution is an FST topology connecting all units in the corpus with associated substitution and concatenation costs that makes the search efficient. Search behavior under Viterbi pruning is explored as well as rescoring of an N-best list generated by a second-stage A^* search.

Then, methods and processes for practice are laid out for semi-automatically designing, creating, and deploying synthesizers. Best-of-breed practices are used to successfully use automatically-trained, FST-based unit selection concatenative speech synthesis in spoken dialogue systems. A wholistic evaluation of human users carrying out a flight planning task finds that the system is "easy to understand". Further listening evaluations drill down and examine the efficacy of automatically trained phonological classes and acoustic costs by relating their performance to opinion scores garnered

from human participants. A simple intonation model is found to have the potential to systematically enhance naturalness. Preliminary experiments with *evaluation by recognition* show how machines can enter the feedback loop for automatically tuning and improving synthesizers.

Finally, Chapter 7 states conclusions and points to directions for future research.

Chapter 2

Unit selection

This chapter introduces the concept of unit selection, an optimization scheme for selecting a best sequence of units, which is utilized in many speech processing applications, including recognition and synthesis. In speech recognition, an optimal sequence of units (e.g., hidden-Markov model states) is selected to best explain a sequence of observations in the hopes of minimizing word error rate. In speech synthesis, an optimal sequence of units is selected to best reconstitute a sequence of speech-describing symbols in the hopes of synthesizing natural-sounding, artifact-free speech.

In this work, the process of unit selection is cast in a communication-theoretic formulation, where the unit selection process is likened to a noisy transmission channel and symbol corruption is minimized in an optimization framework. This is analogous to the model laid out for speech recognition [65]. Because the description of the units may not entirely be captured by symbols alone, a rescoring stage with numerical measurements can be performed afterwards. In this case, the first stage/pass can be thought of as a coarse search which is refined by the second stage/pass. The implementation of this multi-pass system with finite-state transducers is outlined subsequently in Chapter 5.

2.1 Introduction

In the most general of terms, unit selection is the process by which a higher-level description of speech is transformed into a lower-level description of speech given a corpus. This transformation may be as simple as a one-to-one mapping implemented by a table lookup, or it may involve path dependencies and be constructed as an optimization implemented by dynamic programming. Path dependencies necessitate bookkeeping of state, whereas (static) table lookup operates independently of state (i.e., stateless or context-free.) However, since the path-independent process of table lookup can be subsumed by a path-dependent algorithm, the distinction is not necessarily operational, but merely descriptional in nature.

The direction of transformation chosen for our discussion here - from a higher level to a lower level - is arbitrary. Once the direction is chosen, knowledge-bearing information needs to be encoded at the destination level. In speech recognition, the unknown to be inferred is the word sequence, and language models assist in the inference of the word sequence by providing constraints at higher levels. In speech synthesis, the unknown to be inferred is the unit sequence, and acoustic models assist in the inference of the unit sequence by providing constraints at lower levels.

There is a branching factor associated with the selection process from the known to the numerous possibilities of unknowns. In speech recognition, as acoustic models are proposed at each step to best describe an observation, the more discriminating the models are, the less of a branching factor there is. When the scores from the acoustic models are not as diverse, there is heavy competition between the models, and multiple hypotheses need to be considered in parallel, giving rise to a larger branching factor. Likewise, during the next step of stringing sounds (phones) into words (lexical items), when a sub-sequence is non-unique, multiple word hypotheses need to be maintained until a unique (given the history) sound breaks the tie. At the final stage, language models resolve ambiguity by constraining lexical sequences

and possess a measurable branching factor known as perplexity, which quantify the average number of possible words following any word on average.

In corpus-based concatenative synthesis, what is known is the message and what is unknown is the sequence of segments which reconstitute speech. The message may be conveyed by a multitude of different wordings. The alternate sentence arrangements represent the first level of branching. Next, multiple phonemic pronunciations may be specified for each word with additional pronunciation variation captured by phonological rules. At this point - ignoring multiple, parallel paths for now - when a wording is resolved down to a sequence of phonological units, say N of these, to match, there a total number of 2^{N-1} concatenation scenarios to consider. This exponential branching factor is due to the $N-1$ interleaved binary decisions made between each of the N units on whether to concatenate or not. Clearly, sequential decision making (either hard/irreversible or soft/deferred decisions) is required to deal with such an expansionary unfolding of possibilities. For each unit, unit selection will continue to select units from the current utterance barring mismatch on the subsequent unit. Otherwise, unit selection will choose from all the exemplars in the corpus which match the subsequent unit. Say that there are on average L exemplars per unique unit in the corpus. In the worst case when all $N-1$ decisions are to concatenate - or, equivalently, when all N units are non-contiguous - there are a total number of L^N possibilities that match the input unit sequence. Indeed, there is a demonstrable need for combating such growth which increases exponentially with respect to the size of the corpus (L) and the task (N).

The foregoing has paid attention to the selection aspect of *unit selection*. The actual nature of the unit in unit selection is a design choice in of itself. The requirements of the unit are that it encode linguistic, intonational, and (possibly) paralinguistic information. The information can be described at various levels, low and high, or even a hierarchical mixture thereof. Linguistic information would encompass segmental information, from the phonological to the syllabic to the lexical level. Intonational information include supra-segmental information, such as emphasis, rhythm, and stress,

for which pitch, duration, and energy are acoustical correlates, respectively. This correspondence need not be one-to-one though, because of many factors and influences. It can also extend to supra-lexical levels where phrases or multi-word sequences are described. Paralinguistic information can describe notions such as speaker gender, accent, voice quality, and emotional state.

While information that is encoded hierarchically is represented vertically or spatially, information can also be represented horizontally or laterally across time, giving rise to the notion of non-uniform or variable-length units. For example, individual phones can be split into half-phones [8] or diphones [107, 102, 26] or they can be merged into multi-phone units, or individual words can be merged into multi-word units. The decision to merge speech segments into, generally speaking, multigrams [35] can be made *a priori*, or at run-time. Effectively, multigrams can be thought of as a fixed segmentation of smaller speech segments, the result of domain-dependent analyses carried out empirically over a given data set. The merging may be motivated by capturing frequently observed sub-sequences or by discovering building blocks well-shielded from concatenation artifacts for concatenation, motivations to be described next in Chapter 4. Although it represents a hard decision, the design of the selection process is simplified, because the very existence and identity of a multigram preserves the computation that has been performed *a priori*. The creation process of non-uniform or variable-length units can be deferred to run-time by making soft decisions in the sequential decision making process. Here, a greedy search encouraging the selection of variable-length sequences can be utilized.

The final aspect which defines a unit is its resolution. The exact identity of the unit can be resolved with both symbols and numbers, corresponding to discrete and continuous measures, respectively. For example, individual allophonic variations of phones (allophones) can be considered separately as discrete features (e.g., nasalization or palatalization of vowels). When dealing with pitch, a choice can be made between a continuous scale (e.g., 108 Hz) or a discrete scale (e.g., low vs. high).

Given the above examination of what constitutes a unit, it is foolhardy to regard any particular spatio-temporal segmentation of speech units at any level of resolution as optimal, especially in the face of rare events in heavy-tailed distributions [93] and evidence of constant language evolution [92]. The design process of the unit destined for usage in unit selection must be repeated longitudinally in order to adapt to new data.

2.2 Background

Previous work on unit selection with focus on non-uniform unit and algorithmic development can be traced back to the Advanced Telephony Research Institute International (ATR) in the late 1980's, where Sagisaka [118] first developed the notion of non-uniform units for speech synthesis. Later works by Iwahashi [63] introduced a taxonomy of costs for joining units. After the ν-TALK speech synthesizer [119] was completed at ATR, Campbell [24] applied this framework to larger corpora in the CHATR system with implementation carried out by Hunt and Black [61].

Unit selection involves finding an appropriate sequence of units, $u^*_{1:N}$, from a speech corpus given an input specification, $u_{1:N}$, where the subscripts denote a sequence of N units counting from 1. In mathematical terms, this is represented as an optimization with a cost metric, $J(\cdot, \cdot)$.

$$u^*_{1:N} = \underset{\hat{u}_{1:N}}{\operatorname{argmin}} \quad J(u_{1:N}, \hat{u}_{1:N})$$

Typically an independence assumption is made to simplify the joint optimization over the N-length sequence into N individual, term-wise substitution costs, S, with

concatenation costs, C, between the selected units.

$$\begin{aligned} u^*_{1:N} &= \underset{\hat{u}_{1:N}}{\text{argmin}} \quad C(u_0, \hat{u}_1) + S(u_1, \hat{u}_1) + C(\hat{u}_1, \hat{u}_2) + \quad \cdots \quad + \\ &\qquad C(\hat{u}_{N-1}, \hat{u}_N) + S(u_N, \hat{u}_N) + C(\hat{u}_N, u_{N+1}) \\ &= \underset{\hat{u}_{1:N}}{\text{argmin}} \quad C(u_0, \hat{u}_1) \;+\; \textstyle\sum_{i=1}^{N} S(u_i, \hat{u}_i) \;+ \\ &\qquad \textstyle\sum_{i=1}^{N-1} C(\hat{u}_i, \hat{u}_{i+1}) \;+\; C(\hat{u}_N, u_{N+1}) \end{aligned} \tag{2.1}$$

Note that u_0 and u_{N+1} represent initial and final boundary units and are not part of the final selected sequence. Their usage is actually quite subtle, as the concatenation costs at the boundary, $C(u_0, \hat{u}_1)$ and $C(\hat{u}_N, u_{N+1})$, have heterogeneous arguments, whereas the $N-1$ concatenation costs between the selected units, $C(\hat{u}_i, \hat{u}_{i+1})$, have homogeneous arguments. While the latter is quite understandable as it considers concatenating actual instances of speech from the corpus, the former reflects the desire to select starting and ending units, $\hat{u}_1$ and $\hat{u}_N$ with surrounding contexts, u_0 and u_{N+1}. Because the arguments are heterogeneous in the former case, the functional form of C may differ for the heterogeneous case. Unfortunately, some of these subtleties are lost in the notation, but it is used herein nevertheless for its clarity of presentation. Although these contexts are typically chosen to be silence (i.e., in the absence of speech production), choosing speech contexts enables the selected units to blend with running context.

Though this formulation is simple, many issues immediately arise:

- What are the units, u?
- How is the optimization performed?
- Is unweighted addition the proper method of unit cost combination?
- What are the definitions for substitution costs and concatenation costs?
- What is the proper space-time tradeoff between computing the costs on-line and pre-computing (and storing) the costs off-line?

If pre-computation of C is necessary in advance of run-time over all M units in the corpus, then there are a quadratic number, M^2, of $C(\hat{u}_i, \hat{u}_j)$ terms for all i, j.

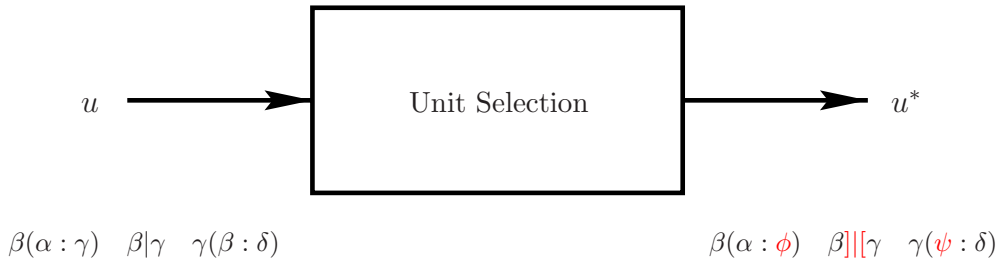
In Equation 2.1, greedy selection of variable-length units is encouraged by setting concatenation costs between adjacent units in the speech corpus to zero. Likewise, when there is an exact match of context, the substitution cost is zero. Of course, this entire setup is based on the premise that objective measures can capture perceptual distances. Chapter 6 describes experiments in which correlations are found.

In practice, the above optimization is handled with a dynamic programming approach, a type of sequential decision making algorithm. Definitions for these cost functions stemming from acoustic and information-theoretic motivations are later presented in Chapter 4. A clustering technique presented in a following section in this chapter linearizes the above quadratic factor by replacing instance-to-instance costs with cluster-to-cluster costs with the introduction of a centroid-based approximation that is accurate on average. What follows first is a reformulation of the above optimization in terms of communication theory.

2.3 Communication-theoretic formulation

Consider a communication-theoretic formulation of the unit selection process in which the unit selector is a black box receiving an input specification and transmitting the best match. This black box can be thought of as a noisy transmission channel [65], because the output may not precisely match the input. Corruption of the message is quantified by description lengths which describe the penalty in approximating the input with the output. When a message is transmitted through the unit selector without degradation, then it must be that the desired units are entirely contiguous within the speech database. The content and format of the message is left unspecified in this generic formulation.

In Figure 2-1, the input specification, u (ignoring subscripts), passes through unit selection and exits as u^*. To adapt this communication-theoretic formulation to the substitution and concatenation costs, S and C, described in Section 2.2, construct the input and output as streams of interleaved unit and transition symbols. The unit symbols may contain contextual (e.g., triphone $\beta(\alpha : \gamma)$ denotes β in the context of α and γ) information as well. Transition symbols are pseudo-units that have no acoustic realization but serve as place-holders between units. Transition symbols are marked differently depending on whether they bridge two contiguous units ($\alpha|\beta$) or not ($\alpha]|[\beta$). Naturally, all transition symbols on the input side are marked as contiguous ($\alpha|\beta$), because ideally that is the desired case. Note that the left context of β and the right context of γ are not corrupted in this transmission example.



Note: $\beta(\alpha : \gamma)$ denotes β with left and right contexts, α and γ, respectively.
$\beta|\gamma$ and $\beta]|[\gamma$ are contiguous and non-contiguous transition symbols, respectively.

Figure 2-1: Communication-theoretic formulation of unit selection.

Because the input and output streams contain unit and transition symbols by construction, it is straight-forward to quantify the corruption of unit and transition symbols with substitution (S) and concatenation (C) costs, respectively. To handle the pathological cases described above, S is zero when the input and output symbols are identical and C is zero when a contiguous transition symbol appears unchanged at the output side (i.e., $\beta|\gamma$ on the input side agrees with $\beta|\gamma$ on the output side.) For the remaining cases, costs are automatically determined from data as described next in Chapter 4. For now, turning back to Figure 2-1, it is seen that the total cost or approximation error, $J(u, \hat{u})$, for the example inputs and outputs is:

$$J(u, \hat{u}) = S_{\beta(\alpha:\gamma)\rightarrow\beta(\alpha:\phi)} + C_{\beta]|[\gamma} + S_{\gamma(\beta:\delta)\rightarrow\gamma(\psi:\delta)} \tag{2.2}$$

Since $J(u, \hat{u})$, a sum of description lengths, is minimized, this setup can also be regarded as a minimum description length (MDL) formulation. The connection here to MDL literature is interesting, because by minimizing the approximation error, unit selection will tend towards a minimum-error transmission channel. When there are no errors in transmission, all the transition symbols have arrived without corruption, which means that all the units were selected contiguously from the corpus. Perfect reconstruction of the input specification with speech recorded in a corpus is the best-case scenario, since the output will be natural by definition as it represents a stretch of speech spoken entirely in succession by a human.

The triphone encoding of unit context (left and right neighbors) described in the foregoing discussion is entirely arbitrary and does not affect the generality of the formulation. However, to illustrate further parameter reductions for the purpose of practical implementation under this triphone context model, another independence assumption can be made, where substitution costs for triphones are decoupled into left-sided and right-sided components. That is, when $\beta(\hat{\alpha} : \hat{\gamma})$ is used in place of $\beta(\alpha : \gamma)$, the total substitution cost is:

$$S_{\beta(\alpha:\gamma)\rightarrow\beta(\hat{\alpha}:\hat{\gamma})} = S^{l}_{[\alpha]\beta\rightarrow[\hat{\alpha}]\beta} + S^{r}_{\beta[\gamma]\rightarrow\beta[\hat{\gamma}]} \tag{2.3}$$

Equation 2.2 can now be simplified with this decoupling relation in Equation 2.3 producing the final cost function, $J(u, \hat{u})$, used from hereon in.

$$\begin{aligned} J(u, \hat{u}) &= S^{l}_{[\alpha]\beta\rightarrow[\alpha]\beta} + S^{r}_{\beta[\gamma]\rightarrow\beta[\phi]} + C_{\beta]|[\gamma} + S^{l}_{[\beta]\gamma\rightarrow[\psi]\gamma} + S^{r}_{\gamma[\delta]\rightarrow\gamma[\delta]} \\ &= S^{r}_{\beta[\gamma]\rightarrow\beta[\phi]} + C_{\beta]|[\gamma} + S^{l}_{[\beta]\gamma\rightarrow[\psi]\gamma} \end{aligned}$$

Note that since the left context of β, α, and the right context of γ, δ, survived transmission, as depicted in Figure 2-1, the substitution costs, $S^{l}_{[\alpha]\beta\rightarrow[\alpha]\beta}$ and $S^{r}_{\gamma[\delta]\rightarrow\gamma[\delta]}$, are identically zero. This contextual reduction of a triphone into a left-sided biphone (e.g., $[\alpha]\beta$) and a right-sided biphone (e.g., $\beta[\gamma]$) is similar to context modelling carried out by Serridge in speech recognition experiments [127].

If the units - described here with Greek letters - contain information across multiple spatial levels, as described in introductory discussion in Section 2.1, there is the potential for additional combinatorial expansion [153] in the number of units. The practical way to deal with this is to decouple the costs and recombine them with addition, as seen here in this example of simplifying triphone costs to biphone costs. Then, for example, intonation symbols would be seen in the input and output streams alongside with unit symbols describing acoustic-phonetic properties. Costs describing mismatch in intonation would also be applied, perhaps, with linear combination. A more visual representation of applying costs at different spatial levels will be described in Chapter 5 with the finite-state transducers.

2.4 Parameter reduction: linearization

This section introduces a very general technique for parameter reduction when distances between individual instances are approximated by distances between distributions or clusters of individual instances. If the space in which the instances lie is well partitioned by the clusters, then the characteristics of each cluster (i.e., location and spread) need only to be determined once. When a new instance is collected, its relation to instances already present is known once its cluster identity is established. Energy[1] devoted to determining the optimal set of clusters is amortized over new, incoming data. The following discussion is tailored to unit selection, but there should be little loss of generality in this choice of application.

In a corpus with M units, a complete characterization of all concatenations (i.e., each unit on the left and right sides of a concatenation) would require M^2 parameters. As new units are incrementally added to this corpus, an entire sweep through the old units is required to calculate the values for the additional parameters. For example,

[1]It would be inaccurate to talk about time that is amortized. Energy, which is the product or combination of work and time, is the more accurate notion.

incrementally adding just one more unit requires $(2M + 1)$ new parameters - there are M concatenations with the new unit on the left of the concatenation boundary, M on the right, and 1 with itself - for a grand total of $M^2 + (2M + 1) = (M + 1)^2$ parameters. For corpora with a large number of units, such an approach does not scale and a method for mitigating the growth of parameters is necessary. The method presented here groups units into a reduced number of clusters which are then further analyzed for parameter calculations. The clusters are assumed to be a representative collection which can be reused as new units are incrementally added to the corpus. While this clustering approach trades off accuracy for speed, the appropriateness of such a tradeoff can be decided by examining the correlation between the true and approximate parameter values.

Consider a space where units of a corpus are represented by points. Figure 2-2 depicts data points which have been grouped into two distinct clusters, A and B, whose member points are marked by triangles and pentagons. The concentric ellipses denote the orientation and the shape of the cluster distributions. The centroids of clusters A and B are pointed to by $\bar{a}$ and $\bar{b}$, respectively. Perturbations from the centers, $\tilde{a}$ and $\tilde{b}$, respectively determine the locations of points a and b relative to their cluster centroids. In other words, it holds that $a = \bar{a} + \tilde{a}$ and $b = \bar{b} + \tilde{b}$.

Determination of the concatenation parameters for all such a and b pairs is a wieldy task as described above. By making a series of approximations, the number of parameters can be made linear with respect to the number of points. Since the triangle inequality states that $||x + y|| \leq ||x|| + ||y||$, then the norm of the sum is decoupled into a sum of norms and pairwise interactions (i.e., summation) need not be considered. The following set of equations list the series of steps beginning from the original comparison between a and b and ending at a final form involving only cluster comparisons and perturbation norms or lengths. Note that an alternate form of the

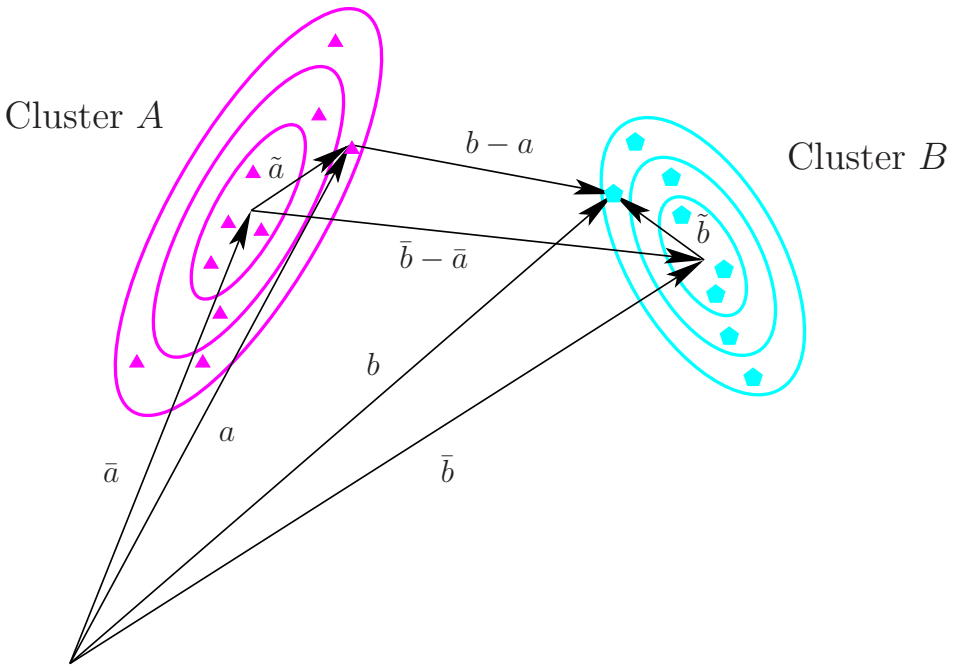


Figure 2-2: Cluster modelling framework for parameter reduction.

Points, a and b, are located at positions determined by perturbations, $\tilde{a}$ and $\tilde{b}$, from cluster centroids, $\bar{a}$ and $\bar{b}$. Via two repeated applications of the triangle inequality, the true distance between a and b, $||a-b||$, is decomposed into an approximate distance represented by the sum of the distance between the centroids, $||\bar{a}-\bar{b}||$, and the perturbation lengths, $||\tilde{a}||$ and $||\tilde{b}||$.

triangle inequality, $||x - y|| \leq ||x|| + ||y||$, is employed in the second application.

$$\begin{aligned} || \, b - a \, || &= || \left(\bar{b} + \tilde{b}\right) - (\bar{a} + \tilde{a}) \, || \\ &= || \left(\bar{b} - \bar{a}\right) + \left(\tilde{b} - \tilde{a}\right) || \\ &\leq || \, \bar{b} - \bar{a} \, || + || \, \tilde{b} - \tilde{a} \, || \\ &\leq || \, \bar{b} - \bar{a} \, || + || \, \tilde{b} \, || + || \, \tilde{a} \, || \end{aligned} \tag{2.4}$$

Although pairwise comparisons are required between the clusters, this is a computation of fixed cost performed just once if the clusters are representative of a language and can be regarded small especially if the number of clusters is significantly less than the number of units in the corpus. Any metric which satisfies the triangle inequality is applicable including the Euclidean and Mahalanobis distances. The former is a simple inner product on a vector space, while the latter considers possibly weighting and rotating dimensions of the vector space. In the equation below, the Mahalanobis

distance is specified with a *concentration matrix* of Σ^{-1}, which parallels the notation in describing Gaussian random variables to be seen shortly. The Euclidean distance is a special case of the Mahalanobis distance when $\Sigma^{-1} = I$, where I is the inverse matrix.

$$\begin{aligned} || \ x \ ||_E &= x^T x \\ || \ x \ ||_M &= x^T \Sigma^{-1} x \end{aligned}$$

The performance of the above approximation in Equation 2.4 can be evaluated by calculating its bias with respect to the true expected value. In particular, consider a multi-variate, d-dimensional Gaussian random variable, X, which is distributed according to mean, μ, and variance, Σ.

$$p_{X \sim \mathcal{N}(\mu,\Sigma)}(x) = \left|(2\pi)^d \Sigma\right|^{-\frac{1}{2}} e^{-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)}$$

Using a formula [116] for expected Mahalanobis distances[2], an expression for the expected Mahalanobis distance between a pair of points is obtained. Note that the joint expectation over X and Y is evaluated with an iterated expectation and that the means and covariances for X and Y are, respectively, m_X and S_X and m_Y and S_Y.

$$\begin{aligned} E\left[|| \ x - y \ ||_M\right] &= E_{X,Y}\left[(x-y)^T \Sigma^{-1} (x-y)\right] \\ &= E_X\left[E_{Y \sim \mathcal{N}(m_Y, S_Y)|X=x}\left[(x-y)^T \Sigma^{-1} (x-y) \, | X = x\right]\right] \\ &= E_{X \sim \mathcal{N}(m_X, S_X)}\left[(x-m_Y)^T \Sigma^{-1} (x-m_Y) \ + \ Tr\left[\Sigma^{-1} S_Y\right]\right] \\ &= (m_X - m_Y)^T \Sigma^{-1} (m_X - m_Y) \ + \ Tr\left[\Sigma^{-1} S_X\right] \ + \ Tr\left[\Sigma^{-1} S_Y\right] \end{aligned} \tag{2.5}$$

Next, the approximation to the true pairwise distance can be calculated in a similar fashion. Relating the former notation of cluster centroids and perturbations to the current description of multi-variate Gaussian means and variances produces the

[2] $E_{X \sim \mathcal{N}(m,S)}\left[(x-\mu)^T \Sigma^{-1} (x-\mu)\right] = (m-\mu)^T \Sigma^{-1} (m-\mu) \ + \ Tr\left[\Sigma^{-1} S\right]$

following equation:

$$\begin{aligned} || \, x - y \, ||_M &\approx || \, m_X - m_Y \, ||_M \; + \; || \, x - m_X \, ||_M \; + \; || \, y - m_Y \, ||_M \\ &= (m_X - m_Y)^T \Sigma^{-1} (m_X - m_Y) \; + \; (x - m_X)^T \Sigma^{-1} (x - m_X) \; + \\ &\quad (y - m_Y)^T \Sigma^{-1} (y - m_Y) \end{aligned} \tag{2.6}$$

The right-hand side of this equation consists of three terms, of which the first is deterministic. For the latter two terms, the joint expectation simplifies to a marginal expectation (over different variables) for each term.

$$\begin{aligned} E\left[|| \, x - y \, ||_M\right] &\approx (m_X - m_Y)^T \Sigma^{-1} (m_X - m_Y) \; + \\ &\quad E_{X \sim \mathcal{N}(m_X, S_X)} \left[(x - m_X)^T \Sigma^{-1} (x - m_X) \right] \; + \\ &\quad E_{Y \sim \mathcal{N}(m_Y, S_Y)} \left[(y - m_Y)^T \Sigma^{-1} (y - m_Y) \right] \\ &= (m_X - m_Y)^T \Sigma^{-1} (m_X - m_Y) \; + \\ &\quad (m_X - m_X)^T \Sigma^{-1} (m_X - m_X) \; + \; Tr\left[\Sigma^{-1} S_X\right] \; + \\ &\quad (m_Y - m_Y)^T \Sigma^{-1} (m_Y - m_Y) \; + \; Tr\left[\Sigma^{-1} S_Y\right] \\ &= (m_X - m_Y)^T \Sigma^{-1} (m_X - m_Y) \; + \; Tr\left[\Sigma^{-1} S_X\right] \; + \; Tr\left[\Sigma^{-1} S_Y\right] \end{aligned} \tag{2.7}$$

The result from this analysis shows that, on average, values of the true distance as defined in Equation 2.5 can be substituted with values of the approximate distance as defined in Equation 2.7 with no apparent loss. This definition is very intuitive as it decomposes the distance between two instances into the sum of distances to their respective cluster centroid (intra-cluster distance) and distances between their respective clusters (inter-cluster distance).

If information about the true locations of a and b is incomplete, then the best way to cope in the absence of any further information is to set $|| \, \tilde{a} \, ||$ and/or $|| \, \tilde{b} \, ||$ to zero. This is the Principle of Maximum Entropy. Whether equivalent expressions can be derived by working from first principles would be an object of future interest. By setting either of the perturbation distances to zero, heterogeneous comparisons are obtained: the distance from an instance to a cluster is the distance of the instance to

its cluster (intra-cluster distance) plus the distance between the clusters (inter-cluster distance).

With K clusters and singular cluster membership for each of the M units, there are M intra-cluster distances and K^2 inter-cluster distances bringing the number of parameters to a grand total of $M + K^2$. When the number of clusters is significantly fewer than the number of units ($K << M$), the savings over the original case of M^2 parameters can be quite large.

2.5 Summary

This chapter has laid out definitions and motivations for unit selection, a communication-theoretic framework for modelling the process, and a groundwork for practical implementation.

The expansionary factors of combinatorial proportions in unit selection highlight the need for sequential decision making algorithms. A particular one, dynamic programming using a Viterbi heuristic, will be described in Chapter 5. A natural fit for finite-state transducers, only symbolic information will be used in the first stage to find the optimal sequence. Then, a second stage using numeric information will be used to re-prioritize or re-sort the top hypotheses. In such a multi-stage setup, the first stage can be regarded as a coarse search which prunes away irrelevant and sub-optimal portions of the search space for a finer search by the second stage.

Modelling the unit selection process with a communication-theoretic formulation paves the way for modelling the costs and it gives clues as to how the costs should be determined from data. Although the parameter reduction method presented here in this chapter assumes a metric obeying the Triangle Inequality, metrics to be examined in Chapter 4 may break this mold in the interest of preserving accordance with the terminology of symbol corruption in communication theory. More specifically, a

generalization of Mahalanobis distance called Kullback-Leibler distance will be used as the workhorse for automatically deriving costs from data.

The nomenclature of substitution and concatenation costs can be slightly generalized to describe the costs of being somewhere and of going somewhere. Calling them unit and transition costs is another suitable way of referring to them. The unit cost is the suitability of using a unit, or being in the state corresponding to that unit. The transition cost is the suitability of moving to another unit, or moving to the state corresponding to that other unit.

By judicious choice of unit and transition costs, other applications may arise. The data sets for the experiments described in Chapter 3 are constructed by using an interactive tool to discover words matching certain lexical patterns. In this case, the beginnings and ends of words are connected to initial and final states, unit costs are set to $\emptyset$, and transition costs within and between words are set to $\emptyset$ and ∞, respectively, to heavily constrain the search - only entire words can be found and switching between words is disallowed as it would create new, and possibly, nonsense, words. Permitting switching could lead to other applications such as pronunciation generation (i.e., letter-to-sound) for novel words [81, 86] which could be used to address out-of-vocabulary (OOV) issues. The inverse task of recognizing novel words (i.e., sound-to-letter) would be another possibility.

Chapter 3

Analysis of sonorant sequences

3.1 Introduction

Previous work by the author on unit selection [162] studied speech production and discovered that by making concatenations at places of source change (i.e., from voiced to unvoiced speech as in vowel to fricative) concatenation artifacts could be minimized. What was also revealed was an inadequate understanding of whether concatenations could be made in running voiced speech (e.g., between vowels and semivowels.) This chapter describes experiments wherein the framework of unit selection introduced in the last chapter is used to pre-select a small corpus of words containing only vowels and semivowels and to subsequently select optimal sub-sequences for constructing novel speech. The rationale behind the pre-selection of such a corpus lies in the following distributional analysis of the English language inspired by Shipman's earlier work [128]. The distributional analysis is used as a way to guide corpus development.

3.2 Distributional properties

An analysis of the English language was conducted [162] using a 90,000-word lexicon from the Linguistic Data Consortium called the *COMLEX English Pronunciation Dictionary* [89], commonly referred to as PRONLEX. The analysis was limited to the non-foreign subset of PRONLEX containing approximately 68,000 words. Focusing on contiguous vowel (16 total without lexical stress, /iʸ/, /ɪ/, /eʸ/, /ɛ/, /æ/, /ɑ/, /ɑʷ/, /ɑʸ/, /ɔ/, /oʷ/, /ɔʸ/, /uʷ/ / ʊ/, /ʌ/, /ə/, /ɝ/) and semivowel (4 total, /ɾ/, /l/, /w/, /y/), sequences and their occurrences within words were enumerated as shown in Table 3.1. The International Phonetic Alphabet (IPA) is used to write phonetic symbols [73].

Length	Example	# of units	% cumulative	# of occurrences	% cumulative
1	/ɪ/	16	0.01	100,513	63.3
2	/rɪ/	167	10.7	40,778	89.0
3	/ərɪ/	484	39.0	12,916	97.1
4	/yəleʸ/	637	76.3	3,392	99.3
5	/oʷriʸəl/	312	94.6	906	99.8
6	/yələrɑʸ/	80	99.2	226	100.0
7	/ærəlɛlɪ/	13	100.0	21	100.0
Total		1,709	100.0	158,752	100.0

Table 3.1: Analysis of vowel and semivowel sequences in PRONLEX.

An analysis of contiguous sequences of vowels and semivowels within words was performed on the non-foreign subset of PRONLEX containing about 68,000 words. Including sequences of up to length 3, 97.1% of all occurrences are covered by only 39.0% of all units.

Sequences of length 1 are trivially the 16 vowels in English as the sequences were constrained to contain one vowel at a minimum. When sequences of length 2 are considered, a total of 167 units - an empirical number that is small compared to the theoretical number, $(16 + 4)^2 = 400$ - are added which accounts for 89% of the cumulative occurrences; with only approximately one-tenth of the units, 89% of occurrences are accounted for. At a sequence length of 3, a large majority (97.1%) of the occurrences is cumulatively observed with still only 39% of the units. These economical operational points of 2 and 3 entreat further analysis. Table 3.2 shows the

breakdown of vowel and semivowel sequences of length 2, where S denotes semivowel, and V denotes vowel. While the SV set (62 units) and the VS set (29 units) are empirically close to their theoretical limits of 64 and 32, respectively, the VV set (76 units) is around one quarter of $16^2 = 256$.

Structure	Finer structure	# of units	%	# of occurrences	%
SV		62		25,416	
	rV	15	24.2	13,535	53.3
	lV	16	50.0	7,700	83.5
	wV	16	75.8	2,764	94.4
	yV	15	100.0	1,417	100.0
VS		29		12,147	
	Vl	16	55.5	7,226	59.5
	Vr	13	100.0	4,921	100.0
VV		76		3,215	
	ʸV	36	47.4	2,543	79.1
	ʷV	22	76.3	500	94.7
	ɝV	9	88.2	143	99.1
	ɔV	1	89.5	12	99.5
	ɑV	3	93.4	8	99.7
	əV	3	97.4	7	99.9
	ʌV	1	98.7	1	100.0
	ɪV	1	100.0	1	100.0
Total		167	100.0	40,778	100.0

Table 3.2: Analysis of vowel and semivowel sequences of length 2 in PRONLEX.

Vowel and semivowel sequences of length 2 can be divided into SV, VS, and VV subsets. The SV and VS sets are empirically close to their theoretical limits, whereas the VV set is much smaller relatively. However, the four top ranking members of the VV set could be considered to belong to the SV set as well, because the left vowel acts like a semivowel. From this viewpoint, a majority of the sequences of length 2 can be represented by a super-SV set.

What is especially striking about the VV set is its substructure: the four top members with the most number of occurrences, ʸV, ʷV, ɝV, and ɔV, are all related to semivowels. ʸV and ʷV refer to the first vowel being a diphthong ([iʸ], [eʸ], [ɑʷ], [ɑʸ], [oʷ], [ɔʸ], and [uʷ] were all represented) with a palatal and bilabial offglide, respectively. It could be argued that ɝV is related to the rhotic liquid ([ɝ] is an extreme version of [r]) and that ɔV is related to the lateral liquid ([l] is an extreme version of [o] which is quite close to [ɔ].) These four top-ranking members of the VV set account for 89.5% of VV units and 99.5% of the occurrences of VV units. Figure 3-1 shows

how the substructure of the VV set is dominated by the first four members.

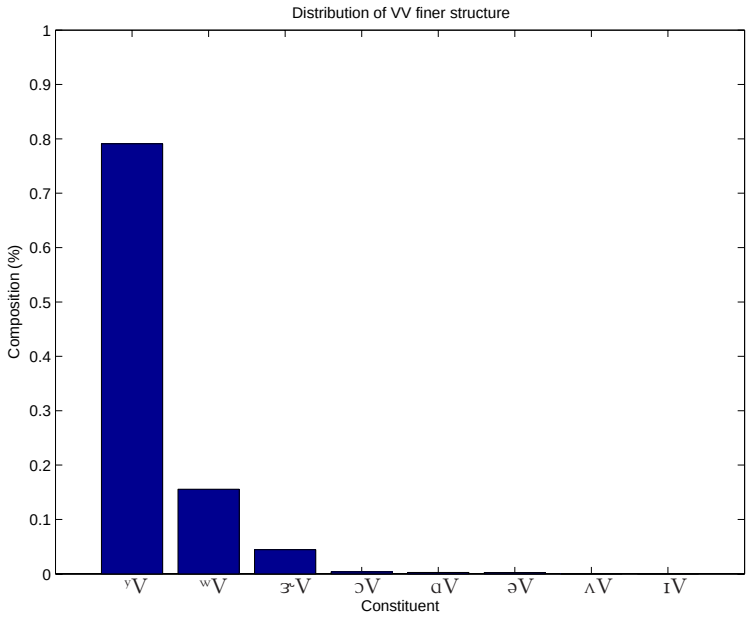


Figure 3-1: Distribution of VV finer structure.

The SV set accounts for 25,416 occurrences, or 62.3% of the total number of occurrences, 40,778, of sequences of length 2. If the four top-ranking members of the VV set, ʸV, ʷV, ɜˑV, and ɔV, are also regarded as part of the SV set due to arguments above, then their additional occurrences of $2543 + 500 + 143 + 12 = 3198$ brings the total occurrences in the super-SV set up to 28,614, or 70.2%. The membership of the SV set increases from 62 to a grand total of 120 $(62+36+22+9+1)$ members, which account for 71.9% of the 167 units that are sequences of length 2.

Drilling down one level further, analysis of the ʸV reveals that the top four members, iʸɪ, iʸɜˑ, iʸə, and iʸeʸ - all starting with iʸ - account for 762, 453, 316, and 218 occurrences, respectively (numbers not separately shown in a table.) These top 4 of the 36 members in the ʸV set have a total of $762 + 453 + 316 + 218 = 1749$ occurrences and account for 68.7% of the 2,543 occurrences within the ʸV set and for 54.4% of the 3,215 occurrences within the VV set.

What the foregoing suggests is that if it were possible to make a concatenation between a semivowel and vowel - to sever their link as it were - a majority (89.0%) of

the cumulative occurrences of sequences of length 2 in the English language could be covered. It is tempting to conjecture that the members of the pseudo-SV subset of the VV set are actually highly evolved SV units in disguise! If this conjecture holds, then perhaps units from the pseudo-SV subset of the VV set can be spliced at the place of maximal constriction. What remains to be answered is the question of how sequences of length 3 break down.

As shown in Table 3.3, a coarser analysis of sequences of length 3 reveals an interesting pattern, that the VSV set accounted for 80.6% of the occurrences. It was conjectured [162] that the sequences of length 3 could be compactly covered with sequences of length 2. More specifically, that VS and SV constituents could be merged to form VSV units - sub-phonetic concatenations within a semivowel would not be detectable. Also conjectured was that SVS could not be similarly constructed because their composition suggests an entire syllable - sub-phonetic concatenations within a vowel would be detectable. The implications of this are that if most (80.5%) of the sequences of length 3 could be synthesized from sequences of length 2, then recording 10.7% (up to length 2) of the units covers up to 97.1% of occurrences (up to length 3). Such savings imply that a compact corpus would serve well in most cases.

Structure	# of occurrences	% cumulative
VSV	10,404	80.6
SVV	1,163	89.6
SVS	1,145	98.4
VVS	174	99.8
VVV	24	100.0
VSS	6	100.0
Total	12,916	100.0

Table 3.3: Analysis of vowel and semivowel sequences of length 3 in PRONLEX.

Enumerating vowel and semivowel sequences of length 3 shows that the VSV set represents a majority of the occurrences. If VSV units can be formed by concatenating VS and SV units, then most of the sequences of length 3 can be reproduced by sequences of length 2.

As described next, tools whose formulation are based on unit selection can be used to pre-select a corpus containing words purely formed from contiguous vowel and semivowel sequences. Recordings of these words are used as raw materials for investi-

gating the hypotheses suggested by above analyses. Novel VSV units are synthesized from constituent VS and SV units. The pseudo-SV members in the VV set are also spliced.

The distributional analysis performed here considered contiguous sequences of vowels and semivowels. Not considered were contiguous sequences of only vowels. It may be that longer sequences of vowels, V^N, can be decomposed into smaller VV sequences. In a separate analysis of PRONLEX shown in Table 3.4, it was found that almost 90% of the occurrences of multi-vowel sequences had the diphthongs, [iʸ], [ɑʸ], [uʷ], and [oʷ], as its first vowel, with [iʸ] accounting for almost two-thirds of the total occurrences.

Phone	# of occurrences	% of occurrences
[iʸ]	2,803	65.02%
[ɑʸ]	458	10.62%
[uʷ]	312	7.24%
[oʷ]	255	5.92%
[ɜˠ]	150	3.48%
[eʸ]	138	3.20%
[ɑʷ]	110	2.55%
[ɔʸ]	46	1.07%
[ɔ]	15	0.35%
[ə]	10	0.23%
[ɑ]	10	0.23%
[ʌ]	2	0.05%
[ɪ]	1	0.02%
[ɛ]	1	0.02%
Total	4,311	100%

Table 3.4: Analysis of multi-vowel sequences in PRONLEX.

When contiguous sequences of vowels are enumerated in PRONLEX, 90% of the occurrences consist of sequences which start with the diphthongs, [iʸ], [ɑʸ], [uʷ], and [oʷ]. In fact, sequences beginning [iʸ] account for almost two-thirds of the total occurrences.

3.3 Crystal tool

Figure 3-2 depicts one particular embodiment of the unit selection framework presented here in the form of an interactive tool, CRYSTAL, which expedites lexicon anal-

ysis and discovery using pattern matching with regular expressions. Search queries can be stated in the form of phonemic, syllabic, lexical stress, or graphemic (letter) patterns. Regular expressions enable the specification of alternatives via the boolean OR (|) operator and of repetition via Kleene star or closure (e.g., the regular expression '$(a|b) * c$' matches strings such as '*abac*'.) Successive applications of regular expression lead to the discovery of smaller subsets of words sharing common structure. Words matching the search query are reported with their pronunciation, stress pattern, and a frequency and rank given by the Brown Corpus [72]. This tool, incidentally, has become part of the laboratory and teaching materials in the Automatic Speech Recognition course, a graduate-level subject numbered 6.345 at MIT.

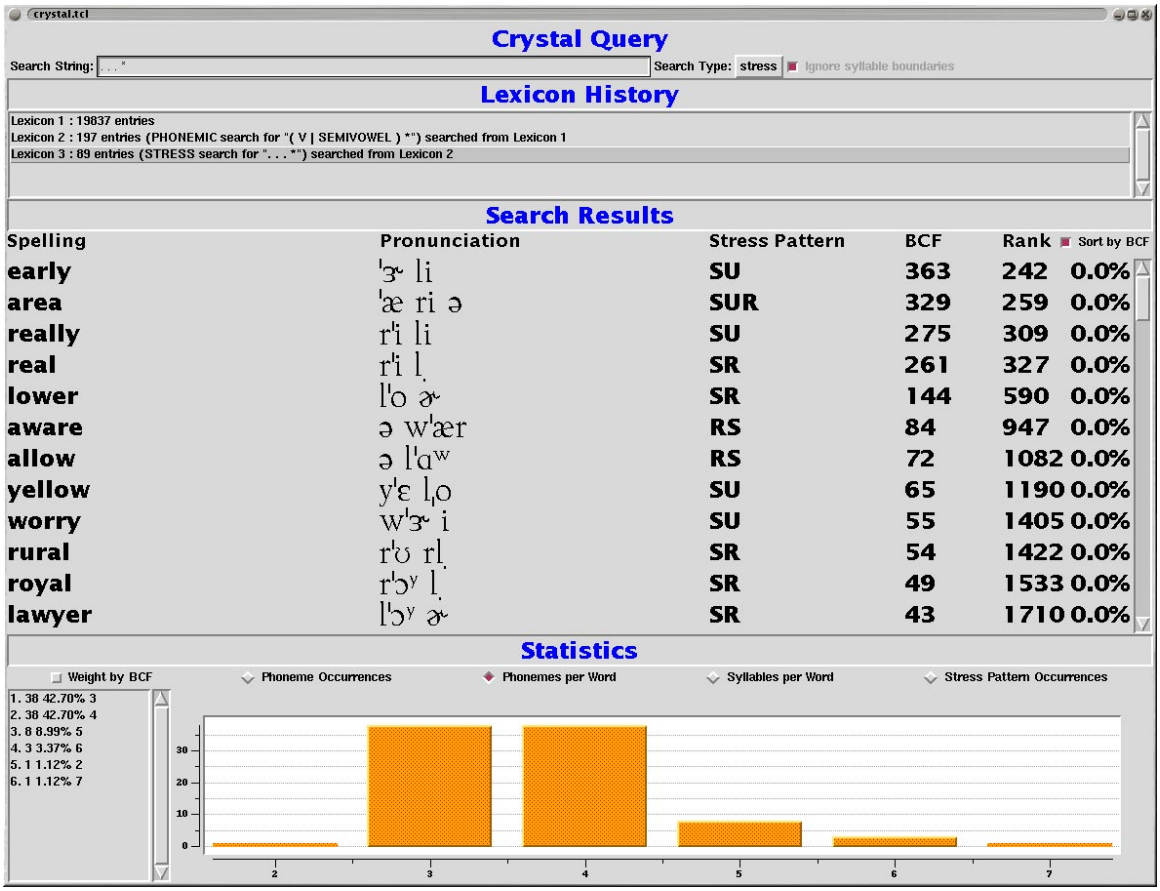


Figure 3-2: Screenshot of the CRYSTAL tool.

Beginning with the 19,837 words in the Merriam-Webster Pocket Dictionary of 1964, a reduced set of 197 words is obtained by retaining only words purely composed of vowels and semivowels. By only allowing poly-syllabic words from the Brown Corpus, a final set of 37 two-syllable and 6 three-syllable was recorded by a human talker.

This particular re-implementation of the original tool, *AleXiS*, designed by Kassel [165] and later ported from LISP to C by Peter Daly was performed by the author using finite-state transducers (FST's), which are described in Appendix A and which receive additional treatment in Chapter 5. This Lexical access given partial phonetic information has been applied to speech recognition in the past [62]. The usage of an FST framework for implementation is a natural choice, because regular expressions are finite-state acceptors, a subset of finite-state transducers. In this application, it is desired to locate only words matching a search criterion and not to create nonsense words outside of the vocabulary. As such, substitution and concatenation costs were set to pathological values of $\emptyset$ and ∞, respectively.

