MINING LINGUISTIC TONE PATTERNS USING FUNDAMENTAL FREQUENCY TIME-SERIES DATA

A Dissertation
submitted to the Faculty of the
Graduate School of Arts and Sciences
of Georgetown University
in partial fulfillment of the requirements for the
degree of
Doctor of Philosophy
in Linguistics

By

Shuo Zhang, M.S.

Washington, DC
June 27, 2017
MINING LINGUISTIC TONE PATTERNS USING FUNDAMENTAL FREQUENCY TIME-SERIES DATA

Shuo Zhang, M.S.

Dissertation Advisors: Amir Zeldes, Ph.D and Elizabeth Zsiga, Ph.D

ABSTRACT

With the rapid advancement in computing powers, recent years have seen the availability of large scale corpora of speech audio data, and within it, fundamental frequency \( f_0 \) time-series data of speech prosody. However, the wealth of this \( f_0 \) data is yet to be mined for knowledge that has many potential theoretical implications and practical applications in prosody-related tasks. Due to the nature of speech prosody data, Speech Prosody Mining (SPM) in a large prosody corpus faces classic time-series data mining challenges such as high dimensionality and high time complexity in distance computation (e.g., Dynamic Time Warping). Meanwhile, the analysis and understanding of speech prosody subsequence patterns demand novel analytical methods that leverage a variety of algorithms and data structures in the computational linguistics and computer science toolkits, prompting us to develop creative solutions in order to extract meaning in large prosody databases.

In this dissertation, we conceptualize SPM in a time-series data mining framework by focusing on a specific task in speech prosody: the analysis and machine learning of Mandarin tones. The dissertation is divided into five parts, each further divided into several chapters. In Part I, we review the necessary background and previous works related to the production, perception, and modeling of Mandarin tones. In Part II, we report the data collection used in this work, and we describe the speech processing and data preprocessing steps in detail.
Part III and IV comprise the core segments of the dissertation, where we develop novel methods for mining tone N-gram data. In Part III, we investigate the use of time-series symbolic representation for computing time-series similarity in the speech prosody domain. In Part IV, we first show how to improve a state-of-the-art motif discovery algorithm to produce more meaningful rankings in the retrieval of previously unknown tone N-gram patterns. In the next chapter, we investigate the most exciting problem at the heart of tone modeling: how well can we predict the tone N-gram contour shape types in spontaneous speech by using a variety of features from various linguistic domains, such as syntax, morphology, discourse, and phonology? The results shed light on the nature of how these factors contribute to the realization of speech prosody in tone production from an information theoretic perspective. In the final part, we describe applications of these methods, including generalization to other tone languages and developing softwares for the retrieval and analysis of speech prosody. Finally, we discuss the extension of the current work to a general framework of corpus-based large-scale intonation analysis based on the research derived from this dissertation.

INDEX WORDS: Mandarin, tone, time-series data mining, machine learning, prosody
Acknowledgments

I would like to thank my advisor, Professor Amir Zeldes and Professor Elizabeth Zsiga, for being fantastic mentors over my PhD years. I would also like to thank Dr. Zeldes, Dr. Zsiga, and Dr. George Wilson for their guidance as my committee members. In my parallel careers in computational linguistics and music information retrieval (as reflected in this dissertation), I owe a great deal to my supervisor and colleagues at the Music Technology Group, Department of Information and Communication Technologies (DTIC), Universitat Pompeu Fabra, Barcelona, Spain, especially Professor Xavier Serra, Sankalp Gulati, and Rafael Caro Repetto. Finally, I could not have finished my PhD without the support of my family and my life partner, Mia.
# Table of Contents

## I Introduction and Background

1 Introduction .......................................................... 2  
   1.1 Introduction .................................................. 2  
   1.2 Research questions and goals .............................. 2  
   1.3 Organization of the dissertation ......................... 4  
   1.4 Contribution of this dissertation ....................... 5  

2 Background and Previous Works ................................. 8  
   2.1 How important is tone? ..................................... 8  
   2.2 Why is tone recognition hard? ............................ 10  
   2.3 Supervised learning of tones ............................... 29  
   2.4 Unsupervised learning of tones ........................... 37  
   2.5 Speech prosody mining using time-series mining techniques ... 41  

## II Data Collection and Speech Processing ...................... 46  

3 Data Collection .................................................. 47  
   3.1 Read speech data set .................................... 47  
   3.2 Newscast speech data sets ................................. 48  

4 Speech Processing ................................................ 53  
   4.1 Fundamental frequency ($f_0$) estimation .................. 53  
   4.2 Speech Processing ............................................ 53  

## III Computing Similarity for Speech Prosody Time-series ...... 68  

5 Time-series Representation and Distance Measures ............ 69  
   5.1 Prosodic modeling feature representation .................. 69  
   5.2 Time-series symbolic representation ....................... 72  
   5.3 Time-series normalization ................................... 75  
   5.4 Distance measure ............................................. 77  

6 Data Mining Mandarin Tones Using Time-series Mining Techniques ... 82  
   6.1 Overview .................................................... 82  
   6.2 Related work ................................................ 83
List of Figures

2.1 Mandarin 4 tones canonical contour shapes .............................................. 12
2.2 Experimentally controlled production of all Rising tones in statement (solid lines) vs question sentences with varying focus position .......................... 12
2.3 Experimentally controlled production of all High tones in statement (solid lines) vs question sentences with varying focus position ......................... 12
2.4 Examples of $f_0$ contours generated by the qTA model with varying values of $m$, $b$, and $\lambda$ ................................................................. 24
2.5 PENTATrainer 2 annotation scheme ....................................................... 26
2.6 Three-phase segmentation of Second Tone ............................................... 37
2.7 Comparison of different clustering algorithms with varying number of clusters and clustering accuracy ................................................................. 41
2.8 Cluster separation of pitch target slope (x) vs height (y) in Mandarin read speech data set ................................................................. 42
4.1 Sample trimmed pitch track and manual check of fundamental frequency $f_0$ ground truth ................................................................. 54
4.2 Long pitch time-series of the entire CMN corpus after naive trimming (normalized) ................................................................. 55
4.3 Long pitch time-series of the entire CMN corpus after more sophisticated trimming (normalized) ................................................................. 56
4.4 A sample pitch track with bimodal distribution (spurious) ......................... 57
4.5 Distribution of a sample bimodal pitch track ............................................. 57
4.6 Difference values between pairs of consecutive points in identifying spurious pitch segments ................................................................. 58
4.7 Original pitch value distribution of a speaker (XII) .................................. 59
4.8 Pitch value distribution of a speaker (XII) after first applying outlier pruning ................................................................. 60
4.9 Sample trimmed and interpolated pitch track ............................................. 62
4.10 Sample result of how smoothing can change the profile of a tone pitch curve ................................................................. 63
4.11 Sample downsampled pitch track with resampling method ....................... 64
4.12 Sample downsampled pitch track with quasi-equidistant method .................. 65
4.13 Sample downsampled pitch track with averaging method ......................... 65
4.14 Speech processing and data preprocessing pipeline .................................. 66
5.1 Types of time-series data representations ................................................. 72
5.2 Piecewise Aggregate Approximation ........................................ 75
5.3 Symbolic Aggregate Approximation ........................................ 76
5.4 Euclidean distance vs. Dynamic Time Warping: example ............. 79
5.5 An algorithm that uses a lower bounding distance measure to speed up the sequential scan search for the query Q ...................... 80
5.6 Pitch contours of three melodic phrases (P1, P2, P3) .................... 81
6.1 KNN classification accuracy depending on $w, a$ ...................... 86
6.2 Average clustering accuracy for 1920 Mandarin tones (%) from 5 iterations ................................................................. 92
6.3 Kmeans clustering objective function by number of iteration ........ 93
6.4 SAX-MINDIST (left) and $f_0$-Euclidean (right) Distance matrix of 1920 Mandarin tones sorted by tone category ..................... 95
6.5 Precision recall curve ............................................................. 96
7.1 Obvious brute force algorithm for motif discovery of 1-Motif .......... 106
7.2 MK algorithm for motif discovery of 1-Motif .............................. 107
7.3 Top five motif pairs (spurious) from the CMN corpus .................. 109
7.4 Five real motif pairs from the CMN corpus ............................... 113
7.5 Shapes and their corresponding “time” series with different levels of complexity ................................................................. 114
7.6 A top-ranked simple (linear) spurious and artificial motif cluster from the CMN corpus .......................................................... 116
7.7 A real motif cluster from the CMN corpus .................................. 116
7.8 Distance distribution of 50,000 randomly sampled pairwise distances in the B300p data set ................................................. 119
7.9 Distance distribution of 50,000 randomly sampled pairwise distances in the B200p data set ................................................. 119
7.10 Distance distribution of 50,000 randomly sampled pairwise distances in the B100p data set ................................................. 119
7.11 Experiments on the value of $X$ and motif cluster quality measures in the B100p data set ...................................................... 120
7.12 Experiments on the value of $X$ and motif cluster quality measures in the B200p data set ...................................................... 123
7.13 Experiments on the value of $X$ and motif cluster quality measures in the B300p data set ...................................................... 123
7.14 A q(uasi)-linear motif cluster from the B100p data set ............... 127
7.15 Stacked bar chart showing the distribution and correlation between motif cluster ranking and classes in the T200p data set .......... 127
7.16 Stacked bar chart showing the distribution and correlation between motif cluster ranking and classes in the T300p data set .......... 128
7.17 Stacked bar chart showing the distribution and correlation between motif cluster ranking and classes in the B200p data set .......... 128
7.18 Stacked bar chart showing the distribution and correlation between motif cluster ranking and classes in the B100p data set
7.19 Example of BWK complexity on z-normalized time-series pair (inconsistent with intuition)
7.20 Example of BWK complexity on z-normalized time-series pair (consistent with intuition)
7.21 Time-series subsequences with different slopes
7.22 Time-series subsequences with different slopes: generalization
7.23 Correlation between rank and complexity in the T300p data set
7.24 Correlation between rank and complexity in the T200p data set
7.25 Correlation between rank and complexity in the B200p data set
7.26 Correlation between rank and complexity in the B100p data set
7.27 Distribution of LSSE complexity scores among the three motif cluster classes
7.28 Boxplot of LSSE complexity scores among the three motif cluster classes
7.29 Distribution of TLC scores among the three motif cluster classes
7.30 Correlation between rank and TLC in the B200p data set
7.31 Correlation between rank and TLC in the B100p data set
7.32 Correlation between rank and TLC in the T200p data set
7.33 Correlation between rank and TLC in the t300p data set
7.34 Correlation between LSSE complexity and TLC in pooled data set
7.35 Average complexity (BWK) distribution between three motif classes in B100p data set
7.36 Iterative Pruning Framework (IPF)
7.37 Motif discovery MAP scores using different methods proposed
8.1 Evolution of clustering coefficients in trigram 3-1-4
8.2 Sample network community structure (Full)
8.3 Sample network community structure (zoomed in)
8.4 Sample network community structure with no outliers
8.5 Feature weights for all unigram data sets
8.6 Feature weights for unigram data sets excluding starting pitch and ending pitch
8.7 Histogram of number of output shape classes in unigram data sets
8.8 Feature weights for all bigram data sets
8.9 Feature weights for bigram data sets excluding sentence position and ending pitch
8.10 Histogram of number of output shape classes in bigram data sets
8.11 Feature weights for all trigram data sets
8.12 Feature weights for trigram data sets excluding starting pitch and ending pitch
8.13 Histogram of number of output shape classes in trigram data sets
8.14 Unigram data sets classification accuracy
8.15 Bigram data sets classification accuracy .................. 191
8.16 Trigram data sets classification accuracy .................. 192
9.1 Mean and std error of mean pitches in Thai ................ 199
9.2 SPQR web tool configuration page screenshot ............... 203
9.3 SPQR web tool QBC results page (partial) screenshot ......... 204
## List of Tables

2.1 Functional load of tones vs other contrasts in Mandarin .......................... 9
2.2 Tone recognition using focus .................................................. 35
2.3 Tone recognition with unsupervised and supervised learning ................. 40
3.1 Three female speakers in the Thai data set .................................... 50
4.1 Data sets overview .................................................................. 67
6.1 K-Nearest Neighbor tone classification results .................................. 88
6.2 Decision tree classification results in Read speech data set ................. 89
6.3 Decision tree classification results in CMN data set .......................... 90
6.4 QBC results MAP scores ....................................................... 98
7.1 Euclidean distance matrix for the geometric figures data set ................. 115
7.2 Number of motif clusters discovered varies with $len_{TS}$ and $X$ ........ 120
7.3 Motif class datasets annotated .................................................. 126
7.4 Features used in classification .................................................... 144
7.5 Classification tasks overview ..................................................... 144
7.6 Classification accuracy B200p (SVM/Decision Tree) ............................. 151
7.7 Classification accuracy B100p (SVM/Decision Tree) ............................. 151
7.8 Classification accuracy T300p (SVM/Decision Tree) ............................. 152
7.9 Classification accuracy T200p (SVM/Decision Tree) ............................. 152
7.10 Classification accuracy pooled data set (SVM/Decision Tree) .............. 152
7.11 Classification accuracy pooled 2C2 data set .................................. 154
7.12 Motif classification results using different parameter settings .............. 155
8.1 Feature set overview ................................................................. 169
8.2 Feature ablation experimental results in data 1-3-4 ............................ 175
8.3 Feature ablation experimental results in data 3-1-4 ............................ 176
8.4 Feature ablation experimental results in data 3-2-2 ............................ 176
8.5 Feature ablation experimental results in data 3-4-2 ............................ 177
8.6 Feature ablation experimental results in data 4-2-1 ............................ 178
8.7 Feature strength from ablation experiments ..................................... 179
8.8 Feature strength from feature weight analysis using all unigram, bigram
and trigram data (un-ordered within each column) ................................ 188
8.9 Comparison of unigram, bigram and trigram classifier performance .... 190
9.1 QBC results MAP scores for Thai ............................................... 201
Part I

Introduction and Background
Chapter 1

Introduction

1.1 Introduction

With the rapid advancement in computing power, recent years have seen the availability of large-scale corpora of speech audio data, and within it, fundamental frequency ($f_0$) time-series data of speech prosody. However, the wealth of this $f_0$ data is yet to be mined for knowledge that has potential theoretical implications and practical applications in prosody-related tasks. Due to the nature of speech prosody data, Speech Prosody Mining (SPM) in a large prosody corpus faces classic time-series data mining challenges such as high dimensionality and high time complexity in distance computation (e.g., Dynamic Time Warping). In the meantime, the analysis and understanding of speech prosody subsequence patterns demand novel analytical methods that leverage a variety of algorithms and data structures in the computational linguistics and computer science toolkits. This prompts us to develop creative solutions in order to extract meaning in large prosody databases.

1.2 Research questions and goals

We purposefully limit the scope of the current project to tone tasks due to its importance, clearly defined evaluation, and the difficulty and complexity of this task in its
own right. In the primary investigations of the dissertation, we will focus on Mandarin tones\(^1\) in order to develop methodologies best adapted to the data mining of tones. In the latter part of the dissertation, we extend the evaluation and application to other tone languages (Thai). Finally, we propose a general framework for working with SPM tasks.

We conceptualize the dissertation within a (time-series) data mining research framework by addressing important issues and tasks in applying time-series data mining to the speech prosody domain. The principal research questions can be divided into two parts:

1. What is the most effective set of time-series data mining methodologies for the computation of speech prosody time-series data? This part is concerned with finding the optimal set of time-series representations, distance measures and other methods for computing time-series similarity in the speech prosody domain (as evaluated on supervised and unsupervised tone learning tasks).

2. How can we develop methods for analyzing tone N-gram patterns in a large database? This research question can be further divided into three parts:

   (a) How can we develop methods to extract previously unknown patterns from a tone N-gram database? (from tone contour shapes to tone categories\(^2\))

\(^1\)In this dissertation, we will use Mandarin to refer to standard Mandarin Chinese spoken in mainland China, as opposed to Taiwanese Mandarin.

\(^2\)In this part, we are first and foremost interested in finding and grouping similar tone N-gram contour shapes, regardless of their categorical memberships. This task is designed to complement the next task, where we start from a given tone N-gram category and explore its actual realized contour shape types in spontaneous speech, i.e., “from categories to contour shapes”.

3
(b) Given a specific tone N-gram category, how can we develop methods for automatically characterizing the different types of tone contour shapes realized in spontaneous speech data? (from categories to contour shapes)

(c) Given (2b), how well can we predict tone contour shape types realized in spontaneous speech, using features/predictors from relevant linguistic domains such as syntax, phonology, morphology, and discourse? What can we learn from this regarding the source of variability problem, as well as the psycholinguistic computational modeling of tone production and perception?

1.3 Organization of the dissertation

The dissertation is divided into five parts, each further divided into several chapters. In Part I, we review the necessary background and previous works related to the production, perception, and modeling of Mandarin tones. In Part II, we report the data collection used in this work, and we describe the speech processing and data preprocessing steps in detail.

Part III and IV comprise the core segments of the dissertation, where we develop novel methods for mining tone unigram and N-gram data to address the research questions proposed above. In Part III we investigate the use of time-series symbolic representation for improving the unsupervised learning of tones. In Part IV, we first show how to improve a state-of-the-art motif discovery algorithm to produce more meaningful rankings in the retrieval of previously unknown tone N-gram patterns. In the next chapter, we investigate the most exciting problem at the heart of tone modeling: how well can we predict the tone N-gram contour shape types in spontaneous speech by using a variety of features from various linguistic domains, such as
syntax, morphology, discourse, and phonology? The results shed light on the nature of how these factors contribute to the realization of speech prosody in tone production from an information theoretic perspective. Moreover, in both chapters of Part IV, we will develop creative solutions to investigate these problems, drawing from methodologies in areas such as motif discovery, network (graph) analysis, machine learning, and information retrieval. In the final part, we describe applications of these methods developed in previous parts, including generalization to other tone languages and developing software for the retrieval and analysis of speech prosody. In the end we discuss the extension of the current work to a general framework of corpus-based large-scale intonation analysis based on the research derived from this dissertation.

1.4 Contribution of this dissertation

It is important to note that since its inception, we have always had a clear intent to carry out the dissertation project within a main methodology framework of data mining, instead of tone recognition in the context of speech recognition. There are several reasons for this design. First, we note that previous works have invested much effort into Mandarin tone recognition, especially the supervised learning of tones for improving tone recognition accuracy. In contrast, the main goal of the current data mining framework is knowledge discovery in large speech prosody time-series data sets (with indirect applications in improving tone recognition). In other words, we are interested in the how and why in the understanding and analysis of tone N-gram patterns, instead of directly focusing on obtaining better tone recognition accuracy. Second, tone recognition research is best combined with segmental recognition that

\footnote{Despite this statement, we must recognize that the current study does in fact benefit from the domain knowledge of tone recognition, and its end result could be eventually useful for improving applications such as tone recognition.}
targets speech recognition as its final goal, as exemplified in Chang et al. [CZD+00]. In contrast, the current project targets speech intonation pattern discovery by focusing on time-series data from only the $F_0$ dimension. Third, whereas the machine learning and automatic recognition of tones have been studied in a quite exhaustive manner (see Surendran dissertation [Sur07]), the time-series data mining approach is a novel approach in speech prosody research, which is the original contribution of the current project. Fourth, we aim to develop methods not only applicable to Mandarin tone time-series data, but generalizable to other tone languages and other SPM tasks. These features of our approach are also reflected in our careful and precise formulation of the title of the dissertation.

In sum, the contribution of this dissertation is two-fold. First, we aim to contribute a set of methodologies associated with the data mining of speech prosody, as the analysis of speech prosody and tones has not been previously carried out within a time-series mining framework (to the best of our knowledge). Second, the size of the data being analyzed is also much bigger than those found in typical speech prosody analysis works [Sur07]. This allows us to discover novel and robust knowledge while developing new methods to address the challenges posed by mining this data. As data mining is being used ubiquitously to advance human knowledge in virtually all scientific fields and application domains, this approach harnesses the power of speech prosody big data, and contributes a set of novel technologies for researchers in speech technology, speech prosody, and linguistics. Finally, it is worth pointing out that to the best of our knowledge, most proposed tasks carried out in this dissertation are novel (especially in the specific ways they are formulated in this domain and the methodologies associated) and have not been attempted in previous works by others. This simultaneously gives us the challenge to develop novel methods to solve these
tasks, and the *opportunity* for us to consider the implications of the results obtained from these novel experiments.
Chapter 2

Background and Previous Works

This dissertation is concerned with the understanding and analysis of speech prosody time-series - especially focusing on Mandarin tones. In this chapter, we provide the general background and literature review regarding the significance and challenges in the computational modeling of tones.

2.1 How important is tone?

Until recently, the state-of-the-art speech recognition of tone languages typically discards tone information altogether. This is mainly attributed to the high error rate of tone recognition algorithms that supersedes the benefit of including tone information. Recent development in tone recognition, however, reveals the importance of tone information in speech recognition from a information theoretic point of view. In Surendran’s dissertation [SL04], the functional load of tones is found to be at least as high as vowels while lower than the consonants. Here, the functional load of tones (FL) is defined as the information we lose if the phonological system loses the contrast posed by tones:

$$FL(\text{tone}) = \frac{H(M_u) - H(M_u\text{-tone})}{H(M_u)}$$  \hspace{1cm} (2.1)

where $M_u$ is a sequence of units of type U in Mandarin, U being either a syllable or word in the entropy function $H(.)$ calculation. $M_u\text{-tone}$ refers to such a sequence
without tone contrast, and \(H(.)\) is the standard entropy function of the system (i.e., a well defined \textit{language}). FL is defined as the functional load of tones. Intuitively, it characterizes how much information the system loses if the tone contrast is lost, therefore how important a contrast is within a phonological system. Table 2.1 shows the functional load of tones versus other (segmental) contrasts in Mandarin.

\begin{table}[h]
\centering
\caption{Functional load of tones vs other contrasts in Mandarin.}
\begin{tabular}{|c|c|c|}
\hline
\textbf{x} & \textbf{FL_{syll}(x)} & \textbf{FL_{word}(x)} \\
\hline
Consonants & 0.235 & 0.081 \\
Tones & 0.108 & 0.021 \\
Vowels & 0.091 & 0.022 \\
Stops & 0.029 & 0.006 \\
Fricatives & 0.021 & 0.005 \\
\hline
Place & 0.065 & 0.014 \\
Manner & 0.034 & 0.006 \\
Aspiration & 0.002 & 0.0003 \\
\hline
\end{tabular}
\end{table}

This analysis captures the importance of tones in Mandarin speech recognition systems. However, it should also be interpreted critically. First, it has the implicit assumption of taking a pure phonological point of view while ignoring all other information that can be used to recognize words and syllables, such as sequence-based language models in speech recognition (which is why Mandarin speech recognition can perform above 90\% accuracy without any tone information \cite{CZD00}). In this regard, Surendran’s \cite{SL04} analysis fails to capture the importance of tones viewed from a broader perspective of information theory and entropy.

Second, results like Chang et al.’s \cite{CZD00} also call for an analysis of how important tone is for Mandarin speech recognition in humans. It is often an implicit assumption in tone recognition literature that humans are always able to perform with reasonably high accuracy in tone recognition \cite{HZ99}, even when contextual tonal,
segmental, and other information is unavailable. However, speech experiments have revealed that native Mandarin speakers could perform below chance in isolated syllable tone recognition tasks when additional information usually available in speech is removed [Xu94], depending on the specific experimental conditions. This is also supported by additional experimental evidence that humans are able to perform with greater than 90% accuracy in speech understanding tasks with tone information removed in Mandarin (i.e., monotone $f_0$ is imposed synthetically) [PXW10].