Beginning with the 19,837 words in the Merriam-Webster Pocket Dictionary of 1964, an initial search for words containing only vowel and semivowel phonemes located a total of 197 (a 100-fold reduction.) By focusing on words containing more than one syllable, a smaller set of 89 words (a 2-fold reduction) was determined. A further restriction was made by only allowing words from the Brown Corpus. This final set contained 37 two-syllable and 6 three-syllable words which were spoken by a human talker and recorded. A graph of the distributional characteristics in Figure 3-2 shows an abundance of words consisting of three and four vowels and semivowels. When ranked by their Brown Corpus frequency, the top-ranking words have three or more phonemes. This concentration is also confirmed from Table 3.1 which shows counts of 483 and 637 for words consisting of three and four vowels and semivowels, respectively. Table 3.5 lists all 43 words and their pronunciations in the order of number of syllables.

3.4 Sub-phonetic: manual experiments

Because the formation of lateral pathways in the vocal tract can result in spectral changes [47], it was hypothesized in a fashion similar to earlier treatment of source changes that a splice may not be perceptible at this point and that speech sounds

airy	eʸ riʸ	era	iʸ rə	rally	ræ liʸ	willow	wɪ loʷ
allay	ə leʸ	error	ɛr ɝ	real	riʸ l̩	wily	wɑʸ liʸ
alley	æ liʸ	laurel	loʷr əl	really	riʸ liʸ	worry	wɝ riʸ
allow	ə lɑʷ	lawyer	lɔʸ ɝ	relay	riʸ leʸ	yellow	yɛ loʷ
ally	æ lɑʸ	leeway	liʸ weʸ	rely	riʸ lɑʸ		
array	ə reʸ	leo	li oʷ	roller	roʷ lɝ	aerial	eʸ riʸ ə
arrow	æ roʷ	liar	lɑʸ ɚ	ruler	ruʷ lɝ	arroyo	ə rɔʸ oʷ
aware	ə wer	lower	loʷ ɝ	rural	rʊ rəl	oriole	oʷ ri oʷl
awry	ə rɑʸ	loyal	lɔʸ əl	wary	weʸ riʸ	warily	weʸ rə liʸ
early	ɝ liʸ	oral	oʷ rəl	weary	wiʸ riʸ	warrior	woʷ riʸ ɝ

Table 3.5: Multi-sonorant sequences.

When analyzing the Merriam-Webster Pocket Dictionary of 1964 for words entirely composed of contiguous vowels and semivowels, there are 37 two-syllable and 6 three-syllable words.

could be naturally concatenated at this boundary. In working with pre-vocalic [l] sounds the possibility of concatenating at these syllable boundaries became apparent. For example, the first syllable, [æ], of the words, "ally", "alley", and "alloy", were found to be interchangeable. Hardly any concatenation artifacts were perceived when care was taken to splice: 1) at waveform zero-crossings which demarcate pitch period boundaries, 2) at the first pitch period whose the amplitude is abruptly reduced due to the lateral constriction of the pre-vocalic [l], and 3) at a pitch period where pitch is continuous across the concatenation. The locations of the pitch periods were manually marked in these manual experiments.

Figure 3-3 shows the word, 'alloy', and the location of the pitch period where the concatenation would be applied. The time cursor is placed at the boundary between [æ] and [l] where the waveform amplitude drops abruptly. A zoomed-in portion of the waveform can be seen in the top pane; more detail can be shown at the expense of the alignment of the top and bottom time cursors, because the time scale is smaller. This screenshot and others following are taken from an interactive tool named Concatenation View (cv) based on the SAPPHIRE framework [56] that allows for the cumulative construction of synthesized speech from user-selected segments of utterances in a given corpus. The formants are tracked with a software package called ESPS+ [143] and the formant tracks have been resampled to a pitch-synchronous

rate. Pitch periods spread out further in time correspond to a lower pitch, while pitch periods packed more densely in time correspond to a higher pitch. The pitch tracking contains some errors as the pitch drops towards the end of the word in the vowel, [ɔʸ], as evidenced by the occasional missing or spurious pitch period.

In examining spectrograms of inter-vocalic glide ([w] and [y]) and retroflexed ([r]) sounds, regions of extreme articulation in which spectral prominences - sometimes referred to as formants - reach extrema during the evolution of their trajectories could be consistently located. Because an earlier hypothesis had stated that areas of extreme articulation or low energy (these two criteria may be correlated) may outline places where splicing can occur without much perceptual distortion, further concatenations were performing at the middle point of [r], [w], and [y] sounds. For example, a natural-sounding 'aerial' could be reconstructed by replacing the start of 'aerial' as uttered by a human speaker with the [eʸ] vowel and the first half of the [r] liquid from 'airy'. The optimal cross-over point was determined by the minimum point reached by the second spectral prominence. In the word 'airy' and 'aerial' shown in Figures 3-4 and 3-5, respectively, the time cursor is placed where the second spectral prominence reaches a minimum. By using the first syllable of 'airy' and the ending syllables of 'aerial', a synthetic 'aerial' is created in Figure 3-6. Although the third spectral prominence is not perfectly aligned at the concatenation boundary, this does not contribute to a perceptual artifact in an informal listening test.

In these two examples of 'alloy and 'aerial', the first part of word A and the second part of word B is used to generate a synthetic version of word B. Although the result is only half-synthetic, technically speaking, what is demonstrated is that the extreme articulation loci reached in the evolution of liquid production demarcate possible boundaries for concatenation.

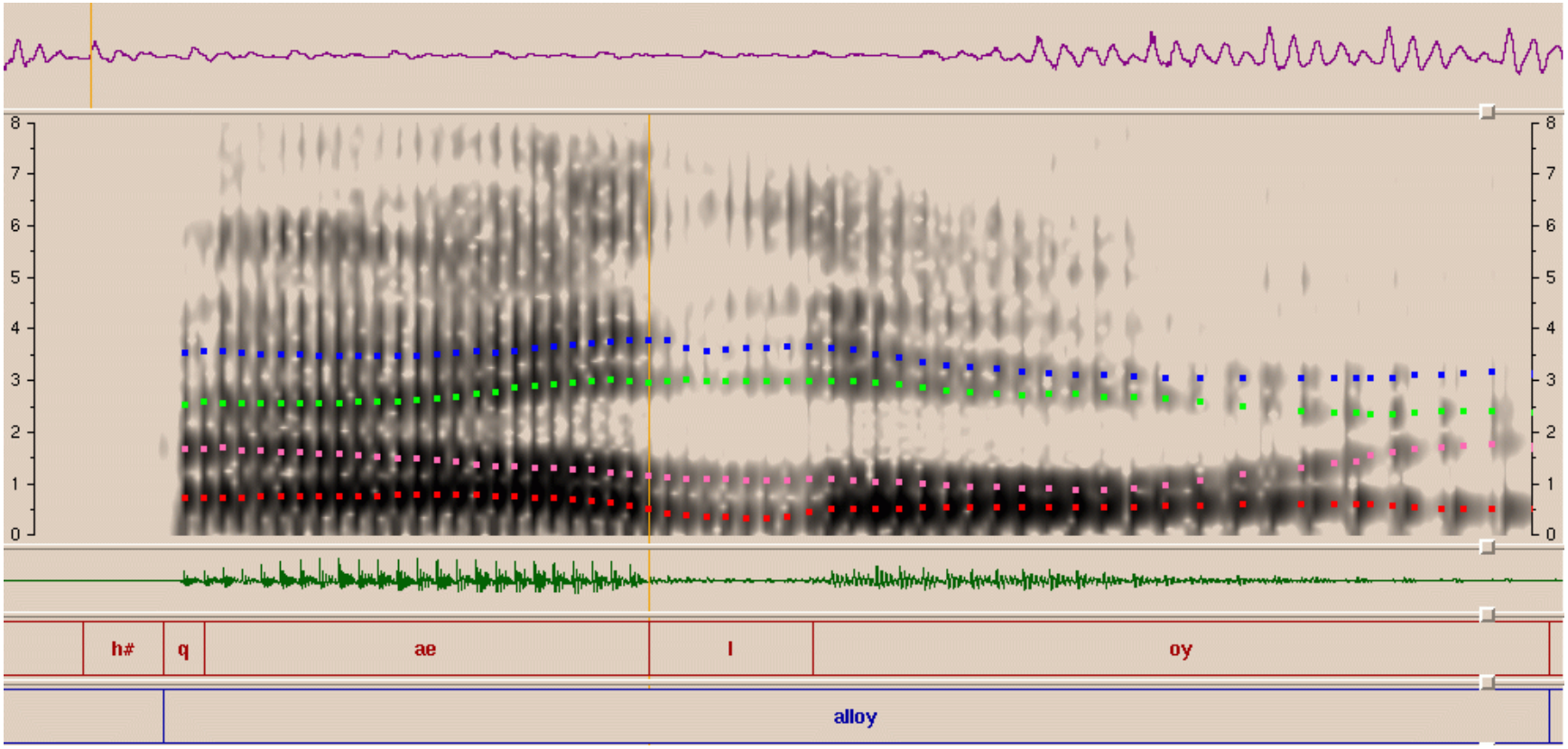


Figure 3-3: Screenshot of the word 'alloy' taken from *cv* tool.

The time cursor is placed at the boundary between [æ] and [l] where the amplitude of the waveform drops abruptly corresponding to the time of lateral constriction. Listening has shown that these locations can be used as splice points and that concatenations made at such points are indiscernible provided that the pitch is continuous. The *cv* tool displays overlaid formant tracks which are sampled at a pitch-synchronous rate (i.e., variable frame rate.) The pitch tracker makes some errors as the pitch drops towards the end of the word in the vowel, [ɔʸ].

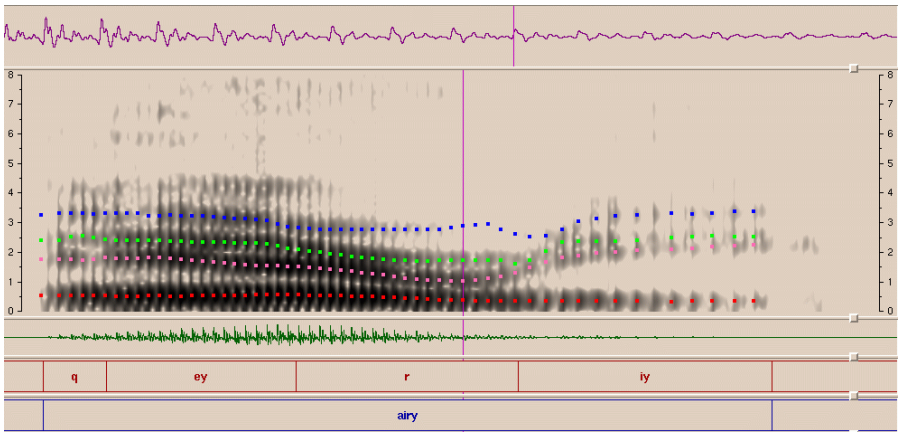


Figure 3-4: Screenshot of the word 'airy' taken from *cv* tool.

The first half of the word 'airy' is selected by splicing in the semivowel, [r], where the second spectral prominence (marked by the magenta color) reaches a minimum in frequency.

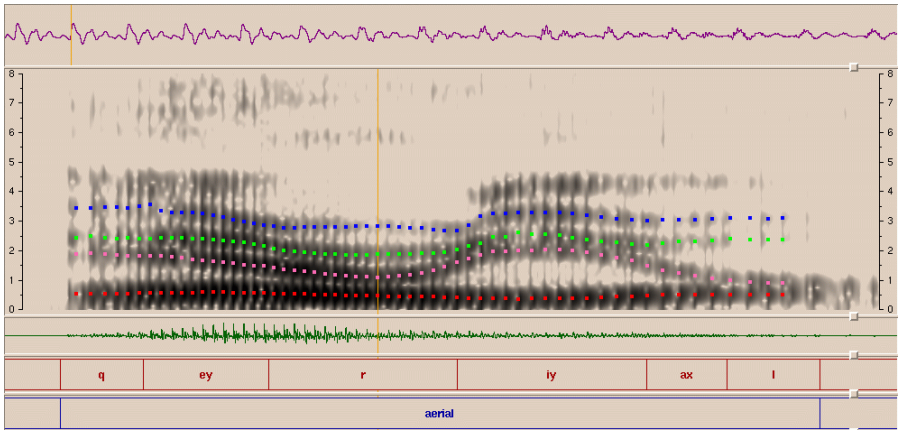


Figure 3-5: Screenshot of the word 'aerial' taken from *cv* tool.

The second half of the word 'aerial is selected by splicing in the semivowel, [r], where the second spectral prominence (marked by the magenta color) reaches a minimum in frequency.

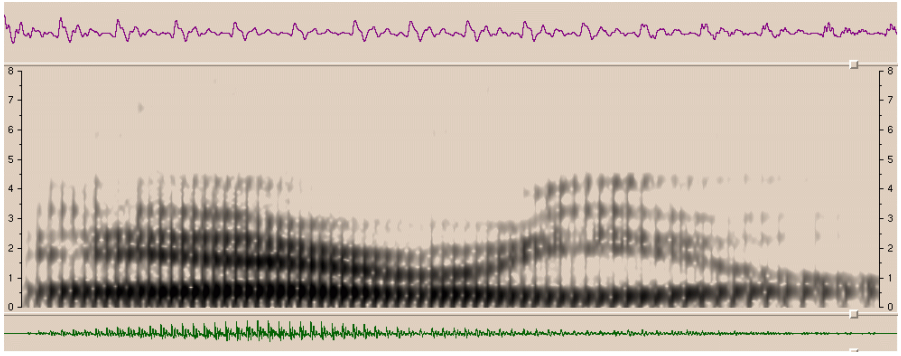


Figure 3-6: Screenshot of the synthetic word 'aerial' taken from *cv* tool.

A synthetic word 'aerial' is created by substituting the first half with the word 'airy'; the second half remains the same. A slight discontinuity in time is noticeable as the third spectral prominence does not line up perfectly. However, this is hardly noticeable in a listening test.

As vowel diphthongs can be realized with [w] or [y] off-glides resulting from bilabial or palatal constriction, respectively, a further source of concatenation points was discovered by working with glides in a diphthong context. This also opens a door to the possibility of synthesizing vowel-vowel sequences from smaller-sized constituents, a notion backed by the data presented in Tables 3.2 and 3.4.

More manual experimentation synthesized natural-sounding speech by concatenating near the areas of extreme articulation in vowel diphthongs. For example, a synthetic 'lawyer' was created from an original 'lawyer' by substituting the first syllable, [lɔʸ], from 'loyal'; the splice was made towards the end of [ɔʸ] where the second spectral prominence reaches a peak. Because these two words share the same stress pattern, strong-weak, the synthetic result sounds natural. A similar example of synthesizing 'lawyer' will be presented in the next section on automatic experiments.

In a counter-example illustrating the effect of incorrect prosody, concatenating the first syllable of 'leeway' with the second syllable of 'aware' in the middle of the pre-vocalic [w] semivowel formed a nonsense word, 'leeware.' Although the formants and pitch were continuous and the splice was not perceptually noticeable, stringing two lexically stressed syllables in succession did not sound natural.

The principles derived in the manual experiments guided the development of a search heuristic which maximized continuity in formant structure and pitch. As a proof of concept, the next section discusses experiments performed automatically with no human guidance.

3.5 Sub-phonetic: automatic experiments

In this section, concatenations are made at the sub-phonetic level [60] and their locations are automatically determined by a search heuristic seeking to maximize continuity in formant structure and pitch. To enable the concatenation to occur at

any point within a phone, the units are designated to be individual pitch period with no phonetic identity. This is the smallest possible unit in the temporal scale and there is no spatial structure - more precisely, it is flat - because phonetic identity is not provided. In other words, no symbolic information is provided and strictly numeric information is matched to determine the concatenation points.

No symbolic and all numeric information represents an extreme in the continuum of input specifications for unit selection. This permits fine control over the desired spectral dynamics, a feature being simultaneously an advantage and disadvantage. The reasons for the former are obvious, while the reasons for the latter include the difficulties of generating the input specification and modelling at search time the perceptual impact of deviations from the input.

To prepare a corpus for a fully numerical search, formants (spectral prominences) and pitch are tracked with dynamic programming procedures. As in the previous experiments, formants are calculated with ESPS+ software which fits linear predictive coefficients (LPC) [84], a form of autoregression, to find the locations of resonances on a sliding window basis. Figure 3-7 gives an example of LPC analysis applied to a single pitch period extracted from a vowel, [u^{w}]. Because the poles of the transfer function are close to the unit circle in the z-plane, estimates of the resonance frequencies can be obtained by mapping the angles of the complex poles to linear frequencies. Note that the first formant can be estimated as shown by the dotted line by the location of the first zero crossing, τ_Z, in the time-domain waveform of the pitch period (i.e., $F_1 = \frac{1}{2\tau_Z}$.) A smooth evolution over time of these resonances is achieved by enforcing smoothness constraints with a heuristic that is maximized using a Viterbi search. This stitches together local LPC-based formant estimates into a smooth trajectory.

Pitch period boundaries are also automatically determined by dynamic programming [144] which optimizes local agreement of pitch period spacings with some notion of pitch such as peaks in the locally windowed auto-correlation function. A pitch trajectory from a pitch tracker may also be used to guide the search heuristic. Figure

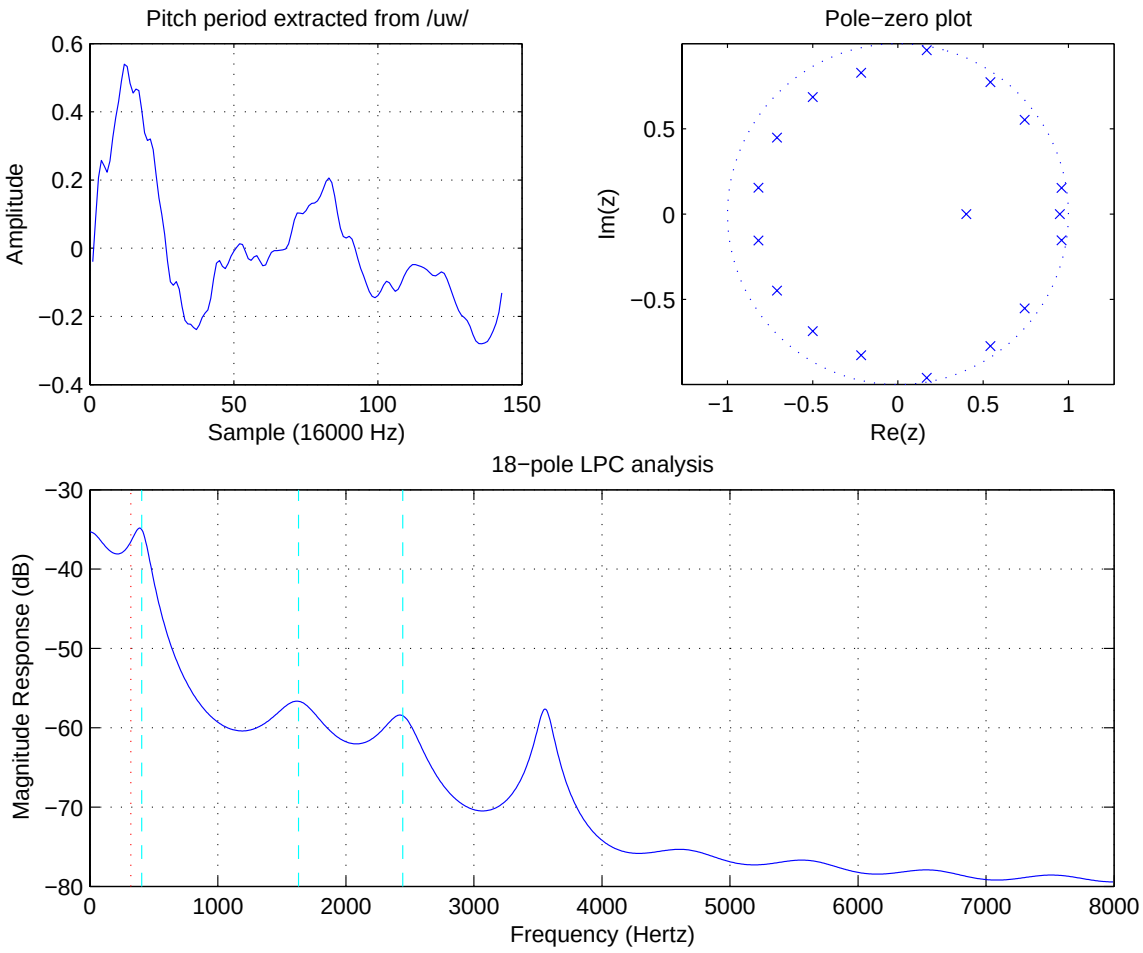


Figure 3-7: LPC analysis applied to a pitch period from [u^{w}].

A single pitch period is extracted from the vowel, [u^{w}], and is processed by LPC analysis. Because the poles of the transfer function are close to the unit circle in the z-plane, the resonance frequencies can be well estimated by mapping their angular locations to linear frequencies. Note that the first formant can also be estimated from the first zero crossing of the time-domain waveform.

3-8 shows the result of automatically determining pitch period boundaries in the word 'lawyer'. The panes correspond to the zoomed-in waveform, spectrogram, zoomed-out waveform, phonetic transcription, orthographic transcription, pitch trajectory from a pitch tracker, pitch period transcription, and pitch trajectory locally estimated from pitch period spacings. Once individual pitch periods are resolved, an estimate of the first formant, overlaid on the spectrogram, can be obtained from the first zero crossing as previously described. Although the algorithm suffers as pitch drops, the accuracy can be quite high as illustrated by a highlighted pitch period.

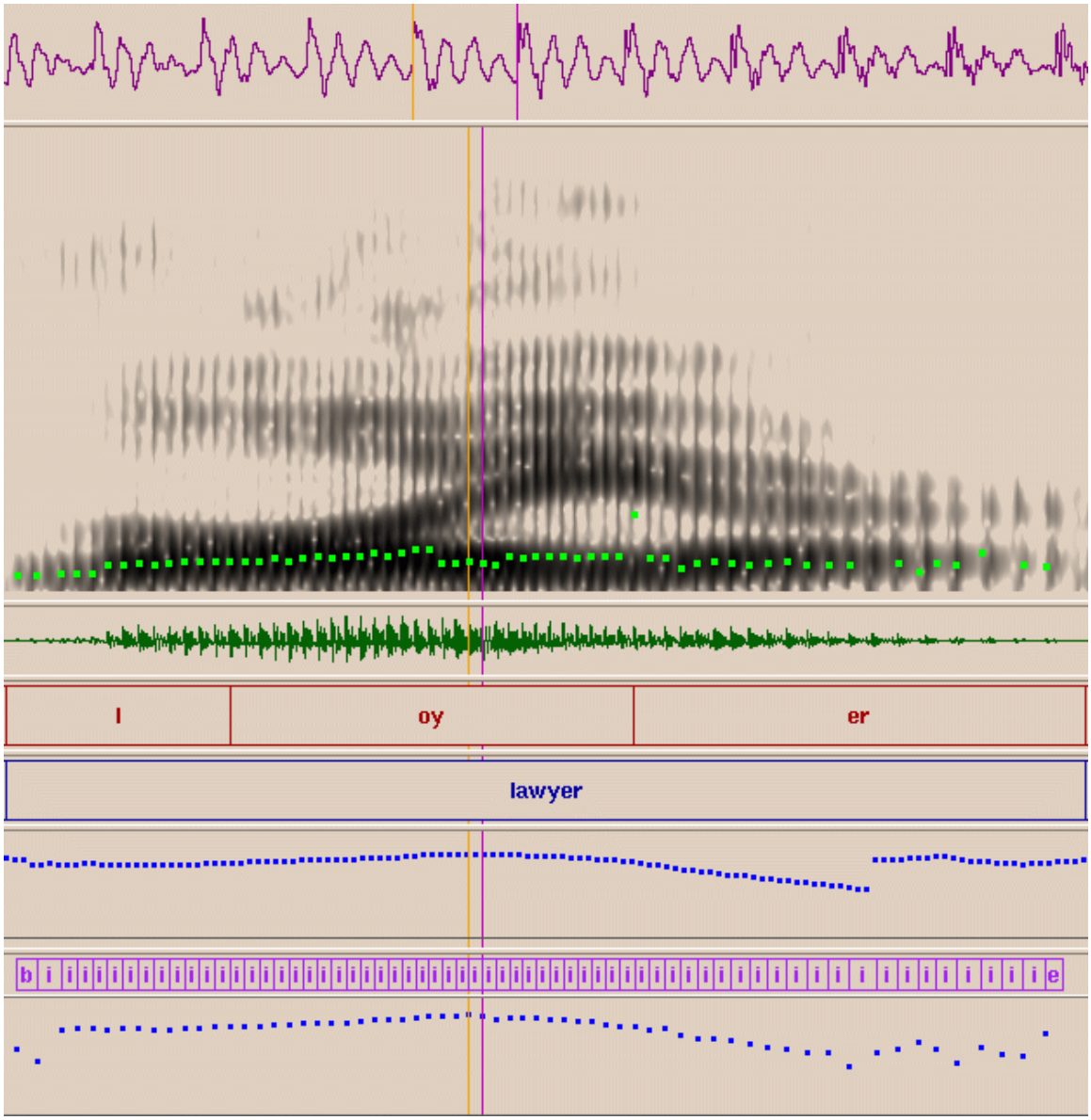


Figure 3-8: Automatic determination of pitch periods in 'lawyer'.

A dynamic-programming search is used to automatically determine the locations of pitch periods by smoothly connecting local estimates. Once the pitch periods are located, local estimates of the pitch and the first formant frequency can also be obtained from the pitch period length and the first zero crossing, respectively.

Individual pitch periods are inserted into the corpus as units without any phonetic identity attached to them. All units in the corpus are available for selection at search time. The unit cost is the distance between the desired and selected formants. The transition cost between two selected pitch periods is zero if they are contiguous, otherwise it is a constant plus the distance between the selected pitches. The squared Euclidean distance was the metric used in these automatic experiments.

In these automatic experiments, jackkniving is performed by leaving out one utterance and reconstructing it from the other utterances in the corpus. The formant and pitch trajectories of the jackknived utterance are used as the input specification. Because the experiments in this chapter are performed on voiced speech, the trajectories are continuous. A search lattice is constructed by licensing all pitch periods in the corpus for each input pitch period. As described in Chapter 2, this *lattice* has N *slices* containing L *nodes* each for a total of L^N nodes, where L is the size of the corpus and N is the length of the input specification.

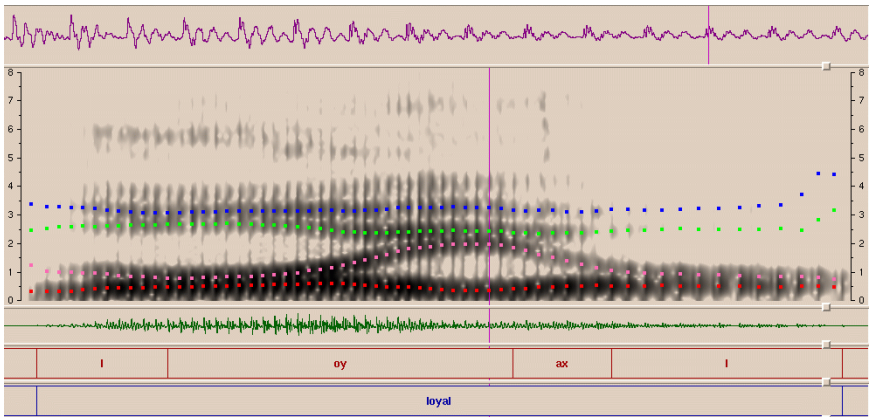
This fully interconnected lattice is efficiently explored by a dynamic programming algorithm. For each of the N slices in the lattice, a node receives incoming connections from each of the L nodes in the previous slice. The score at the current node is the previous unit cost plus the transition cost between the previous and current node. Only the incoming connection corresponding to the lowest score is remembered by a back pointer and once the end of the lattice is reached, the best path is formed by tracing back through the back pointers.

Figure 3-9 shows the process of synthesizing 'lawyer' by concatenating the first and second syllables from 'loyal' and 'layer', respectively. The time cursors in Figures 3-9(a) and 3-9(b) mark the pitch periods of the concatenation boundaries that the search algorithm chooses. In each case, the second spectral prominence appears to be at a maximum, emulating the ideal behavior proposed in the manual experiments described in the previous section. Figure 3-9(c) displays the jackknived utterance, 'lawyer', whose formant and pitch trajectories were used as the input specification.

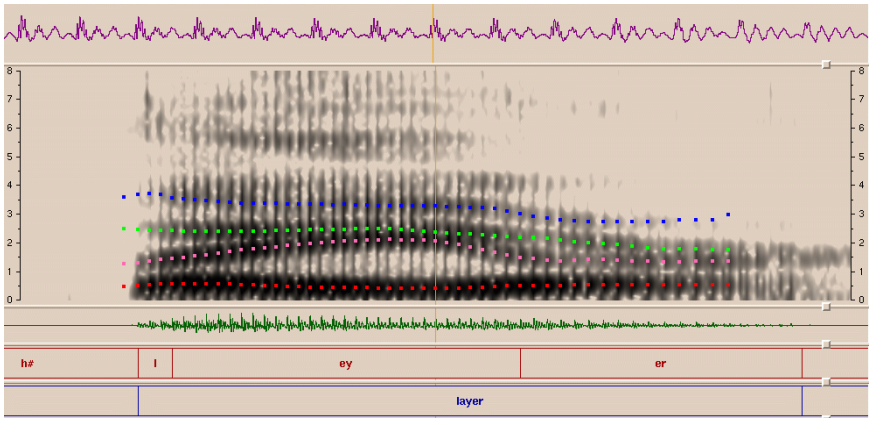
Figure 3-9(d) shows the synthetic 'lawyer' entirely reconstructed from 'loyal' and 'layer'. The spectrograms for the natural and synthetic versions of 'lawyer' are quite similar. The formant structure appears to be aligned at the point of concatenation and the synthetic waveform sounds entirely natural with neither formant nor pitch discontinuities.

Though individual pitch periods offer a great amount of control, it may be possible to retain much of this control by properly segmenting semivowels and diphthongs and by adjusting segment boundaries to coincide with pitch periods [60]. For example, in Chapter 4, it is discovered from data that inter-vocalic [l] can be divided into half-phones at the spectral extremum [51] and that these half-phones can be used for concatenations. The spectral extrema that would have been found anyways by the sub-phonetic search are pushed out to segment boundaries.

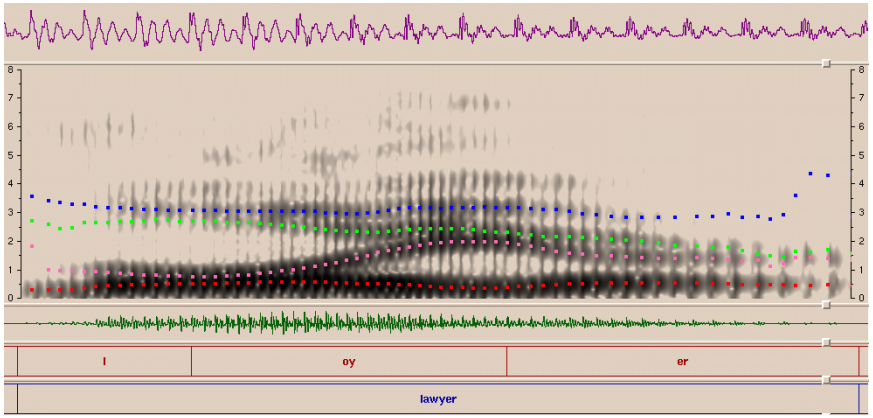
The result produced by this jackkniving experiment with sub-phonetic units is good and gives a proof of existence for such a framework. Because oracle information was provided from an utterance spoken by a human, future work is needed to generate the trajectories and their timing in order to produce a self-standing system.



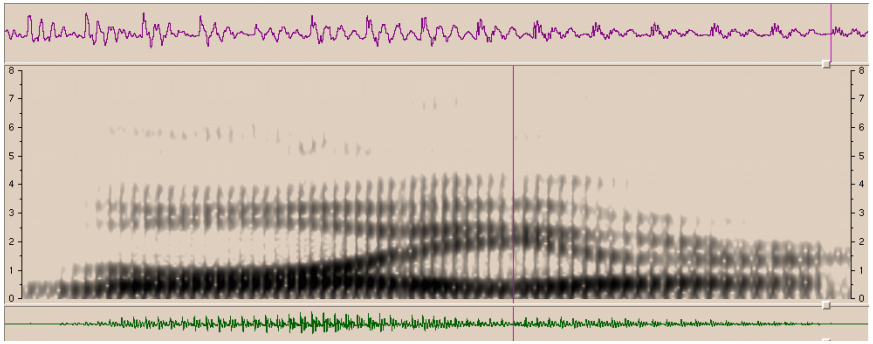
(a) Screenshot of the word 'loyal' taken from *cv* tool.



(b) Screenshot of the word 'layer' taken from *cv* tool.



(c) Screenshot of the word 'lawyer' taken from *cv* tool.



(d) Screenshot of the synthetic word 'lawyer' taken from *cv* tool.

Figure 3-9: Jackknived synthesis of 'lawyer' from 'loyal' and 'layer.'

The word 'lawyer' is jackknived from the corpus, but its formant and pitch trajectories are kept to serve as the input specification. A search algorithm with a heuristic of maintaining continuity in formant structure and pitch concatenates the first syllable of ''loyal' with the second syllable of 'layer' at the optimal location.

3.6 Frame-based units

In the unit selection experiments performed in this chapter, individual pitch periods are marked by either labor-intensive manual or error-prone automatic methods. In the given example, the corpus was prepared by marking pitch periods automatically via a dynamic programming method. This procedure is prone to error as the pitch periods are located by hard decisions. This pitch-synchronous temporal segmentation implies a variable-frame-rate (VFR) analysis of speech. The units are concatenated without signal processing - although abutting signals implicitly assumes rectangular windowing, there is no blending of pitch periods at a concatenation boundary. It can be shown that if pitch periods are blended in the time domain, then it is equivalent to concatenating short-time spectral analysis frames at a constant frame rate (CFR) in the time domain and inverting the sequence of spectral frames back to a time-domain waveform; this is the basis of overlap-and-add (OLA) methods [26]. The analysis stage operates at a variable frame rate and the synthesis stage operates at a constant frame rate. Figure 3-10 shows a portion of voiced speech with a Hanning window (essentially an inverted and raised cosine window) overlaid. The speech waveform modulated by the Hanning window produces a windowed pitch period. Because the units are entire pitch periods which go to zero amplitude at the boundaries, blending in the time domain works well if the pitch is continuous at a concatenation.

If the Hanning window length is fixed, then, when windowed pitch periods are laid in succession, their corresponding windows sum to unity for all time. This is the benefit of CFR synthesis. The challenge lies in the VFR analysis, locating the windows at a pitch-synchronous rate in order to reduce phase mismatch at synthesis time. In this example, the Hanning window is centered at the maximum amplitude of the pitch period. When the pitch period changes, an adaptive Hanning window could be used, wherein the length of the window changes with the pitch, but this complicates the synthesis by requiring an amplitude normalization stage as the windows do not add to unity under changing window lengths. WSOLA [154], for example, is a principled

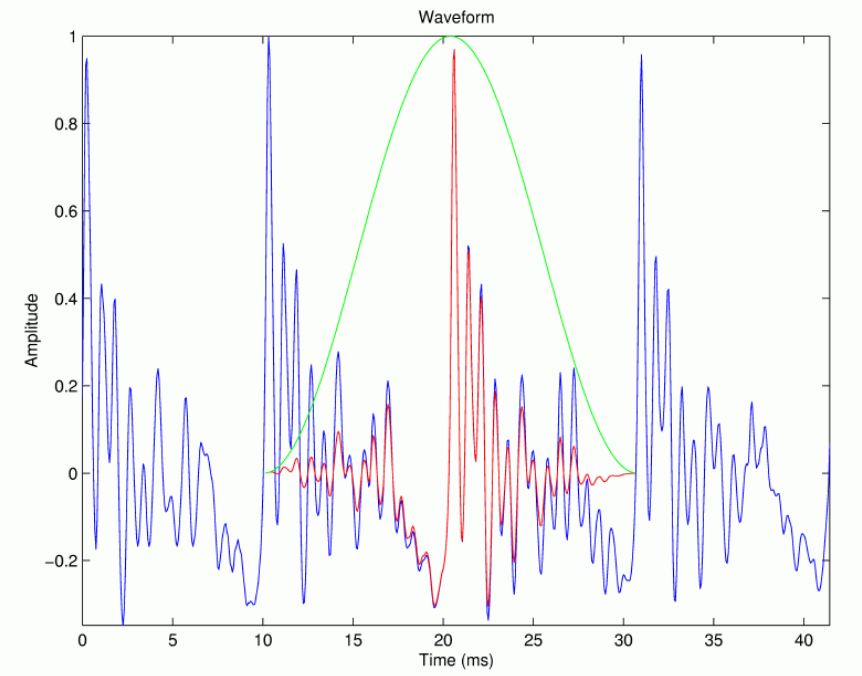


Figure 3-10: Plot of voiced speech waveform and Hanning-windowed pitch period.

When modulated by the Hanning window shown in green, a windowed pitch period shown in red is sectioned out of the voiced speech waveform shown in blue.

approach with VFR analysis and CFR synthesis. When two non-contiguous pitch periods are concatenated, a cross-correlation analysis is performed to find the optimal lag or lead between the window centerings of the source waveforms.

In the method described above, the sectioning of voiced speech into windowed pitch periods can be performed off-line prior to run-time. This computation can be pushed to run time by performing the analysis at a constant rate and the synthesis at a variable rate; this is the basis behind other OLA methods. Because there may be phase mismatch from the CFR analysis stage, the optimal lag or lead is determined at run-time with, for example, cross-correlation analysis. However, now the windows do not sum to unity and amplitude normalization must be performed.

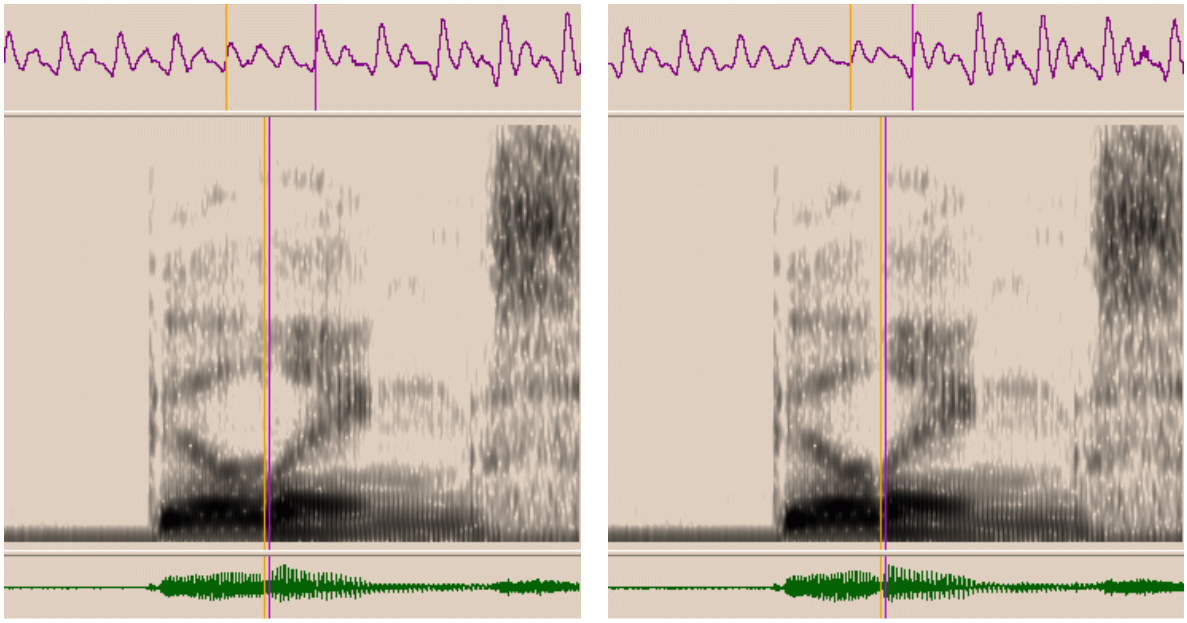
Figure 3-11 demonstrates the need for blending or smoothing pitch periods when concatenating in regions of voiced speech. The word 'Billings' is synthesized from the first syllable of 'billionth', [bɪl], and the second syllable of 'darlings', [lɪŋz]. Although

[l] in ‘billionth’ is inter-vocalic (i.e., sandwiched between two vowels) and the [l] in ‘darlings’ is pre-vocalic (i.e., preceding a vowel), the units are fairly compatible, because the boundaries are chosen at the point of maximum articulation within the [l] where the second and third formants reach a minimum and maximum, respectively. When no smoothing is performed, an irregular pitch period of lower pitch - listening would show a perceivable effect - is produced at the concatenation boundary as can be seen from Figure 3-11(a). When smoothing is performed, the pitch is regular and the amplitude envelope comes to a minimum at the point of maximal constriction in the vocal tract. Also, in the smoothed case the second formant appears to be slightly more continuous. The amplitude envelope of the [l] from ‘billionth’ may be slightly more attenuated than appropriate for a word such as ‘billings’, because the [l] anticipates a tense vowel, [i], in ‘billionth’ as opposed to anticipating only a lax vowel, [ɪ], in ‘billings’; thus, there is feature spreading of tenseness from the vowel to the [l] attenuating it sooner.

3.7 Source-filter separation

In the example of ‘lawyer’, the formant structure and pitch are matched simultaneously during the search. In the source-filter model of speech production, this bifurcation of formant structure and pitch corresponds to the filter and source, respectively. The vocal tract is modelled by a filter and glottal excitations are modelled by a source. If this separation of acoustics (filter) and excitation (source) can be performed successfully, it is possible to catalogue a corpus by its filter and source characteristics independently. Unit selection for the filter and source can proceed independently and in parallel. When selection is completed, the source and filter are re-combined at synthesis time. In this fashion, data sharing is made possible and promoted through the combining of sources and filters from different speech segments.

The particular form of source-filter separation employed here was first used by Seneff



(a) Screenshot of the synthetic word 'billings' when concatenated without smoothing.

(b) Screenshot of the synthetic word 'billings' when concatenated with smoothing.

Figure 3-11: Comparison of pitch continuity without and with smoothing.

The word 'Billings' is synthesized from the first syllable of 'billionth' and the second syllable of 'darlings'. When concatenation is performed without smoothing pitch periods, an irregular pitch period of lower pitch is formed and the effect is negatively perceivable. When smoothing is performed in the second case, the pitch is continuous and is preferable over the first case.

[123] for the purposes of pitch, duration, and voice modification. First, a frequency-domain representation is broken down into a magnitude and phase component: $H(j\Omega) = |H(j\Omega)|e^{\angle H(j\Omega)}$. Then, the magnitude function, $|H(j\Omega)|$, is smoothed in frequency with a (Hanning) window in frequency, $W(j\Omega)$. This smoothed magnitude function, $H_S(j\Omega) = |H(j\Omega)| * W(j\Omega)$, is divided out of the original representation to obtain an excitation function, $H_E(j\Omega) = H(j\Omega)/H_S(j\Omega)$. Note that the smoothed magnitude function is real, while the excitation function is complex.

If the spectral frames, $H(j\Omega)$, are taken from a narrow-band spectrogram, then individual pitch harmonics are resolved in frequency. The purpose of smoothing the magnitude function is to remove these striations in frequency. When the smoothed magnitude function is divided out, the spectral frame is flattened and what remain in the

excitation function are the individual pitch harmonics which resemble a comb filter. The smoothed magnitude function, $H_S(j\Omega)$, and excitation function, $H_E(j\Omega)$, can be separately catalogued in the corpus and re-combined to synthesize novel speech. For example, if another smoothed magnitude function, $H_{S'}(j\Omega)$, is combined with $H_E(j\Omega)$, the resulting spectral frame, $H_{S'}(j\Omega)H_E(j\Omega)$, can be inverted back into a time-domain waveform. When sequence of frames are manipulated on a sliding window basis, methods to invert short-time fourier transforms (STFT) are well understood [112, 53].

Figure 3-12 shows the process of transforming the word 'IMport' (word sense is noun) into a synthetic 'imPORT' (word sense is verb) by using the excitation from a natural 'imPORT'. This matrix of plots was created using MATLAB [95]. The three columns correspond to the natural 'IMport', the natural 'imPORT', and the synthetic 'imPORT'. The four rows correspond to the waveform, the spectrogram (STFT), the smoothed magnitude function, and the excitation function. The striations in the excitation function move closer (especially in 'imPORT') as the pitch lowers at the end of the word - lower pitch means more finely spaced harmonics - but traces of the underlying acoustic-phonetics are still observed when, for example, the second and third formants converge to make the articulatory gesture for the [ɹ]. The smoothed magnitude function from the natural 'IMport' is resampled at the phonetic level to match the phonetic timings of the natural 'imPORT' - this guarantees rhythm and timing. The second and third formants of [ɪ] are seen to fall faster in anticipation of the labial nasal, [m] - frames have been omitted in the resampling process. This resampled smoothed magnitude function is combined with a natural excitation function for 'imPORT' and the new STFT is inverted to produce novel speech. The synthesis sounds very natural and the desired rhythm is achieved. Since the amplitude envelope was not transplanted, the first syllable appears to be somewhat louder than desired in the waveform plot. This may be a secondary effect and could be corrected by modulating the amplitude with the desired envelope.

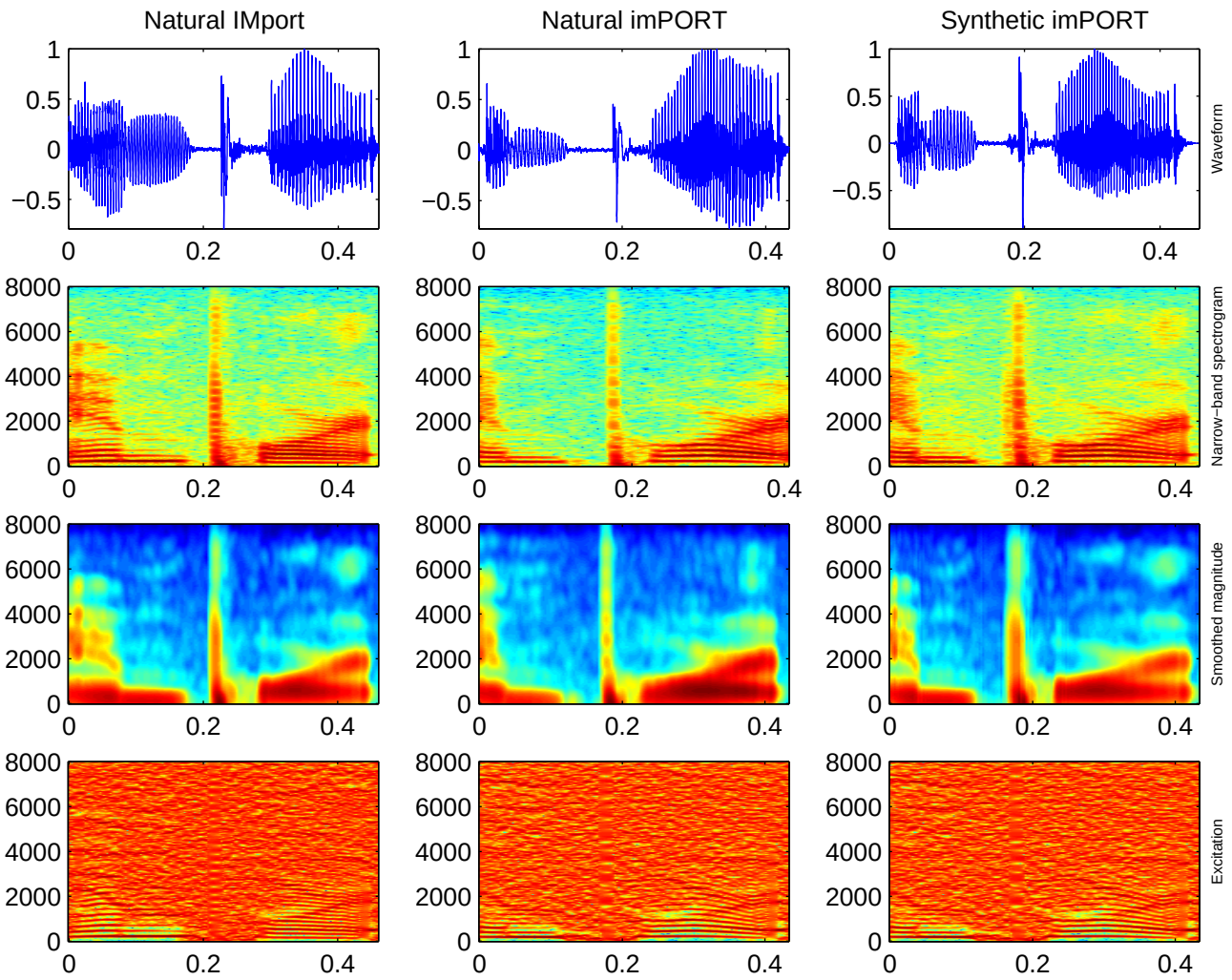


Figure 3-12: Smoothed magnitude from 'IMport' combined with excitation from 'imPORT'.