Therefore, it is worthwhile to investigate this problem from an information theory point of view at a broader perspective (which from the above discussion, would reveal that it is much less important than Surendran [SL04] has suggested). The implication of such an analysis would be that, we cannot expect a machine to perform perfectly on isolated syllable tone recognition when humans cannot do it in the first place. Meanwhile, it suggests the importance to better understand acoustic and non-acoustic contextual information beyond the information present in the acoustic signal of isolated syllables.

2.2 Why is tone recognition hard?

In the canonical forms of Mandarin tone system, four tone categories are present: high-level (High tone), rising (Rising tone), low-dipping (Low tone) and falling (Falling tone). Following convention, these are also referred to as tone 1, 2, 3, and 4. Figure 2.1 shows the canonical contours of four Mandarin tones. In running speech, however, the contour shapes often become much distorted from canonical shapes, making it difficult to identify correct tone categories. Therefore, the challenge of tone recognition is to identify sources of variability and to exploit this knowledge to restore the underlying tone category. In spontaneous speech, there are many local and broader factors that
play a role in producing the final tone contour shapes. These factors are usually rather
convoluted and confounded and are therefore hard to identify. It requires carefully
designed control experiments to reveal the effect and function of each contributing
factor.

As an example, Figure 2.2 shows how a tone-controlled production experiment
reveals variability of tone contour shapes in statement vs. question sentences with
varying focus position (data from Liu et al. [LSX06]). Looking at this simple example,
which is far less complex than real spontaneous speech data, we can begin to see the
reason why tone recognition is hard. First, we can see clearly that even though these
are all Rising tones, most tones in this sentence have a rather flat shape, resembling
the High tone. This is in contrast with Figure 2.3, where a sentence is spoken with
all high tones. Second, the variability of the tone shapes is dependent on the sentence
focus and modality conditions. Ideally, these factors must also be taken into account
in the recognition of tones from real spontaneous speech, where this type of uniform
tone combination is highly unlikely.

In this section, we review previous works on the source of variability problem from
both speech production and prosodic modeling perspective.

2.2.1 SOURCES OF VARIABILITY: EVIDENCE FROM SPEECH PERCEPTION AND
PRODUCTION

Gauthier et al. [GSX07] identified two major sources for the extensive overlaps among
the contour shapes of different tone categories in running speech. The first is the dif-
fERENCE in the pitch range of individual speakers and the second is the variability
introduced by tonal context in connected speech [Xu97]. In tone recognition, the
speaker difference is usually removed by normalization. The tonal context, as men-
Figure 2.1: Mandarin 4 tones canonical contour shapes. (Adapted from Levow [Lev06a]).

Figure 2.2: Experimentally controlled production of all Rising tones in statement (solid lines) vs question sentences with varying focus position. (Adapted from Liu et al. [LSX06]).

Figure 2.3: Experimentally controlled production of all High tones in statement (solid lines) vs question sentences with varying focus position. (Adapted from Liu et al. [LSX06]).
tioned above, is where the majority of the research effort has been devoted to, with a focus on identifying and isolating sources of variability.

Recent works on speech production and perception have made substantial progress on identifying sources of variability in tone production, by carrying out experiments with carefully controlled conditions and designed data sets [XPo14]. Such knowledge were also exploited to improve tone recognition accuracy [SLX05, LSX06]. These developments have led to the context-dependent (CI) models.

In particular, experimental works have identified two levels of variability in continuous tone production: First, a local context refers to the distortion of tones from its canonical shapes due to the co-articulation with adjacent tones. Second, a broad context refers to the further modification of tone contours on the intonation level (i.e., on a larger prosodic unit than the syllable) [WX11a]. Examples of factors related to the broad context include topic, focus, and modality/mood of a sentence. Here, we review works on both the local and the broad context in tone production, and when possible, their application in the machine learning of tones.

**Local context**

The local context of tone production is concerned with the behavior of tone contour $f_0$ trajectories with regard to its immediate environment, i.e., adjacent syllables. To study only the local context, experiments are often conducted with two principles in mind: first, to examine the effect of neighboring tones, speech production tasks are designed to reflect all different combinations of tones as target words. Second, to minimize the effect of higher level intonation, participants are asked to read the target words embedded in carrier sentences with careful speech and neutral focus. Several important results emerge from these studies of the local context.
(1) **Local “conflicting” context distorts tone contour shapes in incompatible environments.** Xu [Xu94] studied the variability of the tone contour shapes due to coarticulation, depending on the nature of the tonal context. By grouping tone environments into “compatible” vs “conflicting” contexts, it was observed that in a context where adjacent tonal values agree (a “compatible” context, such as when the High tone “HH” ending with “H” is followed by a High-Falling tone “HL” beginning also with “H”)\(^1\), the deviation was relatively small. In a context where adjacent tonal values disagree (a “conflicting” context, such as when the said High tone is followed by a Low-Rising tone “LH” starting with “L”), the deviation was much greater, sometimes even to the extent of changing the direction of a dynamic tone.\(^2\)

The author also examined the perception of co-articulated tones when tones are presented out of context in isolation. The results suggest that identification of tones produced in the compatible context was highly accurate with or without the original tonal context. Tone identification for those produced in a conflicting context remained accurate only when the tones were presented with the original context. Without the original context, i.e., in isolation or in artificially altered tonal context, tone identification accuracy dropped below chance. As discussed above, this result provides important counter-evidence to the “myth” in tone recognition literature that human listeners are always able to recognize tones with high accuracy, even when they’re distorted and in isolation.

(2) **Carry-over effect is greater than anticipatory effect in tone co-articulation.** In a pair of consecutively articulated syllables, does the tone shape

---

\(^1\)The “H” (High) and “L” (Low) refer to phonological compositional representation of tones, where for example, a rising tone is represented as “LH” with two pitch targets: low and high.

\(^2\)The authors refer to tones with moving targets as “dynamic tone” (such as a low-rising tone), vs. static tone, where the tone has stable pitch targets (such as a high tone).
of the first syllable have more impact on the tone shape of the second syllable, or vice versa? Xu [Xu97] paid attention to the distortion of tones due to anticipatory and carry-over effects by examining the bigrams of $f_0$ contours in pairwise syllable sequences. Anticipatory effect is defined as the change of the tone contour shapes in the current syllable in anticipation of the next. Carryover effect is seen as the change of tone contour shapes in the current syllable due to the prolonged effect of the preceding tone. Using balanced nonsense sequences produced in different carrier sentences with balanced tonal structures, this study establishes a baseline for local contextual tonal variation in Mandarin. It is found that anticipatory and carry-over tonal influences differ both in magnitude and in nature. Carry-over effects are mostly assimilatory: the starting $f_0$ of a tone is assimilated to the offset value of a previous tone. Anticipatory effects, on the other hand, are mostly dissimilatory: a low onset value of a tone raises the maximum $f_0$ value of a preceding tone. While the magnitude of the carry-over effect is large, anticipatory effects are relatively small. This conclusion has been cited many times subsequently by tone recognition researchers, whose own data analysis also showed support for this asymmetry over and over again [Sur07, ZHN04, Lev05]. There are also many machine learning approaches that take this effect into consideration in tone modeling.

(3) **Physiological constraints of tone co-articulation.** Xu et al. [XS02] studied the maximum speed of pitch change in human speech production, which contributes to the understanding of the underlying mechanism for tone co-articulation and its implication for tone modeling. In this experiment, subjects (native speakers of English and Mandarin) produced rapid successions of pitch shifts by imitating synthesized model pitch undulation patterns. Results show that excursion time is nearly twice as long as response time in completing a pitch shift. Comparisons of this experimental data with real speech data suggest that the maximum speed of pitch
change is often approached in speech, and the role of physiological constraints in tone production is greater than previously appreciated.

(4) $f_0$ peak delay. Fundamental frequency ($f_0$) peak delay refers to the phenomenon that $f_0$ peaks sometimes occur after the syllable it is associated with, either lexically or prosodically. Xu [Xu01] investigated peak delay and its relationship with tone, tonal context, and speech rate. Depending on speech rate, the author found that peak delay occurred regularly in Rising (R) tones and variably in High (H) tones. In general, peak delay occurs when there is a sharp $f_0$ rise near the end of a syllable, regardless of the cause of the rise. The author concluded that much of the variability in the shape and alignment of $f_0$ contours in Mandarin is attributable to the interaction of underlying pitch targets with tonal contexts and articulatory constraints, rather than due to actual misalignment between underlying pitch units and segmental units.

Broad context

This section discusses the broad context of the tone variability phenomenon.

(1) Focus in the temporal domain. Xu et al. [XXS+04] showed that focus not only affects the syllable under focus, but also extensively affects the pitch range of non-focused regions in a sentence, suggesting a wide effect of focus on the temporal domain. Results from this study show that in a declarative sentence, focus is realized not only by expanding the pitch range of the focused item, but also by compressing the pitch range of post-focus items, and possibly requiring that the pitch range of pre-focus items remain neutral. The authors proposed that the domain of a single, narrow focus consists of three temporal zones (pre-focus, on-focus, post-focus), with distinct pitch range adjustment for each. This proposal has received positive support when it was later applied to a machine learning model that improved tone recognition.
accuracy by incorporating focus information [SLX05]. Liu et al. [LSX06] also used focus as an effective input feature to a decision-tree based classifier to predict the modality of a sentence from the prosodic domain (question vs. statement).

In Xu [Xu99], the author further examined how the lexical tone and the focus contribute to the formation and alignment of \( f_0 \) contours, using short Mandarin sentences consisting of five syllables with varying tones on the middle three syllables. The sentences were produced with four different focus patterns: focus on the first word, second word, last word, or with no narrow focus. The results indicate that while the lexical tone is the most important determining factor for the local \( f_0 \) contour of the syllable, the focus extensively modulates the global shape of the \( f_0 \) curve, which in turn affects the height and shape of local contours. Moreover, despite extensive variations in shape and height, the \( f_0 \) contour of a tone remains closely aligned with the associated syllable.

(2) **Long-term \( f_0 \) variations.** Many tone recognition algorithms incorporate features that encode the relative pitch heights of the tones [Lev05, Sur07], in conjunction with the features that encode the contour shapes. In this regard, the pitch height of tones in longer temporal units of spontaneous speech (e.g., a sentence) must be normalized not only to account for individual speaker differences, but also to compensate for the long-term \( f_0 \) pitch movements due to factors such as down-drift and other pragmatic functions such as focus and topic.

Wang et al. [WX11a] reports an experimental investigation of the prosodic encoding of topic and focus in Mandarin by examining disyllabic subject nouns elicited in four discourse contexts. They also looked at how prosodic effects of topic and focus differ from each other and how they interact with sentence length, downstep and newness to determine sentence-initial \( f_0 \) variations. Sixty short discourses were
recorded with variable focus, topic level, newness, downstep, and sentence length conditions by six native speakers. The important conclusions include:

(a) The difference between topic and focus is that focus raises on-focus $f_0$ and lowers post-focus $f_0$, but topic raises the $f_0$ register at the beginning of the sentence while allowing $f_0$ to drop gradually afterward;

(b) Topic has higher pitch register in isolated and discourse-initial sentences than in non-initial sentences;

(c) Longer sentences have higher sentence-initial $f_0$s than shorter sentences, but the differences are small in magnitude and are independent of topic and focus;

(d) The effect of downstep is independent of topic and focus, but is large in magnitude and accounts for a significant amount of the $f_0$ declination in a sentence;

(e) Newness has no $f_0$ manifestations independent of other factors;

(f) The effects of topic, focus, downstep and sentence length are largely cumulative.

As with other experimental findings reviewed in this chapter, the crucial question is how can we incorporate these fine-grained differences in machine learning algorithms to improve tone recognition accuracy. As will be discussed later, current methods tend to compensate for the global pitch shift patterns in a catch-all framework rather than distinguishing different communicative functions.

2.2.2 Prosodic modeling

Research in prosodic modeling provides evidences and often validations to the results obtained from speech production experiments mentioned above. It is also closely related to tone recognition, but with a different goal. The goal of prosodic modeling is to analyze prosodic patterns and synthesize speech prosody that is as natural as possible, while requiring less resources in storage, computation, and supervision. These models are used to generate $f_0$ contours for general speech synthesis. As such,
typically, prosodic models are evaluated by means of its capability to (1) re-synthesize speech melodies that closely approximate the original data (training data); (2) predicatively synthesize / generate speech melodies of unseen data given annotated sentence and syllable conditions (i.e., input features).

Meanwhile, on a different level, prosodic modeling is also associated with the representation of speech intonation in suprasegmental phonology, and the underlying mechanism for the production and perception of speech prosody [Xu11]. While this dissertation is not primarily concerned with either the generation of $f_0$ contours for speech synthesis or the phonological and psycholinguistic theory of tone production and perception, there are aspects of prosodic modeling that can shed light on the computational analysis of tones. Specifically, the representation of intonation is directly relevant to the choice of feature vectors we use to mine tone data. Moreover, the modeling of the $f_0$ generation process can also provide clues as to how tone variability occurs in real-time spontaneous speech.

In the next two sections, I follow the convention in speech prosodic modeling [SX02] and divide the review of relevant works into two parts: phonology-based and phonetic-based models. One of the phonetic-based models of particular interest for Mandarin is the quantitative Target Approximation model [Xu99], which I will discuss separately. This model was developed specifically with Mandarin tones in mind, although it has been also generalized to be a language independent model.

**PHONOLOGY-BASED MODELS**

Phonological models are concerned with the universal organization and underlying representations of intonation, with implications on the theory of speech production.

---

3There has been theoretical discussions and debates [Xu11, RW15] on which type of models is a more truthful representation of the speech production mechanism, which this dissertation is not concerned with.
and perception. Complex intonation patterns are compressed into a set of highly succinct and abstract vocabulary with wide coverage [SX02]. In this framework, with the general notion of tonal targets, production of tones can be thought of as an interpolation between the various targets, and perception of tones is understood as an attempt to identify these targets.

The most influential example of the phonological representation of tone and intonation is the Autosegmental-Metrical (AM) intonational phonology and Pierrehumbert’s model for American English [Pie80]. Ladd [Lad96] states four principles of the AM approach to intonation: (1) Linearity of tonal structure; (2) Distinction between pitch accent and stress; (3) Analysis of pitch accents in terms of level tones; (4) Local sources for global trends. It has also evolved into a standard for transcribing intonation of American English — Tone and Break Indices (ToBI) [Sil92].

The main idea of Pierrehumbert’s model and the AM approach is that tone and intonation can be represented compositionally by two types of tones: High(H) and Low(L). In the autosegmental representation of tones, a Register feature is proposed [Yip] to represent H and L tones in the upper or lower register, allowing representation of up to five levels according to a division in pitch range. The AM representation also entails that associations between tones and TBUs (Tone Bearing Unit) are not necessarily one-to-one. Contours, therefore, can be represented as a sequence of tones associated with a single TBU: HL (Fall) or LH (Rise) [ZZ13].

In terms of intonation, Pierrehumbert’s model has a linear structure in that intonation is solely determined by a local component, which is in contrast to the superpositional approach that treats intonation as resulting from the addition of several components, including a local pitch accent and a global phrase contour. The mapping from phonology onto acoustics and physiology is a dynamic interpretative process [Pie80], where phonetic realization rules are applied to convert abstract tonal repre-
sentation into $f_0$ contours by considering the metrical prominence of the syllables and the temporal alignment of the tones with the accented syllables [Pie80].

In addition to prosody representation and synthesis, phonological models can also be used to learn and recognize tones. For example, Ramadoss et al. [RW15] adopted a phonological view in the probabilistic tone recognition model for Thai (based on the theory from Moren et al. [MZ06]). In this study, since each tone category is modeled to have a specific target template associated with its TBU (mora in this case), categorizing tones reduces to matching the identified targets to the templates.

Overall, few works in tone recognition have considered this phonological approach, possibly due to the general lack of familiarity with phonological theory in the machine learning community, as well as the challenge of incorporating the general abstract symbolic representation of tonal targets into a machine learning framework that targets real-valued $f_0$ trajectories.

**Phonetic-based models**

Phonetic models use a set of continuous parameters to describe intonation patterns observable in $f_0$ contours [Tay00]. An important goal is that the model should be capable of reconstructing $f_0$ contours faithfully when appropriate parameters are given. However, as many researchers have pointed out, a phonetic model should also be linguistically meaningful, since it is not mathematically difficult to approximate $f_0$ contours with some polynomial function. The real challenge is to develop a model whose parameters are predictable from available linguistic information [SX02]. Here, we describe several representative phonetic models.

1. **Fujisaki model.** Fujisaki [Fuj83, Fuj88] developed an intonation model for Japanese (later applied to other languages). The model additively superimposes a phrase component and an accent component on a logarithmic scale. The control
mechanisms of the two components are realized as critically damped second-order systems responding to impulse/rectangular commands. As can be seen, it is a superpositional approach that assumes different intonation components are superimposed on top of each other, which is different from the linear AM approach described above. Mixdorff et al. [MFCH03] has developed an algorithm to automatically extract model parameters for the Fujisak model from large speech corpora.

(2) **Quantitative Target Approximation.** A pitch target approximation model for generating $f_0$ contours in Mandarin Chinese was proposed by Xu [Xu97, Xu01] and quantified in Xu [Xu99] with the quantitative Target Approximation model (qTA). In this model, the surface $f_0$ contour is viewed as the result of asymptotic approximation to an underlying pitch target, which can be a static target (High or Low) or a dynamic target (Rise or Fall). These four pitch targets correspond to the four tones in Mandarin. Here, a pitch target is defined as the smallest unit that is articulatorily operable. The host unit of a pitch target is assumed to be the syllable (for Mandarin, at least). The model is also regarded as a quantitative realization of $f_0$ based on a speech production model (Parallel ENcoding and Target Approximation, or PENTA [PoXT09]), which emphasizes the role of articulatory constraints in intonation modeling. Due to the relevance of the qTA model to the current dissertation, we discuss the relevant literature in more detail in Chapter 2.2.2.

(3) **Non-parametric models.** Both phonological and phonetic frameworks seek to model $f_0$ contours effectively with a set of more abstract representations. However, $f_0$ values themselves are no doubt good indicators of high-level linguistic information [SX02]. As discussed above, the goal in developing parametric models is to find a better representation of $f_0$ contours. However, if inappropriate forms are used, the predictions can be significantly different from the original $f_0$ contours [SX02]. To address this problem, an alternative is to use the original $f_0$ contours directly or $f_0$
with some trivial modifications as the output targets. Such systems are referred to as non-parametric models, and often times they can achieve very competitive results.

**PENTA and Quantitative Target Approximation (qTA)**

As discussed above, the qTA and PENTA model have evolved over the years from its earliest theoretical formulation to the development of quantitative and computational modes. Prom-on et al. [PoXT09] reports the full realization of the quantitative Target Approximation (qTA) model for generating $f_0$ contours of speech, including its mathematical modeling of the pitch target approximation process, and its parameter optimization strategy. The model simulates the production of tone and intonation as a process of syllable-synchronized sequential target approximation [Xu05]. As a speech production model, the qTA and the associated Parallel Encoding and Target Approximation (PENTA) model has generated much debate on its architecture, representation, and predictive powers. In the current context, however, we are only interested in evaluating how qTA’s capability to numerically predict tone contours, as well as its representation of tone targets as input features to our data mining framework. In this model, each tone is produced with a pitch target in mind, defined by a linear equation with a slope and a intercept parameters, m and b:

$$x(t) = mx + b \quad (2.2)$$

However, the realization of this target is often constrained and deviated by the characteristic factors of the human vocal folds, such as the continuity of pitch change (no abrupt changes in the slopes of the trajectories across syllable boundary) and the limitation of the maximum speed of pitch change [XS02]. As a result, actual $f_0$ contours of tones are characterized by a third-order critically damped system:
Figure 2.4: Examples of $f_0$ contours generated by the qTA model with varying values of $m$, $b$, and $\lambda$. The dashed lines indicate the underlying pitch targets, which are linear functions of $m$ and $b$. The vertical lines show the syllable boundaries through which the articulatory state propagates (adapted from Prom-on et al. [PoXT09]).

\[
f_0(t) = x(t) + (c_1 + c_2 t + c_3 t^2) e^{\lambda t}
\]  

(2.3)

Intuitively, this can be seen as casting a noise component on top of the linear pitch target. In this equation, we have in total three parameters to represent a tone contour with the qTA model: slope ($m$) and height ($b$) of the pitch target, and $\lambda$, which represents how fast the pitch change is approaching the target slope and height. Figure 2.4 shows a number of different combinations of the parameters to demonstrate how the actual $f_0$ trajectory behaves depending on the underlying pitch targets.
The qTA model extracts function-specific model parameters from natural speech audio via supervised learning (analysis by synthesis and error minimization\(^4\)). After the parameter extraction, \( f_0 \) contours generated with the extracted parameters can be compared to those of natural utterances through numerical evaluation and perceptual testing (for evaluation).

The computational tools for extracting parameters and performing re-synthesis based on qTA model were developed in Praat Scripting Language (PSL). In the early version of this tool (PENTATrainer1\(^5\)), parameter optimization is achieved by an exhaustive search through the parameter space via error minimization algorithms [PoXT09]. The local parameter sets learned from this process are then summarized into categorical ones by averaging across individual occurrences of the same functional categories [PoXT09]. While the synthesis results did closely approximate the original \( f_0 \) contours, there are a few disadvantages of this strategy. First, the estimated parameters are optimal for the local syllable but not necessarily for the functional categories. Second, the estimation of \( \lambda \) is often not satisfactory because it may be stuck at a local minimum and fails to converge to global minimum [XPo14].

To address this problem and upgrade the qTA model to include more features reflecting function-related variability, Xu et al. [XPo14] developed PENTAtrainer 2 using stochastic learning from real speech data with annotations of metadata information about the sentences (referred to as layered pseudo-hierarchical functional annotation scheme, which requires the manual labeling of only the temporal information of the functional units). More specifically, each syllable is annotated with its syllable boundaries, tone categories of the current and adjacent tones, focus / stress status,

\(^4\)This strategy is used in the early version. Later versions adopted a more sophisticated method.
\(^5\)Downloaded from http://www.homepages.ucl.ac.uk/~uclyyix/PENTAtrainer1/. 
Figure 2.5: PENTATrainer 2 annotation scheme. (Adapted from Xu et al. [XPo14]).

and associated sentential modality (see Figure 2.5). Overall, this version is characterized by its use of a functional annotation scheme, and training from the annotated data to obtain the parameter values, making it more explicitly a standard machine learning approach that encodes various types of input features. In this respect, it is unlike prosodic modeling approaches typically seen in previous literature. It also has the ability to predicatively generate synthesized speech melody on unseen data, given that the test data are also annotated in this set of input feature labels. The authors argue that this set of annotation is much less labor intensive than traditional frameworks.