In this panel of plots, the waveform and its narrow-band spectrogram are shown on the first two rows. The smoothed magnitude function in the third row from 'IMport' is resampled at the phonetic level to match the timing from 'imPORT'. Then, it is combined with the excitation in the fourth row from 'imPORT' to form a synthetic 'imPORT' which sounds natural. Additional improvement could be gained by fitting the energy envelope in addition to the rhythm.

This example of cross-synthesis demonstrates the expressive power - transposing the stress pattern - made possible by source-filter decomposition. However, the problem is relatively constrained, because the excitation function is taken from another word with the exact pronunciation, /ɪm/ /pɔrt/. This previous example adds a degree of freedom in that prosody can be shared when the phonetic pronunciation is the same. What would add another factor of flexibility is sharing prosody when the phones are different and the next example illustrates just this. Table 3.6 lists other pairs with opposite lexical stress patterns for different word senses. These 25 word pairs created by Aull [3, 2] form a pool from which other experiments can be conducted.

ATtribute	atTRIBute	CONtract	conTRACT	INcline	inCLINE	TORment	torMENT
COMpact	comPACT	CONtrast	conTRAST	INsult	inSULT	TRANsport	tranSPORT
COMpound	comPOUND	CONvert	conVERT	OBject	obJECT	UPlift	upLIFT
COMpress	comPRESS	DIgest	diGEST	PERfect	perFECT	UPset	upSET
CONduct	conDUCT	EScort	esCORT	REbel	reBEL		
CONflict	conFLICT	EXport	exPORT	REcord	reCORD		
CONtest	conTEST	IMport	imPORT	SURvey	surVEY		

Table 3.6: Word pairs with transposed lexical stress patterns [2].

Figure 3-13 shows the process of transforming the word 'IMport' (word sense is noun) into a synthetic 'imPORT' (word sense in verb) by using the excitation from a natural 'comPACT'. The three columns correspond to the natural 'IMport', the natural 'comPACT', and the synthetic 'imPORT'. Although the stress pattern has been modified as desired, the final vowel does not sound natural, possibly, because of interaction between the [r] in 'IMport' and the [æ] in 'comPACT'. As before, the separation is not perfect and the smoothed magnitude function for 'IMport' has surrendered some of the [r] coloring to the excitation function. Whereas the [r] in the natural 'IMport' encodes a cue of the following phoneme, /t/, in the form of glottalization - an attribute of the American English dialect - the [æ] in 'comPACT' has no glottalization and the resulting synthesis sounds more like 'imPOR' with the phoneme, /t/, unrealized in acoustic terms. Future work should investigate categorizing speech sounds into classes for which excitations can be transplanted without perceivable effects, and, more generally, creating templates of smoothed magnitude profiles which can be dynamically warped in time and amplitude to fit the desired context at synthesis time.

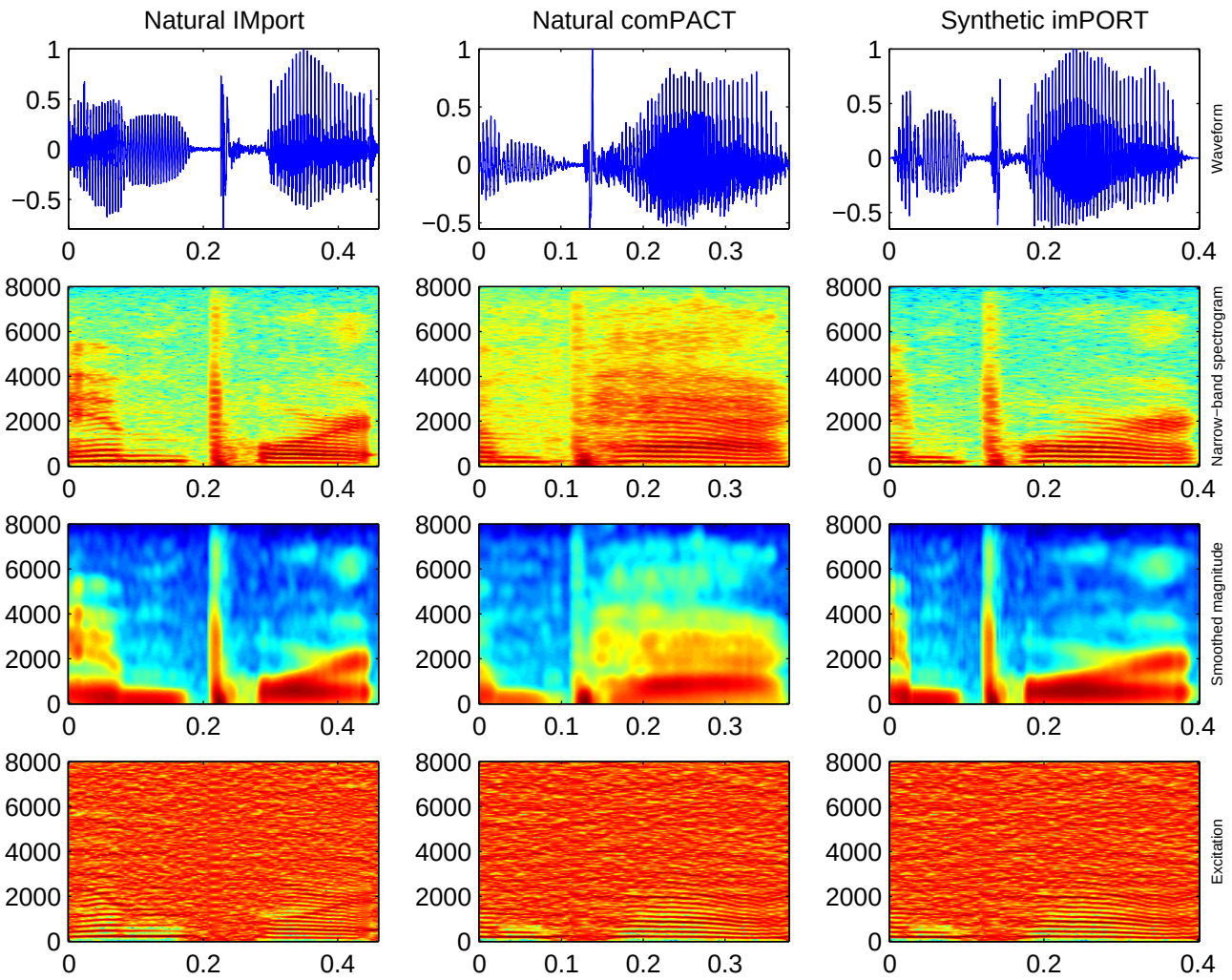


Figure 3-13: Smoothed magnitude from 'IMport' combined with excitation from 'comPACT'.

The smoothed magnitude function from 'IMport' is resampled at the phonetic level to match the timing from 'comPACT'. Then, it is combined with the excitation from 'comPACT' to form a synthetic 'imPORT', the second syllable of which does not sound natural. This may be due to incompatibility between /rt/ from 'IMport' and /æ/ from 'comPACT'.

3.8 Nasals

The foregoing has dealt with vowels and semivowels, yet the broad class of sonorants also includes nasals, [m], [n], and [ŋ], which are produced through the nasal cavities. Because nasals tend to color or nasalize surrounding speech segments, especially vowels more so in anticipation of a following nasal, going forwards in time - their analysis is now briefly performed in this section. Previous work by the author [162] determined that because nasalization is so distinctive, it can be quite jarring in concatenative synthesis when segments are either misleadingly nasalized or when they are not properly nasalized. For example, if a nasalized vowel is concatenated with a non-nasal consonant, the cue encoded in the vowel of a coming nasal segment is dissonant with the spliced non-nasal consonant. On the other hand, if a normal vowel is concatenated with a nasal, the onset of the nasalization in the nasal is quite abrupt as the cue was not present in the vowel. The scenarios described above are already understood and are not addressed any further. Rather simple studies are mentioned here in passing as well as future directions.

Three types of mono-syllabic sequences were studied: nasal-vowel-nasal (NVN), nasal-vowel-stop (NVS), and stop-vowel-nasal (SVN). Nasals and stops were exhaustively cycled, but only five representative vowels were chosen. As depicted in Figure 3-14, the nasals include /m/, /n/, and /ŋ/; the voiced stops are /b/, /d/, and /g/; and, the voiceless stops are /p/, /t/, and /k/. The five vowels are sampled from the outer and central locations of the vowel quadrilateral shown in Figure 3-15: /iʸ/ is high-front, /æ/ low-front, /ɑ/ low-back, and /uʷ/ high-back, and /ʌ/ central. The vowel and consonant charts shown here were created by Rob Kassel and are used as teaching materials at MIT in *Speech Spectrogram Reading: An Acoustic Study of the English Language*, a summer course numbered 6.67s.

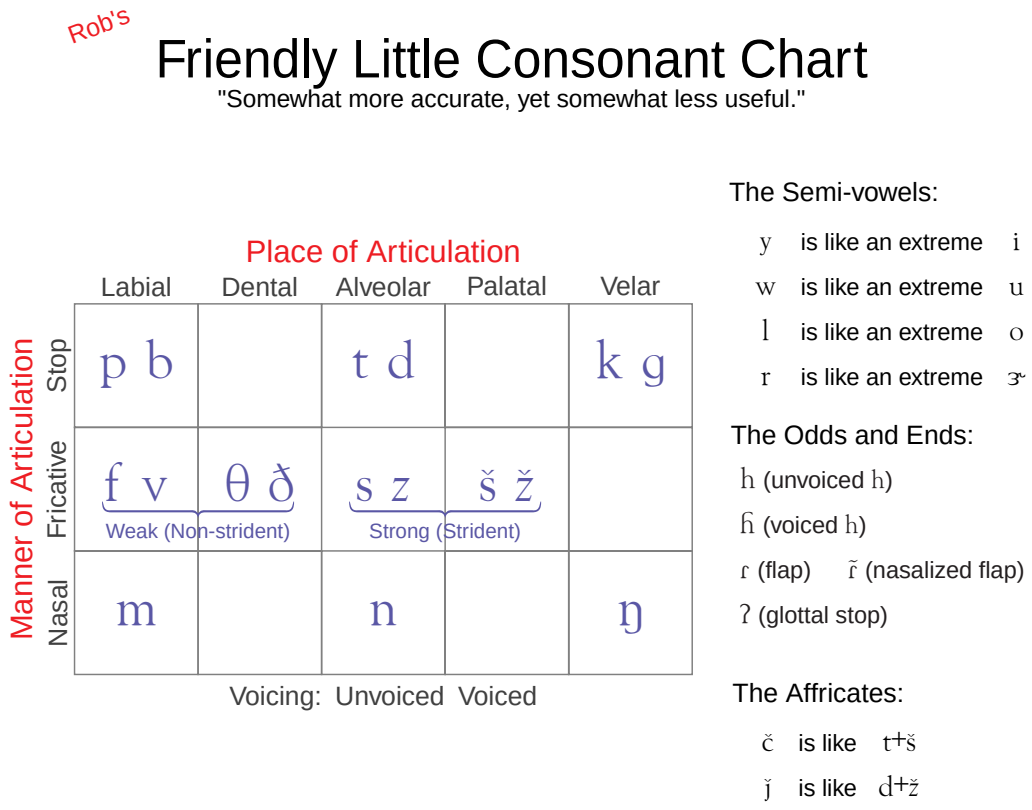


Figure 3-14: Rob Kassel's Happy Consonant chart.

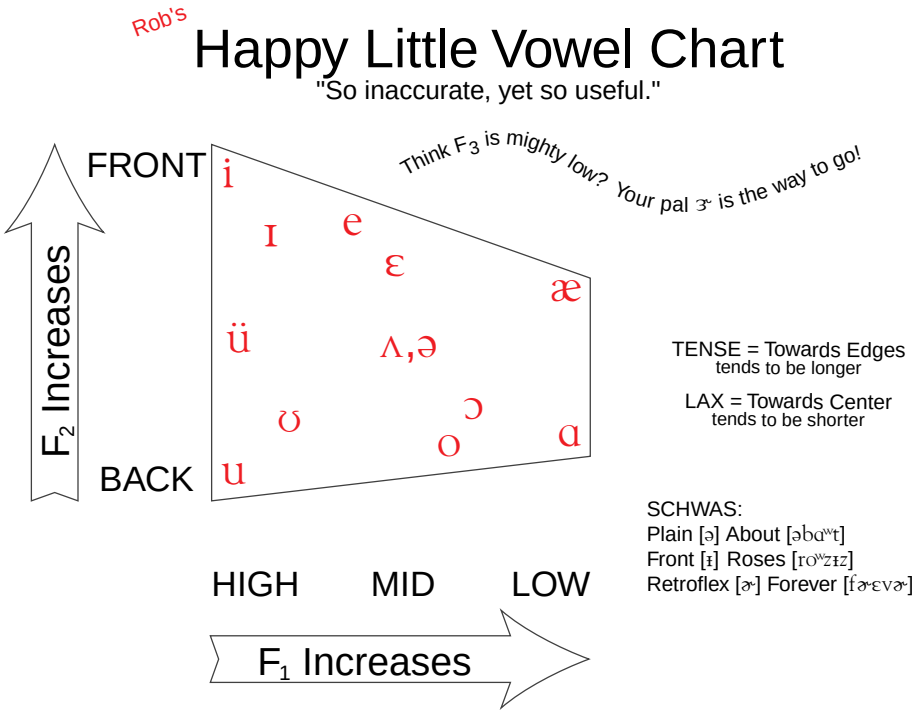


Figure 3-15: Rob Kassel's Happy Little Vowel chart.

With five vowels, six stops, and three nasals, a total of 180 syllables were recorded: 30 (2*5*3) NVN, 60 (2*5*6) NVS, and 90 (6*5*3) SVN. Care was taken to intonate the syllables with a consistent melody. Because /ŋ/ is not allowed to begin a syllable in English, the syllables in the NVN and NVS sets only begin with /m/ and /n/. This data set was used for concatenation experiments that investigated where place of articulation (i.e., labial, alveolar, and velar) is encoded for a nasal, and what properties of nasals are invariant. Because the concatenations were made at the boundaries between vowels and nasals, the exact point of splicing is easily determined from a waveform display; consistent intonation ensures pitch continuity.

Table 3.7 lists a series of concatenation experiments performed with recordings from the NVN set. Phonemes that are shown in parentheses refer to context and their acoustic realizations were not actually concatenated. By replacing nasals in the initial and final positions with nasal produced at other places of articulation, the hypothesis that the nasal murmur encodes place was tested. After concatenating the units and listening to the results, it was determined that, for example, changing the first phone in /mæm/ to /n/ did not change the perception to /næm/; the place of articulation for the intial nasal is encoded as labial co-articulation at the start of /æ/. This would suggest that nasal murmurs can be used interchangeably without regard to their phonetic identity; they can be collapsed into one class.

Figure 3.8 shows the results of experiments conducted with recordings from the NVS and SVN sets. Nasals in the initial and final positions were replaced by nasals from other vowel contexts. For example, the initial nasal in /næp/ was replaced by the nasal in /nip/. As expected, the perception of the syllable did not change, confirming the hypothesis that nasals are fairly stable and do not encode information about bordering vowels; this suggests an invariance of nasals to vowel context. Because both the initial and final positions were studied, these studies suggest that nasals do not receive influence from either their right or left borders. Additional studies beyond those described here are needed to systematically and exhaustively learn other cues and invariances for nasals.

Units concatenated	Expected perception	Actual perception
Swapping nasal in initial position		
/n/(æm) + (m)/æm/	/næm/	/mæm/
/n/(iʸm) + (m)/iʸm/	/niʸm/	/miʸm/
/m/(æm) + (n)/æm/	/mæm/	/næm/
/m/(iʸm) + (n)/iʸm/	/miʸm/	/niʸm/
Swapping nasal in final position		
/mæ/(m) + (mæ)/n/	/mæn/	/mæm/
/mæ/(m) + (mæ)/ŋ/	/mæŋ/	/mæm/
/mæ/(n) + (mæ)/m/	/mæm/	/mæn/
/mæ/(n) + (mæ)/ŋ/	/mæŋ/	/mæn/
/mæ/(ŋ) + (mæ)/m/	/mæm/	/mæŋ/
/mæ/(ŋ) + (mæ)/n/	/mæn/	/mæŋ/

Table 3.7: Nasal experiments with NVN set.

In these experiments, nasals in initial and final positions were replaced by nasals produced at other places of articulation. Phonemes in parentheses denote context and were not concatenated. Changing the place of articulation did not change the perception of the syllable as expected. Rather, the place of articulation was encoded in the vowel and the original perception was not overriden by a different nasal murmur. This would suggest that nasal murmurs can be used interchangeably regardless of their phonetic identity and left and right context.

Units concatenated	Expected perception	Actual perception
Swapping nasal in initial position		
/n/(iʸp) + (n)/æp/	/næp/	/næp/
/n/(ɑp) + (n)/æp/	/næp/	/næp/
/n/(ʌp) + (n)/æp/	/næp/	/næp/
/n/(uʷp) + (n)/æp/	/næp/	/næp/
Swapping nasal in final position		
/tɑ/(m) + (tiʸ)/m/	/tɑm/	/tɑm/
/tɑ/(m) + (tæ)/m/	/tɑm/	/tɑm/
/tɑ/(m) + (tʌ)/m/	/tɑm/	/tɑm/
/tɑ/(m) + (tuʷ)/m/	/tɑm/	/tɑm/

Table 3.8: Nasal experiments with NVS and SVN set.

In these experiments, nasals in initial and final positions were replaced by nasals that were produced at the same place of articulation but in other vowel contexts. Phonemes in parentheses were not concatenated. Whether the vowels were front, back, high, or low was not encoded as cues in the nasal. The nasal segments were invariant to their vowel context and the perception remained unchanged as expected. These results suggest that vowel context can be ignored when substituting nasals from other contexts.

3.9 Discussion

The sub-phonetic and frame-based units are used in this chapter to build an input specification with numerical information and very little or no symbolic information. Many of the ideal concatenation points determined are located at extrema of the spectral prominences. If these stationary points can be reliably transcribed, it may be possible to reduce the input specification to a symbolic one. For example, after forced phonetic alignments are obtained using an automatic speech recognizer, all /w/ sounds could be split into two demi-phones where the splice boundary is the point of extreme articulation.

The work presented in the next chapter focuses on an opposite approach of attack to building an input specification, using primarily symbolic information. Because numeric information is essentially discarded, symbols that are well isolated at their boundaries must be discovered by automatic means. Using notions from information theory, the best boundaries are defined to be those across which the acoustics are maximally unpredictable. While experiments described in Chapter 6 show that this information theoretic framework works favorably, it is currently unclear how defining boundaries to be placed at points of extreme articulation as described above can be reconciled with placing boundaries of maximum uncertainty.

Many of the studies performed in this chapter involved words containing only one or two syllables. A natural extension would examine more complex poly-syllabic words, phrases, and sentences. In the source-filter decomposition, equivalence classes need to be defined for excitations depending on acoustic-phonetics and higher-level information like prosody. Some of the distributional analyses carried out in this chapter point out similarities between semivowel-vowel and vowel-vowel sequences. They lead to conclusions which seem to blur the distinction between vowels and semivowels. This knowledge should be incorporated in order to more accurately locate concatenation boundaries for speech synthesis [33, 149, 151, 78, 83, 25].

Chapter 4

Acoustic modelling

4.1 Introduction

An important component of any speech processing system is an acoustic model which separates speech sounds into categories for subsequent characterization. The categorization can be guided by a combination of rules and data. Knowledge of the phonological organization of a language can be used to create rules that are verified by a corpus of data. Once speech sounds are categorized into separate clusters, characterization entails the extraction of numerical measurements from the data and the summarization - usually statistical in nature - of these measurements. The two processes of categorization and characterization may contain linkage in the form of feedback. The best set of categories may be implicit in the mathematics of characterization. Knowledge of acoustic phonetics may provide hints to the best type of characterization. Once this process, as depicted in Figure 4-1, is completed, a set of models is arrived at and new incoming data can be classified or evaluated for conformance with the models.

Categorization and characterization may be iteratively performed as in the case of

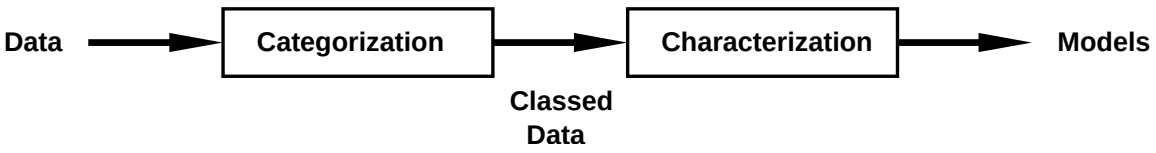


Figure 4-1: Categorization and characterization in modelling.

Incoming data is grouped into distinct clusters by the process of categorization. Subsequently a typical procedure entails characterizing these clusters by statistical summarization where average and deviating behavior are established forming models. New data can be classified by such models blindly or, otherwise, can be evaluated for their conformance to such models.

decision tree clustering where pools of speech unit exemplars are successively divided into a tree of finer classes by phonologically motivated decisions. Once further division does not give additional improvements - as measured by some notion of how well the model fits the data - a node in the tree is reached. Exemplars at the node are grouped as a class and a characterization can be performed. A summarization of the class by its average behavior and deviance from the norm is standard and can be efficiently estimated by accumulating first and second order moments.

This chapter details a general framework for categorizing and characterizing speech units for use in concatenative speech synthesis. Specific choices of parameters and algorithms that are made in the presentation of this material are for ease of exposition and should not detract from the overall concept. Many of these choices are motivated from prior work in speech recognition and have effectiveness which demonstrably carry over to speech synthesis. Future work can examine more tailored choices under the general framework presented here.

Once characterization produces a set of models, they can be compared to produce a set of inter-cluster distances which can be used to approximate instance-to-instance comparisons as previously described in Chapter 2. This forms the basis for the automatic determination for substitution and concatenation costs. The Kullback-Leibler metric is presented for calculating distances among the set of models. Because of parameter reduction arguments made in Chapter 2 inter-instance distances are replaced by distances between their respective corresponding clusters.

4.2 Observation space

A definition of the observation space is necessary for categorization that is driven by data and is given here. Characterization also proceeds from the identical observation space. As depicted in Figure 4-2, an observation space is defined around a boundary between speech units, α and β, which may be phonetic units, or, more generally, classes of phonetic units. The notation, $\alpha[\beta]$, refers to a unit, α, in the context of a β on its right side. Similarly, the notation, $[\alpha]\beta$, refers to a unit, β, in the context of an α on its left side.

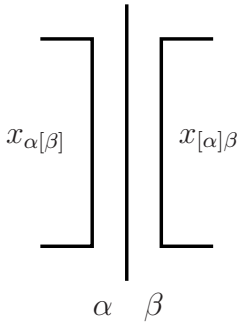


Figure 4-2: Graphical representation of observation space and boundary measurements.

An observation space is defined about the boundary between α and β. Measurements made on the left side are referred to with the notation $\alpha[\beta]$ which denotes α in the presence of β on the right side. Similarly, measurements made on the right side are referred to with the notation $[\alpha]\beta$ which denotes β in the presence of α on the left side. The former is forward looking and is termed the prospective space, while the latter is backward looking and is termed the retrospective space.

The space to the left of the boundary is termed the prospective space and the space to the right of the boundary is termed the retrospective space. The reason for this nomenclature is apparent. As β assumes different values, right-most portions of α close to the boundary may change in anticipation of β. Likewise, as α assumes different values, left-most portions of β close to the boundary may preserve characteristics from α. What is required is a disciplined approach to determining a parsimonious set of right and left contexts bearing, respectively, maximal anticipatory and preservatory effects. This is the process of categorization and will be presented in the next section.

4.2.1 Measurements

Measurements, x, culled from this observation space are divided into prospective measurements, $x_{\alpha[\beta]}$, and retrospective measurements, $x_{[\alpha]\beta}$. These measurements are distilled into prospective and retrospective models which form the basis of comparison for automatically determining costs. This is the process of characterization and will be presented in a later section.

As depicted in Figure 4-3, prototype measurements are computed at a constant frame rate that undergo further processing depending upon target usage. In the case of substitutions, measurements are extracted from the outermost portions of a speech segment. For concatenations, measurements are extracted about a boundary up to an arbitrary distance away, because the location of the next boundary on either side is unknown.

As in the SUMMIT system [50], a segment is divided into roughly thirds (in a 3-4-3 ratio) with the outermost portions being the first and last 30% of a segment. In the lower part of Figure 4-3 these 30% portions are bounded by dotted lines and form regions over which prototype measurements are averaged. Notice that the middle 40% portion of the segment is not used. In the upper part of the figure, boundary measurements are averaged over blocks of time whose widths are unrelated to segment lengths. Arranged in a telescoping fashion these blocks double in size as their distance from the boundary increases. Example locations which bound these blocks are ± 5, ± 15, ± 35, and ± 75 milliseconds away from the boundary.

The prototype measurements used in this work are mel-frequency cepstral coefficients [34], a representation commonly used in speech recognition. First, the waveform is pre-emphasized with first-order differencing (e.g., $H(z) = 1 - 0.97z^{-1}$.) Second, the waveform is windowed (e.g., with a Hamming window of length of 21 milliseconds) and the short-time Fourier transform is computed across a discrete number of bins (e.g., $2^8 = 256$)at a constant frame rate (e.g., 5 milliseconds). Next, mel-frequency

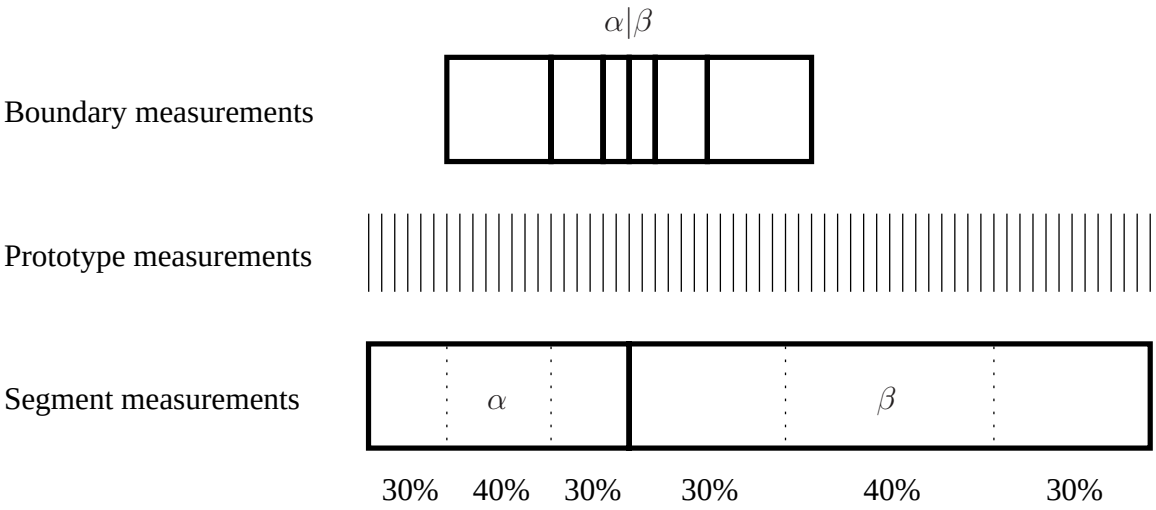


Figure 4-3: Prototype measurements are used to form boundary and segment measurements.

Prototype measurements are taken at a constant frame rate and are then combined to form measurements at the boundary and segment level. Telescoping averages of blocks of increasing (doubling) size are taken around the boundary for the former. Segments are broken into roughly thirds and measurements are averaged over these smaller portions.

spectral coefficients result from processing squared magnitudes through a filterbank consisting of triangular filters spread across a mel-scale (i.e., linearly spaced filters for low center frequencies and logarithmically spaced filters for high frequencies.) Finally, a discrete cosine transform is used to decorrelate these coefficients into the final representation of mel-frequency cepstral coefficients.

4.3 Categorization

Decision tree clustering is used to form categories for grouping data. After all the data is grouped into initial clusters, successive decisions are made to divide the data into smaller, more homogeneous clusters. If the decisions are two-way, then a binary tree results. Growing of the tree is terminated when further improvements in cluster homogeneity, as measured by likelihoods, is minimal and a node or leaf in the tree results. At this point, the node or leaf is termed a cluster and characterization can proceed. Decision tree clustering is a semi-automatic procedure utilizing knowledge

only to form the initial clusters and to structure the types of decisions.

The decisions are based on phonological criteria such as manner of articulation (e.g., vowel, liquid, glide, nasal, stop, fricative, affricate) and place of articulation (e.g., labial, dental, alveolar, velar.) Diphthongs, a monosyllabic speech sound that glides from the articulatory position for one vowel towards the position of another, may be classed differently when considering their impact from left and right locations. For example, [ɑʸ], when viewed from the left side, appears to be a low and back vowel, [ɑ], whereas, when viewed from the right side, it appears to be a high and front vowel, [i]. It will accordingly have different co-articulatory impact on other speech segments depending on whether it is located the left or right side.

4.3.1 Substitution classes

Contexts which have similar effects on the outermost portions of neighboring segments are merged into substitution classes which group interchangeable contexts. For determining substitution classes, the decision tree clustering algorithm is initialized with all the data grouped by phonetic identity. In other words, a separate tree is grown for each phonetic unit. Because substitution costs are decoupled in Equation 2.3 into left-sided and right-sided substitution costs, the classes for categorization are also decoupled into left-sided and right-sided counterparts. Hence, the decision tree clustering algorithm is repeated for left and right contexts for each phonetic unit. Example criteria that are matched at the nodes are listed in Appendix B under Figures B-1 and B-2.

Figure 4-4 shows a sample clustering run of left contexts for [ʌ]. Displayed in the form of a dendrogram, moving down the vertical axis corresponds to increasing likelihood. Divisions with larger likelihood increases occur towards the top of the tree and divisions with smaller likelihood increases occur towards the bottom of the tree. Clustering ends with remaining divisions do not significantly improve the likelihood

of the data. Horizontal branching to the left and right corresponds to a mismatch and match with the knowledge-driven decision, respectively. For example, when node, 1, splits into a node, 3, and a leaf, 4, data points in leaf, 4, correspond to [ʌ] with a left context of [z] or [s]. In this unbalanced tree with only left branches, the members of the leaves, 2, 4, 6, 7, and 8, can be easily read from the knowledge-driven decision made at each node. As another example, leaf, 8, contains all exemplars of [ʌ] whose left context is silence (- is silence between utterances and _ is silence between words).

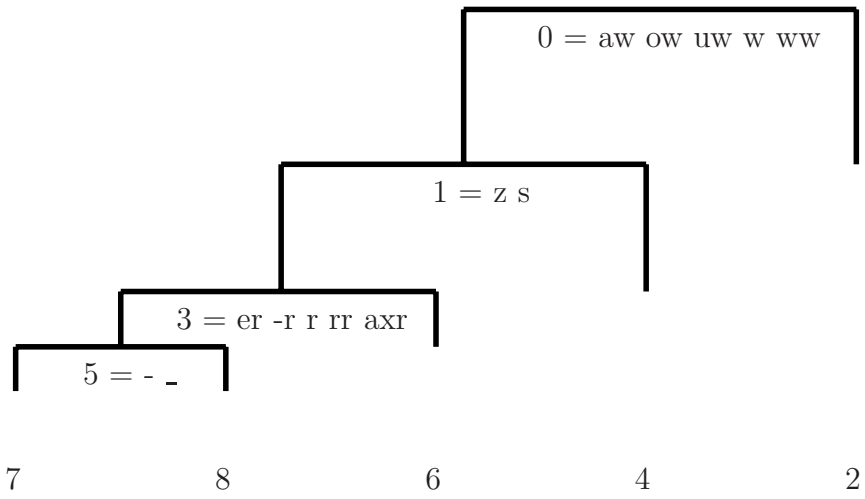


Figure 4-4: Sample clustering run for left context of [ʌ].

The clustering run is depicted as a dendrogram in which decisions branch left and right depending on whether the matching condition is met (right for a match.) The vertical scale is likelihood of the data and greater vertical distance from the top of the tree corresponds to increased likelihood. At each point in the tree, a decision is made to maximize likelihood. For example, node, 1, branches right into leaf, 4, which contains exemplars of [ʌ] whose left context is an alveolar fricative (both voiced, [z], and unvoiced, [s].)

4.3.2 Concatenation classes

A concatenation class merges into a single class multiple phonetic transitions which exhibit similar acoustical patterns around a boundary. This necessitates a fusion of the left and right contexts into a single concatenation class label and will introduce additional complexity in determining concatenation classes beyond that in determining substitution classes.

For clustering transition contexts, the decision tree clustering algorithm can choose to divide either left or right contexts at each node in the tree. This is different from the procedure for substitution classes where only the clustering was restricted to one side per run. Example criteria that are matched at the nodes are listed in Appendix B under Figures B-3 and B-4.

The initialization for clustering phonetic transitions involves, for the sake of tractability, creating an initial set of clusters with broad classes of speech sounds for the left and right contexts. As seen in Figure B-5, the broad classes roughly correspond to vowels/glides, stops, closures, strong fricatives, weak fricatives, lateral liquids, retroflexed liquids, and nasals.

Figure 4-5 depicts a sample run of the decision tree clustering algorithm for transition contexts consisting of stops on the left side and vowels and glides on the right side. This time, the tree is fairly balanced. The left-most leaf, 25, represents the exemplars which did not match any branching criteria and contains the following transitions between velar stops, k and g, and lax vowels, ɪ, ɨ, and ux: g|ɪ, g|ɨ, g|ux, k|ɪ, k|ɨ, k|ux. The right-most leaf, 36, represents the exemplars which match all branching criteria and contains the following transitions between unreleased dental stops, d̚ and t̚, and [ɑ]: d̚|ɑ and t̚|ɑ. For a middle leaf, say, 62, which is arrived at by the nodes, 0, 1, 3, 8, and 16, the left contexts are unreleased dental stops, d̚ and t̚, and the right contexts are æ, ɑʷ, and ɛ. The corresponding transition contexts are: d̚|æ, d̚|ɑʷ, d̚|ɛ, t̚|æ, t̚|ɑʷ, and t̚|ɛ. Leaves, 55, 56, 40, 27, and 28, contain exemplars with a front, high vowel on the right side, because they descend from the right branch of node 1 - matching i and y (in both intra-syllable and inter-syllable positions.) For example, leaf, 55, contains g|i, k̚|i, k|i, and p|i. Leaves 27 and 28 both contain transitions with i and y in the right context, but the left contexts differ in voicing because of the branch made at node 14 between t̚/t and d̚.

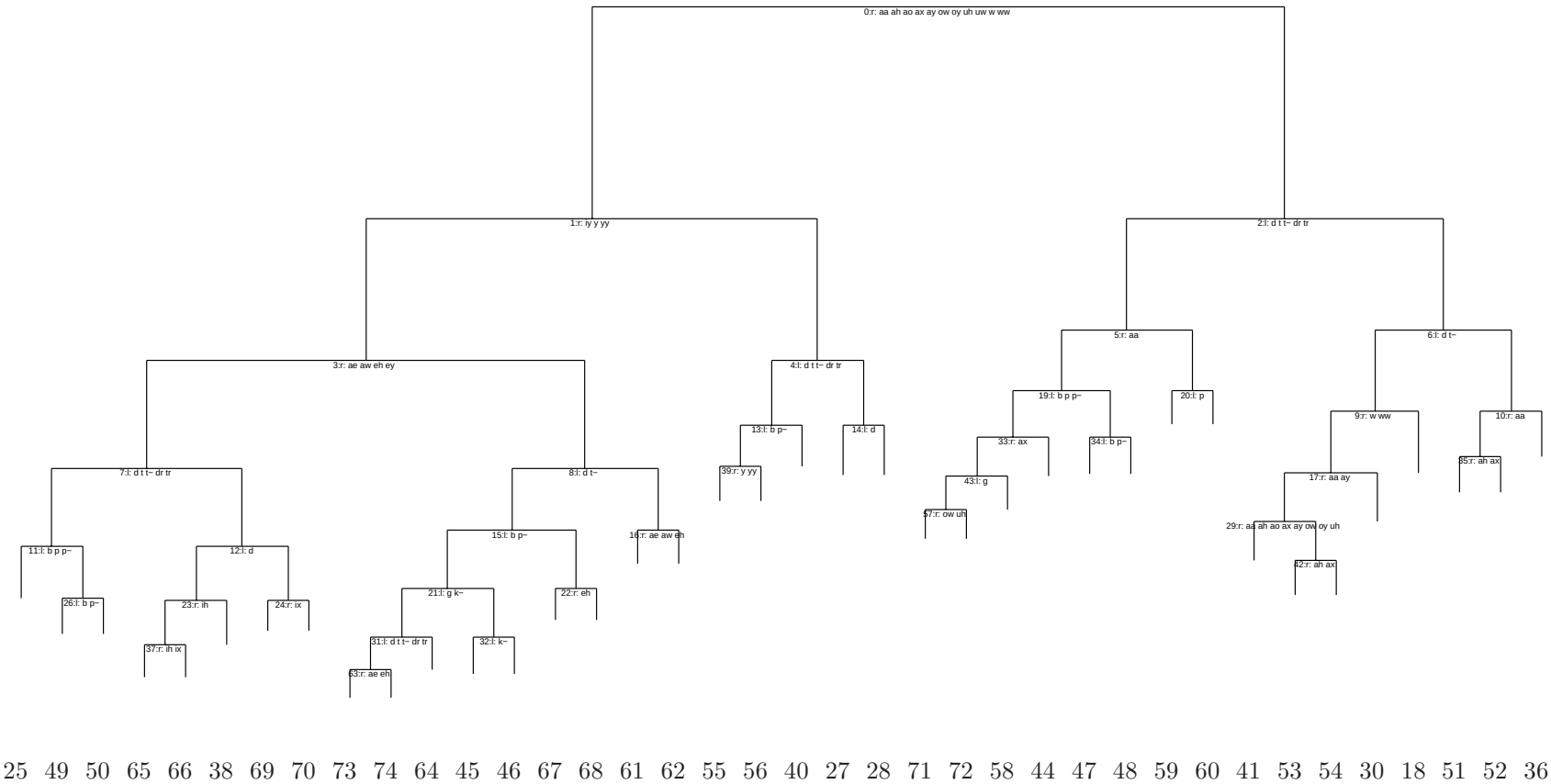


Figure 4-5: Sample clustering run for stop-vowel and stop-glide transitions.

The clustering run presented here produces a fairly balanced tree for dividing stop-vowel and stop-glide transitions into more homogeneous clusters. Decisions made at each node branch either left or right depending on whether matching criteria are satisfied. In the case of clustering transition contexts, nodes can be divided by either left or right contexts unlike before. For example, leaf, 62, is reached by nodes, 0, 1, 3, 8, and 16 and contains exemplars of transitions with unreleased dental stops, d̚ and t̚, on the left of the boundary and vowels, æ, ɑʷ, and ɛ on the right of the boundary. The transition contexts are: d̚|æ, d̚|ɑʷ, d̚|ɛ, t̚|æ, t̚|ɑʷ, and t̚|ɛ.

4.4 Characterization

With data grouped into distinct clusters by the stage of categorization, a characterization is made by summarizing or distilling the data into models. These models are used as the basis of comparison for determining synthesis costs. A model is defined over an observation space and measurements taken from this space are used to build the model. Segment measurements from the prospective observation space (the right-most 30% portion to the left of a boundary) will be used to define the right-sided substitution costs and segment measurements from the retrospective observation space (the left-most 30% portion to the right of a boundary) will be used to define the left-sided costs. Boundary measurements taken jointly from both spaces are used to define the concatenation costs. By making a multi-variate Gaussian assumption, the building of models entails calculating first- and second-order statistics. The measurements are stacked into multi-dimensional vectors and running counts of the first-order moment ($\frac{1}{N}\sum_{i=1}^{N} x_i$) and second-order moment ($\frac{1}{N-1}\sum_{i=1}^{N} x_i^T x_i$) are accumulated.

For the substitution costs, the first F mel-frequency cepstral coefficients ($\mathrm{MFCC}_{1:F}$) averaged over segmental thirds serve as the observation vector. For the right-sided and left-sided substitution model, the coefficients are averaged over the right-most and left-most 30% portions of the segment, respectively. For the concatenation costs, the first F mel-frequency cepstral coefficients are averaged in a telescoping fashion about the boundary. The innermost set of blocks are from 0 to ± 5 milliseconds. The next set of blocks range from ± 5 to ± 15 milliseconds. This continues until the final set of blocks which range from ± 35 to ± 75 milliseconds. A total of 8 blocks span a region of 150 milliseconds about the boundary.

With the segment and boundary measurements segregated by substitution and concatenation classes, characterization by statistical summarization completes the acoustic model for concatenative speech synthesis. The final step of comparison for es-

tablishing inter-model distances is detailed in the next section which automatically determines substitution and concatenation costs from data.

4.5 Comparison

The parameter reduction framework outlined in Chapter 2 paved the way for approximating instance-to-instance costs with cluster-to-cluster costs. With the clusters created by categorization and characterization as described in this chapter, a comparison of the clusters will determine the substitution and concatenation costs.

A particular application of the general parameter reduction framework introduced in Chapter 2 is made by choosing the Kullback-Leibler distance, an information-theoretic metric (see Appendix D for more details), as the measure for inter-cluster distances. Although the framework was derived for a metric, such as the Mahalanobis distance, which obeys the Triangle Inequality, an exception is made here to better fit the communication-theoretic formulation of unit selection also presented in Chapter 2 in which the penalty of symbol corruption is measured by description length costs. Minimizing symbol corruption caused by the communications channel (unit selection) is accomplished by minimizing these description lengths. This can be thought of as an minimum description length (MDL) framework.

4.5.1 Information-theoretic distance metrics

In creating an automatic procedure for determining synthesis costs, the Kullback-Leibler (KL) divergence measure serves as the main workhorse. Although KL distance, $\mathcal{D}(p \mid\mid q)$, has many interpretations in different contexts, the underlying theme is that it represents the asymmetric cost of approximating p with q. In communications, it describes additional information the transmitter must supply about q given

that the receiver already has information about p, $H(P)$. In an MDL framework, it quantifies the additional number of bits required to describe q given p. Equation 4.1 is a definition using the difference between a cross entropy and a self entropy. The second term, self entropy, is $H(P)$ and represents given information. By subtracting it away from the first term, the residual is additional information required to use q in place of p. Because of the communication-theoretic framework introduced in Chapter 2 for unit selection, symbol corruption can be directly measured by the KL distance.

$$\mathcal{D}(P \mid\mid Q) = \int p(x) \log \frac{p(x)}{q(x)} \, dx = E_P \left[\log \frac{1}{q(x)} \right] - E_P \left[\log \frac{1}{p(x)} \right] \tag{4.1}$$

Because Kullback-Leibler distance captures the notion of asymmetric substitution and provides an associated cost, it is well suited to defining substitution costs. When two contexts have a similar acoustical effect on a segment, spectra taken from the segment should be similar. In statistical terms, distributions of spectra conditioned on either of the two contexts should be similar. The similarity is measured by the KL distance; the KL distance between contexts which have similar acoustical effects will be low. By measuring the KL distance between models defined over the prospective space representing different right-sided substitution classes, a set of right-sided substitution costs can be constructed. The reverse argument applies for constructing left-sided substitution costs.

For a unit α, Equation 4.2 defines the right-sided cost for substituting the desired context of β with a backoff context of γ. The probability distribution functions, $p(x_r \mid \alpha[\beta])$ and $p(x_r \mid \alpha[\gamma])$, are conditionally estimated from the right-most 30% portion of the segment, α, under the conditions of the right context being β and γ, respectively.

$$S^r_{\alpha[\beta] \rightarrow \alpha[\gamma]} \equiv \mathcal{D}\left(p(x_r \mid \alpha[\beta]) \mid\mid p(x_r \mid \alpha[\gamma]) \right) \tag{4.2}$$

Similarly, Equation 4.3 defines the left-sided cost for substituting the desired context of β with a backoff context of γ and the probability distribution functions, $p(x_l \mid [\beta]\alpha)$ and $p(x_l \mid [\gamma]\alpha)$, are conditionally estimated from the retrospective observation space,

the left-most 30% portion of the segment, α, under the conditions of the left context being β and γ, respectively.

$$S^{l}_{[\beta]\alpha\rightarrow[\gamma]\alpha} \equiv \mathcal{D}\left(p(x_l \mid [\beta]\alpha) \mid\mid p(x_l \mid [\gamma]\alpha)\right) \tag{4.3}$$

Together, Equations 4.2 and 4.2 define a three-step process for determining substitutions costs:

1. Take segmental measurements under different context conditions.

2. Estimate conditional probability distribution functions.

3. Measure Kullback-Leibler distance.

Because the Greek letters, α, β, and γ, can be phones or any class of phones, any arbitrary categorization can be used to group contexts.

Another information-theoretic measure closely related to KL distance is mutual information which measures statistical independence between two random variables. As shown in Equation 4.4, it can be defined as the KL distance between a joint distribution and the product of its marginal distributions, which describes how well a joint distribution can be factored into marginal distributions. By carefully choosing P and Q later, a direct connection between mutual information and a desired notion of concatenation cost can be made.

$$\begin{aligned} \mathcal{I}(P\ ;\ Q) &= \mathcal{D}(p_{P,Q}(x,y) \mid\mid p_P(x) * p_Q(y)) \\ &= \int\limits_{X,Y} p_{P,Q}(x,y) \log \tfrac{p_{P,Q}(x,y)}{p_P(x) * p_Q(y)}\ dx\ dy \end{aligned} \tag{4.4}$$

Another definition relates mutual information to the difference between unconditional and the conditional entropies. If knowledge about Q does not reduce the uncertainty in P (i.e., $H(P \mid Q) = H(P)$), then there is zero information in common. Equation

4.5 relates to this mathematical statement.

$$\mathcal{I}(P\ ;\ Q) = H(P) - H(P \mid Q) \tag{4.5}$$

Let $\alpha|\beta$ be the boundary between α and β and define P and Q to be the spectral distributions estimated from telescoping averages of the boundary measurements, $p(x_{r_t} \mid \alpha[\beta])$ and $p(x_{l_t} \mid [\alpha]\beta)$. The joint distribution is $p(x_{l_t}, x_{r_t} \mid \alpha|\beta)$. If $\alpha|\beta$ is a good concatenation boundary, then α and β are fairly separable (independent) and can be broken into $\alpha[\beta]$ and $[\alpha]\beta$. In other words, the joint distribution (representing α and β contiguously) can be factored into marginal distributions (representing $\alpha[\beta]$ and $[\alpha]\beta$ non-contiguously.) This is how mutual information as stated in Equation 4.4 relates to concatenative synthesis. An interpretation of Equation 4.5 follows.