Xu et al. [XLPoL15] comments on the advantage of such a learning model using the example of how to learn tone sandhi in this framework (below). This is an expected effect of the model if we consider it from a machine learning perspective:
...tone sandhi. For example, the Mandarin Tone 3 is changed to T2 when followed by another Tone 3. With PENTAtrainer2 this rule can be operationalized as the result of an interaction between two functions: lexical tonal contrast and boundary-marking. That is, the pitch target to be implemented in articulation is jointly determined by the morphemic tone of the current syllable, the morphemic tone of the next syllable, and by the strength of the boundary between the two syllables. Such functional interaction may allow T3 to develop a pitch target variant that happens to be similar to that of another tone, e.g., T2. But the two do not need to be identical, since the functional combinations are not the same. As found in Xu et al. [XPo14], the best modeling result was obtained when the sandhi T3 was allowed to learn its own target, rather than when it was forced to use the T2 target. This result is consistent with the empirical finding of subtle yet consistent differences between the original and sandhi-derived T2 in Mandarin [Xu97]. Thus the obligatoriness of associating a unique target to each functional combination may have led to the development of tone sandhi in the first place. But further research along this line is needed.

2.2.3 Evaluation data set

An important aspect of machine learning is the data set used in training and evaluation. In Mandarin tone learning, the nature of the data set varies depending on the mode of the speech: tone contour shapes in a read speech data set are more faithful to the canonical forms, and is therefore easier to learn (although it is by no means perfectly “clean”, as co-articulation between adjacent syllables would still introduce
variability). Such data sets are usually produced with carefully designed research questions in mind (in the context of a speech production / perception experiment) and is therefore not always appropriate to use in all contexts. The advantage, of course, is that the data set is controlled and may contain more balanced data with regard to the goal of the study, which helps the researcher to better understand a particular problem. In contrast, in a large spontaneous speech data set (such as newscast data sets), the speaking rate is faster, and all parameters of the speech are free to vary. This leads to more distortions to the shapes of the tone contours, and makes them more difficult to recognize when considered in isolation for each syllable. By convention, evaluation of tone recognition systems is usually done with both types of data set, progressing from an easier “clean” data set to a more messy and much bigger “hard” data set [Lev06a, Lev06b].

Xu et al. [XPo14] paid special attention to justifying the use of experimentally produced, smaller, “clean” data sets in training the prosodic models of PENTATrainer 2 with supervised learning methods:

......Note that all these three corpora, due to their experimental nature, may seem more limited than most other corpora used in data-driven modeling, which are typically much less controlled. But speech corpora are merely subsets of all speech and as such they can never be full exhaustive. What really matters is whether a corpus includes sufficient samples (preferably by multiple speakers) of the patterns of interest as well as their triggering contexts. Traditional corpora, typically consisting of many more unique sentences than in a controlled corpus, inevitably have very uneven sample sizes for different patterns. As a result, it is hard to determine in the end which proportion of the modeling errors should be attributed to
the modeling algorithms and which should be attributed to the uneven sample sizes. A further advantage of controlled corpora is that they allow special designs for focusing on difficult problems such as the neutral tone in Mandarin...... it would be very hard to find more than a few (or any at all) samples of similar neutral tone sequence in a traditional corpus. Furthermore, controlled corpora, like those just described, due to their full transparency, makes it easier for investigators to understand what may be the source of a particular problem and how damaging it is, as we will see in the case of the Mandarin corpus used in the present study......

This point is well supported by many experimental works. For example, Liu et al. [LSX06] analyzed sentence global intonation by using a data set where syllabic tone effects are removed. This is achieved through carefully controlled sentences where all words have the same tone (such as the first tone). This type of design is indeed very helpful in understanding the behavior of the global intonation without the effect of tones, and it may indeed by hard to obtain from a large corpus of spontaneous speech. In the meantime, from the perspective of building robust predictive models, the complexity of a larger data set of spontaneous speech is still indispensable.

2.3 Supervised learning of tones

Supervised learning is the predominant approach in Mandarin tone machine learning research. In our current project, a successful unsupervised learning framework must learn from the supervised learning literature. Recent years have seen the renewed interest in improving tone recognition using context-dependent models in supervised learning frameworks. These improvements include better understanding of the features used in tone recognition, and the effort to utilize contextual information to
recover underlying tone targets. These techniques will be reviewed in the next two sections. In Chapter 2.3.3, we also review works that attempt to recover underlying tone targets by identifying the most relevant “nucleus” region in the syllable $f_0$ contour.

2.3.1 Feature selection

Intuitively, $f_0$ contour is the most relevant feature of tones. However, the problem of using only the basic form of $f_0$ features, as discussed above, lies in its variability in running speech. To improve tone recognition accuracy beyond the constraints of using basic $f_0$ features, there have been many efforts to identify and select other useful features that can be extracted from the speech signal.

Surendran [Sur07] concentrated on solving this problem by incrementally testing and identifying effective features in all dimensions. In this dissertation, the author conducted hundreds of experiments on a variety of data sets of broadcast speech to determine the effectiveness of a set of 68 features from multiple dimensions. The results suggest that (1) modifying the pitch and intensity of a syllable based on its neighbors was useful (pitch normalization by subtracting the mean pitch of the preceding syllable); (2) Among the twenty voice quality measures used in tone recognition, energy in various frequency bands was the most useful; (3) A set of 60 band energy features greatly aided the recognition of low and neutral tones; (4) Tone context (knowing the tones of surrounding syllables) did not help as much as one would have expected, suggesting the other features are already capturing a lot of contextual information; (5) Stronger syllables (such as focus) were easier to recognize in lab speech, but the effect is diminishing for broadcast speech.

In another doctoral dissertation, Yu [Yu11] investigated a similar problem by asking how tones are learned (by machines) and acquired (by humans) from the speech
signal through the available information in various phonological spaces (such as $f_0$ and voice quality). The author concentrated her investigation on Cantonese but also included a variety of tone language data such as Mandarin. The results demonstrated that voice source parameters beyond $f_0$ must be included for characterizing phonetic spaces for tonal maps in a wide range of languages.

The correlation between intensity and tones have been demonstrated in early works of speech experiments. In a series of intriguingly designed tone perception judgment experiments, Whalen et al. [WX92] tests the information present in the amplitude contours and in brief segments of Mandarin tones. In the first two experiments, the researchers used a signal-correlated noise (by adding samples with flipped signs to the original samples such that the amplitude is unchanged, but the $f_0$ and formant structure information is removed) to obtain the amplitude contours of the tones without retaining information of $f_0$ and formants. The results showed that Mandarin speakers are able to identify tones with high accuracy using only amplitude contour information (although later it was shown that amplitude values correlate highly with absolute $f_0$ values for tone 2, 3, and 4). In experiment 3 and 4, the authors extracted brief tone segments of variable length using a hamming window, and tested the accuracy of tone identification at each position along the tone contour (e.g., onset at 0ms, 20ms, 40ms, etc., from the beginning of the tone contour). The result suggests that tone 2 and tone 4 are identified with better accuracy when movements of the segments are similar to their respective movement trajectories (i.e., rising for tone 2 and falling for tone 4). For tone 1 and tone 3, the listeners identified more accurately when there are little pitch movement, using differences of absolute $f_0$ (lower sounding pitch judged as tone 3). This indicates the pitch register effect of the tone perception, which is largely unexplored in previous research. The information in intensity
contours are subsequently used in computational works to identify the nucleus region of tones [WX11a], to be discussed in Chapter 2.3.3.

2.3.2 CONTEXT-DEPENDENT MODELING

As mentioned above, recent works have largely distinguished local from broad context in the context-dependent investigation of $f_0$ variability. While tone recognition literature conceptually identifies those two types of contexts, in practice, the two are usually combined to work together inside a single machine learning framework. Therefore I discuss the context-dependent modeling in tone recognition without separating these two types into different sections.

Wang et al. [WS00] improved tone recognition accuracy using contextual information in a Mandarin spoken digit recognition application. The authors focused on two aspects of the contextual intonational effect (one broad and one local context): First, $f_0$ downdrift during the course of an utterance (sentence); Second, the distortion of $f_0$ contour height and slope according to different tonal contexts (i.e., preceding and following tones). To address the first problem, a linear model was built for sentential downdrift and a value is subtracted from the observed $f_0$ values. To take into account the local tone context, the proposed algorithm adjusts the observed $f_0$ contours based on a model trained from the different tonal contexts, in terms of $f_0$ frequency and slope. This system has an overall error reduction rate of 26.1% compared to the base system.

Similarly, Levow [Lev05] incorporates both local context and broader context features in tone recognition with linear kernel SVM, while paying attention to the individual feature sets. The local context is encoded in two types of features: “difference” and “expanded” features. Both types of features have sub-features that encode left or right contexts.
The first set of features (difference features) corresponds to differences between the current syllable and its preceding and following syllables. They include differences between pitch maxima, pitch means, pitch at the midpoint of the syllable, pitch slopes, intensity maxima, and intensity means. The second set of features, known as extended syllable features, is simply the ending pitch values from the end of the preceding syllable and the first values from the beginning of the following syllable, as well as the pitch maxima and means of these adjacent syllables.

The context dependent features are found to consistently outperform context independent features. The results of additional contrastive experiments suggest that left tone context is much more important than right context in tone category identification. In fact, the right context is shown to decrease the performance of the classifier. This result is consistent with speech perception/production experimental results from previous works [Xu01], showing that the tone co-articulation effect is asymmetric.

The broader context feature mainly includes the $f_0$ compensation for downdrift (similar to [WS00]). The author used the median slope per syllable across the entire corpus as phrase-based falling contour compensation. Similar to Wang et al. [WS00], Levow [Lev05] found the alternative compensation strategy based on individual phrase slope (i.e., build a linear model for each phrase, instead of using a global slope across the corpus) overfits to the specific tone configuration and reduced accuracy. In the phrase-based feature representation, each pitch value is thus replaced with an estimate of the pitch value with downdrift removed, by adding back the estimated pitch drop to pitch values later in the phrase. The result showed improvements in classification accuracy. The author commented that since the phrase segmentation employed here was very simple, it is expected that more nuanced approach with finer grained phrase boundary and possibly phrase accent detection would likely yield greater benefits.
Wang et al. [WX11a] proposed a tone recognition approach that employs linear chain Conditional Random Fields (CRF) to model tone contour variations due to intonation effects. Three linear chain CRFs are built, aimed at modeling intonation effects at phrase, sentence and story-level boundaries. All linear-chain CRFs are found to outperform the baseline unigram model, and the biggest improvement is found in recognizing third tones in overall accuracy. In particular, Phrase Bigram CRFs show a 39% improvement in recognizing third tones located at initial boundaries. This improvement shows that the position-specific modeling of initial tones in bigram CRF captures the intonation effects better than the baseline unigram model.

Looking at the broad context, Surendran et al. [SLX05] exploits the focus conditions to improve tone recognition. Pre-focus, post-focus, in-focus, and no-focus conditions are distinguished. The experiment with known focus labels found that pre-focus and no-focus behave similarly in terms of tone recognition error rate, with post-focus having the largest error rates. Meanwhile, on-focus syllables are the easiest to recognize with minimal error rates. Overall, by training and testing SVMs conditioned upon different focus condition groups, the classification error rate reduced 42.9% comparing to the baseline, where no focus group is identified. However, in this experiment, the focus labels had to be manually annotated and they were available on both training and testing sets. This is an unrealistic scenario in real applications. Next, the researchers conducted experiments to incrementally reduce the requirements on manual labeling. The second experiment assumes focus labels are only known during training, and uses \( f_0 \) and intensity based features to predict the focus conditions on testing data set. The third experiment assumes that correct focus labels are not available at all. In this experiment, the focus labels for both training and testing data

---

CRF is a probabilistic graphic model that can be seen as similar to Hidden Markov Model (HMM) for sequence modeling and inference. Crucially, while HMM is a generative model, CRF is a discriminative model.
Table 2.2: Tone recognition using focus. (Adapted from Surendran et al. [SLX05]).