If the mutual information is low across a concatenation boundary, the conditional entropy is high and little information is communicated across the boundary. Knowledge about spectra on the right side of the boundary sheds little light on spectra on the left side of the boundary. Since there is large spectral change at the boundary, previous work described in Chapter 3 shows that a concatenation at this point would be less discernible. An information impasse makes for a good concatenation point. The boundary is a communications barrier and this is how mutual information as stated in Equation 4.5 relates to concatenative synthesis. This notion is also found in statistical machine translation systems [17] where *rifts* [6] are used to segment sentences into chunks that can be individually translated. Mutual information has also been applied to infer boundaries between grammar constituents [82, 16].

With the statistical independence and communications arguments made above, a definition for concatenation cost by mutual information naturally follows. By rewriting a non-contiguous transition unit, $\alpha]|[\beta$, as a concatenation of $\alpha[\beta]$ and $[\alpha]\beta$, the concatenation cost is defined as the approximating the contiguous transition unit, $\alpha|\beta$, with the concatenation of two non-contiguous parts; this approximation is the

definition of mutual information.

$$\begin{aligned} C_{\alpha]|[\beta} &\equiv \mathcal{D}(p(x_{r_t}, x_{l_t} \mid \alpha|\beta) \mid\mid p(x_{r_t} \mid \alpha[\beta]) * p(x_{l_r} \mid [\alpha]\beta)) \\ &= \mathcal{I}\left(p(x_{r_t} \mid \alpha[\beta]) \; ; \; p(x_{l_t} \mid [\alpha]\beta)\right) \end{aligned}$$

When a multivariate Gaussian assumption is made, expressions (see Appendix D for derivations) in terms of nats (natural bits of $\ln 2$ bits) for KL distance and mutual information have a closed form:

$$\begin{aligned} \mathcal{D}(P \mid\mid Q) &= \tfrac{1}{2}\left[(\mu_Q - \mu_P)^T \Sigma_Q^{-1}(\mu_Q - \mu_P) \; + \right. \\ &\qquad \left. Tr\left[\Sigma_P \Sigma_Q^{-1} - I\right] - \log|\Sigma_P \Sigma_Q^{-1}|\right] \\ \mathcal{I}(P \; ; \; Q) &= \tfrac{1}{2}\log\left(\tfrac{|\Sigma_{pp}| \; |\Sigma_{qq}|}{|\Sigma|}\right), \quad \Sigma = \begin{bmatrix} \Sigma_{pp} & \Sigma_{pq} \\ \Sigma_{qp} & \Sigma_{qq} \end{bmatrix} \end{aligned}$$

The expression for KL distance is a sum of three terms: a Mahalanobis distance (under Q), a trace term, and a logarithmic term. The logarithmic term can be regarded as the ratio of two volumes, $\ln(|\Sigma_P|/|\Sigma_Q|)$. $\Sigma_P \Sigma_Q^{-1}$ is a ratio of covariance matrices and measures how similar the dispersions of the random variables, P and Q, are. When $\Sigma_P = \Sigma_Q$, the ratio is unity (the identity matrix) and the trace and logarithmic terms vanish and the KL distance becomes a Mahalanobis distance, obeying the Triangle Inequality. Because of this, the KL distance can, in practice, be used within parameter reduction framework laid out in Chapter 2 for both intra-cluster and inter-cluster distances. For substitution costs, Equation 2.4 becomes:

$$\begin{aligned} || \, x - y \, ||_M &\approx || \, x - m_X \, ||_M \; + \; || \, m_X - m_Y \, ||_M \; + \; || \, y - m_Y \, ||_M \\ &\approx \mathcal{D}(\mathcal{N}(x, \Sigma_X) \mid\mid \mathcal{N}(m_X, \Sigma_X)) \; + \; \mathcal{D}(\mathcal{N}(m_X, \Sigma_X) \mid\mid \mathcal{N}(m_Y, \Sigma_X)) \; + \\ &\qquad \mathcal{D}(\mathcal{N}(m_Y, \Sigma_Y) \mid\mid \mathcal{N}(m_Y, \Sigma_Y)) \end{aligned}$$

We model the observations with Gaussian distributions because closed-form expressions exist for their entropies. Although others have used numerical integration or empirical estimates [155], Gaussian distributions are simple to estimate with only first- and second-order statistics needed. Also, the distribution for a class of units

can be formed by combining statistics of individual units in a bottom-up fashion.

4.6 Analysis

This section describes experiments testing the automatic frameworks for determining classes and costs. First, knowledge-derived classes from previous work are used to test the information-theoretic metrics and to produce a set of semi-automatically determined costs. Second, classes derived from decision tree clustering are used to calculate a set of fully automatically determined costs.

In the first case, a corpus consisting of around 100 minutes of in-house, wideband recordings from one speaker reading typical system responses in weather information, flight status, and air travel domains was used. Because the corpus is small, behavior of the costs was analyzed across broad classes of speech sounds. This is equivalent to tying or fixing the parameters to be the same for members within the broad class.

In the second case, a larger corpus of 124,180, telephony-quality spontaneous utterances from multiple speakers in a weather domain was used. This sample of the English language is larger and classes and costs could be calculated at the phonetic level; no parameter tying was necessary in this case.

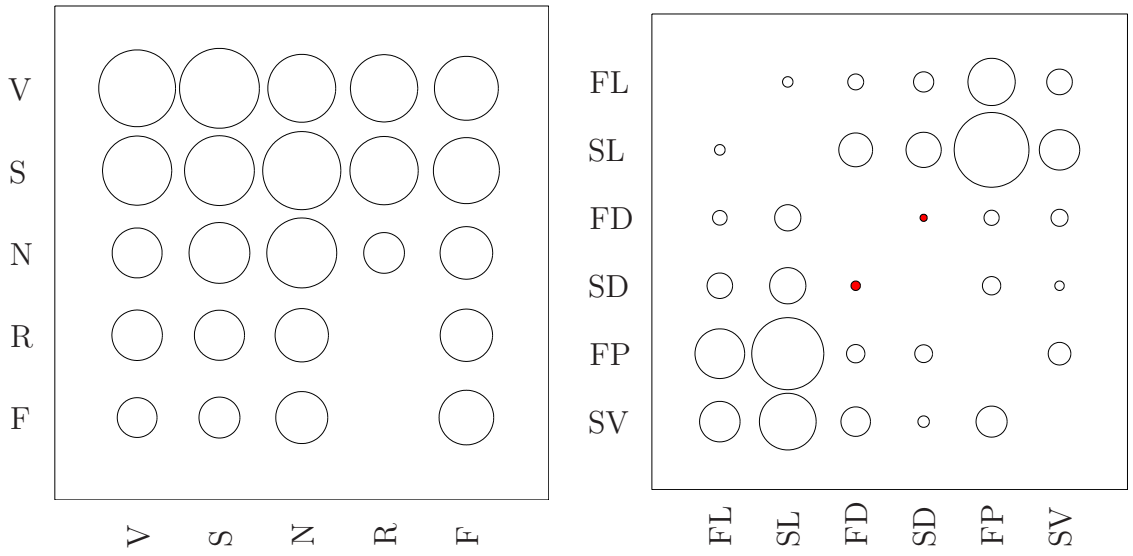
Visual representations of costs produced by the information-theoretic metrics are shown in this section. Experiments carried out with automatically determined costs are described in Chapter 6.

4.6.1 Automatically determined costs

Figure 4-6 displays bubble plots of concatenation and substitution costs. For the purpose of visualization, the radius of the circle (not the area) is used to depict the

magnitude of a cost. The classes are held fixed from previous work so that direct comparisons of the costs can be made.

Figure 4-6(a) shows a bubble plot of concatenation costs where the rows and columns represent the context to the left and the right of the concatenation boundary, respectively. The contexts are vowels, semivowels, nasals, and obstruents. Because concatenations are less discernible at places of source changes, the mutual information is expected to be lower at such places. Indeed concatenations between vowels and semivowels, for example, where the source does not change, are most costly. Semivowel-nasal and vowel-vowel concatenations are the next most costly. Fricative-vowel, nasal-stop, and fricative-semivowel boundaries are the least costly concatenations.



(a) Concatenation cost.

(b) Substitution cost.

Figure 4-6: Bubble plots of concatenation and substitution costs.

Left: Bubble plot of concatenation costs matrix. Rows and columns correspond to context (vowel, semivowel, nasal, stop release, fricative) on the left and right sides of phonetic boundaries. Right: Bubble plot of substitution costs matrix for phonetic context to the right of vowels. Rows and columns correspond to true and approximating contexts (fricatives and stops in labial, dental, palatal, and velar places of articulation).

Figure 4-6(b) displays the right-sided substitution costs for vowels in terms of varying contexts: fricatives and stops in labial, dental, palatal (more precisely, fricatives and affricates), and velar places of articulation. Because the lips are responsive articulators, transitions leading into labial stops are very spectrally distinctive and this is reflected in the large substitution costs seen in the second row. Large costs also appear for interchanging palatal and labial places. Finally, it is undesirable to substitute a velar stop with a labial stop.

Because the dynamic range of the concatenation costs does not appear to be high, it is possible that the concatenation classes are overly broad. In contrast, the classes in the example substitution cost matrix are low in size with only two or four members each. The next section reports results when the classes are also automatically determined from data.

4.6.2 Automatically determined classes

Table 4.1 displays automatically determined classes for left contexts of [i]. AN contains alveolar nasals, FR contains fricatives, RL contains rhotic liquids, BS contains back stops plus [p], LS contains unreleased labial stops, LN contains labial nasals, LL contains lateral liquids, LG contains segments containing labial glides and offglides, LC contains labial stop closures, and OC contains all other contexts.

Figure 4-7 displays left-sided substitution costs for [i] with automatically determined classes. Columns with large bubbles, LG, LC, and LN, primarily correspond to labial contexts which exert strong influence on the beginning of [i]. This suggests that labial contexts leave indelible marks and are poor imposters for other contexts. Substitution costs are fairly small across the board for the row corresponding to fricatives. This suggests that the beginning of [i] is well shielded from the frication and that contexts besides labial glides can be substituted.

Class	Members
AN	n̩ n ŋ ɾ̃
FR	č ð ɾ f h ǰ s š θ v z ž
RL	-r ɚ ɝ ɾ rr
BS	d dr g k k˺ p t t˺ tr
LS	b p˺
LN	m̩ m
LL	-l l̩ l̥ ll
LG	ɑʷ oʷ uʷ w ww
LC	b□
OC	- ‿ ɑ æ ʌ ɔ ə ɛ ɐ ɪ ɨ i ɔʸ ʊ ü y yy epi d□ g□ k□ p□ t□ ʔ

Table 4.1: Substitution classes for [iʸ].

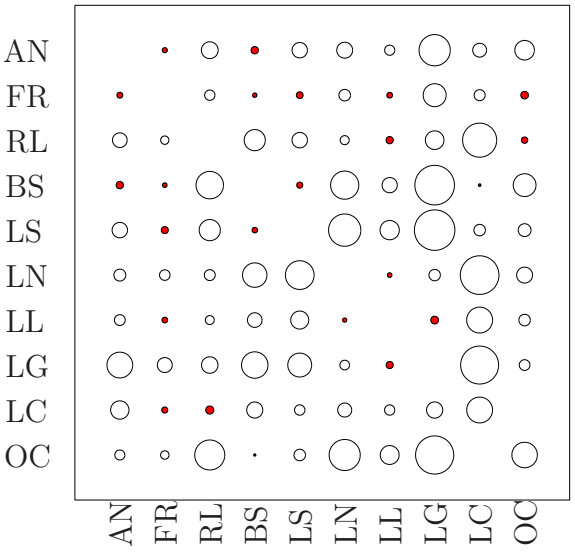


Figure 4-7: Substitution costs for [iʸ].

A total of 1,116 concatenation classes were determined by the decision tree clustering algorithm. While it is not possible to list all the classes, spot checks can be made once a statistical summarization is in hand. Figure 4-8 shows two panels containing the distribution of concatenation costs and a normality test and two panels containing the same for costs that have undergone logarithmic transformation. The mean cost, 1.997, is to the right of the median cost, 1.800, and so there is positive skew (the skew is to the right) as also confirmed by a skewness[1] of +1.361. The distribution is quite peaky (leptokurtotic) with a kurtosis[2] of 5.424. In the normal probability plot, the empirical cumulative distribution function is plotted against the data. Note that the percentiles are normally spaced along the abscissa. If the normality test holds, the scatter plot should fall along a straight line. The costs appear to be well modelled by a lognormal distribution, because after a logarithmic transformation, the normality probability plot is fairly linear and the skewness and kurtosis are close to the theoretical values of 0 and 3, respectively, for a normal distribution.

Another interesting way to view the concatenation costs is presented in Figure 4-9

[1]Skewness is defined by $E[(x-\mu)^3]/\sigma^3$, the ratio of the third central moment and the cube of the standard deviation. For a normal distribution, skewness is zero.

[2]Kurtosis is defined by $E[(x-\mu)^4]/\sigma^4$, the ratio of the fourth central moment and the square of the variance. For a normal distribution, kurtosis is 3.

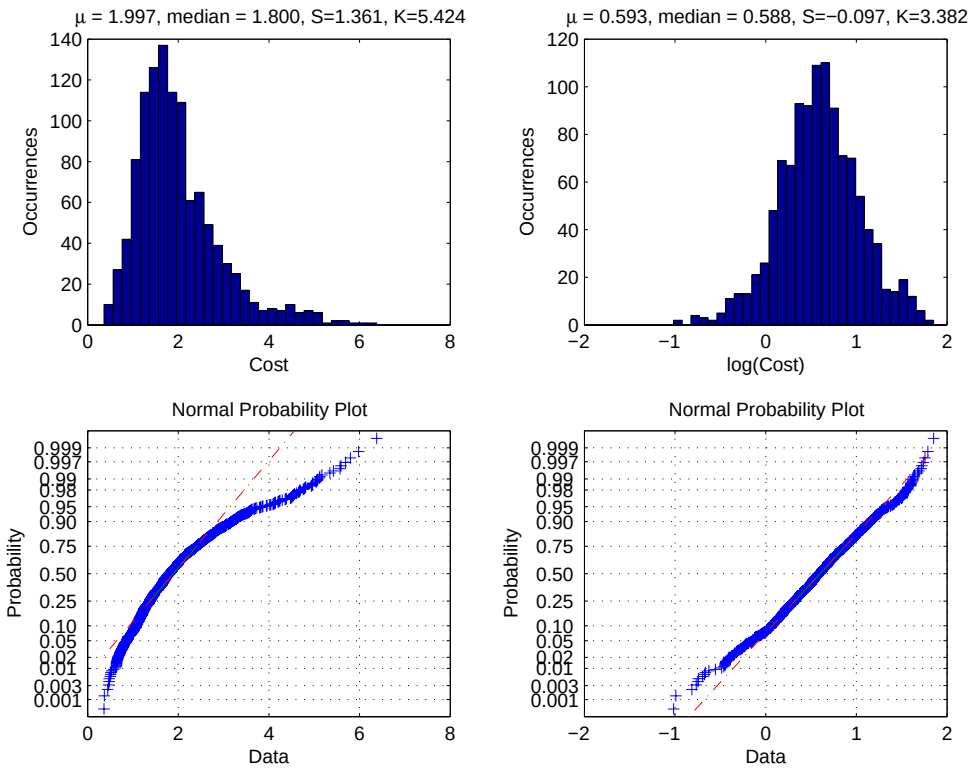


Figure 4-8: Distributional analysis of automatically determined concatenation costs.

Concatenation costs appear to take on a peaky shape with a positive skew, statistical moments which are rather well normalized after a logarithmic transformation. Properties of the distributional are used to focus spot checks on specific concatenation classes, which, for the most part, seem to confirm hypotheses coincident with speech knowledge.

in which the number of observations for a concatenation class is plotted versus the concatenation cost. This direct relationship exists, because the observation space for the boundary model is joint of the prospective and retrospective space or the left and right sides of the boundary, respectively. The inverse relationship suggests that large costs happen for classes with few observations and small costs happen for classes with many observations, although the trend for the former is far more pronounced.

These statistical and distributional criteria can be used to select a small number of the 1,116 concatenation classes for spot checks. The four concatenation classes with the most number of observations are transitions between different types of silences including end-of-utterance silence, epenthetic silence, and unvoiced stop closures. Confirming an earlier hypothesis from Chapter 3, concatenations within inter-vocalic [l]

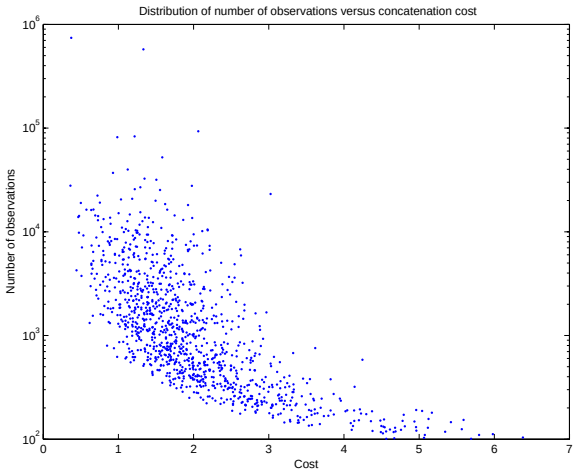


Figure 4-9: Analysis of number of observations in concatenation class versus concatenation costs.

A scatter plot of concatenation costs along the ordinate with the number of observations for the concatenation class along the abscissa seems to suggest an inverse relationship. It is not clear whether poorly sampled concatenation models have a bias towards higher costs.

segments have a cost of 1.610, below the mean of 1.997. However, for reasons unclear, this was not the case for inter-vocalic [r] segments, which have a high cost of 4.534. Perhaps the boundaries were not consistently placed or the high cost was associated to its rare occurrence which placed below the fifth percentile (4.26%). A frequently occurring transition, iʸ|y, had a cost of 1.649, confirming a hypothesis of concatenating in the off-glides of diphthongs. A class containing ɑʸ|y and ɔʸ|y had a concatenation cost of 2.091, while a class containing eʸ|iʸ, eʸ|y, y|y, and y|iʸ had a low cost of 1.206. Transitions at ʷ off-glides had higher costs: for example, oʷ|w, 2.620 and uʷ|w, 2.568. Due to the nature of the clustering procedure in which left and right contexts can be split in arbitrary order, it is difficult to present all 1,116 concatenation costs in an organized manner.

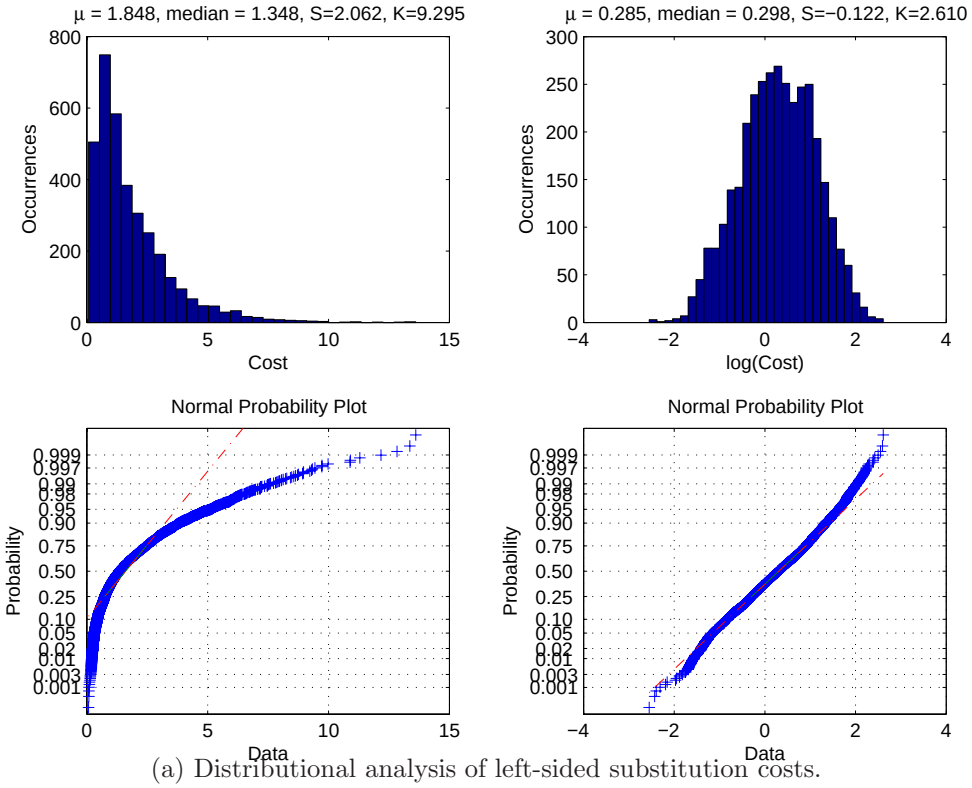
Some of the lowest costs mostly consisted of transitions from phones to silence. Nasal-fricative and fricative-fricative transitions were especially low too. Interestingly enough, the third highest cost of 5.798 was attributed to the transition between silence and [b] in which pre-voicing during the silence interval is highly correlated with the stop. Vowel-vowel as well as retroflex-vowel transitions had especially high

costs too. The highest cost, 6.380, was a class of transitions from the vowels, æ, ɛ, and ü, to w. Other high costs occurred within syllabic rhymes or offsets, uʷl, 5.691, oʷr, 5.124, uʷr, 3.821, ʊl, 3.509.

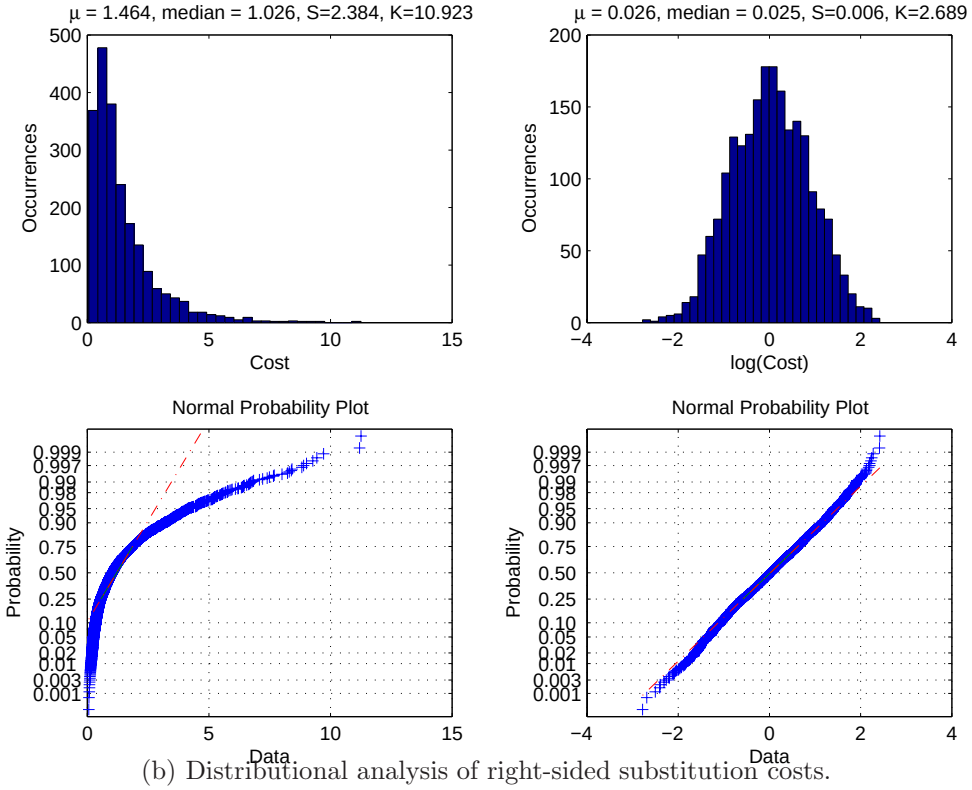
A similar distributional analysis can be performed for substitution costs if they are pooled. For example, the 90 separate costs learned for substitutions of context on the left side of [iʸ] as shown in Figure 4-7 are combined with all left-sided costs from other phones. After pooling, the distributional analysis for left-sided and right-sided substitution costs is displayed in Figure 4-10. Because kurtosis is less than three, the distributions are somewhat more flat (platykurtotic) than a normal distribution. The skew is well removed by the logarithm function for the right-sided case, although not so for the left-sided case.

The absolute values can not be compared with the concatenation costs, because of differences in the observation space. However, the relative values for the left- and right-sided cases, means of 1.848 and 1.464, respectively, are more comparable, because of similar observation spaces and features. Still, the conclusion proffered by the plots would imply that co-articulatory effects are more pronounced in a preservatory (retrospective) than in an anticipatory (prospective) sense.

The analysis presented here has looked at the distributional properties of synthesis costs which appear to take on lognormal distributions. The question of whether entropies take on lognormal distributions remains open. The analysis was used to focus attention on certain concatenation and substitution costs and to make spot checks. Future investigations should examine how best to summarize the information encoded by these costs in a more disciplined manner.



(a) Distributional analysis of left-sided substitution costs.



(b) Distributional analysis of right-sided substitution costs.

Figure 4-10: Distributional analysis of automatically determined substitution costs.

4.7 Related work

Much of the earlier work in the literature has concentrated on instance-level costs that directly compare speech segments, or instantiations of the speech units. Numerical metrics such as Euclidean, Kullback-Leibler, and Mahalanobis distances calculated over spectral features have been considered [21, 22, 39, 61, 69, 160, 137, 70, 161].

The concatenation cost defined here bears similarity to disconcatibility as proposed by Iwahashi [63] and to splicing cost proposed as by Bulyko [22]. The use of mutual information to find boundaries across which information is blocked is related to rifts [6] as studied in statistical machine translation [17]. Automatic determination of contextual classes has been addressed by decision tree clustering [99, 97, 98, 13] and Hidden-Markov Models [41, 37].

4.8 Summary

An acoustic model is built for concatenative speech synthesis by categorization and characterization of speech data. The classes used in grouping data into categories are automatically determined by a decision tree clustering algorithm. Segment models are estimated from measurements taken from the outermost portions of a segment and substitution costs given by the KL distance are used to approximate instance-to-instance costs. Boundary models are estimated from telescoping averages around a boundary and concatenation costs are given by the mutual information of measurements around a boundary.

Because the information-theoretic criteria introduced here are generic, the effectiveness of the synthesis costs will mostly depend on the design of the observation space and subsequent density estimation. As in most speech recognition work, an assumption of normality or Gaussianity is made for simplicity. Although recognition and

synthesis are inverse problems with many commonalities [104], the best performance in each problem may not necessarily be achieved by the same type of features. The current observation vector looks the same amount ahead and backward in time. It might be conceptually attractive to design heterogeneous features [54]. For example, the time windows could adapt to the phonetic context, or they might be biased towards the future for modelling speech sounds in languages that have more anticipatory rather than carry-over co-articulation. A practical issues not addressed here is the balancing of concatenation and substitution costs. The dimensionality of the observation space for the concatenation cost is double that of the dimensionality of the observation space for the substitution cost, suggesting that the entropy of the former is double too. Also, because the observation spaces from which boundary and segment measurements are taken are entirely different, the costs may be of different scale and need to be combined in a weighted fashion.

Chapter 5

Finite-State Transducers

5.1 Introduction

This chapter describes the use of Finite-State Transducers (FST) to encode the networks participating in the unit selection search. The reasons for adopting finite-state methods are legion including that the representation is concise and that the algorithms which operate on transducers are well understood. In this chapter, FST's are introduced as a research tool in the speech community by exploring past applications. Secondly, the cascade of FST's which define the unit selection search space and their individual topologies are described. Next, various search algorithms are defined along with pruning techniques and their empirical performance is examined on unit selection tasks. Then, the Viterbi and A^* search algorithms are integrated with an intervening rescoring stage into a multi-pass search. Finally, possible extensions to the foregoing are discussed.

5.2 Background

Because many processes in speech processing can be modelled with finite-state constraints, finite-state transducers [115] have unleashed a wave of unification and are gaining ever increasing popularity within the speech and language processing community. While the design of FST computational tools may not be trivial, the FST concept itself is one which represents a canonical model for computation. The algorithms are need only be written once and represent a toolbox with which to manipulate transducers. Applications include decision trees [134], pronunciation variation modelling [94, 57], speech recognition [51], and text analysis for text-to-speech synthesis [132].

5.3 Unit selection cascade

As described in Chapter 2, unit selection is a constrained optimization that determines from an input specification the optimal sequence of waveform segments to concatenate. In the following exposition, the input specification is assumed to be composed of a sequence of words which may be optionally marked with information such as supra-segmental knowledge [76]. The mapping from words to waveform segments as shown in Figure 5-1 is captured by a composition (see Appendix A for a definition) or succession of FST's: W, L, P, T, and S. This factoring allows us to independently model, design, and refine the underlying processes. When combined with a search, these components work in succession from an input sequence of words to produce a sequence of waveform segments suitable for concatenation.

Each of the transducers, W, L, P, T, and S, serve a purpose in effecting the transformation from words to waveform segment descriptors. W, is a finite-state acceptor (FSA) which specifies the word sequence. L is a lexicon which maps words to phonemes and can capture multiple pronunciations. P, which maps phonemes to

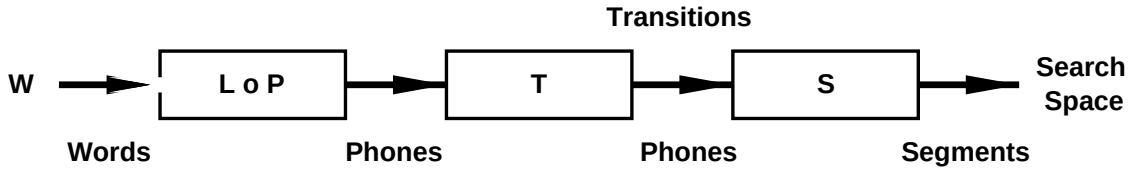


Figure 5-1: FST representation of unit selection.

The search space is represented by the FST cascade, $((W \circ (L \circ P)) \circ T) \circ S$. The lexical network, $L \circ P$, is taken from our speech recognition work. The final composition between $((W \circ (L \circ P)) \circ T)$ and S takes place on the fly as part of a Viterbi search, the efficiency of which is highly enhanced by the network topology of S.

phones, models pronunciation variations. To be described in later sections, T inserts intervening transition [64] (diphone-like) symbols for the primary purpose of search efficiency. While the first four FST's find inverse counterparts in a similar decomposition of the speech recognition process, the topology of the final FST, S, makes unit selection possible and computationally efficient. That phonological units - as opposed to another spatio-temporal representation such as, for example, syllables - are encoded by the transducer has no impact on the validity and generality of the FST framework. The use of the phone as the unit for selection merely clarifies presentation.

In practice, W can be a graph that produces multiple wordings of the same message for variety. For words whose pronunciations may differ in a context-dependent manner (e.g., read $\rightarrow$ r (ɛ | iʸ) d), creating lexical variants of words enables explicit selection, in this example, of word tense (e.g., past versus present) or word sense (e.g., noun versus verb), as in IMport versus imPORT. While the pronunciation variation generated in P should be maximally accommodative in speech recognition, it may be desirable to limit the variation in speech synthesis for the purpose of increasing intelligibility. Soft limits can be achieved by weighting arcs, whereas hard limits can be achieved by removing arcs - effectively infinite weighting. Finally, L, P, and T can be composed and optimized ahead of time (i.e., $opt(L \circ P \circ T)$) to minimize computation at run-time. See Appendix A for more details on transducer optimization.

The final ingredient in FST-based unit selection is the synthesis FST, S, which maps

phones (with transition phones) to waveform segments. The actual composition between $((W \circ (L \circ P)) \circ T)$ and S takes place during a Viterbi search to be described later. Properties of S which expedite this search include low perplexity (average branching factor) and scalability. The solution presented in this work decouples the domain-independent and domain-dependent portions residing in S. Specifically, domain-independent or language-dependent costs relating only to the pertinent language form a constraint kernel that encodes speech knowledge for concatenative speech synthesis. Phonetic transcriptions of individual utterances are represented as finite-state phone chains and are connected to the constraint kernel at the phone level. If the input word string, W, is identical to a particular utterance in S, then that utterance can be retrieved in its entirety by simply walking its finite-state chain, because there will be a one-to-one correspondence between the phones in the input specification and the phones in the utterance. Otherwise, the constraint kernel, by generating transition symbols to match those in the input specification, allows fragments from different utterances to mix and match producing new speech. These transition symbols also have an associated weight and the optimal transition point between two utterances depends on local context in the first utterance before the transition and local context in the second utterance after the transition.

5.4 Input specification

To give an example of how input is specified, Figure 5-2 depicts the word sequence, 'Boston', as represented by an FST, W. Because the input is the same as the output, this FST is actually an FSA, and a singular label is displayed here.

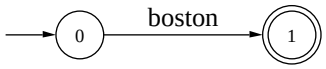


Figure 5-2: FST of the word, 'Boston.'

This FST, W, encodes the word sequence, 'Boston'. Actually an FSA, W absorbs and emits 'Boston' on the left and right side, respectively.

When this example W is composed with the lexicon and pronunciation rules, $(L \circ P)$, on the right, the resulting FST, $W \circ (L \circ P)$, as displayed in Figure 5-3, absorbs 'Boston' on the left side and emits the corresponding phonetic sequence, [bɔstɪn], on the right side. Note that the 'Boston' label on the input side coincides with the first phone, [b], of its pronunciation, and that ASCII symbols are used for the phonetic symbols.

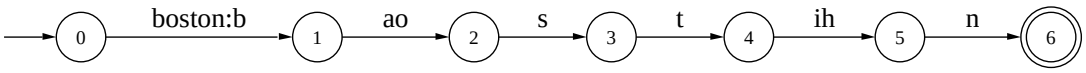


Figure 5-3: FST of the pronunciation of the word, 'Boston.'

This FST, $W \circ (L \circ P)$, encodes the pronunciation for the word, 'Boston'. 'Boston' is absorbed on the left side and the corresponding phonetic sequence, [bɔstɪn], is emitted on the right side.

Up to this point, the pronunciation is the same one as in the speech recognizer used to derive phonetic transcriptions encoded in S for consistency reasons. However, P could be more restrictive and under-generate pronunciations to assume different speaking style. Now the phonetic sequence is populated with intervening transition symbols as shown in Figure 5-4, the purpose of which is to encode context at concatenation boundaries. The resulting FST, $W \circ ((L \circ P) \circ T)$, absorbs the word, 'Boston', on the left side and emits the corresponding phonetic and transition sequence, b b|ɔ ɔ ɔ|s s s|t t t|ɪ ɪ ɪ|n n, on the right side.

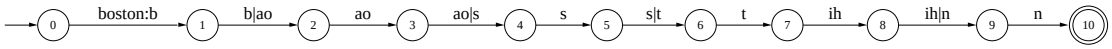


Figure 5-4: FST of the pronunciation and transitions for 'Boston.'

This FST, $W \circ ((L \circ P) \circ T)$, encodes the pronunciation and transition phones for the word, 'Boston'. 'Boston' is absorbed on the left side and the corresponding phonetic and transition sequence, b b|ɔ ɔ ɔ|s s s|t t t|ɪ ɪ ɪ|n n, is emitted on the right side.

In practice, $(L \circ P) \circ T$ is pre-computed and the two steps illustrated above in Figures 5-3 and 5-4 happen in one step. For more than one word in W, the words absorbed on the left side coincide with the first phone of their pronunciations on the right side. $(L \circ P) \circ T$ generates pronunciations for only known words, although this can be remedied by using dynamic FST's to handle out-of-vocabulary words [120].

5.5 Corpus utterances

Figure 5-5 gives an example of how the transcriptions of recorded utterances are encoded in S with finite-state phone chains. Phone are absorbed on the left side and waveform segment descriptors, which are composed of waveform tags, and start and end samples, are emitted on the right side. On the other hand, transition symbols are absorbed with no corresponding emission and serve only a bookkeeping purpose to be described later.

In this FST, S, the input phone and transition symbols have a one-to-one correspondence with the output phone and transition symbols of the input specification in Figure 5-4. When they are composed, the word, 'Boston', can be synthesized. However, since there is no freedom in S to enter late or depart early, it can only be used to synthesize the word, 'Boston'. One way to alleviate this problem is to introduce skip arcs as depicted in Figure 5-6 which enable the selection of sub-word sequences from the word, 'Boston'. For example, the first syllable, [bɔs], could be selected by departing early, taking the arc from state 5 to state 11. Note that early departure arcs follow the emission of waveform segment descriptors and that later entry arcs precede the emission of waveform segment descriptors.

Although sub-sequences can be selected (e.g., [bɔs] or [tɪn]), they are strictly contiguous and chronological in order. Introducing yet more skip arcs will enable the selection of sequences that are non-contiguous and anti-chronological. This time, however, the skip arcs must bear transition symbols which did not naturally occur. For example, in the FST displayed in Figure 5-7, introducing an arc with transition symbol, b|ɪ, from state 1 to state 8 allows the creation of the sequence, [bɪn], from non-contiguous units. The arc with transition symbol, t|ɔ, introduces an anti-chronological transition that could be followed to create the sequence, [tɔs], from non-contiguous units.

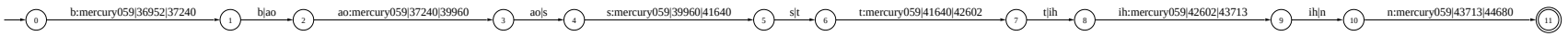


Figure 5-5: FST for corpus representation of utterance, 'Boston.'

This FST depicts a typical finite-state phone chain in S which encodes the transcription for a recorded utterance. Phones are absorbed on the left side and waveform segment descriptors are emitted on the right side. Transition symbols, which are absorbed without emission, play a bookkeeping role to be described later. In this example, an utterance tagged as mercury059 contains speech for the word, 'Boston'. Each waveform segment descriptor consists of the waveform tag, and start and end samples delimited with |.

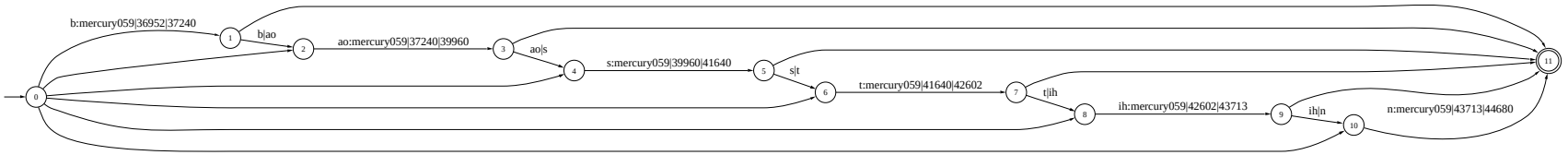


Figure 5-6: FST for corpus representation of utterance, 'Boston', with skip arcs.

The FST from Figure 5-5 is augmented here with skip arcs that enable the selection of sub-word sequences. Early departure arcs follow the emission of waveform segment descriptors and later entry arcs precede the emission of waveform segment descriptors. For example, the first syllable, [bɔs], could be selected by departing early, taking the arc from state 5 to state 11.

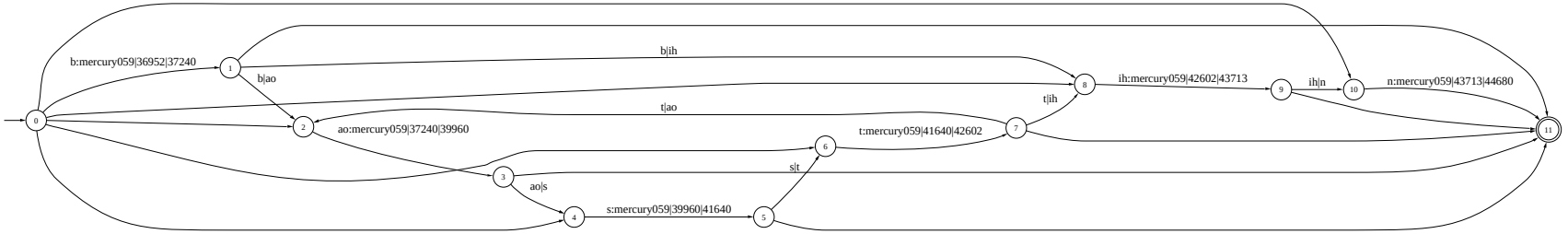


Figure 5-7: FST for corpus representation of utterance, 'Boston', with skip arcs for non-contiguous and anti-chronological selection.

The FST from Figure 5-6 is further augmented here with arcs bearing transition symbols that enable the selection of sequences that are non-contiguous and anti-chronological. For example, the sequence, [bɪn], can be selected by following the arc with transition symbol, b|ɪ, from state 1 to state 8.

Although it is possible to add more and more skip arcs allowing for the selection of more and more sub-sequences that are non-contiguous and anti-chronological in nature, this process will lead towards a quadratic number of arcs. Also, it is not clear that the naïve addition of arcs with transition symbols will select sequences that sound natural. For example, the left context of [ɔ] is a voiced, labial stop, [b], whereas the left context of [ɪ] is an unvoiced, dental stop, [t]. Co-articulatory effects will likely be observed in the leftmost portions of the vowels, [ɔ] and [ɪ], which would likely lead to conflicting co-articulation in novel sequences like [bɪn] and [tɔs]. Penalties, of varying degrees, should be assessed for these types of mismatch and, in the presence of more exemplar units, these penalties would be used to prioritize certain pairings of non-contiguous units over others. The next section describes how all possible skip arcs are enumerated along with associated penalties and packaged into a constraint kernel of manageable size (i.e., independent of corpus size).

5.6 Constraint kernel

As described in Chapter 2 and empirically demonstrated in Chapter 4, contexts which have similar effects on the acoustics of speech segments can be grouped into equivalence classes based on symbolic information alone. The search can be in a finite number of states - determined by past context - as it progresses and future decisions are based on past and present information. This section describes the organization of language-dependent constraints or costs into a constraint kernel which can be reused in other domains. This notion of reuse is in congruence with the desire to amortize the energy of aggregating costs for a particular language into a structure which can be used for future applications in other domains. Although the framework is presented using only one symbol of context before and after (i.e., triphone), it is extensible to additional context.

The constraint kernel organizes costs which connect all M units in a corpus. As

described in Chapter 2, in corpora with large M, the number of costs, M^2, is large, and incrementally adding new units takes $2M + 1$ operations. What the constraint kernel does is to introduce a structure to the costs whose size is independent of the size of the corpus and only dependent on the number of speech sounds and co-articulatory contexts in a language; this can be a big win for corpora with large M.

The diagram in Figure 5-8 depicts the four intermediate layers of the constraint kernel, along with portions of two corpus utterances (top and bottom of figure). It builds upon the example of 'Boston' begun in Figure 5-5. For now some details are omitted for clarity (e.g., output labels of arcs containing waveform segment descriptors corresponding to source and destination utterances). In the example shown, the layers bridge a pair of source ('Bosnia') and destination ('Austin') utterances synthesizing 'Boston' from [bɔ] in 'Bosnia' and from [stɪn] in 'Austin'. The outermost layers of the kernel (1 and 4) serve to multiplex and demultiplex transitions from and to speech segments in the synthesis corpus, respectively. Each state in these two layers connects all instances of a transition between two particular phones in the corpus. The two corpus utterances are only shown to illustrate the function of the constraint kernel and are not considered as part of the constraint kernel.

The units are phonological and the concatenation and substitution costs depend only on the local phonetic context of the transition. For example, the state marked ɔ|z in layer 1 in Figure 5-8 gathers all instances of transitions between [ɔ] and [z] in the corpus (with one of them being in the word 'Bosnia'). Likewise, the state marked ɔ|s in layer 4 connects to all instances of transitions between [ɔ] and [s] in the corpus (with one of them being in the word 'Austin'). The states in layer 1 encode the current context and future context whence the arc comes. The states in layer 4 encode the past and current context thither the arc goes. In general, if there are P phones in the language, there are P^2 states in each of these two layers.

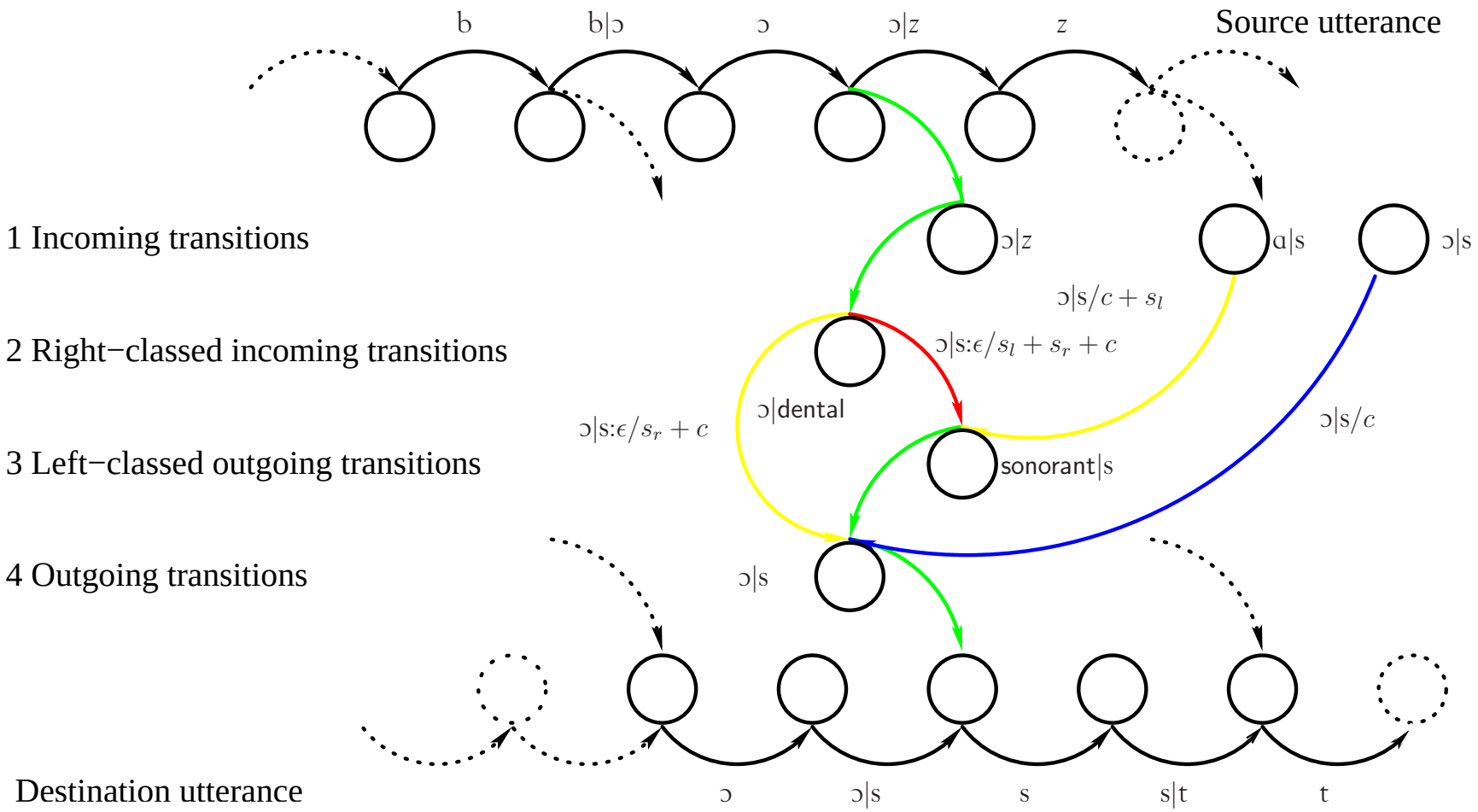


Figure 5-8: FST, S, contains corpus utterances and constraint kernel for efficient unit selection.

Utterances are connected by a language-dependent constraint kernel of a fixed size (i.e., number of states and arcs) which contains four layers to intermediate all possible connections. The outermost layers (1 and 4) connect from and to the utterances in the corpus. When contexts are matching, there are direct connections from layer 1 to layer 4 with only concatenation costs. When contexts are completely mismatched, the innermost layers (2 and 3) are connected by arcs with concatenation and substitution costs. In the case of partially matching contexts, arcs from layers 1 to 3 and arcs from layers 2 to 4 encode costs with one-sided substitution costs.

For complete generality, layers 1 and 4 should be fully interconnected so that any state in layer 1 can connect to any state in layer 4 with an associated concatenation cost, c, and substitution cost, s. Since there are P^2 states in layers 1 and 4 each, a full interconnection would require P^4 arcs between these two layers. By making use of the parameter reduction introduced in Chapter 2, states in layers 1 and 4 can be grouped into clusters from which a significantly smaller number of inter-cluster costs can be calculated. The equivalence classes are learned by clustering based on decision trees as described in Chapter 4 and merge speech sounds which exhibit similar co-articulatory behavior.

This is the motivation for introducing a set of classed layers, layers 2 and 3, which encode prospective and retrospective context, respectively. The assumption made in Equation 2.3 justifies the separation of right and left context into layers 2 and 3, respectively. The substitution cost, s, is broken down into a left and right cost, s_l and s_r, respectively. The left cost is associated with matching the left phone of a layer 1 state with the left phone of a layer 4 state. The right cost is the penalty associated with matching the corresponding right phone labels.

In Figure 5-8 with the ɔ|z transition in layer 1 for example, the right equivalence class for the /z/ is determined by the /ɔ/. In this case, all consonants with an alveolar or dental place of articulation are deemed to have the same effect on the vowel, and are grouped into a dental equivalence class, as shown in layer 2. Similarly, with the ɔ|s transition in layer 4, the left equivalence class for the /ɔ/ is determined by the /s/. In this case, all sonorants are deemed to have the same effect on the fricative, and are grouped into a sonorant equivalence class, as shown in layer 3. The cost of connecting these two states is the cost, s_l, of substituting an /ɔ/ as a sonorant, the cost, s_r of substituting an /s/ as a dental consonant, and the concatenation cost, c, of making a splice between a vowel /ɔ/ and a fricative /s/ (arc weights are delineated by a '/' in Figure 5-8). In this example, the substitution and concatenation are very reasonable, so the overall cost is small.

If there are an average of $\bar{C}_r$ right-sided equivalence classes per phone and an average of $\bar{C}_l$ left-sided equivalence classes per phone, then layers 2 and 3 have a total of $\bar{C}_r P$ and $\bar{C}_l P$ states, respectively. To provide complete generality, these two layers are fully connected with $\bar{C}_l \bar{C}_r P^2$ arcs. Since $\bar{C}_r, \bar{C}_l << P$, this structure requires significantly fewer arcs than fully connecting all states in layers 1 and 4 (P^4 arcs). Table 5.1 summarizes the number of states in the four layers. When $\bar{C} = \bar{C}_l = \bar{C}_r$, this expression for the total number of states, $P(\bar{C}_l + \bar{C}_r + 2P)$, can be simplified to $2P(P + \bar{C})$.

Layer	Function	Number of states
1	Multiplex transitions	P^2
2	Class right context	$P\bar{C}_r$
3	Class left context	$P\bar{C}_l$
4	Demultiplex transitions	P^2
Total		$P(C_l + C_r + 2P)$

Table 5.1: Number of states in each layer of constraint kernel.

Layers 1 and 4 contain one state per transition, of which there are P^2 where P is the number of phones. Layers 2 and 3 are classed versions of layers 1 and 4 containing $P\bar{C}_r$ and $P\bar{C}_l$ states, respectively.

Each state in layer 1 connects to only one state in layer 2, and each state in layer 4 connects to only one state in layer 3. Although inter-cluster costs are used in the connections between layers 2 and 3, intra-cluster costs have not been utilized. These would correspond to the generalizations made by the arcs between layers 1 and 2 (e.g., how does the ɔ|dental cluster differ from the ɔ|*z* cluster) and the specializations made by the arcs between layers 3 and 4 (e.g., how does the ɔ|s cluster differ from the sonorant|s.) Going from layer 1 to 2 loses information and going from layer 3 to 4 gains information. These generalization and specialization costs correspond to the intra-cluster costs in the parameter reduction framework and merit future investigations as sparse data issues (i.e., training P^2 transition models) may come into play.

In order to account for making splices where the contexts match exactly, P^2 direct connections are made between states in layers 1 and 4 which have identical labels. The only cost associated with these arcs is the concatenation cost, c, which depends on

the particular transition context. An example arc in Figure 5-8 travels from the ɔ|s in layer 1 directly to the corresponding state in layer 4. Similarly, there are $\bar{C}_r P^2$ direct connections between layers 2 and 4 for all cases where the left labels are identical. An example arc is illustrated in Figure 5-8 for this case since the /ɔ/ vowel is common between the two states. In this case the total cost is the right substitution cost, s_r, and the concatenation cost. Finally, there are also $\bar{C}_l P^2$ direct connections between layers 1 and 3 for all cases where the right labels are identical. An example arc in Figure 5-8 connects the ɑ|s state in layer 1 to the sonorant|s in layer 3. The cost on these arcs is a concatenation cost plus a left substitution cost.

As shown in Table 5.2, the total number of arcs in the constraint kernel, $P^2(\bar{C}_l\bar{C}_r + \bar{C}_l + \bar{C}_r + 1)$, is therefore fixed, and is independent of the size of the speech corpus itself. When $\bar{C} = \bar{C}_l = \bar{C}_r$, this expression can be simplified to $(P(\bar{C}+1))^2$. This along with the total number of states, $2P(P+\bar{C})$, as tallied in Table 5.1 determines the storage requirements of the constraint kernel.

Layers	Left match	Right match	Number of arcs
$2 \rightarrow 3$	No	No	$P^2\bar{C}_l\bar{C}_r$
$1 \rightarrow 3$	No	Yes	$P^2\bar{C}_l$
$2 \rightarrow 4$	Yes	No	$P^2\bar{C}_r$
$1 \rightarrow 4$	Yes	Yes	P^2
Total			$P^2(\bar{C}_l\bar{C}_r + \bar{C}_l + \bar{C}_r + 1)$

Table 5.2: Number of arcs in between layers of constraint kernel.

Because states in layers 1 and 4 are labelled by transitions, they can be directly connected on a one-to-one basis with only P^2 arcs. Between states in layers 2 and 4 are labelled by one-sided classed transitions, there is an expansionary factor of $\bar{C}_l\bar{C}_r$. Between layers 1 and 3 and between layers 2 and 4, there are only expansionary factors of $\bar{C}_l$ and $\bar{C}_r$, respectively, because one side of context matches.

When M, the number of units in the corpus, is large, using the constraint kernel to organize the costs may be preferable for two reasons: 1) $(P(\bar{C}+1))^2$ arcs is far less than M^2 arcs, 2) adding new units incrementally does not entail additional computation - $(P(\bar{C}+1))^2$ is independent of M. The constraint kernel trades off precision for small size and fast incremental speed. For every utterance with M phones, there are $2M$ connections made to the constraint kernel: M arcs connecting from the end of

every phone to the matching transition state in layer 1, and M arcs connecting from the outgoing transition states of layer 4, to the start of the matching phone in the utterance.

In real terms, using rough estimates for the number of phones and classes for any given language, $P = 100$ and $C = 9$, the number of arcs is 1000^2. The constraint kernel begins to pay dividends when $M > 1000$: when more than 1000 units are present in the corpus, using the proposed constraint kernel greatly reduces the number of arcs. Given a typical syllabic speaking rate of 4Hz and 2.5 phones per syllable, the typical phonetic speaking rate is 10Hz. In other words, the constraint kernel begins to pay dividends after $100s$ of speech - roughly a minute and a half - is recorded.

Although this estimate may be somewhat of an oversell for the constraint kernel - as phonotactics, the set of all possibly allowed arrangements of speech sounds in a given language, would imply that not necessarily each of the M^2 phone pairs would be actually be instantiated - this ballpark figure of $100s$ demonstrates the spoils of using a framework motivated by parameter reduction wherein instance-to-instance costs are replaced by cluster-to-cluster costs. Of course, the presupposition is that inter-cluster costs indexed by symbolic context are sufficient for a first-pass, coarse search possibly followed up by a second-pass, fine search. As alluded to earlier, generalization (from layer 1 to layer 2) and specialization (from layer 3 to layer 4) costs should be targeted for future studies. Arcs arriving at layer 1 and departing layer 4 can have intra-cluster costs that explain how well exemplar speech transitions are fit by the transition models.

5.7 Design considerations for search

As mentioned earlier, the constraint kernel enumerates all possible substitutions of transitions and associated penalties. In Figure 5-8 the arc between state, ɔ|dental, in

layer 2 and state, sonorant|s, in layer 3 contains the transition symbol, ɔ|s, and an associated penalty consisting of concatenation and left and right substitution costs. With only 'Bosnia' and 'Austin', the corpus lacks an ɔ|s transition, the constraint kernel provides that very missing transition. This is, however, not the only ɔ|s transition the constraint kernel is capable of supplying. Depending on class definitions, it could supply an ɔ|s transition between, for example, state, ɔ|velar, in layer 2 and state, sonorant|s. If ɔ has $\bar{C}_r$ such right classes in layer 2 and if s has $\bar{C}_l$ such left classes in layer 3, there will be $\bar{C}_l\bar{C}_r$ arcs between layers 2 and 3 bearing the transition symbol, ɔ|s. Table 5.3 displays similar analyses for arcs between other layers when context is partially matched.

Layers	Left match	Right match	Number of transitions
$2 \rightarrow 3$	No	No	C_lC_r
$1 \rightarrow 3$	No	Yes	$\bar{C}_l$
$2 \rightarrow 4$	Yes	No	$\bar{C}_r$
$1 \rightarrow 4$	Yes	Yes	1
Total			$C_lC_r + \bar{C}_l + \bar{C}_r + 1$

Table 5.3: Expansionary factors between layers that impact search.

When the search traverses between layers across arcs which contain transition symbols, there are expansionary factors which quantify the replication of identical transition symbols. These numbers can be calculated by dividing the numbers in Table 5.2 by P^2.

Notice that the number of transitions enumerated in Table 5.3 is the result of analysis for ɔ|s. Repeating the analysis for all P^2 transitions would scale these numbers by P^2 and result in numbers identical to those in Table 5.2. The numbers here are, in fact, expansionary factors for the increase in simultaneous search paths when the constraint kernel is entered. The constraint kernel is, to use a monetary metaphor, a lender of last resort and provides liquidity where there is none. To the extent that the corpus covers all the units at least once, those transitions that are missing from the corpus are manufactured by the constraint kernel.

It should be noted that the cross-connections between layers 2 and 3, 1 and 3, 2 and 4, and 1 and 4, absorb the transition labels which are inserted between phones via the T FST described earlier. The use of the transition labels in this manner adds significant

constraint to the topology of S, greatly reducing the number of active search nodes, and enabling real-time synthesis. Without transition symbols to identify the arcs, all arcs in the constraint kernel would have an ϵ label and would have to be expanded during the search.

Besides being, metaphorically speaking, a lender of last resort, the constraint kernel also produces transition symbols that break the chronology of the recorded utterances; non-contiguous segments of speech can be mixed and matched to produce novel speech sequences. The search will find low-cost transition points where non-contiguous segments can be joined to form a new, contiguous sequence. To the extent that speech requested by the input specification lies in a contiguous sequence in the corpus, the search algorithm is encouraged to pick up contiguous sequences, because the finite-state phone chains have zero weight and can be walked indefinitely barring mismatch with respect to the input specification; once a mismatch in the transition symbol is encountered, the search path is diverted off into the constraint kernel. Subsequences of the finite-state phone chains can be traversed without penalty and the search is therefore greedy. A side benefit of the constraint kernel bearing arcs with transition symbols (as opposed to none) is that the search will avoid infinite looping through the constraint kernel, because a transition symbol must be paired with a matching transition symbol in the input specification.

5.8 Search

The role of the search is to find the least-cost sequence of speech utterance segments for a given input specification. Specifically, the search finds the least-cost path through the composition of $(W \circ (L \circ P) \circ T)$ with S. If this composition were to be performed in its entirety, an exponentially large search *trellis* would be created as explained in Chapter 2. The need for expanding this trellis incrementally one *slice* at a time and for making decisions sequentially is obvious. Fortunately, a well-known

dynamic programming technique called the Viterbi search fits the bill perfectly.

The input specification, $W \circ (L \circ P) \circ T$, is optimized and topologically sorted so that the states are numbered in increasing order from the initial state to final state. To perform the composition on the fly, the Viterbi search algorithm walks through the states of the first transducer, the input specification, in topological order exploring compatible arcs in the second transducer, S, in tandem. Arcs in the first transducer which emit output labels that match input labels absorbed by the second transducer are deemed to be compatible. Valid end points of the search when the first and second transducer both transition to final states.

As previously described, the search algorithm proceeds to pick up phonetic units and output corresponding waveform segment descriptors from an utterance until a transition mismatch is detected. Then, it is forced off the current utterance path into the constraint kernel to gather the missing transition. Finally, it is off once again in search of another portion of the corpus containing the subsequent phonetic unit. The search path will contain a sequence of waveform segments that best reconstitute the speech symbols in the input specification.

At each state in the first transducer, the second transducer may be in a multitude of states, because of the expansion in search paths. Call this set of states a slice. The dynamic programming heuristic is that for each state or *node* in the slice, only the best path up to that node needs to be stored in the form of a back pointer. Once the final state in the first transducer is reached, a backtrace is constructed by starting at the node with the best score in the final slice. Then, the trellis is traversed backwards in time, one slice at a time, following the trail of back pointers which represent local decisions.

The dynamic programming algorithm as described above will find the best path with the lowest score as if the entire composition was fully performed. This guarantee exists even when weights on the arcs are negative - the path score may not be monotonically

increasing. This guarantee does not necessarily hold true when the larger cost savings afforded by the Viterbi search are realized - when the slice at each state in the first transducer is pruned of poorly scoring nodes. Not only does this speed up the search by discarding paths from the search, it also saves memory for storing the paths.

Pruning works by eliminating nodes from a slice in two ways: score-based and count-based pruning. Score-based pruning only keeps nodes whose scores lie within a given threshold from the best scoring node in the slice. Count-based pruning only keeps a certain maximum number of nodes in each slice. Nodes which survive the pruning process are referred to as being in the pruning beam. The larger the thresholds are, the wider the beam is. When pruning is used, accuracy is traded off for speed and the best path is not guaranteed to be found.

The score threshold and maximum count are parameters which can be tuned for real-time operation. The effects of pruning on speed and accuracy are examined next in an empirical fashion. In the special case when a slice is pruned down to a single node, the Viterbi backtrace can be initiated early and a binding decision for the best subsequence of waveform segment descriptors up to that point can be made. This can be used to pipeline synthesis and reduce the latency in response time and is described in a following section.

5.9 Pruning

The purpose of the experiments performed in this section is to illustrate the effects of pruning on search accuracy and speed. The task used in these pruning experiments, training and test sets consisting of 1,496 and 22 utterances taken from an air-travel domain, respectively, is described in more detail in the next chapter. The score and count pruning thresholds are swept through different settings, while the accuracy and speed are measured.

To calculate accuracy, a reference path is first obtained by running the search without pruning. This path will contain a sequence of N waveform segment descriptors for subsequent waveform generation. A segment error rate (SER) is then defined as the degradation from this reference path in terms of descriptors that are substituted, inserted, or deleted: $SER = \frac{S+I+D}{N}$. With pruning activated, the more faithful the answer is to the reference path, the lower the segment error rate is.

Speed is defined as a real-time ratio (RTR) which is the search time divided by the speaking time of the synthesized waveform. A real-time ratio of less than unity means that the search produces an answer faster than the answer can be spoken. The real-time ratio acts somewhat as a measure like bandwidth does in data communications. However, this is not the only component that determines interactive response time. Not measured here is latency, which is defined as the time from the start of the search to the first output of waveform segment descriptors, which also plays an important role. This will be addressed in the next section.

Figure 5-9 shows the segment error rates and real-time ratios measured over the test set as the score threshold is swept from 0 to 7 in increments of 0.1. As the beam is tightened, the best path under pruning conditions deviates from the reference path and SER rises as indicated by the 'x' symbols. As the beam is relaxed, SER tends towards zero. In contrast, the real-time ratio indicated by the 'o' symbols decreases (the search speeds up) as pruning is stricter and more nodes are discarded in each slice from the trellis. The SER and RTR curves appear to be fairly smooth.

Figure 5-10 shows the segment error rates and real-time ratios measured over the same test set, but this time count threshold is swept from 0 to 5000 in increments of 100. As the search progresses and a new slice is expanded, that slice is referred to as the *active frontier* in this plot. As more nodes are retained on the active frontier, the more accurate , but slower the search is. This time, the SER and RTR curves do not appear to be smooth. The SER curve appears to contain tipping points when the SER rises quickly only to fall to lower lows as the count threshold is increased. When

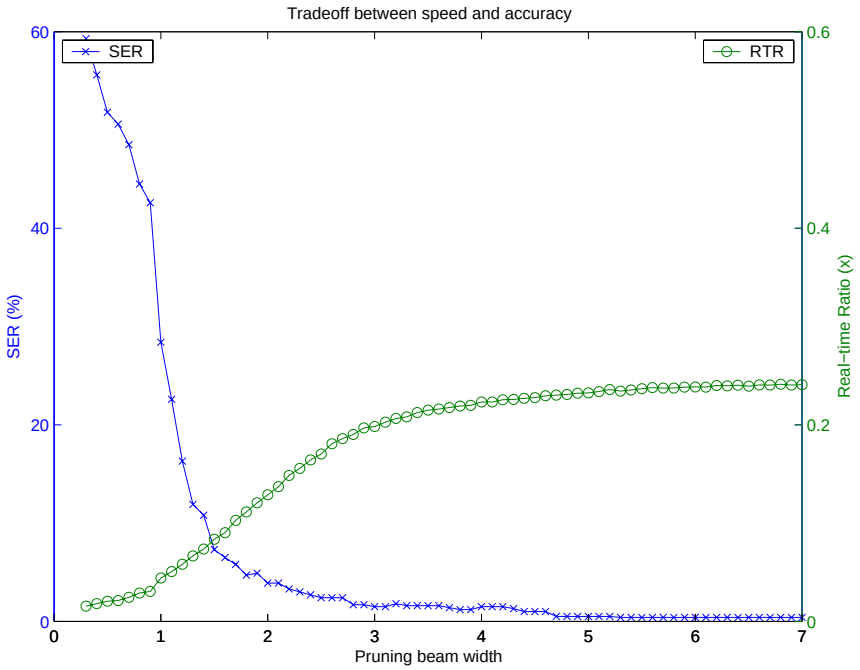


Figure 5-9: Tradeoff between search speed and accuracy as pruning beam varies.

This figure depicts the tradeoff between search speed indicated by the 'o' symbols and accuracy indicated by the 'x' symbols as the width of the pruning beam is varied. Searching can be sped up (falling real-time ratio) as the pruning beam is tightened to the detriment of search accuracy as evidenced by a falling segment error rate (SER is an error rate calculated with respect to the no-pruning best path).

tipping points are reached, a new portion of the search space is suddenly included that does not necessarily decrease SER. The RTR curve is fairly smooth until around 3500 when the ultimate tipping point is reached as also evidenced by the drastic drop in SER. Now, a sufficient number of nodes is included in the active frontier and the reference path is within reach of the search. After slight bump in the RTR curve, the real-time ratio levels off as the reference path always stays inside the search beam to ward off false, would-be competitors.

5.9.1 Pushed weights

In working with finite-state transducers, when weights can be pushed outside so that they can be assessed earlier, the effectiveness of pruning may be further enhanced. By

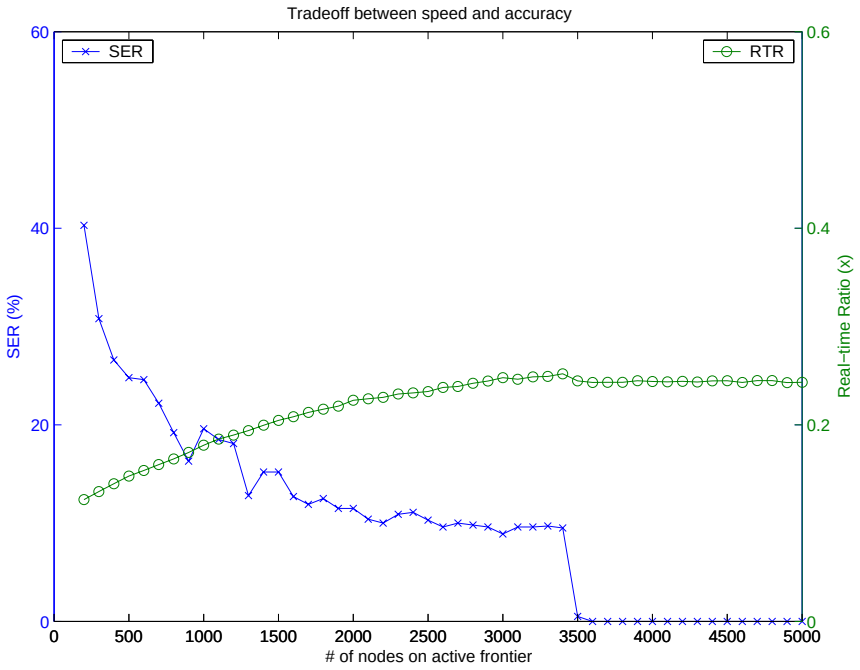


Figure 5-10: Tradeoff between search speed and accuracy as number of nodes varies.

This figure depicts the tradeoff between search speed and accuracy as the number of nodes on the Viterbi frontier is varied. Searching can be sped up (falling real-time ratio) as the number of nodes is reduced to the detriment of search accuracy as evidenced by a falling SER. In contrast to Figure 5-9 which shows a smooth drop in SER to zero percent, this chart shows that SER abruptly drops to zero past a certain number of nodes. This is due to the correct answer remaining in the Viterbi frontier after surviving the pruning process.

observing that the concatenation cost depends only on the transition, its associated cost can be pushed out of the constraint kernel in S and into the transition FST, T. The transition FST, T, can be modified to output two types of transition symbols to distinguish between transitions that occur between contiguous (e.g., $\alpha|\beta$) and non-contiguous (e.g., $\alpha]|[\beta$) units. The brackets,][, around the | delimiter create another type transition symbol which signifies that the units are not contiguous. Figure 5-11 illustrates a possible input specification. The concatenation cost, c, can be assessed ahead of entering the constraint kernel and used to prioritize and prune paths which are considering a non-contiguous transition. What is also required and not shown here is that concatenation costs need to be removed from the constraint kernel leaving only left and right substitution costs. In S, transition symbols in the constraint kernel would be purely non-contiguous (e.g., $\alpha]|[\beta$) and transition symbols in the corpus

utterances would be purely contiguous (e.g., $\alpha|\beta$) and zero cost.

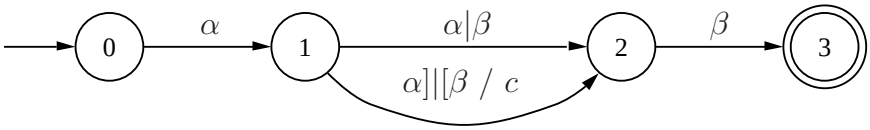
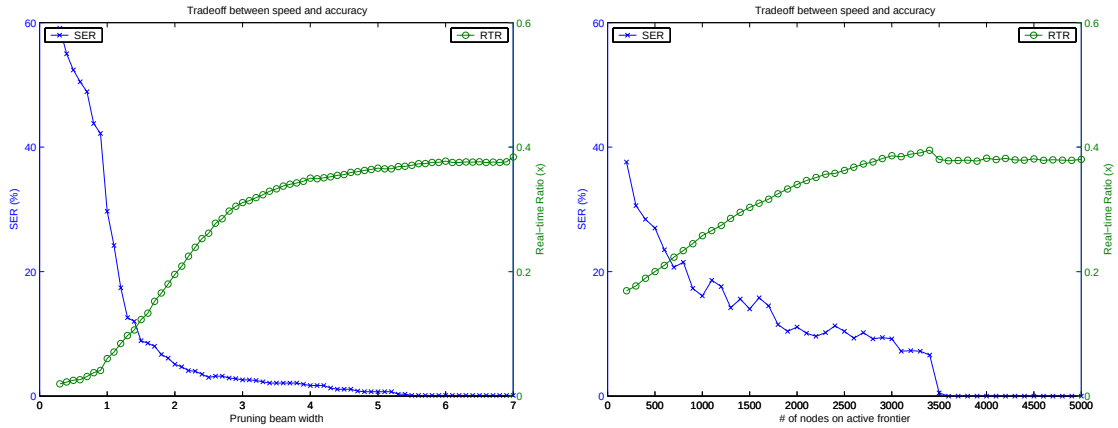


Figure 5-11: Example of pushing weights from S to T.

Because the concatenation cost is a parameter that is effectively tied across many states in the constraint kernel, it can be pushed outside of the constraint kernel for (possibly) better pruning behavior. An alternate way of generating transitions would involve the explicit generation of symbols pertaining to contiguous and non-contiguous transitions. The latter would contain a weight for the concatenation cost.

Figure 5-12 shows SER and RTR curves calculated with the same setup as the plots shown earlier in Figures 5-9 and 5-10 except that concatenation costs have been pushed from S to T. By themselves they are rather unremarkable as they bear a great resemblance to the curves obtained in the original setup. Figure 5-12(a) corresponds to 5-9 and Figure 5-12(b) corresponds to 5-10.



(a) Tradeoff between search speed and accuracy as pruning beam is varied for pushed weights configuration.

(b) Tradeoff between search speed and accuracy as number of nodes is varied for pushed weights configuration.

Figure 5-12: Tradeoff between search speed and accuracy as score-based and count-based thresholds are varied for pushed weights configuration.

This figure reports similar segment error rate and real-time ratio curves as those illustrated in Figures 5-9 and 5-10 except that the weights are pushed.

Figures 5-13 and 5-14 overlay the SER and RTR curves for the original setup and the

setup with pushed weights (marked by the '+' and '*' symbols, respectively). Differences in RTR are clearly revealed, whereas differences in accuracy are not conclusive. The real-time ratio for the setup with pushed weights is higher, because the search is slowed down. The additional arc bearing a non-contiguous transition symbol and a weight requires additional path exploration in the search. It is not clear why accuracy does not always benefit from pushed weights from S into T. For example, for a given level of accuracy, sometimes it can be achieved with less nodes in the original setup than in the setup with pushed weights, or vice versa. When the score threshold is adjusted, the original setup appears to be more accurate most of the time. Further studies should examine weighting the concatenation costs which reside in T versus the substitution costs which reside in S and its impact on accuracy.

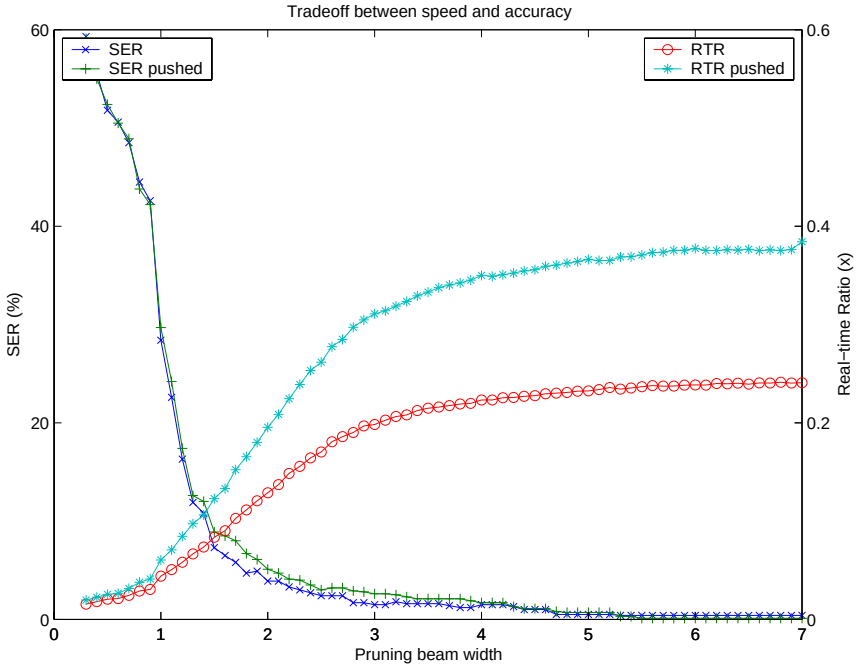


Figure 5-13: Overlay of tradeoff between speed and accuracy when pruning beam is varied for original and pushed weights configurations.

This figure overlays the plot from the left side of Figures 5-12 with Figure 5-9. The '+' and '*' symbols indicate the SER and RTR for the configuration with pushed weights.

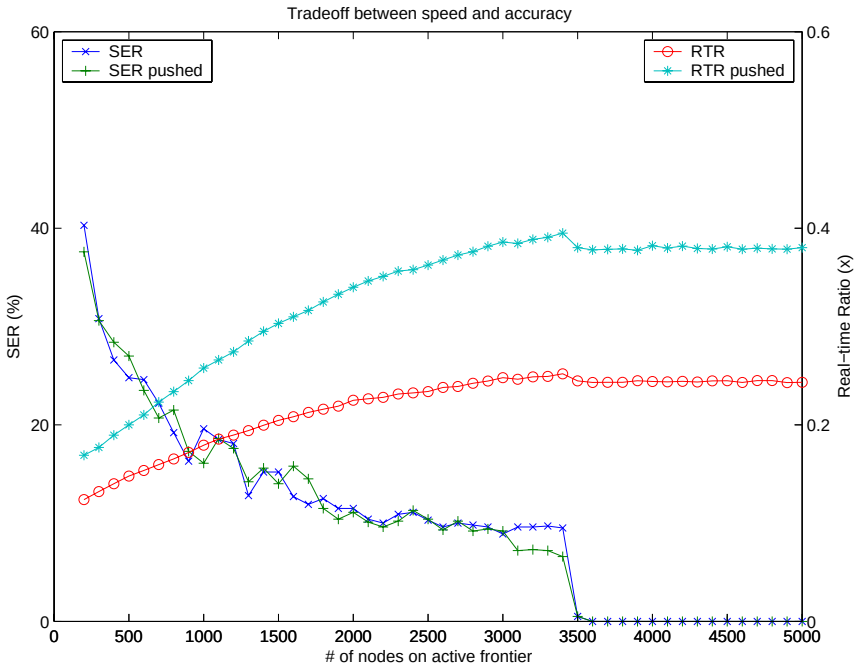


Figure 5-14: Overlay of tradeoff between speed and accuracy when number of nodes is varied for original and pushed weights configurations.

This figure overlays the plot from the right side of Figures 5-12 with Figure 5-10

5.10 Pipelined search

The two determinants of responsiveness are latency and bandwidth. To the extent that the search is real-time (i.e., RTR < 1) and can keep up, the synthesizer will be maximally responsive when the latency is zero. In practice, whatever can be effected to minimize the wait time until the first waveform segment descriptor will greatly enhance responsiveness. The guiding principle is to divide the search into multiple sub-searches which can be pipelined.

When an entire paragraph is to be synthesized, the first step is to divide it into separate sentences. Then, for each sentence, frequently used phrases can be identified for which the search can be computed once and cached for later use. This was the strategy used by the author in previous works [162, 163]. Common phrases called "shortcuts" were determined manually and were used to cut short the search

latency. While a "shortcut" is being played back, the search can proceed in parallel to determine sub-paths on the fly for the less commonly used words and phrases. Because the phonetic units at the start and end of a "shortcut" are known, transition symbols into and out of the "shortcut" can be used to gird or seed the sub-searches surrounding the "shortcut".

This process of determining commonly used phrases was manual, involved a fair amount of knowledge engineering, and represented significant development time. Preliminary studies presented show the path to a, perhaps, more disciplined approach to pipelining the search. Figure 5-15 tracks the growth of the Viterbi trellis by plotting the size of each slice along a logarithmic axis versus states in the input specification in topological order. Different growth curves are shown for score-based pruning beams of 1, 3, and ∞ (i.e., no pruning.) and are marked with 'x', 'o', and '+' symbols, respectively. The top curve marked with '+' symbols corresponds to no pruning and as the pruning beam narrows, the curve is pushed down and fewer nodes survive pruning in each slice. Ultimately, in the bottom curve marked with 'x' symbols, some slices are pruned down to just one node. While the general trend described here can be gathered from this figure, another closer look can be taken at a smaller time scale.

Figure 5-16 zooms in to the range of states from 200 to 400. The bottom curve touches the horizontal axis on more than one occasion revealing the presence of multiple search *islands of reliability* [164]. These islands are bordered by bottlenecks in the search space where the graph has been pinched down to one node. The best path must traverse these bottlenecks and so a partial backtrace can be initiated from the single node. The bottlenecks appear to occur regularly and each island can be considered as a sub-search. By tracing back whenever a bottleneck is reached, the search can be pipelined and the latency can be reduced down to the time when the search hops off the first island. To the extent that islands are hopped at a sufficient rate, the search can keep up with playback. Finding the optimal tradeoff between the score-based pruning threshold and latency is crucial for the deployment of real-time systems and is a topic for future studies. Parallel approaches to speech recognition

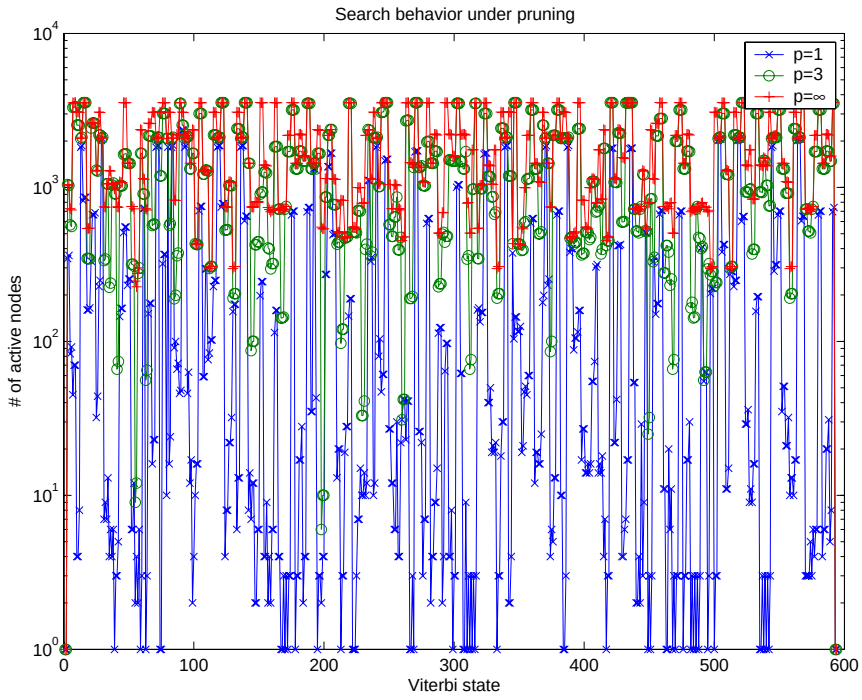


Figure 5-15: Growth of Viterbi trellis as pruning beam varies

The top curve marked with '+' symbols tracks the number of nodes per slice at each state in the Viterbi trellis. As the pruning beam narrows, the curve is pushed down until some slices are pruned down to a single node. A partial Viterbi backtrace may be initiated immediately at this point for pipelined search with reduced latency.

by hardware [42], Markov random fields [101], and finite-state machines [108] may also offer alternative ways of achieving real-time performance.

5.11 Rescoring

The foregoing has discussed the representation of the search space with a cascade of finite-state transducers and the Viterbi search which carries out unit selection with units defined as symbols and symbolically defined costs. Because of parameter reduction arguments made in Chapter 2, instance-to-instance costs were approximated by a significantly smaller set of cluster-to-cluster costs. The first Viterbi pass can be regarded as a coarse search which reasonably prunes the search space with symbolic guidance only.

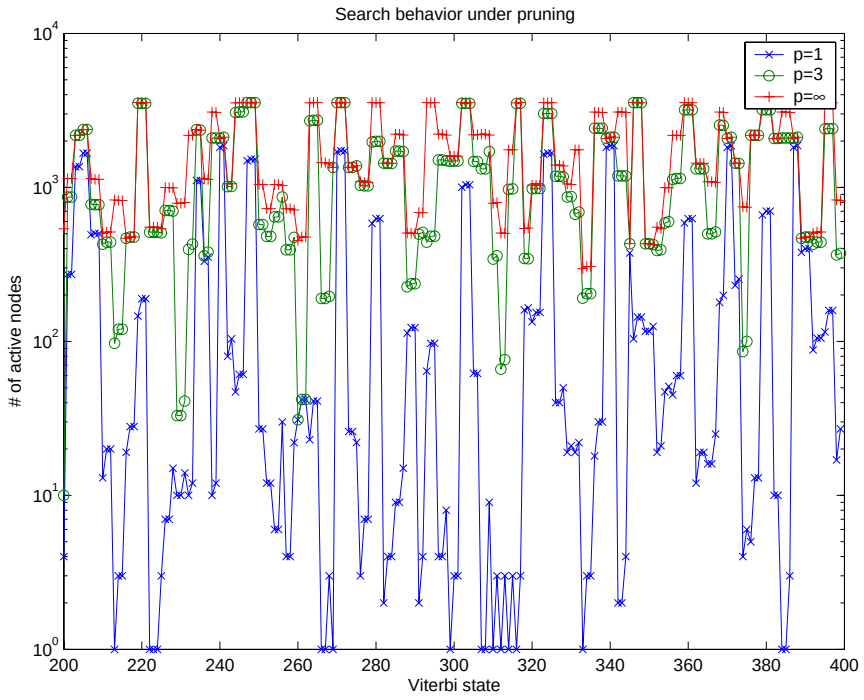


Figure 5-16: Growth of Viterbi trellis as pruning beam varies (zoomed view).

This is a zoomed-in portion of Figure 5-15 between states 200 and 400.

This section discusses a second pass which incorporates numeric information for a finer, more detailed search. The A^* algorithm, a best-first search with a priority queue, is used to generate a short list of top scoring paths. Then, scores on the paths in the short list can be replaced with numerical measurements made on the path. Finally, the A^* algorithm is used again to re-sort the short list and output the final top scoring path which incorporates both symbolic and numeric information. Pipelining which entails incremental Viterbi and A^* sweeps on sub-sections is not investigated here, but has been previously explored elsewhere for speech recognition [75].

Figure 5-17 describes the flow of a two-stage unit selection search. The Viterbi search in the first step is identical to previous discussions except that it is modified to output information beyond the best path. In fact, after the forward pass of the Viterbi search is complete and nodes are pruned at each slice, the entire trellis is wired up; that is,

instead of retaining only a single, back pointer to the incoming path with the best score, all possible incoming paths are kept with an associated score. This populated or connected Viterbi trellis is a graph which can be walked by an A^* search. Without pruning, the Viterbi scores can be used as the underestimate in the A^* search and the search is admissible. Otherwise, with pruning the Viterbi scores may not be accurate underestimates and the A^* search is not admissible. However, this, in practice, may not be important. Complete paths are repeatedly popped from the priority queue until a short list of N top hypotheses is accumulated.

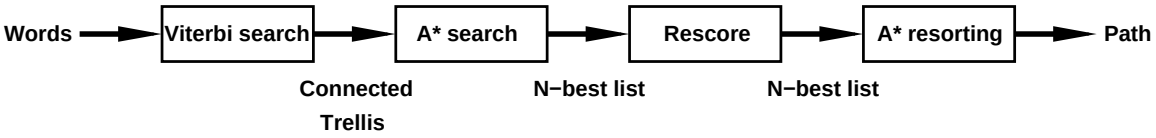


Figure 5-17: Process flow for generation, rescoring and resorting of N-best lists.

This figure depicts the process by which finer models can be used to refine search results in a subsequent pass. The interface is an N-best list which is extracted from a fully-connected Viterbi trellis by an A^* search. After the hypotheses are rescored, a second A^* search is performed to (cheaply) resort the list. At this point, the best-scoring path is kept as the result.

The N-best list produced by the first A^* search is ordered by scores determined purely by symbolic information. By adjusting the scores with numeric information *in situ*, it is possible to perform a second A^* search to re-order or re-sort the list. This is the point at which instance-to-instance costs can be integrated for finer modelling of, for example, acoustic and pitch continuity. One possible model for rescoring is displayed in Figure 5-18. Where two non-contiguous units, α and β, are proposed for concatenation, the model also takes into account the following context of α, α^+, and the preceding context of β, β^-.

Besides considering the continuity between α and β at the point of their join, the rescoring model considers whether α^+ bears any similarity to β and whether β^- bears any similarity to α. If the left portion of α^+ is fairly predictable from the right portion of α, then the left portion of β had better be similar of the left portion of α^+ to avoid the prospective perception (going forwards) that a concatenation has taken place. Likewise, if the left portion of β is fairly predictable from the right

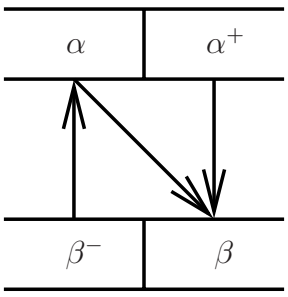


Figure 5-18: Model for rescoring of concatenations.

When a segment α is considered to be concatenated with segment β (as denoted by the diagonal arrow), the model for rescoring concatenations considers predictability of α^+ from α and of β from β^-. Also considered is the similarity of α^+ to β and of β^- to α.

portion of β^-, then the right portion of α had better be similar to the right portion of β^- to avoid the retrospective perception (looking backwards) that a concatenation has taken place. Such a rescoring model is fairly general and does not make any assumptions on how predictability and similarity are defined. However, this rescoring model only takes into account segmental continuity, or, more precisely, continuity at the boundary between segments and does not take into account longer-distance constraints such as supra-segmentals. Experiments with narrow-band spectrographic features are described in Chapter 6.

5.12 Extensions

5.12.1 Hierarchical unit selection

The examples of unit selection provided in this chapter have dealt mainly with phonetic units. Incorporating spatial information (e.g., syllable boundaries) was previously relegated to a pre-selection phase detailed in Chapter 2 wherein an optimal spatio-temporal segmentation of speech units was determined prior to unit selection search. In that case, a hard decision is made as to the hierarchical organization of input specification. While this conveniently transfers spatial analysis to a separate

component, it complicates unit selection, because substitutions and concatenations need to be modelled at all levels between all levels leading to many expansionary factors for the number of parameters. By deferring the decision of spatial organization to the search, hierarchical information can be incorporated in a scalable approach across as many desired spatial levels.

Spatial information is incorporated by splitting transition symbols into transitions across different boundaries such as phonetic boundaries, syllabic boundaries, and word boundaries. Whereas transitions symbols previously described only encoded the identity of the units to either side of the boundary, the level-dependent transition symbols depicted in Figure 5-19 also encode the nature of the boundary using different delimiters for different boundaries (e.g., ^ and # for syllable and word boundaries, respectively.) The finite-state phone chain for corpus utterances are modified too. At the boundary between the syllables of 'Boston', a transition symbol, s^tcl, denotes that a syllable boundary lies between the phonetic unit, s, and the phonetic unit, tcl. After the arc bearing the phonetic unit, s, which lies to the left of a syllable boundary, another arc departs the corpus utterance for the syllable-level constraint kernel, which brokers connections to phonetic units which lie to the right of a syllable boundary. Connections are made to and from the word-level constraint kernel at the end and the beginning of the word, 'Boston', respectively.

By encoding spatial information into the transition unit and not the speech units, multiple levels of hierarchical organization can be modelled in a scalable fashion. There are a total LP^2 transition symbols, where L represents the number of spatial levels. The only possible drawback is that context at all boundaries is effectively only one phonetic unit. While this may not matter much in practice, one workaround is to temporally merge units to capture more contextual information near boundaries of import. However, this will increase the number of substitution and concatenation costs as temporally merging introduces new units.

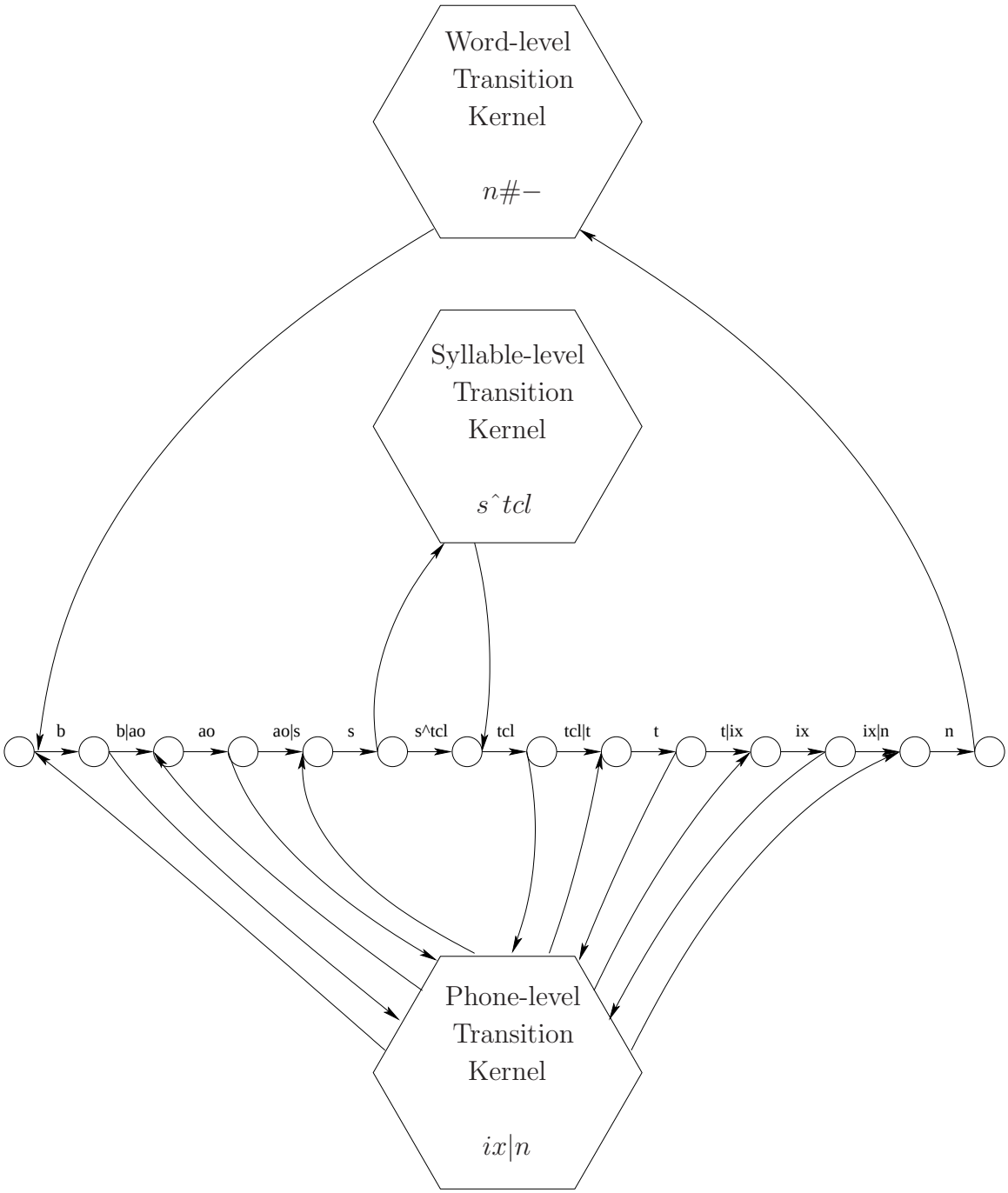


Figure 5-19: Hierarchical configuration for synthesis FST, S.