<table>
<thead>
<tr>
<th>Condition</th>
<th>Error Rates</th>
</tr>
</thead>
<tbody>
<tr>
<td>Combined: Not using focus (baseline)</td>
<td>15.16%</td>
</tr>
<tr>
<td>No-focus syllables</td>
<td>7.74%</td>
</tr>
<tr>
<td>Pre-focus syllables</td>
<td>7.74%</td>
</tr>
<tr>
<td>In-focus syllables</td>
<td>0.80%</td>
</tr>
<tr>
<td>Post-focus syllables</td>
<td>18.37%</td>
</tr>
<tr>
<td>Combined: Conditional on correct focus</td>
<td>8.66%</td>
</tr>
</tbody>
</table>

are predicted from the confidence rating on the tone recognition algorithm without using focus information. It is observed and hypothesized that the tones classified with the highest confidence score is the location of the focus. In both of these subsequent experiments using predicted focus label, it is interesting to observe that even though the prediction errors were high on the focus label (more than 30%), the error reduction of the ultimate tone recognizer is still comparable with the first experiment where the correct focus label is known (error rate below 10%). The authors attribute this to the similar behavior of the tone classifier on pre-focus and no-focus conditions, where most of the confusion happens in the focus label prediction phase. Table 2.2 shows the summary of results in this study assuming known focus.

Liu et al. [LSX06] uses the B-Spline coefficients\(^7\) plus acoustic features to train decision trees for question/statement classification from intonation contours in Mandarin (referred to as the modality of the sentence), using a highly controlled experimental data set. For ten-syllable utterances, the highest correct classification rate

\(^7\)B-Spline is a type of piecewise polynomial that functionally approximates the intonation curve.
(85%) is achieved when normalized final $f_0$s of the seventh and the last syllables are included in the tree construction. The results confirm the previous finding that the difference between statement and question intonations in Mandarin is manifested by an increasing departure from a common starting point toward the end of the sentence. Meanwhile, this paper also raises the question regarding the effectiveness of using compact model coefficients to represent $f_0$ contours in supervised and unsupervised learning (as other works such as Zhang [ZRS15] have found that ultra-low-dimension polynomial and qTA coefficients do not perform well in unsupervised learning for Mandarin tones).

2.3.3 Tone nucleus region modeling

Other researchers have sought to identify the most relevant regions in a syllable for tone recognition. Such regions are hypothesized to better approximate the true underlying tone target. This is in part motivated by tone production models such as the PENTA model, which assume that carryover coarticulation dominates tone realization and thus the true tone is more closely approximated in the latter half of the syllable. Sun et al. [SX02] used partial tone segments from the midpoint to the end of the syllables for pitch accent recognition. Subsequently, Zhang et al. [ZHN04] proposed a model that successfully identifies tone nucleus regions for canonical tone production. The tone region is segmented by applying k-means clustering on pitch contour units; the nucleus itself is identified based on features including segmental time and energy.

Wang et al. [WL06] proposes a strategy to identify nucleus regions of tones using the amplitude and pitch plot segmentation with computational geometry techniques. Given syllable boundaries, this approach employs amplitude and pitch information
Figure 2.6: Three-phase segmentation of Second Tone. Pitch contour (top left); Amplitude contour (top right); Convex hull of amplitude-frequency plot (bottom left); Final segmentation (bottom right), adapted from Wang et al. [WL06].

to generate an improved sub-syllable segmentation and feature representation, essentially segmenting the syllable into several regions (one of which is the nucleus region). This sub-syllable segmentation is derived from the convex hull of the amplitude-pitch plot, based on criteria such as the slope (illustrated in Figure 2.6). This approach achieves a 15% improvement using the said segmentation strategy over a simple time-only segmentation.

2.4 Unsupervised learning of tones

There has been limited effort on the unsupervised learning of tones. It is well known that most supervised learning frameworks must rely on a large amount of manual annotation effort that is costly in time and money [Lev06b]. This annotation bottle-
neck, along with the theoretical interest in the learning of tones, motivates the use of unsupervised or semi-supervised approaches to tone recognition [Lev06a]. Another motivation to explore unsupervised learning is to model the process of language acquisition, as child learners must identify these linguistic categories without explicit instruction but only by observing natural language interactions [Lev06b]. As such, the goal of unsupervised learning frameworks is to improve the accuracy of tone learning algorithms with minimum supervision and human labeled data.

2.4.1 Unsupervised and semi-supervised learning of Mandarin tones

Some preliminary unsupervised works by Gauthier et al. [GSX07] employed self-organizing map\(^8\) by use of \(f_0\) velocity as input features for tone learning. Gauthier et al. [GSX07] used a raw 30-point pitch vector and the pointwise first derivatives (D1) of the \(f_0\) values as feature vectors on some 2000 observations of tone contours. In particular, they found that the D1 feature vectors yielded an almost perfect result in classifying unseen stimuli, an improvement over using raw 30-point \(f_0\) values. This shows the internal structures that can be exploited in the tone space via unsupervised learning. Meanwhile, since this study did not use spontaneous speech data set, it is yet to be seen how it performs on more realistic data sets\(^9\).

Levow [Lev06a, Lev06b] concentrated on the problem of unsupervised learning and semi-supervised learning in Mandarin tone recognition. Levow [Lev06a] employed

\(^8\)A self-organizing map (SOM) is a type of artificial neural network (ANN) that is trained using unsupervised learning to produce a low-dimensional (typically two-dimensional), discretized representation of the input space of the training samples, called a map.

\(^9\)Christo Kirov (Johns Hopkins University) and I have performed an initial trial experiment of using SOM on some small samples of spontaneous speech data and obtained mixed results. This is yet to be more systematically evaluated on a larger data set of spontaneous speech.
asymmetric k-lines clustering, a spectral clustering algorithm, as the primary unsupervised learning approach. Rather than assuming that all clusters are uniform and spherical, this approach enhances clustering effectiveness when clusters may not be spherical and may vary in size and shape. The author argues that this flexibility yields a good match to the structure of Mandarin tone data where both the shape and the size of clusters vary across tones. A comparison is made between k-means clustering, symmetric k-lines clustering, and Laplacian Eigenmaps with k-lines clustering.

This algorithm is evaluated on a clean read data set and a spontaneous broadcast news data set. Table 2.3 summarizes the results using the unsupervised vs. supervised approaches.

Contrastive experiments showed that the asymmetric k-lines clustering approach consistently outperforms the corresponding symmetric clustering learner, as well as Laplacian Eigenmaps with binary weights for English pitch accent classification (shown in Figure 2.7). To the author’s surprise, k-means clustering outperforms all of the other approaches when producing 3 to 14 clusters (i.e., \( k = 3, 4, ..., 14 \)).\(^\text{10}\) Accuracy when using the optimal choice of clusters and parameters is comparable for asymmetric k-lines and k-means, and somewhat better than all other techniques considered. The author attributes this similar performance to the careful feature selection process. Moreover, for the four-tone classification task in Mandarin using two stage clustering, asymmetric k-lines strongly outperforms k-means, at 87% vs. 74.75% accuracy.

The feature set used in this study is well informed by previous works on tone production and perception, which included multiple types of features beyond the \( f_0 \) and features that reflect the context-dependent nature of tone contour shapes. The

\(^\text{10}\)In fact, it has been shown in time-series mining literature [MKZ+09] that k-means and Euclidean distance are extremely powerful techniques as the data gets bigger, despite their simplicity.
Table 2.3: Tone recognition with unsupervised and supervised learning.
(adapted and modified from Levow [Lev06a, Lev06b]).

<table>
<thead>
<tr>
<th>Condition</th>
<th>Unsupervised</th>
<th>Supervised</th>
<th>Semi-supervised</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lab, In-focus</td>
<td>87%</td>
<td>99%</td>
<td>94%</td>
</tr>
<tr>
<td>Lab, Pre and In-focus</td>
<td>77%</td>
<td>93%</td>
<td>n/a</td>
</tr>
<tr>
<td>Broadcast News</td>
<td>78%</td>
<td>81.3%</td>
<td>70%</td>
</tr>
</tbody>
</table>

basic features include $f_0$ features (five equidistant points sampled from the $f_0$ contour of the syllable nucleus, and mean $f_0$) and intensity features (both are normalized per speaker and log scaled). The final region of each syllable is identified as the nucleus region. To account for co-articulation effects, nucleus region slope features are computed according to qTA’s assumptions [PoXT09]. These are further log-scaled and normalized to compensate for the greater speeds of pitch falls than pitch rises [XS02]. Figure 2.8 shows the well-separatedness of the four tones in the read speech data set in terms of the slope and height of their pitch targets. Overall, while this yields valuable evidence for feature experimentation, it is doubtful that this pattern can be generalized to the spontaneous speech data.

2.4.2 Semi-supervised learning of Mandarin tones

Levow [Lev06b] further explored semi-supervised learning for the tasks in Levow [Lev06a], using the Manifold Regularization framework. This framework postulates an underlying intrinsic distribution on a low dimensional manifold for data with an observed, ambient distribution that may be in a higher dimensional space with pairwise distances preserved. More specifically, this paper uses Laplacian Support
Vector Machines, a semi-supervised classification algorithm, which allows training and classification based on both labeled and unlabeled training examples. For each class in a Mandarin data set, the model uses a small set (40) of labeled training instances in conjunction with 60 unlabeled instances, and then tests on 40 instances.

The semi-supervised classifier achieved comparable results with the unsupervised algorithm (see Table 2.3). Surprisingly, the semi-supervised classifier also reliably outperforms an SVM classifier with an RBF kernel trained on the same labeled training instances.

2.5 Speech prosody mining using time-series mining techniques

A less explored area related to the computational modeling of tones is the data mining of speech prosody in a large spoken or intonation corpus [RK]. The goal of this endeavor is to improve the understanding of speech intonation and tone contour
patterns through data mining and pattern discovery algorithms using a large quantity of real speech data.

Previous works in corpus-based intonation research [RK, ZRS15] have shown the challenges of mining a large intonation corpus. At the core of this task is the expensive computation of a large amount of high-dimensional pairwise distance (especially with the Dynamic Time Warping or DTW distance measure for time-series data) to obtain the distance matrix, and to find the most effective low-dimension feature representation for the $f_0$ time-series data that faithfully preserve the true distances among objects, with increased efficiency for storage.

2.5.1 Overview of time-series data mining

Formally, a time series $T = t_1, \ldots, t_p$ is an ordered set of $p$ real-valued variables, where $t_i$ is the time index. Time-series data mining focuses on data mining tasks using time-
series data. Lin et al. [LKWL07] outlined the main tasks that time-series data mining research is concerned with:

1. Indexing: Given a query time series $Q$, and some similarity/dissimilarity measure $D(Q,C)$, find the most similar time series in database $DB$;

2. Clustering: Find natural groupings of the time series in database $DB$ under some similarity/dissimilarity measure $D(Q,C)$;

3. Classification: Given an unlabeled time series $Q$, assign it to one of two or more predefined classes;

4. Summarization: Given a time series $Q$ containing $n$ data points where $n$ is an extremely large number, create a (possibly graphic) approximation of $Q$ which retains its essential features but fits on a single page, computer screen, executive summary etc;

5. Anomaly Detection: Given a time series $Q$, and some model of normal behavior, find all sections of $Q$ which contain anomalies or surprising/interesting/unexpected/novel behavior.

Due to the typical large size of data mining tasks and the high dimensionality of time-series data, a generic time-series data mining framework is as follows [FRM94b]: (1) Create an approximation of the data, which will fit in main memory, yet retains the essential features of interest; (2) Approximately solve the task at hand in main memory; (3) Make (hopefully very few) accesses to the original data on disk to confirm the solution obtained in Step 2, or to modify the solution so it agrees with the solution we would have obtained on the original data.

However, in practice, the success of this generic framework depends on the efficient time-series representation and distance measure in the approximated space that allows
the lower bounding of true distances in the original space [LKWL07]. The distance measure also needs to effectively capture the true (and meaningful) distances among objects, and also allows reasonably efficient computation (tractable) often by using special techniques to prune off impossible candidates. In Part III of this dissertation, I discuss some of the most relevant key issues in these areas and focus on finding the optimal methods for computing similarity for speech prosody time-series data.

2.5.2 Data Mining $f_0$ time-series data

The data mining of $f_0$ (pitch) contour patterns from audio data has recently gained success in the domain of Music Information Retrieval (a.k.a. MIR, see [GS14, GSS15a, GRP+15] for examples). In contrast, the data mining of speech prosody $f_0$ data (here on referred to as Speech Prosody Mining (SPM)\footnote{As the previous research in this specific area is sparse, we have coined this term as we conceptualize the time-series data mining based framework for the pattern discovery, similarity computation and content retrieval from speech prosody databases.}) is a less explored research topic [RK]. Fundamentally, SPM in a large prosody corpus aims at discovering meaningful patterns in the $f_0$ data using efficient time-series data mining techniques adapted to the speech prosody domain. Such knowledge has many potential applications in prosody-related tasks, including speech prosody modeling and speech recognition. Moreover, a Speech Prosody Query and Retrieval (SPQR) tool can be also of great utility to researchers in speech science and theoretical phonology/phonetics (tone and intonation).

Due to the nature of speech prosody data, SPM in a large prosody corpus faces classic time-series data mining challenges such as high dimensionality, high feature correlation, and high time complexity in operations such as pair-wise distance computation. Many of these challenges have been addressed in the time-series data mining
literature by proposing heuristics that make use of cheaper and more efficient approximate representations of time-series (e.g., symbolic representations). However, a central question to be addressed in SPM is how to adapt these generic techniques to develop the most efficient methods for computing similarity for the speech prosody time-series data (that also preserves the most meaningful information within this domain). In Part III and Part IV of the dissertation, we will investigate a variety of approaches and toolkits to develop methods targeted at the understanding and analysis of speech prosody time-series data, including time-series representation and distance measure, motif discovery, network analysis, and cross-domain machine learning\textsuperscript{12}.

\textsuperscript{12}By “cross-domain” we mean using features from a variety of linguistic domains to predict problems from another linguistic domain. See Part IV for details.
Part II

Data Collection and Speech Processing
Chapter 3

Data Collection

Following conventions in machine learning literature for Mandarin tones [Lev06a], we experiment with two sets of corpora for our preliminary investigation: a class of smaller, cleaner read speech corpus, and another class of large speech-recognition-size corpora containing spontaneous newscast speech typically found in the evaluation for Mandarin speech recognition and tone recognition tasks. See Chapter 2.2.3 for a discussion on the characteristics of these two types of data.

3.1 Read speech data set

We use the read speech data set from Xu’s study on the contextual variation of Mandarin tones [Xu97] (subsequently used in Gauthier et al. [GSX07]). Many properties of this data set make it a good starting point for time-series similarity experiments.

The design of this data set was originally targeted at revealing only the variability of tone shapes induced by contextual factors, i.e., position of tones, and to reduce the effect of other factors to minimum. To achieve this, the author designed a word list consisting of two syllables (/ma+ma/), and all $4^2 = 16$ combinations of tones are read by the subjects. Most of these two-syllable sequences would be nonsense words. Each two-syllable sequence is read within the context of four different carrier sentences, giving rise to further contextual variation, recorded five times by each speaker, yielding $16 \times 5 \times 4 = 320$ tokens per speaker. Three speakers are recorded in
this data set (two male and one female), yielding a total of 320*3=960 words, or 960*2=1920 tokens of single syllable tone instances. We obtained the data set from the first author of Xu [Xu97] in the format of csv files, including un-normalized and smoothed, downsampled 30-point $f_0$ contour pitch tracks of the 3 speakers. In addition, it contained the corresponding 30-point first derivative (D1) data computed from the original pitch track used in Gauthier et al. [GSX07].

At first glance, this data set seems too clean, as it maintains a very constrained set of combinations of variables. However, simple clustering experiments in previous works demonstrated the challenge of learning this data set [ZRS15]. In the first study that used this data set, Xu [Xu97] concluded that in a two-syllable sequence, the influence of the contexts on tone shapes are mutually inequivalent. Specifically, the anticipatory effect of the second syllable on the first syllable is smaller than the carryover effect from the first to the second. Meanwhile, Zhang [ZRS15] observed that clustering with the subset of data from the first syllable achieves significantly better results than using the data set from both syllables. Therefore, one interesting property of this data set is that it’s possible to observe the behavior of features and methods using a cleaner subset of the data, i.e., first syllable only, and contrast it with the performance on the full data set.

3.2 Newscast speech data sets

We describe the Mandarin newscast speech corpus obtained from Linguistic Data Consortium (LDC) in this study. In addition, we also discuss speech corpora for Thai, which will be used in extended evaluation.
3.2.1 Mandarin Chinese Phonetic Segmentation and Tone corpus
(CMN corpus)

The Mandarin Chinese Phonetic Segmentation and Tone (MCPST) corpus\(^1\) was developed by the Linguistic Data Consortium (LDC) and contains 7,849 Mandarin Chinese “utterances” and their phonetic segmentation and tone labels, separated into training and test sets. The utterances were derived from 1997 Mandarin Broadcast News Speech and Transcripts (HUB4-NE) (LDC98S73 and LDC98T24, respectively). The collection consists of approximately 30 hours of Chinese broadcast news recordings from Voice of America, China Central TV and KAZN-AM, a commercial radio station based in Los Angeles, CA.

“Utterances” in this corpus are time-stamped between-pause units in the transcribed news recordings. Those with background noise, music, unidentified speakers and accented speakers were excluded. A test set was developed with 300 utterances randomly selected from six speakers (50 utterances for each speaker). The remaining 7,549 utterances formed a training set. In the current work we combine all data sets, obtaining 7,849 utterances, totaling about 100,000 syllables (tones) to work with.

This data set is unique in that it contains annotations on the segmentation and identity of syllabic tones (whereas other newscast data sets contain only audio and transcripts). The utterances in the test set were manually labeled and segmented into initials and finals in romanized pinyin form. Tones were marked on the finals, including Tone1 through Tone4, and Tone0 for the neutral tone. The Sandhi Tone3 was labeled as Tone2. The training set was automatically segmented and transcribed using the LDC forced aligner\(^2\), which is a Hidden Markov Model (HMM) aligner trained on the same utterances. The aligner achieved 93.1% agreement (of phone

\(^1\)https://catalog.ldc.upenn.edu/ldc2015s05

Table 3.1: Three female speakers in the Thai data set.

<table>
<thead>
<tr>
<th>Speaker ID</th>
<th>Mean Pitch</th>
<th>Size (syllable)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPA</td>
<td>212.767628995</td>
<td>294</td>
</tr>
<tr>
<td>SPB</td>
<td>268.64804386</td>
<td>208</td>
</tr>
<tr>
<td>SPC</td>
<td>263.441188041</td>
<td>185</td>
</tr>
</tbody>
</table>

boundaries) within 20 ms on the test set compared to manual segmentation. The quality of the phonetic transcription and tone labels of the training set was evaluated by checking 100 utterances randomly selected from it. The 100 utterances contained 1,252 syllables: 15 syllables had mistaken tone transcriptions; two syllables showed mistaken transcriptions of the final, and there were no syllables with transcription errors on the initial. Each utterance has three associated files: a .flac compressed sound file, a transcript text file, and a phonetic boundaries and labels file (.phons).

This is the primary large spontaneous speech data set we will be working with in this dissertation. Henceforth we denote this data set the CMN (Chinese segMenta-tioN) data set.

3.2.2 Corpora for Thai

In Part V of the dissertation, we extend the supervised and unsupervised tone learning algorithms proposed in Part III and evaluate them on a data set of Thai tones. For this experiment, we need data that satisfies these conditions: (1) segmented and annotated with tone information; (2) the quality of the audio should be good, with decent signal-to-noise ratio (SNR); (3) the data set should be of a relatively large size that merits data mining tasks.
In reality, we have found that such large-scale corpora (or Thai speech corpora in general) are much harder to acquire than Mandarin. This may be related to the relatively large amount of interest, high activity and commercial markets present in the Mandarin speaking world. There are several speaker recognition corpora at LDC that claims to include multiple languages including Thai, however, we are not able to locate the Thai portions of the corpora (nor their transcripts) in any of these data sets made available at LDC.

On the other hand, we have tracked down some data sets used in previous Thai tone works such as Ramadoss et al. [RW15]. However, these data sets are very small in size (data size < 50 tones) and therefore do not meet our criteria for performing data mining experiments.

In the end, we have decided to utilize a subset of the CRSLP-MARCS corpus\(^3\) Luk-saneeyanawin [Luk00] made available through CHILDES\(^4\), the child language acquisition data platform. The CRSLP-MARCS corpus consists of video-linked transcriptions of 18 Thai adult-child interactions from the child age of 6 to 24 months, at three monthly intervals. Sessions were of 20 minutes of duration and for CHILDES these have been split into 10 minute files, for a total of 242 files. This corpus has several advantages for our use: (1) it is a large corpus containing more than 40 hours of video recordings (not all of which contain speech), from which we only need to extract a fraction to obtain a good size in terms of number of syllables for our analysis; (2) all adult speech (which we are using) is transcribed with clear phonetic symbols and each syllable is marked with the a tone label; (3) the data is already segmented with utterance-level timestamps (however, additional manual work is needed to segment syllables and mark their tone labels); (4) the entire corpus is publicly available on the

\(^3\)http://childes.talkbank.org/access/EastAsian/Thai/CRSLP.html.
\(^4\)http://childes.talkbank.org.
CHILDES platform. The major shortcoming of this data is that the audio is not of ideal quality, which often contains a lot of background noise.

An additional consideration when using this data is that being a data set on CHILDES, we observe a lot of child-directed speech which may include prosodic profiles that deviate from regular speech found in a newscast corpus, for instance.

Due to these considerations and limitations, we have segmented, annotated and extracted about 700 tones from a subset of the corpus following these criteria: (1) we primarily chose speech that is directed at adults (whenever available), and secondly, directed at children; (2) we chose speech with relatively low ambient noise; (3) we chose to annotate tones with a consideration of the balance of the five tone categories in the data set. The resulting data set comes from 3 different female adult speakers. Table 3.1 shows the statistics of the three speakers in the data set.

This corpus marks tones in the following convention: mid level = tone 0; low level = tone 1; falling = tone 2; high level = tone 3; rising = tone 4.
4.1 Fundamental frequency ($f_0$) estimation

One challenge of obtaining high-quality time-series $f_0$ data from speech is the presence of voiceless segments as well as pitch estimation errors (e.g., pitch doubling and halving, etc). To this end, previous literature have used standard pitch estimation algorithms such as autocorrelation [Boe93] found in Praat, paired with pitch smoothing techniques. In the current study we use the autocorrelation pitch detection algorithm in Praat as our pitch estimation algorithm, and we describe post-processing steps to ensure the quality of the tone time-series data. To evaluate the estimated pitch, we randomly sampled pitch tracks and manually checked that they are identical to the ground truth in spectrogram (i.e., fundamental frequency), as shown in Figure 4.9.

4.2 Speech Processing

Following Surendran [Sur07], we then apply an intuitive trimming algorithm to eliminate large jumps in pitch, and use simple linear interpolation to fill in values of $f_0$ for unvoiced frames. Others have used more complex methods of interpolation, such as splines [Lei06].

Figure 4.1: Sample trimmed pitch track and manual check of fundamental frequency $f_0$ ground truth. Estimated pitch track, marked in dark thick lines, matches the fundamental frequency in the background spectrogram in red.

4.2.1 Pitch Trimming

Due to the unavoidable pitch detection errors in autocorrelation algorithm, such as octave doubling, we must carefully trim pitch values obtained from the original algorithm before they can be used for any data mining purposes. We discuss several problems and strategies in pitch trimming.