Hierarchical information can be encoded into the transitions by introducing extra transition symbols for level-dependent variants. If the transition symbols are perfectly matched, so will the hierarchy be matched without the necessity to introduce extra unit symbols. Concatenations at different levels are managed by separate kernels which generate level-dependent transition symbols. Examples of level-dependent transition symbols are illustrated within each level-dependent kernel here.

5.12.2 Segmental diacritics

Speech segments in the corpus can be optionally annotated with diacritics, which might describe, for example, quantized levels of pitch. By placing the diacritics on self loops, as shown in Figure 5-20, it is possible to include additional search costs while entering and leaving a segment. In this simple example, the segment α has attribute $L3$ at its beginning and attribute $R2$ at its end.

When the input does not specify attributes, the diacritic information is ignored and normal unit selection proceeds with only substitution and concatenation costs included in the total path cost. If the input specifies attributes, starting and ending costs can be incorporated at the segment level. Figure 5-21 depicts an input specifying $L1$ and $R3$ at the start and end of segment α. Because there is a mismatch with the corpus utterance in Figure 5-20, the backoff costs accumulate to a total of 3: the mismatch between $L1$ and $L3$ has a cost of 2 and the mismatch between $R3$ and $R2$ has a cost of 1.

The costs of mismatch in diacritics are similar to substitution costs and can be automatically learned from data in a similar fashion by computing inter-cluster distances. These costs are not stored in the corpus (only the true underlying diacritic is), but are inserted by a modified T into the input specification.

Diacritics representing additional sources of information can be chained in series. For example, duration or lexical stress could be another diacritic to be matched by the unit selection search along with pitch. An independence assumption underlies this additive chaining of costs. In work by Bulyko [21, 20], simplified ToBI [131] prosodic markers as part of unit selection.

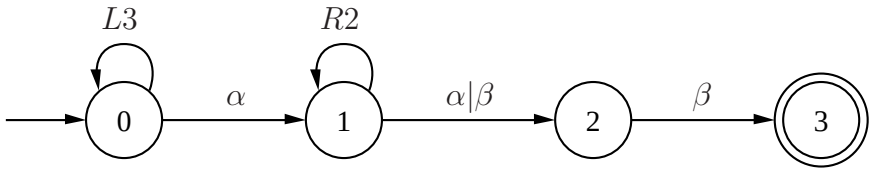


Figure 5-20: FST of corpus utterance in S incorporating diacritics.

In S, diacritics can be added around units to describe intonational or paralinguistic attributes. In this example, the unit, α, belongs to classes of 3 and 1 on the left and right, respectively. It is backwards compatible with input specifications without diacritics, because the diacritics are added on self loops; these self loops can have weights describing the fit of the model (i.e., how well $L3$ describes the attributes for the left portion of α.)

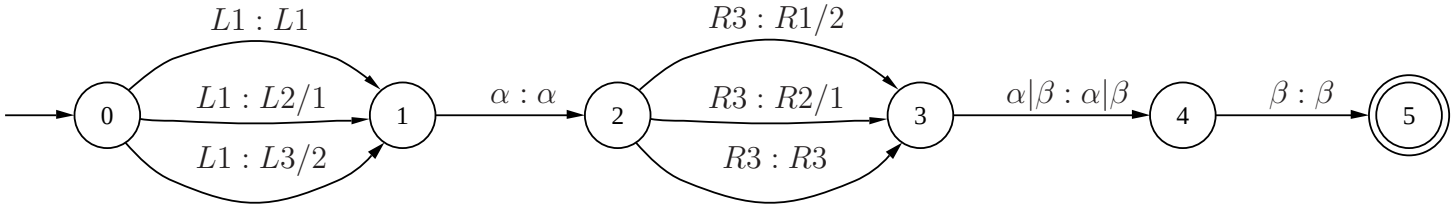


Figure 5-21: FST of input specification incorporating diacritics.

This input specifies a unit, α, with left and right attributes of $L1$ and $R3$, respectively. However, it can back off to other attributes with associated costs (e.g., inter-cluster distances.) When composed with Figure 5-20, the mismatch between $L1$ and $L3$ has a cost of 2 and the mismatch between $R3$ and $R2$ has a cost of 1.

5.12.3 Dynamic determination of location of concatenation

The discussion thus far has assumed that the boundary of speech units are determined and reliably so. However, if the location of the boundary is imprecise, an additional stage of boundary setting can be conducted in parallel with the rescoring stage. As described in Chapter 3 the boundary between a vowel and a semivowel is less simple to define and is highly variable to conditions and context. When joining a vowel with a semivowel, the best point of hand-off can be determined by examining the spectral information at the boundary and shifting the boundaries of the vowel and semivowel until the spectral information aligns [139]. This notion of a soft, malleable boundary defers the precise determination of the (phonetic) boundary to run-time when concatenating with another unit.

5.13 Summary

This chapter has outlined the implementation of a multi-pass unit selection search using finite-state transducers. A first pass is made with a dynamic programming search which produces a fully connected Viterbi trellis with scores. Without pruning the trellis scores are admissible for an A^* search that generates a shortlist of N hypotheses. With pruning accuracy is traded off for speed and real-time pipelining is possible. The N-best list can be rescored and re-sorted with a subsequent A^* pass. Many extensions are possible including hierarchical unit selection, segmental diacritics, and dynamic determination of the location of concatenation.

The search is factored as a cascade of transducers, the last of which, S, is a highly compact structure called the constraint kernel that encodes all concatenation and substitution costs for a particular language. Only a linear number of connections is made between the utterances in the corpus - as represented in the form of finite-state phone chains - and the constraint kernel; the size of S grows linearly with the size of

the corpus. Once the size of the corpus surpasses the order of magnitude of a minute or more of speech, the constraint kernel begins to pay for itself. The constraint kernel has a number of properties which also make the expansion of search paths efficient.

In the unit selection literature, many works describe making hard decisions in a pre-selection phase to limit the size of the search space. In the AT&T's NextGen synthesizer [7], over 99% of the transitions are determined to be rarely used after profiling over a training set and are simply omitted to make their unit selection algorithm tractable [9]. In other works that make explicit use of decision trees [13, 148, 38], only the exemplars at the leaf nodes are licensed to populate the trellis in the Viterbi search. Although a decision tree could be incorporated into L - decision trees can be represented by FST's [134] - the benefit of using the proposed constraint kernel is that soft decisions are employed and all possibilities are considered at search time. In fact, the states corresponding to equivalence classes in the constraint kernel compose a one-level decision tree.

Recent work on lexical and pronunciation modelling in an FST-based speech recognizer has investigated reductions and contractions [55] and learning pronunciation weights with an Expectation-Maximization algorithm [130]. In theory, these could be incorporated into L and P as mentioned before to control speaking style (i.e., casual versus formal). Out-of-vocabulary words could be spliced into dynamic FST's on the fly [120] with pronunciations generated by stochastic letter-to-sound rules [30].

Chapter 6

Design and evaluation

The frameworks and notions presented thus far have addressed certain technical challenges presented by the desire for efficient unit selection concatenative speech synthesis. At each step, computational arguments have been made for the design choices at each stage and claims have been mathematically backed by objective reasoning. Lacking at this point is an assessment of system performance and subjective evaluation of the speech synthesis. This chapter presents a set of processes that organize a set of speech synthesis components for use in an end-to-end conversational system. With real data taken from human users conversing with the system to solve tasks in a given scenario, the speech synthesis framework, ENVOICE, can be empirically evaluated.

An underlying theme in this chapter is the goal of relating objective and subjective measures. Objective measures can be made by the machine from, for example, the model and parameter settings of synthesizer. On the other hand, subjective measures can only be made by humans. Discovering any relationship between the two would greatly advance our knowledge of how to tune a system for maximum perceived benefit. Figure 6-1 summarizes the difficulty of doing so with a simple example of where the mapping does not preserve local distances. Points close in one space do

not correspond to points that lie close in the other space.

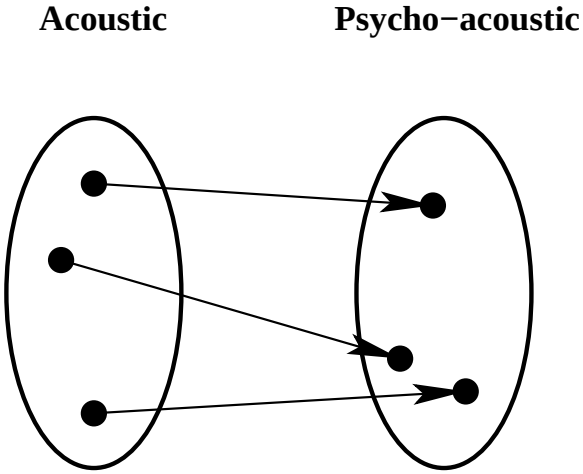


Figure 6-1: Mapping of objective distances to subjective distances.

A challenge in building speech synthesizers is understanding and discovering the mapping between objective measures and subjective opinion. If such a relationship exists, then objective measures can be automatically optimized to maximize subjective opinion.

6.1 Related work

Speech synthesis can be evaluated along many subjective dimensions including intelligibility (is the speech understandable?), naturalness (does the speech sound like a human?), and quality (does the voice sound smooth?) [71]. Flexibility can be an important determining factor for practical applications (what is the input vocabulary size? does it read free-form text?). As described by Pols in [32] speech synthesis can be diagnosed on both modular and global levels. Recent evaluations of synthesizers have been performed both independently of an application [160, 111, 28, 70, 106] and as part of a spoken dialogue system [156, 157]. Chu et al was able to correlate Mean Opinion Scores (MOS) taken from human subjects with objective concatenation cost functions [28, 106]. When the cost function is used as the metric in unit selection, the MOS of the resulting synthesis can be predictable. Syrdal et al examined acoustic correlates of voice quality and was able to predict which human voices are more suitable for concatenative speech synthesis [142, 141]. Psychoacoustic experiments at a

smaller time scale have investigated whether discontinuities at concatenation boundaries are perceptible and what measures can be used to predict signal discontinuities [39, 69, 160, 137].

6.2 Processes

Crucial to designing and building systems is an understanding of the underlying processes. This section is devoted to delineating the steps taken to rapidly prototype a system from scratch, including how to train a synthesizer for a new language and how to determine the set of prompts to record for a new domain. Once a prototype is designed, it can be deployed and tuned. The tuning can be automatic if a desired objective function to optimize (e.g., minimizing word error rate) is provided. A common theme of the processes presented here is that minimal interaction is required once properly configured. The processes are semi-automatic, because once the proper set of information is provided by a speech expert, the process is entirely automatic.

Figure 6-2 illustrates the steps involved in preparing synthesis costs for a new language. This process is entirely automatic given a corpus and clustering questions. Starting with a representative corpus of speech utterances, clustering questions are used to group exemplars into categories. These clustering questions are provided by a speech expert and example questions for the acoustic clustering procedure are given in Appendix B as described in Chapter 4. The categorization step produces a set of substitution and concatenation classes which group the exemplars into clusters of examples. The characterization step involves training models for each cluster distilling the characteristics of the cluster into a statistical model. These models are compared by information-theoretic metrics to calculate synthesis costs. Inter-model distances between substitution models are used to form substitution costs and concatenation costs are formed by distances between joint and marginal models. These substitution and concatenation costs are efficiently encoded into a constraints finite-state

transducer as described in Chapter 5. All the speech knowledge contained within the representative corpus as clustered by expert guidance is captured within this language-dependent constraints FST that can be re-used across different domains.

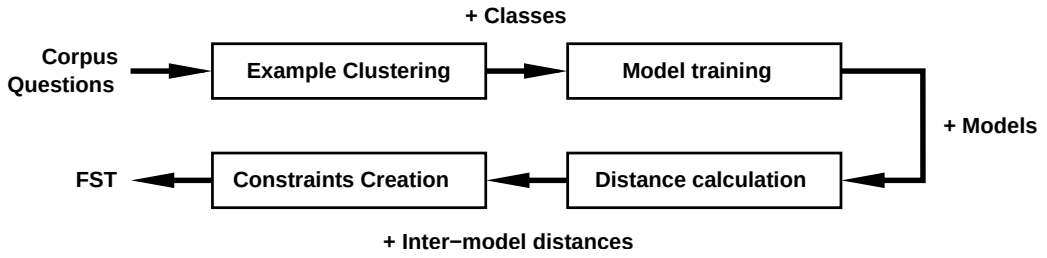


Figure 6-2: Process for adapting synthesizer to new language.

Porting the ENVOICE framework to a new language requires the analysis of a representative corpus of speech utterances and the distillation of synthesis costs into a constraints FST. A speech expert provides speech knowledge in the form of phonological rules and questions which are used to categorize exemplars into equivalence classes for subsequent model training. Inter-model distances are then used to define connection costs in the constraints FST.

Figure 6-3 illustrates the process for building a new domain. Once again, only the initial step requires human attention and the remaining steps are fully automated. The first step is prompt generation and produces a set of prompts to record given the response requirements of the conversational system. The system designer will assemble a list of typical system responses. Prompt generation takes this list and produces another list, possibly disjoint, of utterances to record. After recording these utterances in quiet conditions, the corpus of speech is automatically transcribed by, typically, an automatic speech recognizer configured to locate, for example, phonetic boundaries given only the word sequence. These phonetic transcriptions are instrumented for search by connecting them in a topologically linear fashion to the constraint kernel as described in Chapter 5 producing the full synthesis FST, S. The waveforms are placed into a binary large object (BLOB) which is memory mapped at run-time for efficient access. Because systems are typically deployed over the telephone, the size of the BLOB is just under 30 megabytes per hour of speech when using an 8-bit u-law sample encoding and a 16-kHz sampling rate.

Prompt generation is defined as the process of selecting a set of prompts to provide unit coverage over the expected set of system responses [40]. The pathological case

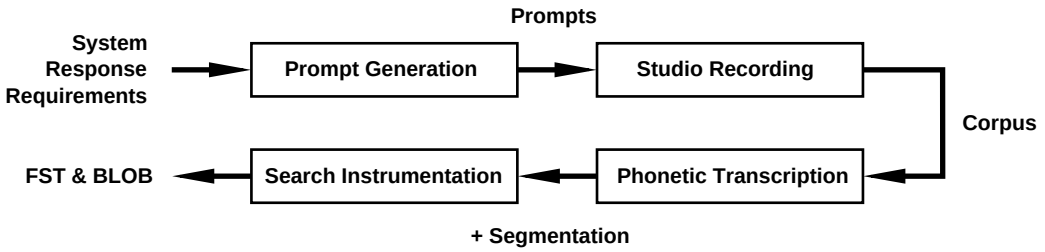


Figure 6-3: Process for adapting synthesizer to new domain.

Porting the ENVOICE to a new domain requires the analysis of a representative corpus of typical system responses and selective recording of these responses as uttered by a human speaker. A prompt generation phase selects a compact, domain-specific corpus to cover the response requirements. Once the recordings are transcribed, the waveforms and their transcription are instrumented into final forms for use by ENVOICE.

is straight recording, where everything that is needed is recorded once. It is, of course, desirable to record less and some measure is required to ascertain to what extent the prompts can be compacted. The measure can be made in two ways: static and dynamic analysis. In static analysis, the units in the expected system responses are tabulated and prompts are iteratively selected to maximize coverage of the required units. The description of the units can be, for example, phonological and/or intonational. Previous work by the author [162] examined static analysis. In dynamic analysis, prompts are iteratively selected in a closed loop to minimize unit selection cost over the expected set of system responses. It is dynamic, because it actually considers the concatenation of units and their associated costs over the expected set. The iterations occur in a closed loop by a greedy selection of the prompt at each step expected to best minimize total cost. Figure 6-4 illustrates this process.

Regardless of whether the analysis is static or dynamic in prompt generation, to counter the possibility of over-fitting, it is possible to divide the input set into training and development sets and to achieve coverage on the training set while monitoring coverage of the development set. Over-fitting to the training set occurs when coverage continues to improve on the training set with no benefit on the development set. Cross-validation is a way to insure generalization of the prompts to unseen data. Corpus selection for creating voices from scratch [40] or domain adaptation for incrementally adding domain-specific utterances [27] remain to be active areas of research.

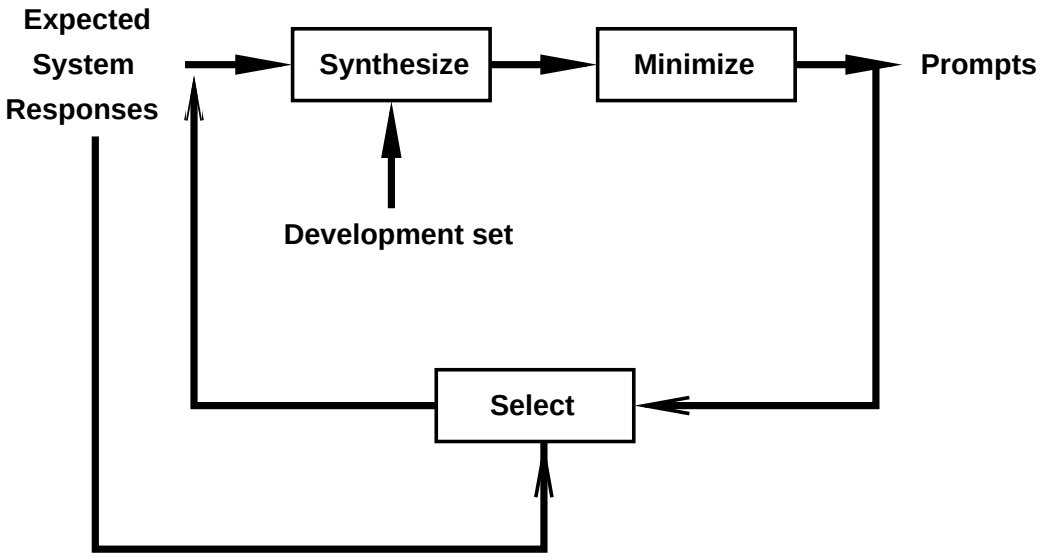


Figure 6-4: Process for selecting generating prompts.

Prompt generation entails iterative selection of prompts guided by dynamic analysis of how the units in a selected prompt benefit the overall unit selection cost over a expected set of system responses. It is a closed loop which at each step minimizes the total cost by greedy selection of the next best prompt. To avoid data mining biases, cross-validation can be accomplished by leaving out utterances into a development set over which generalization to unseen data can be measured.

After the system is initially prototyped, it can be immediately deployed as shown in Figure 6-5. This figure illustrates the process flow in our GALAXY conversational system framework, in which meaning is converted into speech. After a language generation component, GENESIS [5], generates well-formed sentences from the semantics or meaning, the word sequences are used in a unit selection search as described in Chapter 2. The optimal sequence of speech segments that result from the search are reconstituted into fluent speech by waveform concatenation, for which a frame-based method was described in Chapter 3.

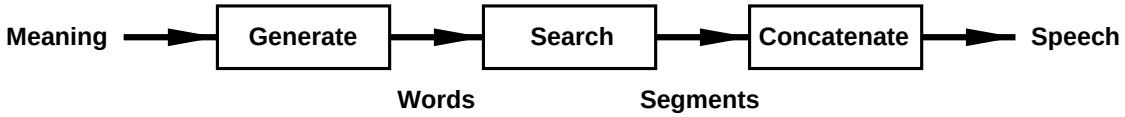


Figure 6-5: Process for meaning-to-speech conversion.

In our spoken dialogue systems, speech is synthesized directly from a meaning representation. GENESIS generates a response, for which the words are fetched from a corpus by an efficient FST-based unit selection search. The resulting speech segments are concatenated by a waveform processor.

Because the operation of the conversational components is scripted in the GALAXY framework, it is rather straightforward to define the (low-bandwidth) communications flow between the language generation, unit selection, and waveform generation components. A short-circuit loop can also be created around the unit selection search to directly play a waveform to the user. This functionality is used, for example, in the ORION system [125] where users can record reminder messages for later playback. It is also possible to transmit word boundaries to the audio server for multi-modal applications. Multi-modal support was added to GALAXY by Doreteo Torre Toledano. This functionality is found in the Spoken Language Learning System [74] for word-level playback of what language learners speak in foreign language lessons.

In displayless systems deployed over the telephone, the search and concatenation components are located in the back-end abstracted away from the user. Because the search is compute intensive, the search engine may reside on a computer where compute and power concerns are non-existent. The speech segment descriptors (i.e., waveform tag and start/end times) are handed to the waveform generator which assembles the synthetic waveform for transmission over the Public Switched Telephone Network (PSTN). The computation needs for the waveform generator are far less, while bandwidth - between where the BLOB is stored and where the interface to the PSTN begins - is more of an issue.

If the conversational system is deployed over mobile, handheld devices, it is beneficial to locate the waveform generator on the device end. If the concatenation corpus, the BLOB, can be determined *a priori*, it can reside on the handheld device and only the speech segment descriptors need to be received for synthesis to occur. This considerably lessens the bandwidth requirements for wireless communications to the mobile device. With the search engine located in the "cloud", this setup may be referred to as distributed speech synthesis (DSS). This is a direct analogue of distributed speech recognition (DSR), in which front-end processing lessens the bandwidth requirement for transmitting speech waveforms or parameters to the recognizer located in the "cloud".

The task of evaluating speech synthesizers is one of a rather subjective nature. In the final analysis, humans are the consumers of speech synthesis and as end users their opinions are important. However, for developmental purposes it is convenient to have an automatic process by which the effects of small incremental changes in synthesizer configurations on quality, intelligibility, and naturalness can be measured. Similar in spirit to prompt generation, Figure 6-6 illustrates a process for tuning synthesizer models. The premise is that a change is for the better if it improves the understanding of synthetically generated speech. This ultimately needs to be verified by correlating the results to human opinion. A speech recognizer is used to measure the word error rate on synthetic speech. A model built from training data can be tuned on development data to minimize word error rate as automatically measured by a speech recognizer. A later section will report some preliminary results from using this evaluation framework.

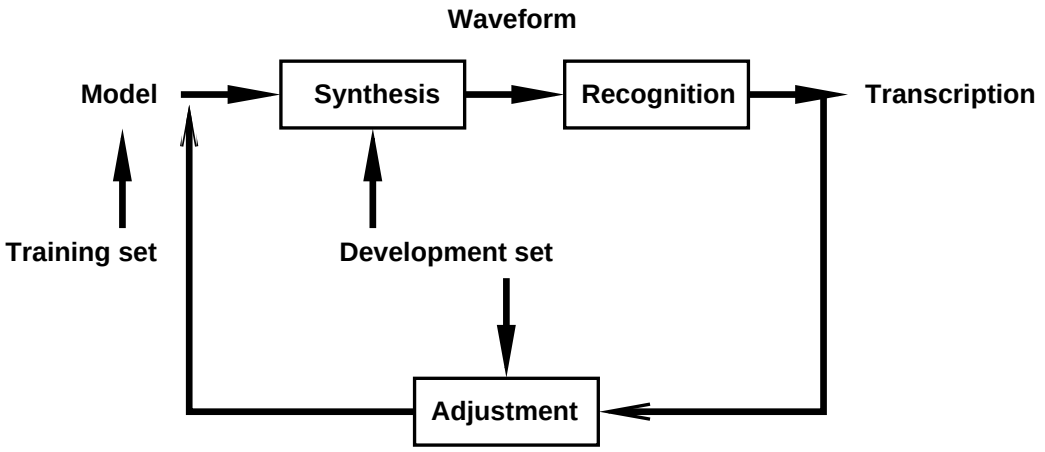


Figure 6-6: Process for tuning synthesis parameters.

In automatic evaluation by machine, initial synthesis models built from a training set are tuned and adjusted with feedback obtained from utterances synthesized from a development set. Synthesized waveforms are fed into a speech recognizer which delivers a sentence hypothesis. This is compared with the input to the synthesizer and forms the basis for model adjustment.

6.3 Empirical evaluation

An initial step towards evaluating a synthesizer is establishing a baseline with which modifications can be compared. This section describes a baseline system and qualitative assessments by a fairly large user base. ENVOICE was evaluated within a conversational system deployed over the telephone for making air travel plans [126]. Flights could be browsed by asking for airline, city, and date. After narrowing down the flight choices by a series of negotiations between the user and system, flexible itineraries could be planned including one-way, round-trip, and other more complex (open-jaw) specifications. Evaluations of the DARPA Communicator took place over a 9-day period in 2000 [156] and over a 6-month period in 2001 [157]. A total of nine sites participated in the first evaluation: American Telephone and Telegraph (AT&T), Bolt Beranek and Newman Inc. (BBN), University of Colorado, Carnegie Mellon University (CMU), International Business Machines (IBM), Lucent Bell Laboratories, Massachusetts Institute of Technology (MIT), MITRE, and Stanford Research Institute International (SRI); MITRE did not participate in the 2001 evaluation.

In the 2001 DARPA Communicator Evaluation, naïve users were recruited from across the country and asked to call a flight travel system deployed by one of the above eight different participating sites. Approximately 20 subjects called each system four times each, and another 15-16 subjects called each system eight or more times. After each call, users were asked to fill out a Likert scale questionnaire [77] in which they were asked to rate, among other things, the degree to which they agreed with the statement, "I found the system easy to understand in this conversation." Each of the eight participating sites used some form of concatenative synthesis in their deployed systems. Among the eight, the MIT system, which used the ENVOICE synthesizer, was ranked the highest (4.07/5) in user agreement with that statement. It is unknown whether what factors played a role in this ranking. To be fair, it could be the case that language generation or corpus design is better for the MIT system.

The qualitative assessment of how easy the system is to understand encompasses many different factors, such as dialogue management (i.e., does the system response make sense in the context of the sequence of interactions?), language generation (i.e., are the sentences individually well-formed?), and speech synthesis (i.e., does the synthetic speech sound natural and is it intelligible?). Because this assessment is wholistic, rather than reductionistic, it may not follow that the speech synthesis is natural-sounding. Furthermore, this particular configuration of ENVOICE concatenated words and phrases and used unit selection search (with manual weights) more sparingly. As such, the performance of the unit selection search may also be difficult to separate out. Nonetheless, the 2001 DARPA Commmunicator Evaluation provides a reference point for naturalness for the baseline system used in these experiments.

As an example of how the system may ask for confirmation of a negotiated flight itinerary, Figure 6-7 shows a synthetic utterance with concatenation boundaries denoted by apostrophes. Selected units of varying lengths are seen to occur ranging from sub-word syllabic units to long phrases. The last sentence is extracted from the corpus in its entirety. Flight numbers and times are composed of smaller pieces.

US Airways' flight' 11'45' leaves at' 4:'54' pm' from' Charleston' and arri'ves' in' Charlotte' at' 5:'45' am.' After' that,' you depart' on' US Airways' flight' 7'9'7' at' 7:'0'5' pm' and arrive in' Boston' at' 9:0'7' am.' The total flight time including stops and connections is' 4' hours' and' 15' minutes.' Shall I add this flight to your itinerary?

Figure 6-7: Sample system response for air travel domain.

6.4 Automatically determined costs

This section describes a listening evaluation for testing the efficacy of using automatically determined costs in the synthesis FST. A small test set of twenty-two utterances shown in Table 6.1 in an air travel domain is re-synthesized on a leave-one-out basis from a training set consisting of around 100 minutes of in-house recordings of typical system responses in weather information, flight status, and air travel domains.

Twelve of the utterances are system responses describing flights from a source to a destination cities, while the ten remaining utterances describe returning flights. The sentence structure is very rigid and the changing content includes times and dates, airline and city names, and flight numbers.

```
01) On Monday, January 1st at 1:23am you fly on Northwest flight 401 from Boston, Massachusetts to Atlanta arriving at 12:59pm.
02) On Wednesday, March 3rd at 3:21am you fly on Taiwan Airlines flight 408 from Ontario, California to Albuquerque arriving at 10:37pm.
03) On Thursday, April 4th at 4:20am you fly on Tower Air flight 201 from Ottawa, Canada to Washington arriving at 9:26pm.
04) On Friday, May 5th at 5:19am you fly on Vanguard Airlines flight 208 from London, England to Anchorage arriving at 8:19pm.
05) On Saturday, June 6th at 6:18am you fly on Virgin Atlantic flight 101 from Frankfurt, Germany, to Aspen arriving at 7:18pm.
06) On Sunday, July 7th at 7:17am you fly on Sun World International flight 108 from Yokohama, Japan to Atlantic City arriving at 6:17pm.
07) On Monday, August 8th at 8:16am you fly on Sun Country Air flight 309 from Augusta, Georgia to Austin arriving at 5:16pm.
08) On Tuesday, September 9th at 9:15am you fly on Alitalia flight 409 from Augusta, Maine to Ottawa arriving at 4:15pm.
09) On Wednesday, October 10th at 10:14am you fly on US Airways flight 109 from Charleston, West Virginia to Amsterdam arriving at 3:14pm.
10) On Thursday, November 11th at 11:13am you fly on TWA flight 209 from Bali to Athens arriving at 2:13pm.
11) On Friday, December 12th at 12:12pm you fly on Reno Air flight 501 from Dubai to Auckland arriving at 1:12pm.
12) On Saturday, January 13th at 11:11pm you fly on Sabena flight 508 from Hollywood to Osaka arriving at 12:11pm.
13) On Tuesday, April 16th at 4:08pm you return on Nordic East Airways flight 808 arriving in Kalispell at 9:08pm.
14) On Wednesday, May 17th at 5:07pm you return on Midwest Express flight 809 arriving in Midland at 8:07pm.
15) On Thursday, June 18th at 6:06pm you return on Midway Airlines flight 888 arriving in Ontario at 7:05pm.
16) On Friday, July 19th at 7:05pm you return on America West flight 281 arriving in Orange County at 6:01pm.
17) On Saturday, August 20th at 8:04pm you return on KLM Airlines flight 382 arriving in Paris Orly Airport at 5:58pm.
18) On Sunday, September 21st at 9:03pm you return on Air New Zealand flight 483 arriving in St. Martin at 4:48pm
19) On Monday, October 28th at 10:02pm you return on British Airways flight 584 arriving in Singapore at 3:38pm.
20) On Tuesday, November 29th at 11:01pm you return on Alaska Airlines flight 785 arriving in Tokyo Narita airport at 2:28pm.
21) On Wednesday, December 30th at noon you return on Air Lingus flight 886 arriving in Washington National at midnight.
22) On Thursday, January 31st at midnight you return on Delta flight 987 arriving in Shenzhen at noon.
```

Table 6.1: Twenty-two utterances form the test set for synthesis evaluations.

These utterances are sample responses taken from an air travel conversational system. The first twelve describe flights from city X to city Y, while the second ten describe flights returning from city X. The sentence structures are rigid and the changing content only includes times and dates, airline and city names, and flight numbers.

In order to evaluate the information-theoretic measures, it is necessary to isolate all other factors and to keep them constant. For example, the input specification must be a single path, or otherwise different costs could encourage the search down other paths. For this reason, the phonetic sequence of the original utterance, and not the word sequence, is used for the input specification in experiments testing automatically determined costs. In other words, the specific phonetic realization of the phonemic sequence implied by the word sequence is constrained and the phonological information for the synthesizer is provided by an oracle.

After unit selection is complete, the segments are concatenated by a waveform processor. Concatenation boundaries are smoothed with windowed overlap-and-add processing using constant-frame rate analysis and variable-frame rate synthesis. The shift corresponding to maximum correlation between abutting short-time spectral frames is used to appropriately offset the windows at the concatenation boundary.

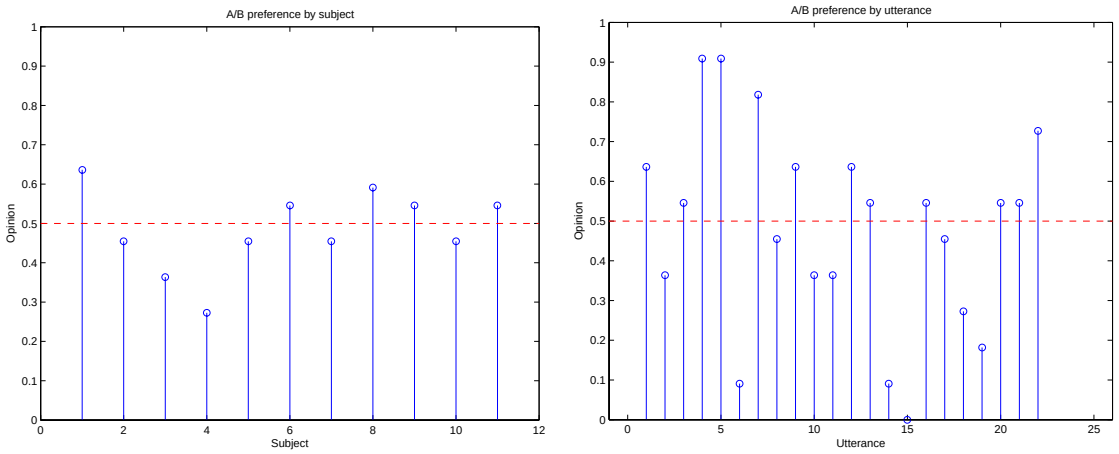
Since window shifts at boundaries introduce variable timing of the window centers, normalization is required to preserve energy relations.

In comparing synthesizers using manually tuned and automatically derived synthesis costs, eleven subjects participated in a listening evaluation and performed A/B comparisons [79] on an utterance-by-utterance basis. The utterances were presented in random order and, for each utterance, the presentation of A and B systems was also randomized. As seen from Figure 6-8(a), only five of the eleven subjects found the second system preferable as noted by an opinion score greater than one-half that was obtained by averaging over all twenty-two utterances. Pooling votes the other way, Figure 6-8(b) shows that the second system was found to be preferable (opinion $>$ 0.5) for twelve of the twenty-two utterances, or 55% of the time. The opinion score was obtained by forming a voting committee from the eleven subjects and averaging numerical values, 0 for A, and 1 for B, representing their votes.

A 50% preference level suggests that subjects are indifferent and that one system is indistinguishable from the other. Since the baseline system is considered quite natural, this result can be viewed positively as it reduces the amount of manual heuristics necessary to tune a synthesizer. However, what is somewhat worrisome is how averaging and thresholding can mask extreme values. Utterances #6 and #14 received only one vote, while utterance #15 received none. It may be worthwhile to investigate *min-max* criteria, in which the magnitude of the worst case is minimized. This can be likened to the saying that rising waters raise the level of all the boats, except just the opposite. Utterances #4 and #5 received high votes - for #4, the airline and city names sound more natural, whereas for #5 it was the weekday.

For the three utterances mentioned above where the second system received one vote or less, spurious pitch and irregular durations were detracting factors. On the other end of the spectrum, three utterances received nine votes or more. The automatically determined costs reduces the concatenation rate (i.e., number of concatenations per second of speech) from 2.14 to 1.81, a 15.3% relative reduction. It is not clear whether

this observed effect can be causally related to the automatic learning procedure, although in general, concatenation rate can be directly controlled by the weighting between concatenation and substitution costs. However, it suggests that reducing the concatenation is preferable, a hypothesis to be examined in greater detail next.



(a) Evaluation results by subject.

(b) Evaluation results by utterance.

Figure 6-8: Results of listening evaluation of automatically determined costs.

This evaluation compares system, A, using manually determined costs with system, B, using automatically determined costs. Eleven subjects were asked to listen to twenty-two utterances in random order and the system order was also presented in random order for each utterance. Although only five of the eleven subjects found the automatically determined costs preferable, using the subjects as a voting committee found the new system to be preferable for twelve out of the twenty-two utterances.

Figure 6-9 shows a panel of plots of opinion of automatically determined costs against several measures. Subjective opinion is shown on the abscissa against various objective measures on the ordinate. The measures are extracted from the synthetic waveforms of system B. Correlation coefficients are shown as well as associated p-values for statistical significance testing using $\alpha = 0.05$. Regression lines are shown for significant correlations. Concatenation rate of waveforms synthesized with automatically determined costs did not appear to be a significant predictor of opinion. However, the concatenation rate spread, obtained by subtracting the concatenation rate of baseline waveforms synthesized with manually determined costs, did appear to be a significant predictor of opinion. The result is somewhat counter-intuitive,

because the correlation coefficient, $\rho = 0.46$, is positive, suggesting that a positive increase in concatenation rate is related to a higher opinion score. Not all concatenations have the same cost, and so, another plot relates opinion to the cost of the concatenation divided by the length of the concatenated segment. This rate in time of concatenation costs appears to be positively correlated with opinion, another counter-intuitive result. On the other hand, the total cost of concatenations over the entire utterance does not see to be correlated with opinion.

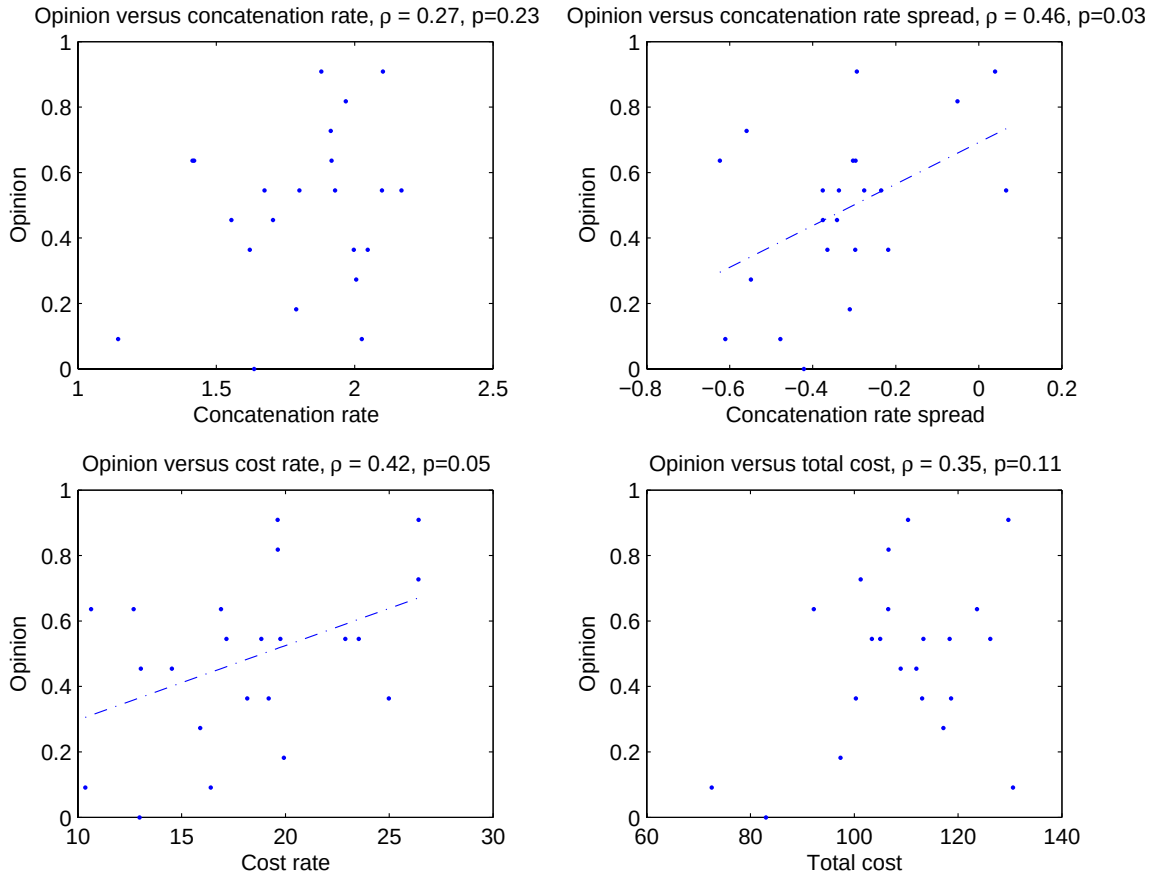


Figure 6-9: Detailed analysis of evaluation of automatically determined costs.

This panel of plots examines correlation between subjective opinion of the voting committee and various objective measures extracted from the system using automatically determined costs and whether the measures are significant predictors. An increase in concatenation rate over the baseline system and a high absolute rate of concatenation cost per unit time were found to be, counter-intuitively, significantly correlated to high opinion scores. Since intonation was not explicitly modelled, perhaps the high concatenation rate precludes settling locally on intonation from any particular segment of speech.

The obvious question from looking at the plots in Figure 6-9 is why an increase in

concatenation rate and a high rate of concatenation cost per unit time are significantly correlated with high opinion scores. Because intonation is not modelled here, transitioning between waveforms at a higher rate, perhaps, avoids settling for too long on the local intonation of any particular segment. This would explain why an increase of concatenation rate might be preferable as well as a high concatenation rate in absolute terms.

An explicit intonation model would help to address this issue. Taylor modelled rise and fall of local F_0 contours in [146, 147]. The Boston University radio news corpus [105] is often used in intonation studies in concatenative synthesis [12], because it has hand-labelled ToBI [131] markers, a system based on the Pierrehumbert annotation system [109]. Another popular model is the Fujisaki convolution model which generates F_0 contours with phrase and accent impulse response functions [49]. Linear and non-linear regression models have also been used to generate F_0 contours as well [12, 140].

6.5 Intonation modelling

A simple intonation model was assembled in an unsupervised fashion by bottom-up clustering of feature vectors. The features extracted including prosodic factors such as pitch, duration, and energy. For all sonorant segments (i.e., vowel, semivowel, and nasal), F_0 was extracted as average over the entire segment and also over the first and last third (30%) of the segment. The energy and duration of the segment was also included on a logarithmic scale. These five dimensions constituted the feature vector.

After extracting feature vectors from the 100-minute training set mentioned above, a principal component analysis was performed to whiten the data into decorrelated, unit-variance dimensions. Because no intonation symbols were pre-ordained, the optimal number of intonation symbols was learned using a mixture of Gaussian statistical model. Training a mixture involves an initial step of K-means clustering [52] and subsequent refinement via iterations of Expectation-Maximization (EM) [36]. The number of clusters, 10, was chosen using the Bayesian Information Criterion (BIC) [122] measured over a mixture model of full-covariance Gaussian components: additional mixture components were added until the BIC no longer increased. When the BIC levels off, it indicates overfitting, or that the number of parameters suggests a model more complex than can be supported by the number of data points.

Figure 6.2 shows the class size and model means for each of the 10 intonation clusters. The five dimensions of the mean correspond to average pitch, start pitch, end pitch, duration, and energy. The dimensionality for pitch is in Hertz, and for duration, milliseconds. The clusters have all types of pitch contours: rising, falling, and flat. Clusters with rising pitch include 1 and 7, while clusters with falling pitch include 0 and 3. However, the remaining clusters with "flat" pitch still retain a downward bias. The pitch contours are visually displayed in Figure 6-10(a) where the downward trends become apparent. Also, the flatter pitch contours appear to spaced in quantized steps

attributable, perhaps, to the clustering procedure. Note that clusters, 8, 4, and 2 have short durations, whereas clusters, 6, 0, and 3 are longer in duration. Clusters 8 and 9 have low energy.

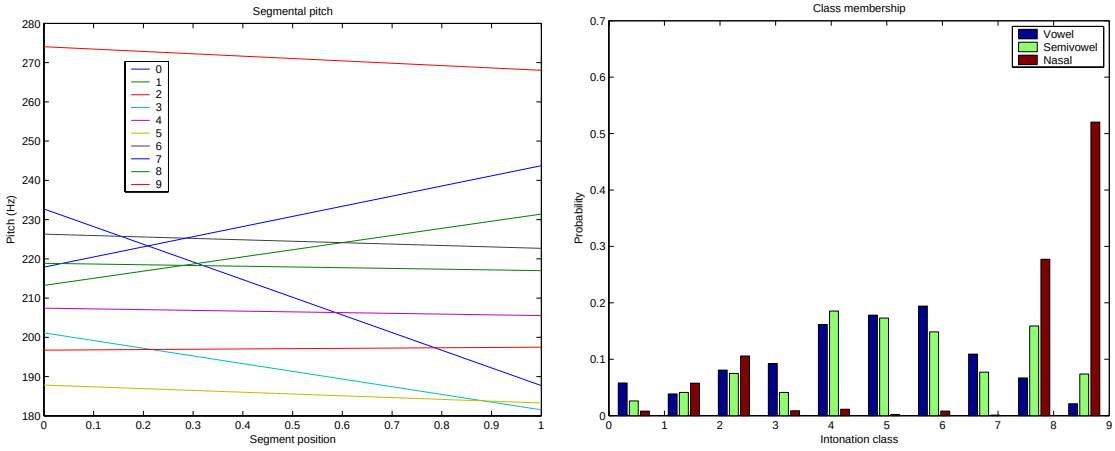
	0	1	2	3	4	5	6	7	8	9
#	983	1,045	2,433	1,923	4,052	4,538	5,109	2,120	3,666	3,553
P	212.48	225.08	271.20	187.40	206.52	185.21	224.49	229.31	217.89	197.10
P_s	232.70	213.23	274.05	201.15	207.43	187.84	226.30	217.90	218.87	196.73
P_e	187.70	231.40	268.05	181.54	205.54	183.28	222.69	243.73	217.00	197.51
D	158.57	100.40	57.24	146.15	36.15	100.56	76.58	139.47	32.24	67.08
E	86.78	81.42	83.03	84.44	85.44	85.14	86.67	87.81	77.65	72.70
ΔP	-45.00	+18.17	-6.00	-19.61	-1.89	-4.56	-3.61	+25.83	-1.87	+0.78

Table 6.2: Makeup of automatically acquired intonation classes.

These ten intonation classes were automatically acquired by iteratively fitting a mixture model of increasing complexity until stopped by the Bayesian Information Criterion (BIC). The training of mixtures at each step from pre-whitened intonation (pitch, duration, and energy) feature vectors is done by Expectation Maximization (EM). Interesting clusters are formed by this process: for example, rising pitch is seen in clusters 1 and 7, while falling pitch is seen in 0 and 3.

Figure 6-10(b) shows the makeup of each intonation class in terms of vowels, semivowels, and nasals. Clusters 8 and 9 are mostly represented by nasal segments, while clusters 3 through 7 are composed predominantly of vowels and semivowels. Interestingly enough, the pitch in nasals, as represented by clusters 8 and 9, is fairly steady - the start and end pitch is close to the average pitch. In cluster 8, the pitch is high (217.89 Hz) and duration is short (32.24 ms), while in cluster 9, the pitch is low (197.10 Hz) and duration is long (67.08 ms). It is likely that cluster 9 occurs most often in the final position of an utterance. With clusters 8 and 9 also possessing the lowest energy, it may be worthwhile to model intonation for nasals separately. Other clusters with steady or flat pitch are 4 (206.52 Hz), 5 (185.21 Hz), and 6 (224.49 Hz), which are dominated by vowels and semivowels and are spaced in approximately 20 Hz steps.

The set of 10 intonation clusters derived above represents a quantization of the entire space represented by the five dimensions of average pitch, start pitch, end pitch, log duration, and log energy. By using Maximum Likelihood classification, sonorant segments can be assigned to one of the 10 quantized clusters given their feature



(a) Pitch contours by class.

(b) Sonorant composition by class.

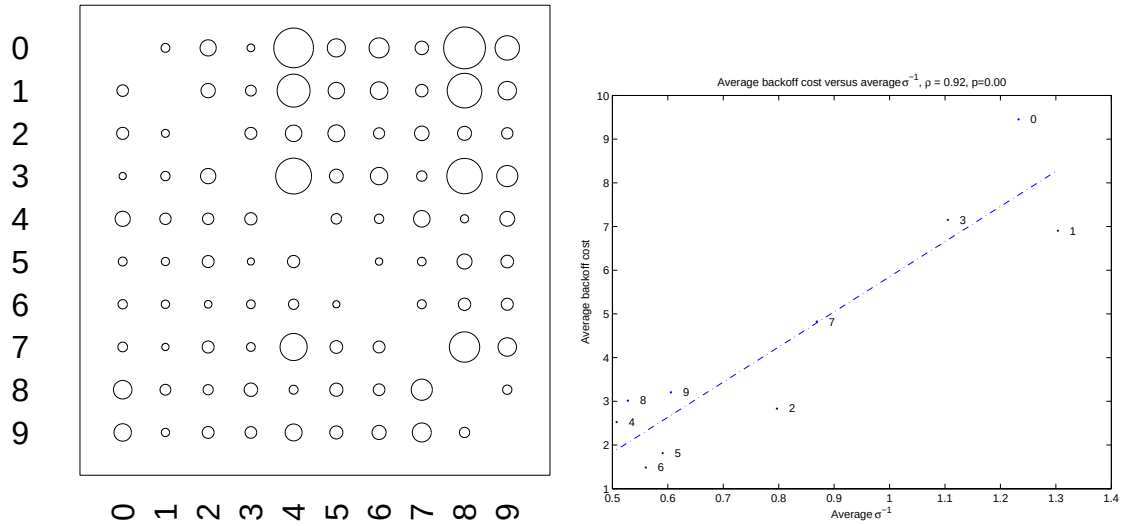
Figure 6-10: Breakup of intonation classes by pitch contour and sonorant composition.

When the intonation classes are viewed by pitch contour, they are either rising, falling, or flat (with slight downward bias). The flat contours apparently span the pitch space equal intervals. When viewed by sonorant composition, clusters 8 and 9 are predominated by nasals, whereas clusters 4 through 6 are mostly vowels and semivowels.

vector by picking the most likely model. Note that this procedure discards any prior information on the model such as the number of training tokens per model. To handle data sparseness, back-off inter-cluster distances are computed by the Kullback-Leibler distance between intonation models.

Figure 6-11(a) displays the back-off costs in the form of a bubble plot in which rows and columns represent the true and approximating intonation clusters and the area of the bubble represents the magnitude of the approximation cost. There are large costs down columns 4 and 8, because intonation clusters 4 and 8 (high, flat pitch and short duration) are poor substitutes for clusters 0, 1, 3, and 7 (changing pitch and long duration). One reason for the large costs is the mismatch in intonation attributes between the two cohorts. Another reason is seen from the correlation plot in Figure 6-11(b) in which there is direct relation (high correlation, zero p-value) between the back-off cost averaged across the column of the bubble plot and inverse-variance, or precision, of the intonation cluster averaged across the five dimensions of

its observation space. This is a direct consequence of the definition of the Kullback-Leibler distance which includes determinant terms of the variance-covariance matrix. Interestingly enough, the first cohort (4 and 8) has the lowest precision, whereas the second cohort (0, 1, 3, and 7) has the highest precision.



(a) Intonation back-off costs.

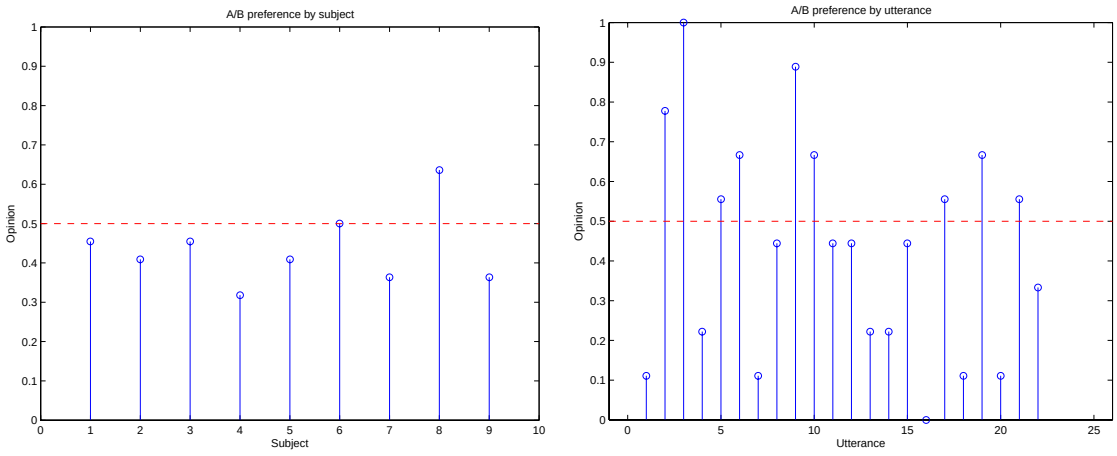
(b) Back-off cost vs. precision.

Figure 6-11: Results of listening evaluation of automatically determined costs.

In the bubble plot, the rows and columns correspond to the true and approximating intonation classes and the area of the bubble denotes the magnitude of the approximation cost. Clusters 4 and 8 (high, flat pitch and short duration) are poor substitutes for clusters 0, 1, 3, and 7 (changing pitch and long duration). A reasonable explanation for this overwhelming trend in back-off cost lies in the variance of the clusters as shown in the correlation plot. Cluster 0, 1, 3, and 7 have high backoff costs, because of their high precision or low variance. Clusters 4 and 8 have low precision or high variance and are poor approximations.

Intonation classes assigned by classification to each sonorant segment are encoded symbolically into the search as segmental diacritics in the synthesis FST, S, described in Chapter 5. The back-off costs incurred when intonation diacritics can not be matched are encoded into the input specification. The testing set of twenty-two utterances is synthesized by jackkniving each utterance out of the corpus one at a time and using its phonological and intonation sequence as oracle information for the search.

To measure the subjective performance, nine of the previous eleven subjects judged the synthesizer using an intonation model against the synthesizer using automatically determined costs. In other words, the better of the two systems in the previously described listening evaluation is used here as the baseline system. On average, use of the intonation model increased concatenation rate from 1.81 from 1.85, a relative increase of 2.4%. Because this is far less than the 15.3% reduction in the previously described evaluation, this may not be as significant. Figure 6-12 shows opinion scores for the new system profiled by subject and utterance. Only two of the nine subjects found the new system indifferent or preferable. Nine of the twenty-two utterances using the intonation model were deemed preferable.



(a) Evaluation results by subject.

(b) Evaluation results by utterance.

Figure 6-12: Results of listening evaluation of automatically acquired intonation model.

This evaluation compares system, A, using automatically determined costs with system, B, using an automatically acquired intonation model. Although the new system was not found to be preferable on average (two out of nine subjects, nine out of twenty-two utterances), it received complete approval for utterance #3 in which naturalness was considerably improved by proper matching of the intonation.

Even though the new system did not fare as well on average, what was surprising was the unequivocal vote for utterance #3 under the new intonation model. With the intonation model in place, the search found a final syllable, [də], for "Canada"

with better duration and pitch by correctly matching the intonation class. In the natural utterance, the final syllable was assigned to cluster 3. Without the intonation model, the search blindly chose a [də] of cluster 5. With the intonation model, the search correctly chose a final [ə] from Georgia in the phrase "from Augusta, Georgia to Austin." The phrase structure has been implicitly matched with only quantized information about intonation! This single utterance gives an existence proof for the ability to automatically mine a large corpus for matching intonation.

Figure 6-13 shows a panel of plots repeating analysis similar to the previous evaluation. None of the correlations are significant from which it can be concluded that other factors must be at play. By incorporating intonation into the search, concatenations may be made at different places in order to satisfy the additional intonation constraint. Because the search path has changed, the effect of the intonation model can not be adequately isolated by these plots. Also, the total cost includes both intonation and acoustic-phonetic costs which should be separated.

Figure 6-14 shows additional correlation plots against measures more directly related to the newly added intonation model. Intonation match rate is the proportion of sonorant segments for which intonation is correctly matched. Although a higher match rate should obviously correspond to higher naturalness, the trend does not appear in the data. Though the concatenation decisions made by the search algorithm change when intonation costs are introduced, the second plot indicates that there is no significant correlation between the opinion and the change in the acoustic-phonetic costs. Looking at intonation cost normalized by the length of the utterance in the third plot, intonation cost rate does not seem to be an important factor. Finally, total intonation cost shown in the fourth plot is not correlated with opinion either.

If an intonation model is effective, the costs it generates should be (negatively) correlated with opinion. When intonation is accurately modelled, increasing intonation mismatch/cost, *ceteris paribus*, should correspond to decreasing naturalness. In the intonation modelling experiments described here, the actual search path changes when

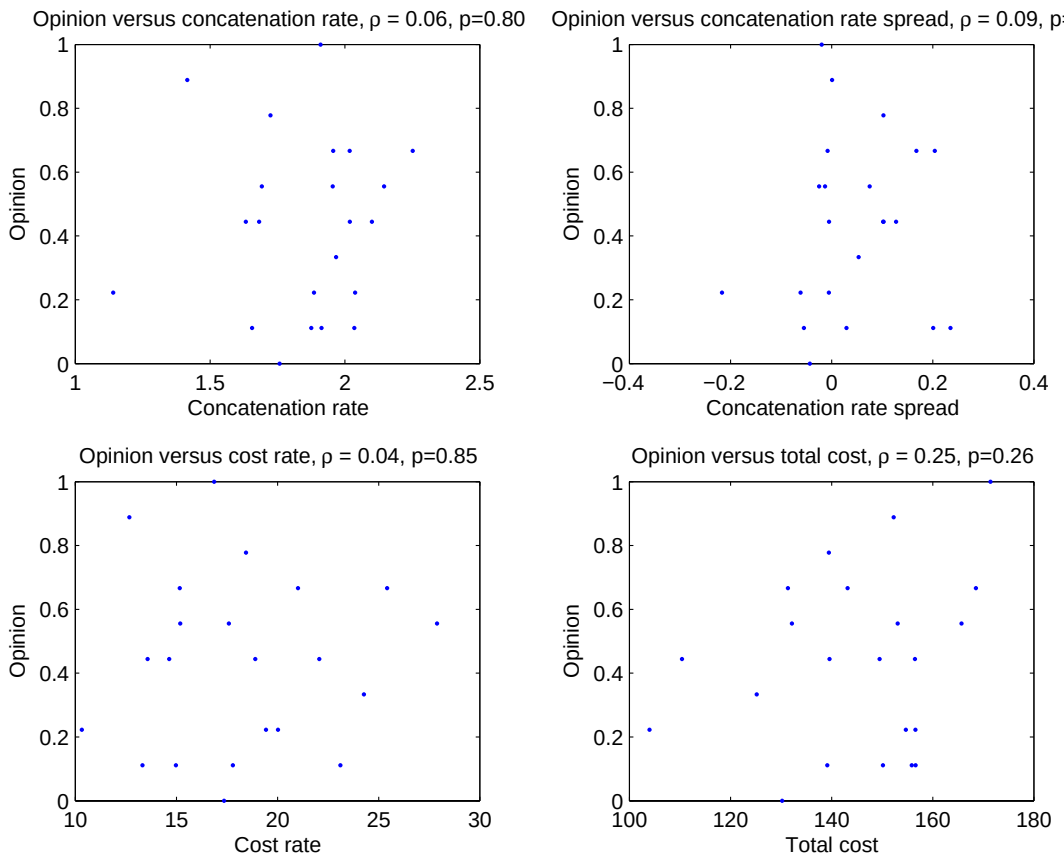


Figure 6-13: Detailed analysis of evaluation of automatically acquired intonation model.

This panel of plots examines correlation between subjective opinion of the voting committee and various objective measures extracted from the system using an automatically acquired intonation model and whether the measures are significant predictors. Unlike the previous evaluation of automatically determined costs, none of the correlation coefficients here are significant which suggests other factors are at play.

the costs change and so the experimental setup fails in keeping other experimental conditions constant. One possible workaround would be to use the intonation model to score the utterances produced by the system with automatically determined costs. The opinion could then be regressed against the acoustic-phonetic cost and intonation score. Notice in this setup that the intonation model is not directly used in the search, but as part of a two-factor regression model for predicting human opinion.

Future work should investigate a more sophisticated application of the intonation model developed in this section by also using intra-cluster distances. A mere match of

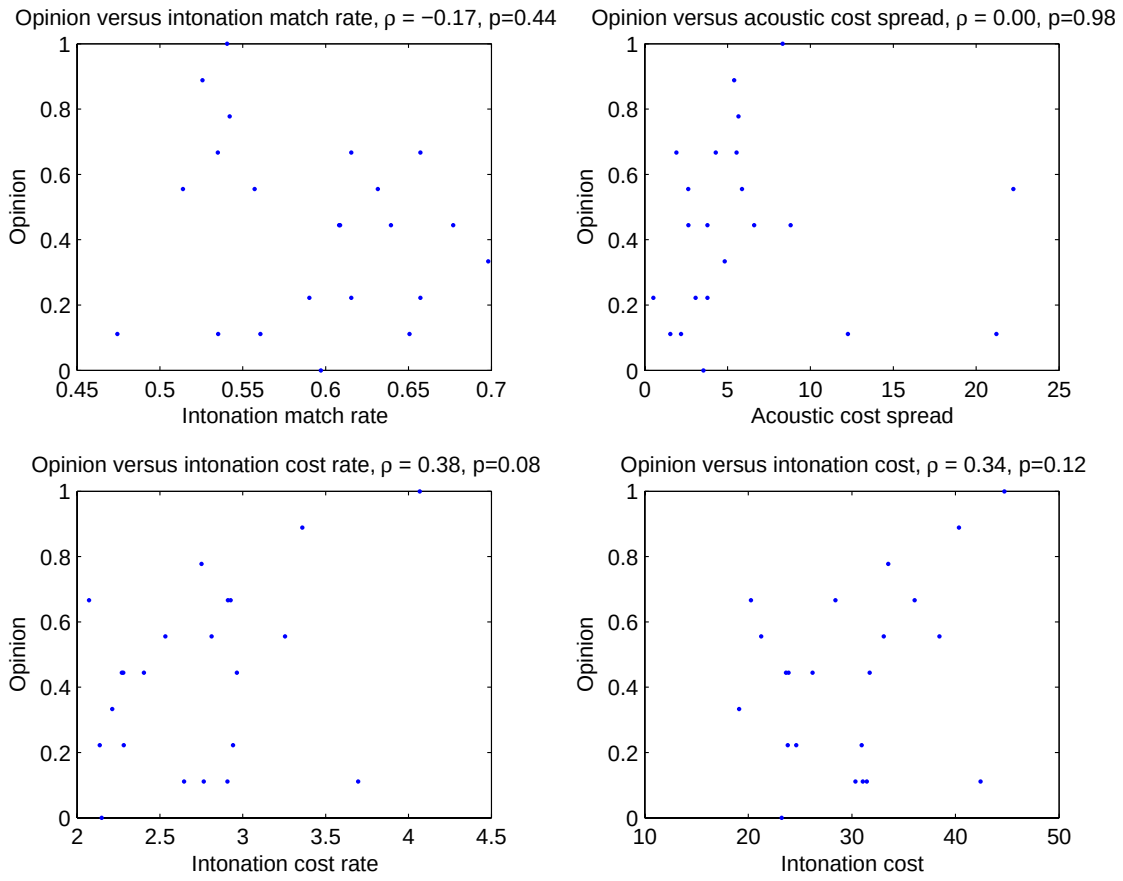


Figure 6-14: More detailed analysis of evaluation of automatically acquired intonation model.

This panel of plots examines correlation between subjective opinion of the voting committee and various objective measures extracted from the system using an automatically acquired intonation model and whether the measures are significant predictors. Unlike the previous evaluation of automatically determined costs, none of the correlation coefficients here are significant which suggests other factors are at play.

intonation cluster identity says nothing about the distance of the intonation exemplar to the cluster centroid. A more accurate measure of the true underlying cost should help to discern between multiple exemplars from the same cluster and to improve intonation matching. Some other necessary procedural changes would include the incorporation of prior information for Maximum A Posteriori (MAP) classification and the balancing of intonation costs with acoustic-phonetic costs. Finally, objective measures for intonation such as pitch continuity of the synthetic utterance can be formulated for predicting human opinion.

6.6 Evaluation by recognition

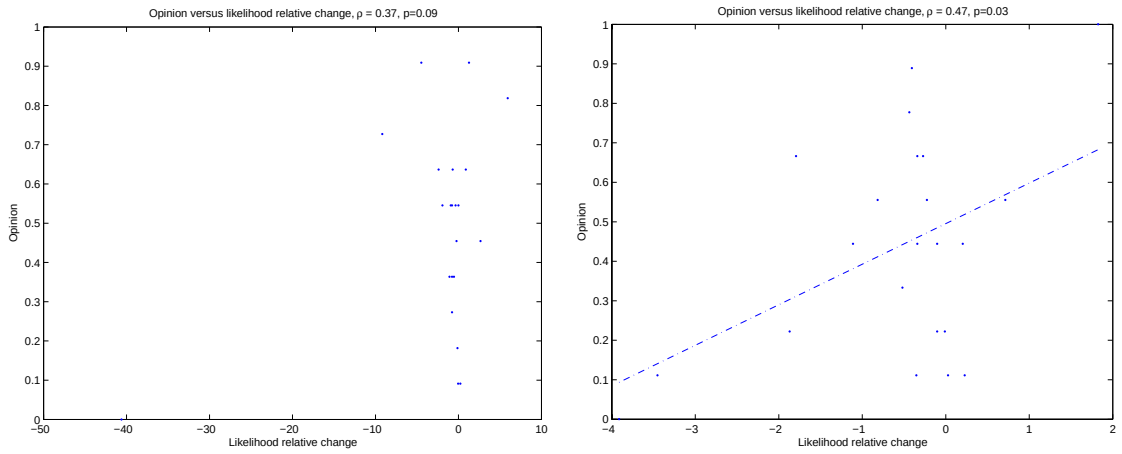
The experiments described thus far in this chapter are aimed at gathering evidence of a link between the path of costs generated by the unit selection search and human opinion in listening evaluations. Because the costs are inter-model distances and do not include intra-model distances, they may not be sufficiently descriptive to serve as a proxy of opinion for the resulting synthetic utterances. Direct measurements taken from the synthetic waveforms ala speech recognition are described and their correlation to human opinion is studied as well.

In this section techniques taken from speech recognition are used to extract acoustic measurements and calculate recognition features directly from the synthetic waveform and to score the features with pre-specified acoustic models. The model scores or likelihoods are examined in conjunction with human opinions obtained from listening evaluations to determine whether human opinion is predictable from a measure like acoustic likelihood. The goal is to move towards a more automatic framework for evaluating synthesis.

Because the synthesizer produces a time-aligned transcription of the speech along with the synthetic waveform, the locations of the phonetic boundaries are known and the boundaries can be scored by a boundary model. The use of segment models to score the phonetic segments could also apply in this evaluation framework, but is not described here. As in SUMMIT, telescoping averages of MFCCs form a feature vector whose likelihood is evaluated by an acoustic model. For a synthetic utterance with N phones, there are $N-1$ boundaries and likelihoods. These boundary likelihoods are used in regression analyses as independent variables to predict human opinion, the dependent variable.

Figure 6-15 shows plots of relative change in likelihood between experimental conditions, A and B, versus opinion or preference between the two systems. Relative change in likelihood from system A to system B, $\Delta L_{A \rightarrow B}$, is defined as the change

in likelihood normalized by the starting likelihood, $\frac{L_B - L_A}{L_A}$. This relative change is computed for each boundary and is averaged across each boundary to produce a single number per utterance. This utterance-level relative change in likelihood is paired with the human opinion for each utterance to produce the correlation plots shown here. The hypothesis is that a relative increase in likelihood should correspond to an increase in preference.



(a) Automatically determined costs.

(b) Automatically acquired intonation model.

Figure 6-15: Regression of results of listening evaluation and likelihood.

This correlation plot regresses the average relative change of likelihood with human opinion. Significant correlation was observed for the comparison of synthesis with and without an intonation model. This suggests that an increase in likelihood predicts an increase in human opinion.

Figure 6-15(a) shows the correlation plot for moving from manually determined costs to automatically determined costs. The correlation is not quite significant, but is positive as hypothesized. Figure 6-15(b) shows the correlation plot for adding intonation modelling to a system with automatically determined costs. The correlation, 0.47, is positive and significant which supports the hypothesis that an increase in likelihood produces better synthesis.

Because speech recognition is typically based on a maximum likelihood framework, a natural extension of this work is to engage the remainder of the speech recognizer,

lexical decoding. If the word sequence encoded in a speech waveform by a synthesizer can be decoded by a recognizer, then a round trip of conversion would be complete and the synthesizer can be deemed intelligible and understandable. Because the speech recognizer used here does not incorporate a prosody model, future work should study automatic evaluation of naturalness.

Table 6.3 shows recognition performance for various synthesis configurations: natural speech spoken by human, manually determined costs (baseline), automatically determined costs, costs and automatically acquired intonation model, costs and automatically determined classes, costs and classes and pronunciation. The last condition, pronunciation, uses the orthographic, not the phonetic, transcription as the input specification for the synthesizer, and can model phonological variation to gain access to additional units in the synthesis corpus. Word error rate (WER) is calculated as $\frac{S+I+D}{N}$, where $N = 631$ is the number of reference words. The language model for the recognizer is a bigram trained on approximately 6,000 in-domain sentences. There is for these sentences, however, mismatch between the representation of numbers in the language model and the lexicon which accounts for the high substitution rate. Although the insertions and deletions are not balanced, the figure of primary concern is the word error rate. Interestingly, word error rates for the artificial systems do not approach that of the natural system (i.e., human).

Configuration/%	Substitutions	Insertions	Deletions	WER
1) Natural	13.8	4.6	0.3	18.7
2) Baseline	14.6	9.2	0.5	24.2
3) Costs	13.5	9.2	0.6	23.3
4) Costs + intonation	13.3	11.1	1.1	25.5
5) Costs + classes	13.0	7.4	0.8	21.2
6) Costs + classes + pronunciation	14.1	10.1	0.2	24.4
7) Costs + classes + pronunciation - smoothing	14.3	5.2	0.3	19.8

Table 6.3: Recognition error analysis for different synthesis configurations.

Recognition errors are shown for seven different synthesis configurations, the first of which being human. Although the substitution rate is high and insertions and deletions are not balanced, primary focus should be placed on the aggregate word error rate.

Word error rate is plotted against average likelihood in Figure 6-16. The average

likelihood for each synthesizer configuration is obtained by averaging utterance-level likelihoods which themselves are produced by averaging over all phonetic boundary likelihoods per utterance. What is revealed in this plot is a statistically significant, negative correlation, -0.87, between likelihood and word error rate. That is to say, as likelihood worsens, word error rate increases.

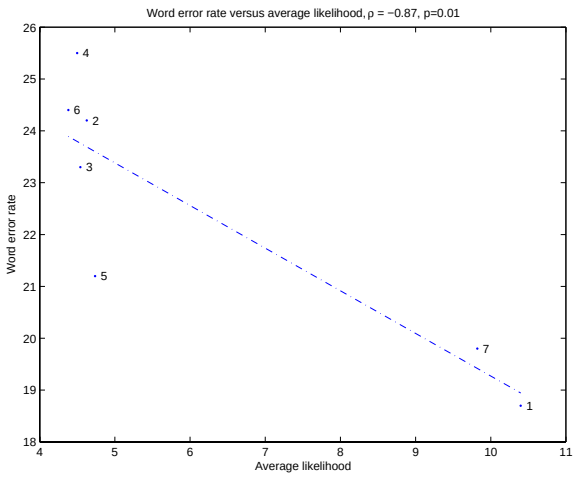


Figure 6-16: Regression of average utterance likelihood and word error rate.

An utterance-level likelihood is calculated by averaging over all phonetic boundary likelihoods. These likelihoods are averaged over all utterance to produce an average likelihood for each synthesizer configuration. The regression of these likelihoods against word error rate shows a statistically significant negative correlation, -0.87. This suggests that as likelihood worsens, the word error rate increases.

An apparent shortcoming in the implementation of the smoothing is seen from the high WER decrease between system 6 and system 7 in Figure 6.3. The high insertion rate of 10.1% in system 6 is caused by zero padding at the start of the synthetic waveform. The segmentation algorithm in SUMMIT triggers a segment boundary at the end of the zero padding. That the frame rate of synthesis is variable is not compensated for in the time alignments of the phonetic transcription. As a result, the accuracy of the phonetic boundaries in the transcription becomes progressively worse towards the end of the utterance as the offsets from the variable frame rate accumulate. This is reflected in a significant negative bias of likelihood seen in Figure 6-16. Although the correlations appear to be strongly positive and statistically significant, the analysis would benefit from improving the smoothing algorithm.

A weak link hinging on likelihood can now be established between word error rate and human opinion. Relative increases in likelihood produced increases in preferences. Increases in average likelihood produced decreases in word error rates. The weak link is that decreases in word error rate suggest increases in preference. If this is true, a closed feedback loop such as the one depicted in Figure 6-6 can be used to evaluate synthesis by recognition. Whether or not this link exists does not detract from the validity of these two independent results. The foregoing examines likelihood increases given the true phonetic transcription. In recognition, the phonetic sequence is unknown and models compete in the maximum likelihood decoding process. Future work should examine whether the likelihoods of competing models also increase with human opinion. This issue can also be approached by examining rank-order statistics, or whether the correct model retains a high rank.

Relative increase in likelihood can serve as a measure for tuning synthesis configurations. It could also be used as a re-scoring component in a two-stage unit selection search. However, as described here, likelihood computation requires access to a local section of the waveform (e.g., $\pm 75ms$ in SUMMIT) at each phonetic boundary. Instead, rescoring is performed as described in Chapter 5 and its effectiveness is evaluated by a speech recognizer. Figure 6.4 presents preliminary results of rescoring. Rescoring with 10,000 top hypotheses reduces WER down to 17.9, below that of natural speech, 18.7. While word error rates are seen to improve beyond that of the case for natural speech, listening reveals that rescoring does not necessarily sound natural. What makes synthetic speech more understandable for a machine may not correspond to what makes more natural sounding for a human.

Configuration/N-rescoring	1	100	500	1000	5000	10000
1) Natural	18.7	NA	NA	NA	NA	NA
7) Costs + classes + pronunciation - smoothing	19.8	19.0	19.8	20.0	18.4	17.9

Table 6.4: Word error rates for speech synthesized with rescoring.

Word error rates are seen to decrease as larger N-best lists are rescored. Beyond $N = 5000$ the WER is better than the WER obtained on natural speech. However, listening reveals that synthetic utterances produced by rescoring do not necessarily sound better. Clearly, what improves word error rate may not improve naturalness.

6.7 Summary

This chapter has presented the embodiment of ideas presented in this thesis as tools. When the tools are applied in a disciplined process, synthesis systems can be designed, built, and evaluated. Human subjects participate in listening evaluations by auditioning synthetic utterances and providing opinion and preference scores. Once their subjective opinion can be correlated to an objective measure, the objective measure can be incorporated into the search to select more natural sounding units. An additional benefit is that humans could be freed from the loop and a machine can take over the time consuming task of tuning a synthesizer.

Chapter 7

Conclusions and future work

This thesis presents frameworks for formulating, parameterizing, and representing a unit selection search for concatenative speech synthesis. Values for all parameters can be derived on either a semi-automatic or fully automatic basis from theoretical guiding principles. The choice of a finite-state transducer representation makes the implementation both efficient and expressive. This entire speech synthesis framework, ENVOICE, is compliant with the GALAXY architecture for conversational systems and operates in a distributed fashion. That it has been deployed in a wide variety of systems across multiple voices, domains, and languages is a testament to its flexibility. Feedback from users trying to accomplish tasks in given scenarios through interactions with spoken dialogue systems indicates that the output responses synthesized by ENVOICE are easy to understand. Subjective opinions gathered from further listening evaluations of various finer aspects of synthesis configuration correlate with objective measures which the machine can automatically assess.

The key contributions of this work can be briefly listed as the following:

1. Communication-theoretic formulation and approximation framework for general unit selection.
2. Analysis of sonorant speech segments with regard to their distributional prop-

erties and conduciveness to concatenation.

3. Finite-state transducer architecture for efficient and expressive implementation of unit selection search.

4. Automatic learning and acquisition of equivalence classes and synthesis costs via construction of acoustical models.

5. Processes for designing, adapting, building, deploying, tuning, and evaluating speech synthesizers.

Practical implementation of the processes has provided validation and even motivation for the technical aspects addressed in this work. The ENVOICE framework has been used across many domains and languages. At current count, it is found in an air travel planning system in English [126, 124], weather information systems in Mandarin Chinese [158] and Japanese [100], task delegation systems in English [125] and Mandarin [31], and a bi-lingual spoken language learning system in Mandarin and English [74]. Although these are research systems, they are often kept running for on-going data collection and contemporaneous demonstrations. An autonomous agent capable of speaking to these systems by way of speech synthesis could take over the tedious task of making sure the systems are up.

Integrating video-realistic facial animation [46] with speech synthesis would be a natural next step towards multimodal speech synthesis [121]. Moving from developing unit selection synthesizers for separate domains to perfecting unit selection for generic use is challenging and requires much more research, but techniques such as *blending* domains into a single voice and *tiering* different voices across domains may be appropriate solutions in the meantime [11]. Voice transformation techniques may enable the creation of new voices from existing voices [145, 66, 67]. System back-channel while the user is still speaking would increase the naturalness of the interactions extending previous work on intelligent barge-in [136]. The question remains whether expressive or emotional speech synthesis [23, 96] can be achieved with concatenative methods [58, 113, 18]. Although speech synthesis has considerably improved in the past century or so, it is difficult to predict whether synthetic speech will ever equal

or surpass human speech in naturalness. If it indeed does become indistinguishable or preferable to human speech, it will have then passed the Turing test [150].

The power of expression offered by the FST representation clears the path to future research. The hierarchical transition units and intonation diacritics described earlier are just two examples of extensions that can be handled by the FST framework. The constraint kernel topology can be further refined by generalization and specialization costs between layers of exemplars and categories. More sophisticated second-pass search and rescoring algorithms should be investigated along with practical features such as low-latency, island-based pipelining. A completely orthogonal approach to real-time processing would be to adopt a parallel distributed processing paradigm (e.g., neural networks) [117]. Recurrent neural networks (i.e., output feeds back into the input) have been constructed to map single sequences of equal length from an input language to an output language [48, 103]. This idea of encoding finite-state automata into an RNN has recently been refined into what is called neural transducers [135]. To complete the transformation of a linear architecture into a parallel architecture, it would be necessary to introduce multiple paths and associated weights in both the input and output.

The experiments performed on sonorant speech segments reveal a new source of concatenation boundaries located at extrema of spectral prominences that can be exploited for creating more compact synthesis corpora. Decomposition of sub-phonetic spectral frames into spectral magnitude and excitation phase functions at a small time scale has the potential of producing natural-sounding speech synthesis. Full automation could be achieved by a set of parallel searches for the optimal magnitude and excitation functions that re-combine to reconstitute speech. Preliminary implementation of this setup has been done for small corpora with exact calculations. Moving to larger corpora would require approximations.

There are some shortcomings in the development of the acoustic model. The very divide-and-conquer nature of a decision tree introduces a requisite for a large data

set to avoid sparse data issues. Perhaps Maximum Entropy methods [6] which do not "split the data" could be applied to learn clusters for smaller data sets. The substitution and concatenation costs are currently derived from *plug-in* entropy estimates, a process that involves initial density estimation and subsequent closed-form entropy calculation from the parametric density (e.g., Gaussian.) Empirical estimators based on order statistics and spacings [91] offer a direct path for deriving entropies from a set of data. Better methods are needed for the proper interpretation of the automatically acquired equivalence classes and synthesis costs as it is always important to relate results of engineering back to well understood principles in speech science.

Finally, further work in understanding the perceptual artifacts caused by concatenation will ultimately benefit the naturalness of concatenative speech synthesis. From a dynamical system viewpoint, concatenating speech is like concatenating observations without paying attention to, or, not even to mention, smoothing the underlying state sequence. Also, anecdotal evidence seems to point to the existence of a "chasing effect" [129] in which rapidly occurring concatenations cause a reversal of the perceived temporal order of speech segments with respect to the actual order of presentation. Perhaps work in auditory scene analysis [15] would have an explanation for this.

One interesting bit offered here in parting concerns itself with the, possibly, split perception of linguistic and intonation information. Figure 7-1 shows the waveform and spectrogram of the sentence, "There are 49 Honda Civics." Sinewave analysis and resynthesis [4, 114, 45] replaces each formant with a frequency-modulated oscillator and produces a waveform bereft of intonation with most of the acoustic-phonetic content intact, albeit sounding somewhat inhuman. The dual of sinewave analysis is *mumble synthesis*, in which individual pitch periods are replaced by a sine wave of the proper magnitude and length (i.e., $T = \frac{1}{F_0}$). The percept is that of muffled speech in which the acoustic-phonetic content is virtually impossible to ascertain, whereas the intonation is remarkably preserved. This is an absolutely startling example that simply highlights the need for a better understanding of speech production and perception in humans that should benefit speech synthesis as well.

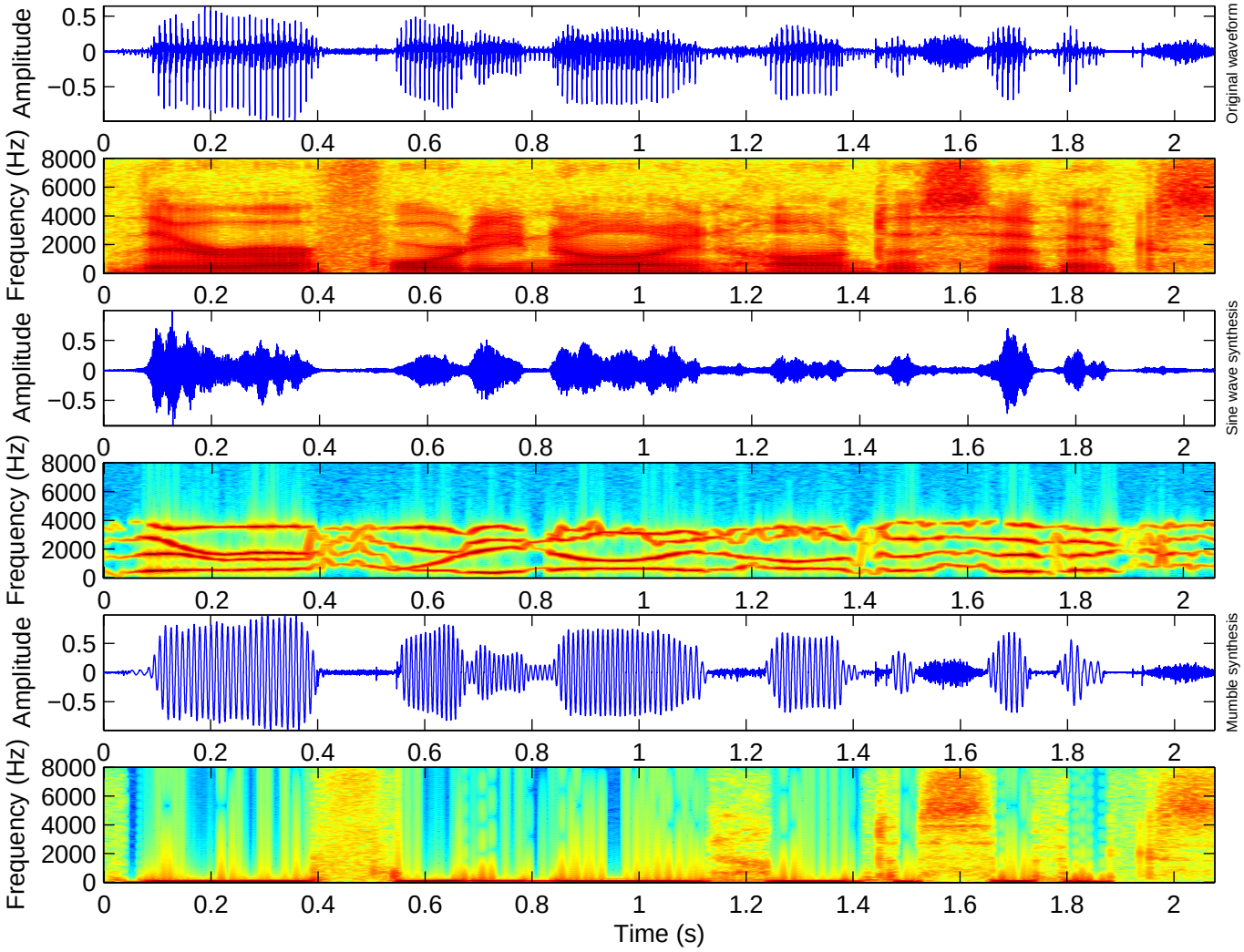


Figure 7-1: Sinewave and mumble resynthesis: There are 49 Honda Civics.

A waveform, its sinewave resynthesis, and its mumble resynthesis is displayed in MATLAB. The sinewave resynthesis was performed using scripts written by Dan Ellis. The mumble resynthesis was performed by replacing individual pitch periods with sinewaves of appropriate magnitude and length. Upon listening the acoustic-phonetics and intonation appear to be well isolated in the sinewave and mumble waveforms, respectively.

Bibliography

[1] J. Allen, S. Hunnicutt, and D. Klatt, *From Text to Speech: The MITalk System*, Cambridge University Press, 1976.

[2] A. M. Aull, "Lexical stress and its application in large vocabulary speech recognition," M.S. thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, 1984.

[3] A. M. Aull and V. Zue, "Lexical stress determination and its application to large vocabulary speech recongnition," in *Proc. ICASSP '85*, Tampa, USA, Mar. 1985, pp. 1549–1552.

[4] P. Bailey, Q. Summerfield, and M. Dorman, "On the identification of sine-wave analogues of certain speech sounds," Tech. Rep. Status Report on Speech Perception, SR 51-52, Haskins Laboratories, 1977.

[5] L. Baptist and S. Seneff, "GENESIS-II: A versatile system for language generation in conversational system applications," in *Proc. ICSLP '00*, Beijing, China, Oct. 2000, vol. 3, pp. 271–274.

[6] A. L. Berger, S. A. Della Pietra, and V. J. Della Pietra, "A maximum entropy approach to natural language processing," *Computational Linguistics*, vol. 22, no. 1, pp. 39–68, 1996.

[7] M. Beutnagel, A. Conkie, J. Schroeter, Y. Stylianou, and A. Syrdal, "The AT&T Next-gen TTS system," *Joint Meeting of ASA, EAA, and DAGA*, pp. 18–24, 1998.

[8] M. Beutnagel, M. Mohri, and M. Riley, "Rapid unit selection from a large speech corpus for concatenative speech synthesis," in *Proc. Eurospeech '99*, Budapest, Hungary, Sept. 1999, pp. 607–610.

[9] M. Beutnagel, A. Syrdal, and P. Brown, "Preselection of candidate units in a unit selection-based text-to-speech synthesis system," in *Proc. ICSLP '00*, Beijing, China, Oct. 2000, vol. 3, pp. 314–317.

[10] A. Black and K. Lenzo, "Limited domain synthesis," in *Proc. ICSLP '00*, Beijing, China, Oct. 2000, vol. 2, pp. 411–414.

[11] A. Black, "Perfect synthesis for all of the people all of the time," *Proc. of the IEEE Workshop on Speech Synthesis*, 2002.

[12] A. Black and A. J. Hunt, "Generating F0 contours from ToBI labels using linear regression," in *Proc. ICSLP '96*, Philadelphia, PA, Oct. 1996, vol. 3, pp. 1385–1388.

[13] A. Black and P. Taylor, "Automatically clustering similar units for unit selection in speech synthesis," in *Proc. Eurospeech '97*, Rhodes, Greece, Sept. 1997, vol. 2, pp. 601–604.

[14] A. Black and P. Taylor, "The Festival speech synthesis system," Tech. Rep. HCRC/TR-83, University of Edinburgh, Jan. 1997.

[15] A. S. Bregman, *Auditory scene analysis: the perceptual organization of sounds*, The MIT Press, London, 1990.

[16] E. Brill, D. Magerman, M. Marcus, and B. Santorini, "Deducing linguistic structure from the statistics of large corpora," in *Proc. DARPA Speech and Natural Language Workshop*, 1990.

[17] P. F. Brown, J. Cocke, S. A. Della Pietra, V. J. Della Pietra, F. Jelinek, J. D. Lafferty, R. L. Mercer, and P. S. Roossin, "A statistical approach to machine translation," *Computational Linguistics*, vol. 16, no. 2, pp. 79–85, 1990.

[18] M. Bulut, S. Narayanan, and A. Syrdal, "Expressive speech synthesis using a concatenative synthesizer," in *Proc. ICSLP '02*, Denver, Colorado, Sept. 2002.

[19] I. Bulyko and M. Ostendorf, "Efficient integrated response generation from multiple targets using weighted finite-state transducers," *Computer Speech and Language*, vol. 16, no. 3/4, pp. 533–550, July 2002.

[20] I. Bulyko, *Flexible speech synthesis using weighted finite-state transducers*, Ph.D. thesis, Electrical Engineering, University of Washington, Seattle, WA, Mar. 2002.

[21] I. Bulyko and M. Ostendorf, "Joint prosody prediction and unit selection for concatenative speech synthesis," in *Proc. ICASSP '01*, Salt Lake City, UT, May 2001, vol. 2, pp. 781–784.

[22] I. Bulyko and M. Ostendorf, "Unit selection for speech synthesis using splicing costs with weighted finite state transducers," in *Proc. Eurospeech '01*, Aalborg, Denmark, Sept. 2001, vol. 2, pp. 987–990.

[23] J. Cahn, "Generating expressions in synthesized speech," M.S. thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, 1989.

[24] N. Campbell, "CHATR: A high-definition speech re-sequencing system," *Acoustical Society of America and Acoustical Society of Japan, Third Joint Meeting*, Dec. 1996.

[25] D. Chappell and J. Hansen, "A comparison of spectral smoothing methods for segment concatenation based speech synthesis," *Speech Communication*, vol. 36, no. 3–4, pp. 343–373, Mar. 2002.

[26] F. Charpentier and E. Moulines, "Pitch-synchronous waveform processing techniques for text-to-speech synthesis using diphones," in *Proc. Eurospeech '89*, Paris, France, Sept. 1989, vol. 2, pp. 13–19.

[27] M. Chu, C. Li, H. Peng, and E. Chang, "Domain adaptation for TTS systems," in *Proc. ICASSP '02*, Orlando, FL, May 2002.

[28] M. Chu and H. Peng, "An objective measure for estimating MOS of synthesized speech," in *Proc. Eurospeech '01*, Aalborg, Denmark, Sept. 2001.

[29] M. Chu, H. Peng, and E. Chang, "A concatenative Mandarin TTS system without prosody model and prosody modification," in *Proc. ESCA Workshop on Speech Synthesis*, Scotland, 2001, number 115.

[30] G. Chung, "Automatically incorporating unknown words in Jupiter," in *Proc. ICSLP '00*, Beijing, China, Oct. 2000.

[31] C. Chuu, "LIESHOU: A Mandarin conversational task agent for the Galaxy-II architecture," M.Eng. thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, Dec. 2002.

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

[33] A. Conkie and S. Isard, "Optimal coupling of diphones," in *Progress in speech synthesis*, J. van Santen, R. Sproat, J. Olive, and J. Hirschberg, Eds., pp. 293–304. Springer Verlag, New York, 1996.

[34] S. Davis and P. Mermelstein, "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences," *IEEE Trans. Acoustics, Speech and Signal Processing*, vol. ASSP-28, no. 4, pp. 357–366, Aug. 1980.

[35] S. Deligne and F. Bimbot, "Inference of variable-length linguistic and acoustic units by multigrams," *Speech Communication*, vol. 23, no. 3, pp. 223–241, Nov. 1997.

[36] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society, Series B*, vol. 39, pp. 1–38, June 1977.

[37] R. Donovan, *Trainable speech synthesis*, Ph.D. thesis, Cambridge University Engineering Department, Cambridge, UK, June 1996.

[38] R. Donovan, "Segment preselection in decision-tree based speech synthesis systems," in *Proc. ICASSP '00*, Istanbul, Turkey, June 2000, vol. 2, pp. 937–940.

[39] R. Donovan, "A new distance measure for costing spectral discontinuities in concatenative speech synthesis," in *Proc. ESCA Workshop on Speech Synthesis*, Scotland, 2001, number 123.

[40] R. Donovan, "Optimal data selection for unit selection synthesis," in *Proc. ESCA Workshop on Speech Synthesis*, Scotland, 2001, number 129.

[41] R. Donovan and P. Woodland, "Automatic speech synthesiser parameter estimation using HMMs," in *Proc. ICASSP '98*, Seattle, WA, May 1995, pp. 640–643.

[42] R. Dujari, "Parallel Viterbi search algorithm for speech recognition," M.S. thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, 1992.

[43] T. Dutoit, Ed., *An introduction to text-to-speech synthesis*, Kluwer Academic Publishers, 1996.

[44] T. Dutoit and H. Leich, "MBR-PSOLA: text-to-speech synthesis based on an MBE re-synthesis of the segments database," *Speech Communication*, vol. 13, no. 3–4, Nov. 1993.

[45] D. Ellis, "Sinewave speech analysis/synthesis in Matlab," URL http://www.ee.columbia.edu/ dpwe/resources/matlab/sws/.

[46] T. Ezzat, *Trainable Videorealistic Speech Animation*, Ph.D. thesis, Massachusetts Institute of Technology, Cambridge, MA, June 2002.

[47] G. Fant, *Descriptive analysis of the acoustic aspects of speech*, pp. 17–31, The MIT Press, Cambridge, MA, 1973.

[48] P. Frasconi, M. Cori, M. Maggini, and G. Soda, "Representation of finite state automata in recurrent radial basis function networks," *Machine Learning*, vol. 23, pp. 5–32, 1996.

[49] H. Fujisaki and H. Kawai, "Modeling the dynamic characteristics of voice fundamental frequency with applications to analysis and synthesis of intonation," in *Working Group on Intonation, 13th International Congress of Linguists*, Tokyo, 1982.

[50] J. Glass, J. Chang, and M. McCandless, "A probabilistic framework for feature-based speech recognition," in *Proc. ICSLP '96*, Philadelphia, PA, Oct. 1996, pp. 2277–2280.

[51] J. Glass, T. J. Hazen, and I. L. Hetherington, "Real-time telephone-based speech recognition in the JUPITER domain," in *Proc. ICASSP '99*, Phoenix, AZ, Mar. 1999, pp. 61–64.

[52] A. Gray and J. Markel, "Distance measures for speech processing," *IEEE Trans. Acoustics, Speech and Signal Processing*, vol. ASSP-24, no. 5, pp. 380–391, Oct. 1976.

[53] D. W. Griffin and J. S. Lim, "Signal estimation from modified short-time Fourier transform," *IEEE Trans. Acoustics, Speech and Signal Processing*, vol. ASSP-32, no. 2, pp. 236, 1984.

[54] A. Halberstadt, *Heterogeneous Measurements and Multiple Classifiers for Speech Recognition*, Ph.D. thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Nov. 1998.

[55] T. J. Hazen, I. L. Hetherington, H. Shu, and K. Livescu, "Pronunciation modeling using a finite-state transducer representation," in *Proc. of the ISCA Workshop on Pronunciation Modeling and Lexicon Adapation*, Estes Park, Colorado, Sept. 2002, pp. 99–104.

[56] I. L. Hetherington and M. McCandless, "SAPPHIRE: An extensible speech analysis and recognition tool based on tcl/tk," in *Proc. ICSLP '96*, Philadelphia, PA, Oct. 1996, pp. 1942–1945.

[57] I. L. Hetherington, "An efficient implementation of phonological rules using finite-state transducers," in *Proc. Eurospeech '01*, Aalborg, Denmark, Sept. 2001, pp. 1599–1602.

[58] B. Heuft, T. Portele, and M. Rauth, "Emotions in time domain synthesis," in *Proc. ICSLP '96*, Philadelphia, PA, Oct. 1996.

[59] H. Hon, A. Acero, X. Huang, J. Liu, and M. Plumpe, "Automatic generation of synthesis units for trainable text-to-speech synthesis," in *Proc. ICASSP '98*, Seattle, WA, May 1998, vol. 1, pp. 293–296.

[60] X. Huang, A. Acero, J. Adcock, H. Hon, J. Goldsmith, J. Liu, and M. Plumpe, "Whistler: A trainable text-to-speech system," in *Proc. ICSLP '96*, Philadelphia, PA, Oct. 1996, vol. 4, pp. 2387–2390.

[61] A. Hunt and A. Black, "Unit selection in a concatenative speech synthesis system using a large speech database," in *Proc. ICASSP '96*, Atlanta, GA, May 1996, pp. 373–376.

[62] D. Huttenlocher and V. Zue, "A model of lexical access from partial phonetic information," in *Proc. ICASSP '84*, San Diego, CA, Mar. 1984, pp. 391–394.

[63] N. Iwahashi, N. Kaiki, and Y. Sagisaka, "Concatenative speech synthesis by minimum distortion criteria," in *Proc. ICASSP '92*, San Francisco, CA, Mar. 1992, vol. 2, pp. 65–68.

[64] F. Jelinek, "Continuous speech recognition by statistical methods," *Proceedings of the IEEE*, vol. 64, no. 4, pp. 532–556, 1976.

[65] F. Jelinek, L. R. Bahl, and R. L. Mercer, "Design of a linguistic statistical decoder for the recognition of continuous speech," *IEEE Trans. Information Theory*, vol. 21, no. 3, pp. 250–256, 1975.

[66] A. Kain, *High resolution voice transformation*, Ph.D. thesis, Oregon Health and Science University, Portland, OR, Oct. 2001.

[67] A. Kain and J. van Santen, "Compression of acoustic inventories using asynchronous interpolation," *Proc. of the IEEE Workshop on Speech Synthesis*, 2002.

[68] J. L. Kelly and C. C. Lochbaum, "Speech synthesis," in *Proc. of the Fourth Intl. Congress on Acoustics*, Copenhagen, Sept. 1962, vol. 4, pp. 1–4.

[69] E. Klabbers and R. Veldhuis, "On the reduction of concatenation artefacts in diphone synthesis," in *Proc. ICSLP '98*, Sydney, Australia, Nov. 1998, pp. 1983–1986.

[70] E. Klabbers and R. Veldhuis, "Reducing audible spectral discontinuities," *IEEE Trans. Acoustics, Speech and Signal Processing*, vol. 9, no. 1, pp. 39–51, Jan. 2001.

[71] D. Klatt, "Review of text to speech conversion for English," *Journal of the Acoustical Society of America*, vol. 82, pp. 737–793, 1987.

[72] H. Kucera and W. N. Francis, *Computational analysis of present-day American English*, Brown University Press, Providence, RI, 1967.

[73] P. Ladefoged, "The revised international phonetic alphabet," *Journal of the International Phonetic Association*, vol. 19, no. 2, pp. 67–80, 1990.

[74] T.-L. J. Lau, "SLLS: An online conversational spoken language learning system," M.Eng. thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, May 2003.

[75] S. Lee, "Probabilistic segmentation for segment-based speech recognition," M.Eng. thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, June 1998.

[76] I. Lehiste, *Suprasegmentals*, The MIT Press, Cambridge, MA, 1970.

[77] R. Likert, *New patterns of management*, McGraw-Hill, 1961.

[78] A. Ljolje and M. Riley, "Automatic segmentation of speech for TTS," in *Proc. Eurospeech '99*, Budapest, Hungary, Sept. 1999, pp. 1445–1448.

[79] S. Logan and D. Pisoni, "Preference judgements comparing different synthetic voices," *Speech Technology*, vol. Suppl. 1, pp. S24, 1979.

[80] P. Luce, T. Feustel, and D. Pisoni, "Capacity demands in short-term memory for synthetic and natural speech," *Human Factors*, vol. 25, pp. 17–31, 1983.

[81] R. W. P. Luk and R. I. Damper, "Stochastic phonographic transduction for english," *Computer Speech and Language*, vol. 10, no. 2, pp. 133–153, Apr. 1996.

[82] D. M. Magerman and M. P. Marcus, "Parsing a natural language using mutual information statistics," in *National Conference on Artificial Intelligence*, 1990, pp. 984–989.

[83] M. Makasha, C. Wightman, A. Syrdal, and A. Conkie, "Perceptual evaluation of automatic segmentation in text-to-speech synthesis," in *Proc. ICSLP '00*, Beijing, China, Oct. 2000.

[84] J. Makhoul, "Linear prediction: a tutorial review," *Proc. of the IEEE*, vol. 63, no. 4, pp. 561–580, Apr. 1975.

[85] L. M. Manous, D. Pisoni, M. J. Dedina, and H. C. Nusbaum, "Perception of speech: the human listener as a cognitive interface," Tech. Rep. Speech Research Laboratory Progress Report 11, Indiana University, 1985.

[86] Y. Marchand and R. I. Damper, "A multistrategy approach to improving pronunciation by analogy," *Computational Linguistics*, vol. 26, no. 2, pp. 195–219, June 2000.

[87] H. D. Maxey, "Smithsonian Speech Synthesis History Project (SSSHP)," URL http://www.mindspring.com/ ssshp/ssshp_cd/ss_home.htm.

[88] R. J. McAulay and T. F. Quatieri, "Speech analysis/synthesis based on a sinusoidal representation," *IEEE Trans. Acoustics, Speech and Signal Processing*, vol. ASSP-34, no. 4, pp. 744, 1986.

[89] C. McLemore, "COMLEX English pronunciation dictionary," URL http://www.ldc.upenn.edu/ldc/catalog/html/lexical_html/comlexep.html.

[90] H. Meng, S. Busayapongchai, J. Glass, D. Goddeau, L. Hetherington, E. Hurley, C. Pao, J. Polifroni, S. Seneff, and V. Zue, "WHEELS: A conversational system in the automobile classifieds domain," in *Proc. ICSLP '96*, Philadelphia, PA, Oct. 1996, pp. 542–545.

[91] E. G. Miller and J. W. Fisher, "ICA using spacing estimates of entropy," in *Proc. Fourth International Symposium on Independent Component Analysis and Blind Signal Separation*, Nara, Japan, Apr. 2003, pp. 1047–1052.

[92] B. Möbius, "Corpus-based speech synthesis: methods and challenges," *Arbeitspapiere des Instituts für Maschinelle Sprachverarbeitung*, vol. 6, no. 4, pp. 87–116, 2000.

[93] B. Möbius, "Rare events and closed domains: two delicate concepts in speech synthesis," *International Journal of Speech Technology*, vol. 6, no. 1, pp. 57–71, 2003.

[94] M. Mohri and R. Sproat, "An efficient compiler for weighted rewrite rules," in *Proc. 34th Mtg. of the Assoc. for Computational Linguistics*, Santa Cruz, CA, June 1996, pp. 231–238.

[95] C. Moler, J. Little, and S. Bangert, "Pro-Matlab user's guide," 1987.

[96] I. Murray and J. Arnott, "Toward the simulation of emotion in synthetic speech: a review of the literature on human vocal emotion," *Journal of the Acoustical Society of America*, vol. 93, pp. 1097–1108, 1993.

[97] S. Nakajima, "English speech synthesis based on multi-layered context oriented clustering; towards multi-lingual speech synthesis.," in *Proc. Eurospeech '93*, Berlin, Germany, Sept. 1993, vol. 3, pp. 1709–1712.

[98] S. Nakajima, "Automatic synthesis unit generation for english speech synthesis based on multi-layered context oriented clustering," *Speech Communication*, vol. 14, pp. 313–324, Sept. 1994.

[99] S. Nakajima and H. Hamada, "Automatic generation of synthesis units based on context oriented clustering," in *Proc. ICASSP*, New York, NY, Apr. 1988, vol. 1, pp. 659–662.

[100] M. Nakano, Y. Minami, S. Seneff, T. J. Hazen, D. S. Cyphers, J. Glass, J. Polifroni, and V. Zue, "Mokusei: a telephone-based Japanese conversational system in the weather domain," in *Proc. Eurospeech '01*, Aalborg, Denmark, Sept. 2001, pp. 1331–1334.

[101] H. Noda and M. N. Shirazi, "A MRF-based parallel processing algorithm for speech recognition using linear predictive HMM," in *Proc. ICASSP '94*, Adelaide, Austrailia, Apr. 1994, vol. 1, pp. 597–600.

[102] J. P. Olive, "Rule synthesis of speech from dyadic units," in *Proc. ICASSP '77*, New York, NY, 1977, pp. 568–570.

[103] C. Omlin and C. Giles, "Constructing deterministic finite-state automata in recurrent neural networks," *Journal of the ACM*, vol. 43, no. 6, pp. 937–972, Nov. 1996.

[104] M. Ostendorf and I. Bulyko, "The impact of speech recognition on speech synthesis," *Proc. of the IEEE Workshop on Speech Synthesis*, 2002.

[105] M. Ostendorf, P. J. Price, and S. Shattuck-Hufnagel, "The Boston University radio news corpus," Tech. Rep. ECS-95-001, Boston University, 1995.

[106] H. Peng, Y. Zhao, and M. Chu, "Perceptually optimizing the cost function for unit selection in a TTS system with one single run of MOS evaluation," in *Proc. ICSLP '02*, Denver, Colorado, Sept. 2002.

[107] G. E. Peterson, W. S.-Y. Wang, and E. Sivertsen, "Segmentation techniques in speech synthesis," *Journal of the Acoustical Society of America*, vol. 30, no. 8, pp. 739–742, Aug. 1958.

[108] S. Phillips and A. Rogers, "Parallel speech recognition," *International Journal of Parallel Programming*, vol. 27, no. 4, pp. 257–288, Aug. 1999.

[109] J. Pierrehumbert, *The Phonology and Phonetics of English Intonation*, Ph.D. thesis, Massachusetts Institute of Technology, 1980.

[110] D. Pisoni, "Perception of speech: the human listener as a cognitive interface," *Speech Technology*, vol. 1, pp. 10–23, 1982.

[111] M. Plumpe and S. Meredith, "Which is important in a concatenative text-to-speech system - pitch, duration, or spectrla discontinuity?," in *Proc. ESCA/COCOSDA Workshop on Speech Synthesis*, 1998, pp. 231–236.

[112] M. R. Portnoff, "Time-Scale modification of speech based on short-time fourier analysis," *IEEE Trans. Acoustics, Speech and Signal Processing*, vol. ASSP-29, no. 3, pp. 374, 1981.

[113] E. Rank and H. Pirker, "Generating emotional speech with a concatenative synthesizer," in *Proc. ICSLP '98*, Sydney, Australia, Nov. 1998, pp. 671–674.

[114] R. E. Remez, P. E. Rubin, D. B. Pisno, and T. D. Carrell, "Speech perception without traditional speech cues," *Science*, vol. 212, pp. 947–950, 1981.

[115] E. Roche and Y. Shabes, Eds., *Finite-state language processing*, The MIT Press, Cambridge, MA, 1997.

[116] S. Roweis, "Gaussian identities," July 1999, URL http://www.cs.toronto.edu/ roweis/notes/gaussid.pdf.

[117] D. E. Rumelhart and J. L. McClelland, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Vol. I: Foundations.*, MIT Press, Cambridge, MA, 1986.

[118] Y. Sagisaka, "Speech synthesis by rule using an optimal selection of non-uniform synthesis units," in *Proc. ICASSP*, New York, NY, Apr. 1988, pp. 679–682.

[119] Y. Sagisaka, N. Kaiki, N. Iwahashi, and K. Mimura, "ATR - ν-TALK speech synthesis system," in *Proc. ICSLP '92*, Banff, Canada, Oct. 1992, pp. 483–486.

[120] J. Schalkwyk, L. Hetherington, and E. Story, "Speech recognition with dynamic grammars," Geneva, Switzerland, Sept. 2003, submitted to Proc. Eurospeech '03.

[121] J. Schroeter, J. Ostermann, H. P. Graf, M. Beutnagel, E. Cosatto, A. Conkie, and Y. Stylianou, "Multimodal speech synthesis," *Proc. IEEE International Conference on Multimedia and Expo*, vol. 1, pp. 571–574, 2000.

[122] G. Schwarz, "Estimating the dimension of a model," *The Annals of Statistics*, vol. 6, no. 2, pp. 461–464, 1978.

[123] S. Seneff, "System to independently modify excitation and/or spectrum of speech waveform without explicit pitch extraction," *IEEE Trans. Acoustics, Speech and Signal Processing*, vol. ASSP-30, no. 4, pp. 566, 1982.

[124] S. Seneff, "Response planning and generation in the Mercury flight reservation system," *Computer Speech and Language*, vol. 16, pp. 283–312, 2002.

[125] S. Seneff, C. Chuu, and D. S. Cyphers, "Orion: from on-line interaction to off-line delegation," in *Proc. ICSLP '00*, Beijing, China, Oct. 2000, pp. 142–145.

[126] S. Seneff and J. Polifroni, "Formal and natural language generation in the Mercury conversational system," in *Proc. ICSLP '00*, Beijing, China, Oct. 2000, vol. 3, pp. 767–770.

[127] B. Serridge, "Context-dependent modeling in a segment-based speech recognition system," M.S. thesis, Massachusetts Institute of Technology, 1997.

[128] D. W. Shipman and V. Zue, "Properties of large lexicons: implications for advanced isolated word recognition systems," in *Proc. ICASSP '82*, New York, NY, 1982, pp. 546–549.

[129] H. Shu, "Personal communications," 2002.

[130] H. Shu and I. L. Hetherington, "EM training on finite-state transducers and its application to pronunciation modeling," in *Proc. ICSLP '02*, Denver, Colorado, Sept. 2002, pp. 1293–1296.

[131] K. Silverman, M. B. Beckman, J. Pirelli, M. Ostendorf, C. Wightman, P. Price, J. Pierrehumbert, and J. Hirschberg, "ToBI: A standard for labeling English prosody," in *Proc. ICSLP'92*, Banff, Canada, 1992, pp. 867–870.

[132] R. Sproat, "Multilingual text analysis for text-to-speech synthesis," *Journal of Natural Language Engineering*, pp. 369–380, 1997.

[133] R. Sproat, Ed., *Multilingual text-to-speech synthesis: The Bell Labs Approach*, Kluwer Academic Publishers, Boston, 1997.

[134] R. Sproat and M. Riley, "Compilation of weighted finite-state transducers from decision trees," in *Proc. 34th Mtg. of the Assoc. for Computational Linguistics*, Santa Cruz, CA, June 1996, pp. 215–222.

[135] I. Stoianov, *Connectionist lexical modelling*, Ph.D. thesis, University of Gronigen, The Netherlands, Feb. 2001.

[136] N. Ström and S. Seneff, "Intelligent barge-in in conversational systems," in *Proc. ICSLP '00*, Beijing, China, Oct. 2000.

[137] Y. Stylianou and A. Syrdal, "Perceptual and objective detection of discontinuities in concatenative speech synthesis," in *Proc. ICASSP '01*, Salt Lake City, UT, May 2001, vol. 2, pp. 837–840.

[138] Y. Stylianou, "Applying the harmonic plus noise model in concatenative speech synthesis," *IEEE Trans. Acoustics, Speech and Signal Processing*, vol. 9, no. 1, pp. 21–29, Jan. 2001.

[139] Y. Stylianou, "Removing linear phase mismatches in concatenative speech synthesis," *IEEE Trans. Acoustics, Speech and Signal Processing*, vol. 9, no. 1, pp. 39–51, Jan. 2001.

[140] X. Sun, "F0 generation for speech synthesis using a multi-tier approach," in *Proc. ICSLP '02*, Denver, Colorado, Sept. 2002.

[141] A. Syrdal, A. Conkie, and Y. Stylianou, "Exploration of acoustic correlates in speaker selection for concatenative synthesis," in *Proc. ICSLP '98*, Sydney, Australia, Nov. 1998.

[142] A. Syrdal, A. Conkie, Y. Stylianou, J. Schroeter, L. Garrison, and D. Dutton, "Voice selection for speech synthesis," *Journal of the Acoustical Society of America*, vol. 102, no. 5, pp. 3191, 1997.

[143] D. Talkin, "Speech formant trajectory estimation using dynamic programming with modulated transition costs," Tech. Rep. 11222-870720-07TM, AT&T Bell Laboratories, 1987.

[144] D. Talkin, "Voicing epoch determination with dynamic programming," *Journal of the Acoustical Society of America*, vol. 85, no. Suppl. 1, 1989.

[145] M. Tang, C. Wang, and S. Seneff, "Voice transformations: from speech synthesis to mammalian vocalizations," in *Proc. Eurospeech '01*, Aalborg, Denmark, Sept. 2001.

[146] P. Taylor, *A phonetic model of English intonation*, Ph.D. thesis, University of Edinburgh, 1992.

[147] P. Taylor, "Analysis and synthesis of intonation using the tilt model," *Journal of the Acoustical Society of America*, vol. 107, no. 3, pp. 1697–1714, 2000.

[148] P. Taylor and A. Black, "Speech synthesis by phonological structure matching," in *Proc. Eurospeech '99*, Budapest, Hungary, Sept. 1999, vol. 2, pp. 623–636.

[149] D. Torre Toledano, M. Rodriguez Crespo, and J. Escalada Sardina, "Trying to mimic human segmentation of speech using HMM and fuzzy logic post-correction rules," in *Proc. ESCA/COCOSDA Workshop on Speech Synthesis*, 1998.

[150] A. Turing, "Computing machinery and intelligence," *Journal of the Mind Association*, vol. LIX, no. 236, pp. 433–460, 1950.

[151] J. van Santen and R. Sproat, "High-accuracy automatic segmentation," in *Proc. Eurospeech '99*, Budapest, Hungary, Sept. 1999.

[152] J. van Santen, R. Sproat, J. Olive, and J. Hirschberg, Eds., *Progress in speech synthesis*, Springer Verlag, New York, 1996.

[153] J. P. H. van Santen, "Combinatorial issues in text-to-speech synthesis," in *Proc. Eurospeech '97*, Rhodes, Greece, Sept. 1997, pp. 2511–2514.

[154] W. Verhelst and M. Roelands, "An overlap-add technique based on waveform similarity (WSOLA) for high quality time-scale modification of speech," in *Proc. ICASSP '93*, Minneapolis, MN, Apr. 1993, pp. 554–557.

[155] P. Viola, N. Schraudolph, and T. Sejnowski, "Empirical entropy manipulations for real-world problems," in *Advances in Neural Information Processing Systems*

9, D. Touretzky, M. Mozer, and M. Hasselmo, Eds., Denver, CO, 1996, The MIT Press.

[156] M. Walker, J. Aberdeen, J. Boland, E. Bratt, J. Garofolo, L. Hirschman, A. Le, S. Lee, S. Narayanan, K. Papineni, B. Pellom, J. Polifroni, A. Potamianos, P. Prabhu, A. Rudnicky, G. Sanders, S. Seneff, D. Stallard, and S. Whittaker, "DARPA Communicator dialog travel planning systems: The June 2000 data collection," in *Proc. Eurospeech '01*, Aalborg, Denmark, Sept. 2001, pp. 1371–1374.

[157] M. Walker, A. Rudnicky, J. Aberdeen, E. Bratt, J. Garofolo, H. Hastie, A. Le, B. Pellom, A. Potamianos, R. Passonneau, R. Prasad, S. Roukos, G. Sanders, S. Seneff, and D. Stallard, "DARPA Communicator evaluation: progress from 2000 to 2001," in *Proc. ICSLP '02*, Denver, Colorado, Sept. 2002, pp. 273–276.

[158] C. Wang, S. Cyphers, X. Mou, J. Polifroni, S. Seneff, J. Yi, and V. Zue, "MUXING: A telephone-access Mandarin conversational system," in *Proc. ICSLP '00*, Beijing, China, Oct. 2000, vol. 2, pp. 715–718.

[159] D. R. Williams, "Synthesis of initial (/s/-) stop-liquid clusters using HLsyn," in *Proc. ICSLP '96*, Philadelphia, PA, Oct. 1996, vol. 4, pp. 2219–2222.

[160] J. Wouters and M. Macon, "A perceptual evaluation of distance measures for concatenative speech synthesis," in *Proc. ICSLP '98*, Sydney, Australia, Nov. 1998, vol. 6, pp. 2747–2750.

[161] J. Wouters and M. Macon, "Control of spectral dynamics in concatenative speech synthesis," *IEEE Trans. Acoustics, Speech and Signal Processing*, vol. 9, no. 1, pp. 30–38, Jan. 2001.

[162] J. Yi, "Natural-sounding speech synthesis using variable-length units," M.Eng. thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, June 1998.

[163] J. Yi and J. Glass, "Natural-sounding speech synthesis using variable-length units," in *Proc. ICSLP '98*, Sydney, Australia, Nov. 1998, pp. 1167–1170.

[164] V. Zue, "The use of speech knowledge in automatic speech recognition," *Proceedings of the IEEE*, vol. 73, no. 11, pp. 1602–1615, Nov. 1985.

[165] V. Zue, D. S. Cyphers, R. H. Kassel, D. H. Kaufman, H. C. Leung, M. Randolph, S. Seneff, J. E. Unverferth, III, and T. Wilson, "The development of the MIT LISP-Machine based speech research workstation," in *Proc. ICASSP '86*, Tokyo, Japan, Apr. 1986, pp. 329–332.

Appendix A

Finite-state transducer (FST) composition

When mapping from one representation to another, it is often times easier to divide the task into a series of mappings whose composition achieves the desired transformation. This is witnessed in, for example, function theory where functions map a domain, the set of possible inputs, into a range, the set of possible outputs, and where functions can be successively composed. Consider an example where $y = f(x)$ defines a function, f, mapping x to y and where $z = g(y)$ defines a function, g, mapping y to z. A new function, h, mapping x to z can be created by the successive application of f and g: $z = h(x) = g(f(x))$. This can also be written as $h(x) = (g \circ f)(x) = g(f(x))$ or, more compactly, as $h = g \circ f$. The $\circ$ operator can itself be thought of as a function - a higher-order function, in fact, whose domain and range are sets of functions - which maps the two functions on its left and right side to a third composite function.

In more general terms where the range of f is a subset of the domain of g:

$$\begin{aligned} f: &\quad X \to Y \\ g: &\quad W \to Z, \quad Y \subset W \\ \circ: &\quad (X \to Y) \times (W \to Z) \to (X \to Z) \\ h: &\quad X \to Z \end{aligned}$$

An example application of the functions would be:

$$\begin{aligned} x \in X, \quad y \in Y: &\quad y = f(x) \\ w \in W, \quad z \in Z: &\quad z = g(w) \\ x \in X, \quad z \in Z: &\quad z = h(x) = (g \circ f)(x) = g(f(x)) \end{aligned}$$

The above discussion applies to FST's in similar terms. If an FST, F, maps an input regular language, X, to an output regular language, Y, and an FST, G, maps an input regular language, W (a superset of Y), to an output regular language, Z, the composition of F and G, H, maps X to Z. H may now be used to directly effect mappings from X to Z. This composition represents a benefit, because it has simplified the mapping process. The set of all symbols of a language is referred to as the alphabet. For example, the set of all $x \in X$ is the input alphabet of F.

In practice where computation time affects system performance, additional benefit may be gained by optimizing H, which will minimize the number of states and arcs, by subjecting the transducer to determinization, minimization, and ϵ-removal. For example, if there exists replication of substructures in F due to internal tracking of state and if the mapping performed by G is not sensitive or indifferent to the state, the replication can be compacted in H.

A.1 Example of composition

Consider a toy language, $\mathcal{L}_{toy}$, where the only phones are /b/ and /ɑ/. An example utterance would be /bɑ/ as shown in Figure A-1. This transducer, call it U, consists of three states and two arcs. The initial state is 0 and the final state is 2 as marked by the double circle. The first arc connects states 0 and 1 and the second arc connects states 1 and 2. U happens to also be a finite-state acceptor (FSA), because all arcs perform identity mappings (i.e., absorb input label and emit identical output label.) FSA's are usually rendered without the colon and the redundant label. Note that the phonetic symbols are rendered in an ASCII version of the International Phonetic Alphabet (IPA) [73].

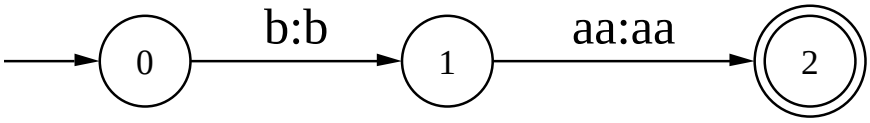


Figure A-1: FST for example utterance in toy language.

An example utterance, /bɑ/, is represented by an FST, U, with a trivial chain topology containing one arc per phonetic symbol. The transducer performs an identity mapping by absorbing and emitting the identical label displayed on either side of the colon.

A common task in speech recognition is the evaluation of context-dependent models, which requires the mapping from context-independent labels to context-dependent labels. A transducer, called D for diphones, that performs this task for the toy language is shown in Figure A-2. It is applied or composed on the right side of the above utterance FST and outputs left-sensitive context-dependent labels.

Continuing the example, applying D on the right side produces the FST in Figure A-3. The phonetic sequence, bɑ, has been mapped to the diphone sequence, <> |b b|ɑ, where <> represents nil context.

Because mapping to context-dependent labels is not a lossless process (it actually adds redundancy), further composition with the inverse of D, D^{-1}, would restore context-independent labels and produce the original utterance. In other words, $U =$

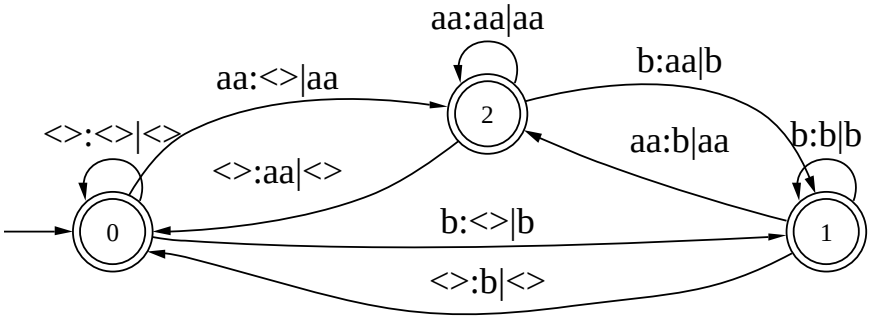


Figure A-2: FST for diphone rewriting in toy language.

An FST, D, rewrites a sequence of phones into a sequence of left-sensitive context-dependent labels by maintaining internal state which tracks the previous phone. The states, 0, 1, and 2, correspond to a previous phone whose identity is <> (nil context), ɓ, and ɑ, respectively. The transducer starts in the initial state, 0, because there is nil context at the start of a sequence. After entering the transducer, for example, absorbing an ɑ symbol will emit a <> |ɑ symbol, while making a transition from state 0 to state 2. Arcs departing state 2, which encodes a previous context of ɑ, will emit labels containing ɑ| concatenated with the next phonetic symbol.

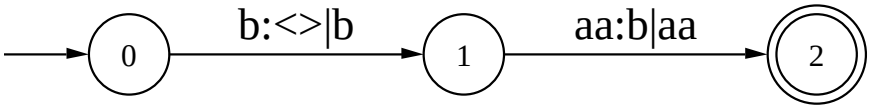


Figure A-3: FST for previous example utterance as rewritten with diphones.

The result of composing U with D creates an FST with a trivial chain topology mapping from phonetic symbols to diphone symbols. Specifically, this composition maps the phonetic sequence, ɓɑ, to the diphone sequence, <> |ɓ ɓ|ɑ.

$(U \circ D) \circ D^{-1}$ or $U = U \circ (D \circ D^{-1})$ by associativity of composition. The second term on the right side of the second equation would suggest that $D \circ D^{-1}$, as displayed in Figure A-4, is an identity mapping, $I_{\mathcal{L}_{toy}}$, Although this is confirmed in Figure, $D \circ D^{-1}$ contains triplication of substructure and can be optimized. Subsequent use of optimized transducers translates directly into reduced computation as redundant sub-structures are not wastefully explored. Note that the input and output alphabets of the optimized identity transducer are augmented with the nil context symbol, <>, because it was present in the input alphabet of D.

A.2 Two-sided composition

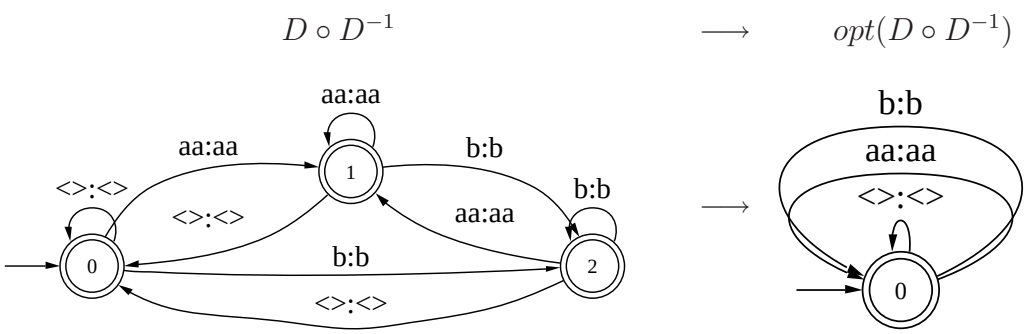


Figure A-4: FST for $D \circ D^{-1}$ before and after optimization.

The FST resulting from the composition of D and D^{-1} should be an identity mapping over the input alphabet of D. The transducer on the left contains triplicate substructure, because D maintains state based on the previous phonetic input. The optimized version on the right has undergone minimization which removes the redundant substructure.

Appendix B

Clustering questions

```
LEFT: aa ah ah_fp ao aw ax ow uh uw ae ay eh ey ih ix iy oy ux w ww y yy el -l l ll er -r r rr axr
LEFT: aa ah ah_fp ao aw ax ow uh uw ae ay eh ey ih ix iy oy ux w ww y yy
LEFT: aa ao ah ah_fp ax uh aw ow uw w ww
LEFT: ae eh ih ix ux ay oy ey iy y yy
LEFT: aa ao ah ah_fp ax uh
LEFT: aw ow uw w ww
LEFT: ay oy ey iy y yy
LEFT: ae eh ih ix ux
LEFT: ah ah_fp ax uh
LEFT: ey iy y yy
LEFT: ih ix ux
LEFT: aa ao
LEFT: aw ow uw
LEFT: w ww
LEFT: aw ow
LEFT: ae eh
LEFT: ay oy
LEFT: ey iy
LEFT: y yy
LEFT: ih ix
LEFT: b p p- d dr t t- tr g k k- jh zh ch sh z s dh th v f dx hh m em n en ng nx
LEFT: b p p- d dr t t- tr g k k-
LEFT: m b p p-
LEFT: b p p-
LEFT: b p-
LEFT: s z n d t t- dr tr
LEFT: d t t- dr tr
LEFT: dr tr
LEFT: d t t-
LEFT: d t-
LEFT: ng g k k-
LEFT: g k k-
LEFT: g k-
LEFT: bcl dcl gcl epi kcl pcl tcl tq - _
LEFT: bcl dcl gcl
LEFT: epi kcl pcl tcl tq - _
LEFT: epi kcl pcl tcl tq
LEFT: epi tcl tq
LEFT: tcl tq
LEFT: - _
LEFT: jh zh ch sh z s dh th v f dx hh
LEFT: jh zh ch sh z s
LEFT: jh zh ch sh
LEFT: ch sh
LEFT: jh zh
LEFT: z s
LEFT: dh th v f dx hh
LEFT: dh th v f
LEFT: dh th
LEFT: v f
LEFT: dx hh
LEFT: el -l l ll
LEFT: el -l
LEFT: l ll
LEFT: er -r r rr axr
LEFT: er axr -r
LEFT: er axr
LEFT: r rr
LEFT: m em
LEFT: m em n en ng nx
LEFT: n en nx ng
LEFT: n en nx
LEFT: n en
```

Figure B-1: Left-sided triphone questions.

```
RIGHT: aa ah ah_fp ao aw ax ow uh uw ae ay eh ey ih ix iy oy ux w ww y yy el -l l ll er -r r rr axr
RIGHT: aa ah ah_fp ao aw ax ow uh uw ae ay eh ey ih ix iy oy ux w ww y yy
RIGHT: aa ah ah_fp ao ax ay ow oy uh uw w ww
RIGHT: ae aw eh ey ih ix iy ux y yy
RIGHT: aa ah ah_fp ao ax ay ow oy uh
RIGHT: uw w ww
RIGHT: aa ay
RIGHT: ah ah_fp ax
RIGHT: ao oy
RIGHT: ow uh
RIGHT: ah ax
RIGHT: w ww
RIGHT: ae aw eh ey
RIGHT: ih ix ux
RIGHT: iy y yy
RIGHT: ae aw eh
RIGHT: ae eh
RIGHT: ih ix
RIGHT: y yy
RIGHT: b p p- d dr t tr t- g k k- ch jh s z sh zh dh th v f dx hh
RIGHT: b p p- d dr t tr t- g k k-
RIGHT: b p p-
RIGHT: b p-
RIGHT: d dr t tr t-
RIGHT: d t t-
RIGHT: dr tr
RIGHT: d t-
RIGHT: g k k-
RIGHT: g k-
RIGHT: bcl dcl gcl epi kcl pcl tcl tq - _ em en m n ng nx
RIGHT: bcl dcl gcl epi kcl pcl tcl tq - _
RIGHT: bcl dcl gcl em en m n ng nx
RIGHT: bcl em m
RIGHT: dcl en n nx
RIGHT: gcl ng
RIGHT: epi kcl pcl tcl tq - _
RIGHT: epi kcl pcl tcl tq
RIGHT: - _
RIGHT: epi tcl tq
RIGHT: tcl tq
RIGHT: ch jh s z sh zh dh th v f dx hh
RIGHT: ch jh s z sh zh
RIGHT: ch jh
RIGHT: sh zh
RIGHT: s z
RIGHT: dh th v f dx hh
RIGHT: dh th v f
RIGHT: dh th
RIGHT: v f
RIGHT: dx hh
RIGHT: em m
RIGHT: em en m n ng nx
RIGHT: n en nx ng
RIGHT: n en nx
RIGHT: n en
RIGHT: er axr r rr -r
RIGHT: er axr rr -r
RIGHT: er axr
RIGHT: rr -r
RIGHT: l ll -l el
RIGHT: ll -l el
RIGHT: ll -l
```

Figure B-2: Right-sided triphone questions.

```
LEFT: aa ao ah ah_fp ax uh aw ow uw w ww
LEFT: ae eh ih ix ux ay oy ey iy y yy
LEFT: aa ao ah ah_fp ax uh
LEFT: aw ow uw w ww
LEFT: ay oy ey iy y yy
LEFT: ae eh ih ix ux
LEFT: ah ah_fp ax uh
LEFT: ey iy y yy
LEFT: ih ix ux
LEFT: aa ao
LEFT: aw ow uw
LEFT: w ww
LEFT: aw ow
LEFT: ae eh
LEFT: ay oy
LEFT: ey iy
LEFT: y yy
LEFT: ih ix
LEFT: b p p-
LEFT: b p-
LEFT: d t t- dr tr
LEFT: dr tr
LEFT: d t t-
LEFT: d t-
LEFT: g k k-
LEFT: g k-
LEFT: bcl dcl gcl
LEFT: epi kcl pcl tcl tq - _
LEFT: epi kcl pcl tcl tq
LEFT: epi tcl tq
LEFT: tcl tq
LEFT: - _
LEFT: jh zh ch sh
LEFT: ch sh
LEFT: jh zh
LEFT: z s
LEFT: dh th v f
LEFT: dh th
LEFT: v f
LEFT: dx hh
LEFT: el -l
LEFT: l ll
LEFT: er axr -r
LEFT: er axr
LEFT: r rr
LEFT: m em
LEFT: n en nx ng
LEFT: n en nx
LEFT: n en
```

Figure B-3: Left-sided diphone questions.

```
RIGHT: aa ah ah_fp ao ax ay ow oy uh uw w ww
RIGHT: ae aw eh ey ih ix iy ux y yy
RIGHT: aa ah ah_fp ao ax ay ow oy uh
RIGHT: uw w ww
RIGHT: aa ay
RIGHT: ah ah_fp ax
RIGHT: ao oy
RIGHT: ow uh
RIGHT: ah ax
RIGHT: w ww
RIGHT: ae aw eh ey
RIGHT: ih ix ux
RIGHT: iy y yy
RIGHT: ae aw eh
RIGHT: ae eh
RIGHT: ih ix
RIGHT: y yy
RIGHT: b p p-
RIGHT: b p-
RIGHT: d dr t tr t-
RIGHT: d t t-
RIGHT: dr tr
RIGHT: d t-
RIGHT: g k k-
RIGHT: g k-
RIGHT: bcl dcl gcl
RIGHT: epi kcl pcl tcl tq - _
RIGHT: epi kcl pcl tcl tq
RIGHT: - _
RIGHT: epi tcl tq
RIGHT: tcl tq
RIGHT: ch jh
RIGHT: sh zh
RIGHT: s z
RIGHT: dh th v f
RIGHT: dh th
RIGHT: v f
RIGHT: dx hh
RIGHT: em m
RIGHT: n en nx ng
RIGHT: n en nx
RIGHT: n en
RIGHT: er axr rr -r
RIGHT: er axr
RIGHT: rr -r
RIGHT: ll -l el
RIGHT: ll -l
```

Figure B-4: Right-sided diphone questions.

```
LEFT: aa ah ah_fp ao aw ax ow uh uw ae ay eh ey ih ix iy oy ux w ww y yy
LEFT: b p p- d dr t t- tr g k k-
LEFT: bcl dcl gcl epi kcl pcl tcl tq - _
LEFT: jh zh ch sh z s
LEFT: dh th v f dx hh
LEFT: el -l l ll
LEFT: er -r r rr axr
LEFT: m em n en ng nx
LEFT: <>
RIGHT: aa ah ah_fp ao ax ay ow oy uh uw ae aw eh ey ih ix iy ux w ww y yy
RIGHT: b p p- d dr t tr t- g k k-
RIGHT: bcl dcl gcl epi kcl pcl tcl tq - _
RIGHT: ch jh s z sh zh
RIGHT: dh th v f dx hh
RIGHT: em en m n ng nx
RIGHT: er axr r rr -r
RIGHT: l ll -l el
RIGHT: <>
```

Figure B-5: Initial diphone questions.

Appendix C

Gaussian random variables

Consider a multivariate Gaussian random variable, x, whose distribution can be written as:

$$p(x) = \frac{1}{(2\pi)^{N/2}|\Sigma|^{1/2}} e^{-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)} \sim \mathcal{N}(\mu, \Sigma)$$

where N is the number of dimensions, x and μ are 1 x N vectors, and Σ is a N x N matrix. A Gaussian probability density is completely specified by its mean and covariance, μ and Σ, respectively. They can be estimated in a maximum likelihood sense from a corpus of data by collecting first- and second-order statistics. In other words, the estimates of μ and Σ are just the sample mean and covariance:

$$\hat{\mu} = \frac{1}{M}\sum_{i=1}^{M} x_i, \qquad \hat{\Sigma} = \frac{1}{M-1}\sum_{i=1}^{M}(x_i - \mu)(x_i - \mu)^T.$$

The following identity is very useful in identifying closed-form expressions for quadratic products under a Gaussian expectation.

$$\begin{aligned} E_{X\sim\mathcal{N}(m,S)}\left[(x-\mu)^T \Sigma^{-1}(x-\mu)\right] &= \int_X (x-\mu)^T \Sigma^{-1}(x-\mu)\,\mathcal{N}(m,S)dx \\ &= (m-\mu)^T \Sigma^{-1}(m-\mu) \;+\; Tr\left[\Sigma^{-1}S\right] \end{aligned}$$

Appendix D

Information theory

D.1 Introduction

This section describes an information-theoretic framework in which synthesis classes and costs can be automatically derived from data. Scalar metrics quantify how much two arbitrary probability distribution functions differ. Gaussian distribution are assumed below since closed-form expressions exist for Kullback-Leibler divergence and mutual information. The costs can be interpreted as Minimum Description Length (MDL) costs which quantify the amount of information (measured in bits) required to describe a distribution. In the context of unit selection search, the total path length is the total number of bits required to describe deviations - due to corpus limitations - from the ideal input specification.

D.2 Information-theoretic metrics

The Kullback-Leibler distance for two probability densities, $P \sim p(x)$ and $Q \sim q(x)$, is defined to be:

$$\mathcal{D}(P \mid\mid Q) \;=\; \int p(x) \log \frac{p(x)}{q(x)}\; dx = -\int p(x) \log q(x)\; dx - H(P)$$

where the first term is cross entropy, $H_P(Q) = E_P\left[\log \frac{1}{q(x)}\right]$, and the second term, $-H(P)$, is negative differential entropy, $H(P) = E_P\left[\log \frac{1}{p(x)}\right] = -\int p(x) \log p(x)\; dx$. Kullback-Leibler distance describes additional information the transmitter must supply about q given that the receiver already has p.

Mutual information is defined as:

$$\begin{aligned} \mathcal{I}(P\;;\;Q) &= H(P) - H(P \mid Q) \\ &= H(Q) - H(Q \mid P) \\ &= H(P) + H(Q) - H(P,Q) \\ &\leq min(H(P), H(Q)) \end{aligned}$$

and it describes how much information is known about one random variable given knowledge of the other. While it can be expressed in terms of conditional entropy, $H(P \mid Q) = -\int p_{P|Q}(x) \log p_{P|Q}(x)\; dx \leq H(P)$, and joint entropy, $H(P,Q) = -\int\int p_{P,Q}(x) \log p_{P,Q}(x)\; dx\; dy = H(P) + H(Q|P) = H(Q) + H(P|Q)$, an alternative form involves the Kullback-Leibler distance between a joint distribution and its factored marginals:

$$\mathcal{I}(P\;;\;Q) \;=\; \mathcal{D}(p_{P,Q}(x,y) \mid\mid p_P(x) * p_Q(y)).$$

Jensen's Inequality can be used to derive an upper bound for KL distance, $\mathcal{D}(P \,||\, Q) \leq \sum_i \mu_i \mathcal{D}(P \,||\, Q_i)$, when the approximating distribution is a mixture:

$$\begin{aligned} H_P\left(\textstyle\sum_i \mu_i q_i(x)\right) &= -E_P\left[\log\left(\textstyle\sum_i \mu_i q_i(x)\right)\right], \quad \textstyle\sum_i \mu_i q_i(x) = 1 \\ &\leq -E_P\left[\textstyle\sum_i \mu_i \log q_i(x)\right] \\ &= -\textstyle\sum_i \mu_i E_P\left[\log q_i(x)\right] \\ &= \textstyle\sum_i \mu_i H_P(Q_i) \end{aligned}$$

D.2.1 Expressions for Gaussian condition

For multivariate Gaussian random variables, entropy, mutual information, and Kullback-Leibler distance are:

$$H(\mathcal{N}(\mu, \Sigma)) = \tfrac{1}{2} \log\left((2\pi e)^N |\Sigma|\right)$$

$$\mathcal{I}(P\,;\,Q) = \tfrac{1}{2} \log\left(\tfrac{|\Sigma_{pp}|\ |\Sigma_{qq}|}{|\Sigma|}\right), \qquad \Sigma = \begin{bmatrix} \Sigma_{pp} & \Sigma_{pq} \\ \Sigma_{qp} & \Sigma_{qq} \end{bmatrix}$$

$$\mathcal{D}(P \,||\, Q) = \tfrac{1}{2}\left[(\mu_Q - \mu_P)^T \Sigma_Q^{-1} (\mu_Q - \mu_P) + Tr\left[\Sigma_P \Sigma_Q^{-1} - I\right] - \log|\Sigma_P \Sigma_Q^{-1}|\right].$$

Note that entropy or uncertainty is only dependent on the spread of the distribution. Mutual information involves joint statistics between two distributions which can be written in a block matrix of variances and co-variances. The Kullback-Leibler distance involves (half of) the Mahalanobis distance of μ_P in the context of Q. Note that I is the identity matrix and the logarithmic term can be regarded as the ratio of two volumes, $\log \frac{|\Sigma_P|}{|\Sigma_Q|}$. In fact, when $\Sigma_P = \Sigma_Q$, the trace and logarithmic terms disappear, leaving only the Mahalanobis distance. For this reason, the Kullback-Leibler distance can be regarded as a generalized version of the Mahalanobis distance.

D.2.2 Derivations

$$
\begin{aligned}
H(\mathcal{N}(\mu,\Sigma)) &= E\left[\tfrac{1}{2}\log\left((2\pi)^N|\Sigma|\right) + \tfrac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right] \\
&= \tfrac{1}{2}\log\left((2\pi)^N|\Sigma|\right) + \tfrac{1}{2}(\mu-\mu)^T\Sigma^{-1}(\mu-\mu) + Tr\left[\Sigma^{-1}\Sigma\right] \\
&= \tfrac{1}{2}\log\left((2\pi)^N|\Sigma|\right) + N \\
&= \tfrac{1}{2}\log\left((2\pi e)^N|\Sigma|\right)
\end{aligned}
$$

$$
\begin{aligned}
\mathcal{I}(P\ ;\ Q) &= H(P) + H(Q) - H(P,Q) \\
&= \tfrac{1}{2}\log\left((2\pi e)^{\frac{N}{2}}|\Sigma_{pp}|\right) + \tfrac{1}{2}\log\left((2\pi e)^{\frac{N}{2}}|\Sigma_{qq}|\right) - \tfrac{1}{2}\log\left((2\pi e)^N|\Sigma|\right) \\
&= \tfrac{1}{2}\log\left(\tfrac{|\Sigma_{pp}|\ |\Sigma_{qq}|}{|\Sigma|}\right) + \tfrac{1}{2}\log\left(\tfrac{(2\pi e)^N}{(2\pi e)^N}\right) \\
&= \tfrac{1}{2}\log\left(\tfrac{|\Sigma_{pp}|\ |\Sigma_{qq}|}{|\Sigma|}\right)
\end{aligned}
$$

$$
\begin{aligned}
\mathcal{D}(P\ ||\ Q) &= E_P\left[\log\tfrac{p(x)}{q(x)}\right] = E_P\left[\log\tfrac{1}{q(x)}\right] - H(P) \\
&= E_P\left[\tfrac{1}{2}\log\left((2\pi)^N|\Sigma_Q|\right) + \tfrac{1}{2}(x-\mu_Q)^T\Sigma_Q^{-1}(x-\mu_Q)\right] - \tfrac{1}{2}\log\left((2\pi e)^N|\Sigma_P|\right) \\
&= \tfrac{1}{2}\log\left((2\pi)^N|\Sigma_Q|\right) + E_P\left[\tfrac{1}{2}(x-\mu_Q)^T\Sigma_Q^{-1}(x-\mu_Q)\right] - \tfrac{1}{2}\log\left((2\pi e)^N|\Sigma_P|\right) \\
&= \tfrac{1}{2}(\mu_P-\mu_Q)^T\Sigma_Q^{-1}(\mu_P-\mu_Q) + \tfrac{1}{2}Tr\left[\Sigma_Q^{-1}\Sigma_P\right] - \tfrac{1}{2}\log\left(\tfrac{(2\pi e)^N|\Sigma_P|}{(2\pi)^N|\Sigma_Q|}\right) \\
&= \tfrac{1}{2}(\mu_Q-\mu_P)^T\Sigma_Q^{-1}(\mu_Q-\mu_P) + \tfrac{1}{2}Tr\left[\Sigma_P\Sigma_Q^{-1}\right] - \tfrac{N}{2} - \tfrac{1}{2}\log|\Sigma_P\Sigma_Q^{-1}| \\
&= \tfrac{1}{2}\left[(\mu_Q-\mu_P)^T\Sigma_Q^{-1}(\mu_Q-\mu_P) + Tr\left[\Sigma_P\Sigma_Q^{-1} - I\right] - \log|\Sigma_P\Sigma_Q^{-1}|\right]
\end{aligned}
$$

D.3 Automatic determination of synthesis costs

The framework for automatically determining synthesis costs from given classes involve the above information-theoretic metrics. Substitution costs are based on Kullback-Leibler divergence which describes the penalty of using q in lieu of p. For concatenation costs, it is hypothesized that concatenations are least discernible at boundaries of greatest spectral change. Since mutual information is low when knowledge of one random variable provides little knowledge of the other, mutual information of spectral observations across boundaries is used for deriving concatenation costs.