Naive pitch trimming based on outlier removal

Intuitively, we can apply outlier removal to trim spurious pitch values estimated from a sentence, and we will interpolate pitch values for both removed and unvoiced frames. Outliers can be detected by looking at values above the upper whisker in a boxplot. However, this results in a large amount of remaining spurious pitch values,
Problem with centrality-based approaches

The problem with the naive approach is that it assumes unimodal central tendency of the pitch data. While this is in general true for well estimated pitch tracks, it has problems with the spurious pitch tracks which can appear to have a bimodal distribution. We show such a sample pitch track in Figure 4.4, where pitch segments are discontinuous and is therefore bimodal between true pitch values and spurious

\footnote{In this plot the periodic chunks of fluctuations reflect the different pitch ranges of different speakers.}

and the interpolation makes the pitch distribution even more deviated from the true distribution. Figure 4.2 shows the results obtained by this method. This plot shows the entire pitch time-series in the CMN corpus (normalized). Values around and above 1.5 should be considered spurious (corresponding to above 600 Hz in original space). We see that after our proposed trimming method is applied (described next), we obtain a much cleaner pitch time-series\textsuperscript{1} (Figure 4.3).

\textbf{Figure 4.2: Long pitch time-series of the entire CMN corpus after naive trimming (normalized).}
(very high) pitch values. We show a sample bimodal pitch distribution histogram in Figure 4.5. Therefore, in order to effectively trim spurious pitch values, we need to consider this kind of data, and carefully design a more sophisticated trimming algorithm.

**A SEGMENT-BASED SPEAKER-DEPENDENT TRIMMING ALGORITHM**

Our proposed algorithm starts from the observation that in a pitch track like Figure 4.4, we can easily identify spurious pitch segments if we divide the pitch into discrete segments first based on timestamps and then compute the difference $d$ in pitch values between the end of a segment and the beginning of the next (and vice versa). A spurious pitch segment would have a very large positive value of $d$ followed by a very large negative value, whereas other true pitch values would have a $d$ that is fairly constant. We implemented this intuition and made some observations in Figure 4.6, where we plotted the difference values of all consecutive pairs of points in the pitch
Figure 4.4: A sample pitch track with bimodal distribution (spurious).

Figure 4.5: Distribution of a sample bimodal pitch track.
Figure 4.6: Difference values between pairs of consecutive points in identifying spurious pitch segments. Dark green shows the original pitch track, with red cross showing its difference values.

The algorithm has one hyper-parameter, i.e., it depends on setting a threshold $T$ for $d$ difference values so that we can be confident to trim of pitch values beyond that threshold of difference. However, this again depends on our understanding of the pitch movements in speech and the distribution of $d$, which can vary a lot depend on speaker and other functions. Concretely, we have found that the speaker’s overall pitch standard deviation (std) is an useful indicator for their pitch movement patterns. Therefore we propose a speaker dependent value for $T$. But before we can compute the std values for each speaker, we must also apply a first round trimming of their outlier pitch values, otherwise the std would be a distorted representation of their most typical pitch movement patterns. Figure 4.7 shows a sample speaker’s (XIY)
original pitch value distribution and the std value is suspiciously large. However, after we performed an outlier pruning step, the distribution becomes more reasonable (Figure 4.8). We implemented the outlier pruning step by computing a histogram with density estimation and empirically decided to remove any pitch value with a density value less than $10^{-4}$. In the end, we chose the threshold $T = 130$ for $std \leq 70$, and $T = 200$ for $std > 70$ (only applies to one speaker). Besides, we also remove very short pitch segments with a duration less than 60 samples (with a time step of 0.001s using the autocorrelation algorithm).

Lastly, we used a remedy to the centrality problem by first interpolating the pitch track so that the central tendency can be regained before applying any outlier-based trimming strategy. Figure 4.1 shows the results of trimmed pitch.
4.2.2 Normalization

Speaker-dependent pitch normalization is necessary to account for the different range of different speakers. We normalized all pitch values by individual speakers using the subtract mean normalization (see Chapter 5.3 for detail).

4.2.3 Log transformation

Generally, semitones (a logarithmic function of fundamental frequency) are a more robust measure of pitch than Hertz. Therefore, we used $\log F_{0\text{hz}}$ instead of $F_{0\text{hz}}$ following Surendran [Sur07]. More specifically, we use Bark scale, a log scale of pitch shown to correlate with perception better than linear scales such as Hertz. Bark scale is defined as below and will be further discussed in Chapter 5.1.

$$F_{CENT} = 1200 \times \log_2 \frac{F_{HZ}}{F_{REF}}$$ (4.1)
Bark scale is near linear in lower values and deviates faster from linear in higher values. While this is true for log scale in general, we found that Bark scale is more effective at many tasks comparing to simply using $\log_2$ on pitch values (as some previous works did). This also shows the importance of the quality of preprocessing.

4.2.4 Pitch reliability percentage

The outcome of pitch estimation contains many frames without reliable pitch values. For each syllable we collect a simple statistic describing the percentage of the syllable duration being reliably estimated, and we logged this information as an extra column in the .phons file in the corpus, which contains time and segmental information of the utterances. This enables us to estimate the confidence of pitch estimation of a given syllable and easily filter unreliable ones.

4.2.5 Interpolation

Following convention, we use the scipy.interpolate1d function in Python to perform linear interpolation on missing pitch values. A sample pitch track after applying interpolation is shown in Figure 4.9.

4.2.6 Smoothing

We experimented with two smoothing strategies: first, a simple moving average window; second, a Gaussian window that assigns more weight to points that are closer and decaying weight for those that are far away\(^2\). However, in subsequent time-series mining experiments (Part III of this dissertation), we found that this sophisticated smoothing technique consistently decreased the performance of the Query by Content (QBC) task in the unsupervised learning of tones. We speculate

\(^2\)https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.gaussian.html
Figure 4.9: Sample trimmed and interpolated pitch track.

that this is due to that fact that often times, this technique may produce too big a deviation from original values. Therefore, we use only the simple moving average sliding window to smooth the pitch curves in subsequent tasks. Figure 4.10 gives us a peek into how a rising contour profile, coupled with some discontinuities in pitch estimation, can turn into falling when more sophisticated smoothing is applied.

4.2.7 Downsampling

We test three strategies for downsampling. First, we test scipy.resample which resamples x to a specified number of samples using Fourier method along the given axis\(^3\). This is a sophisticated method using signal processing techniques. Second, we test a “padding-and-averaging” method, a classic downsampling method by first padding the signal with 0s to the length of \(L\) so that it can be divided by the desired

\(^3\)https://docs.scipy.org/doc/scipy-0.16.0/reference/generated/scipy.signal.resample.html
downsampled length \(l\) with no remainders. Then we average each \(L/l\) samples to derive \(l\) samples as the downsampled vector. Third, we implement a “(quasi)equidistant” downsampling algorithm where we compute a combination of hop sizes \(a\) and \(b\) to do (quasi)equidistant sampling, where \(a, b\) are very close integers. Concretely, we want to use a linear combination of hop sizes \(a\) and \(b\) in order to maximize \(f = a \times x + b \times y\), where \(x, y\) are integers, \(x + y = l\) (where \(l\) is again the desired length of the vector after downsampling), with the constraint \(f \leq L\). The result is almost equidistant sampling of the original vector. In practice, we found \(a = 5, b = 4\) to work well, and we’ll be able to programmatically solve for \(x\) and \(y\) given that \(x \leq (L - 30 * b) / (a - b)\). Intuitively, this means that when absolute equidistant sampling is not possible because the remainder of \(L/l\) is not zero, we use a combination of hop size steps \(a\) and \(b\) to achieve quasi-equidistant sampling.
Figure 4.11: Sample downsampled pitch track with resampling method.

Empirically, the quasi-equidistant method performs the best with the QBC task (Part III of this dissertation). Here, we show the results of applying these three methods to a group of sample pitch tracks in Figures 4.11, 4.12, 4.13. From comparing these plots we can see that the quasi-equidistant method is the most faithful to the original (by design), while the other two have different amounts of distortion (the averaging method is a natural smoothing mechanism). Based on the empirical evidence, we conjecture that these distortions are harmful to the signal carried in the original pitch tracks for tone recognition purposes.

Figure 4.14 shows the pipeline of speech and data preprocessing.

4.2.8 Extracting ngrams

We extract tone unigrams from the preprocessed pitch data, producing two versions: voiced-only and whole-syllable unigram data. A downsampled version is produced.
Figure 4.12: Sample downsampling pitch track with quasi-equidistant method.

Figure 4.13: Sample downsampling pitch track with averaging method.
after the originals. We then extracted tone N-gram data (bigrams and trigrams) from the whole-syllable unigram data set by concatenating tone unigrams and then downsample them to different sampling rate (100, 200, 300, and 400 point vectors). The metadata (which speaker and sound file the N-gram is coming from and its index position in the corresponding .phons file, tone labels, token sentence boundary, etc.) is also appended to the end of each line, enabling us to access the original data points in a later stage.
Table 4.1: Data sets overview. U=unigram, B=bigram, T=trigram, W=whole syllable, V=voiced only, nN=non-neutral tones. UW is the only original data set without downsampling.

<table>
<thead>
<tr>
<th>N</th>
<th>Subsequence Length</th>
<th>ID</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>varies</td>
<td>UW</td>
<td>100161</td>
</tr>
<tr>
<td></td>
<td>30</td>
<td>UW30p</td>
<td>78546</td>
</tr>
<tr>
<td></td>
<td>30</td>
<td>UV30p</td>
<td>74141</td>
</tr>
<tr>
<td></td>
<td>30</td>
<td>UVnN30p</td>
<td>70815</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>UW100p</td>
<td>90524</td>
</tr>
<tr>
<td>2</td>
<td>100</td>
<td>B100p</td>
<td>88360</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>B200p</td>
<td>85573</td>
</tr>
<tr>
<td>3</td>
<td>100</td>
<td>T100p</td>
<td>77813</td>
</tr>
<tr>
<td></td>
<td>200</td>
<td>T200p</td>
<td>77562</td>
</tr>
<tr>
<td></td>
<td>300</td>
<td>T300p</td>
<td>76513</td>
</tr>
<tr>
<td></td>
<td>400</td>
<td>T400p</td>
<td>67644</td>
</tr>
</tbody>
</table>

4.2.9 Data sets

We give a summary of the data sets we produced after preprocessing. We organize this information in Table 7.3 by the value of $N$ (number of syllables in the N-gram) and we show the variety of data sets produced for tone unigrams, bigrams, and trigrams. Subsequence length refers to the number of data points in a tone N-gram. In the rest of the dissertation we refer to these data sets by their IDs. With the exception of UW data set, all other data sets are downsampled and therefore suffered from a loss of data points comparing to the original (UW). Concretely, these lost data points have an original subsequence length less than the downsample subsequence length. Therefore the downsampling also serves as a pruning step to ensure better quality of the data set.
Part III

Computing Similarity for Speech

Prosody Time-series
A preliminary task for SPM (Speech Prosody Mining) is the computation of similarity for time-series objects in the speech prosody domain. In order to efficiently and meaningfully capture this similarity, in this chapter, we consider a variety of parameters in the time-series data mining framework: (1) time-series representation; (2) distance measure; (3) normalization. Then, in the next chapter, we evaluate these methods in a series of time-series data mining tasks in the speech prosody domain.

5.1 Prosodic modeling feature representation

5.1.1 Polynomial regression

The most straightforward way to model $f_0$ contour curves is to use polynomial functions to fit the $f_0$ contour of each utterance. A $f_0$ contour can thus be represented by the coefficient vector $[c_1, c_2, ..., c_{n+1}]$ of a $n$-th order polynomial. This has been done in a number of studies for tone and intonation [LSX06]. Alternatively, one could use a spline function, a piece-wise polynomial to model different sections of a complex contour. This approach greatly reduces the dimensionality of the original $f_0$ contour.

However, Xu [Xu11] points out that the critical question about polynomial representations is that whether they are linguistically meaningful, and whether they can be used in predictive modeling, i.e., serving as categorical parameters that can be generalized to other instances of the same category. However, this method has not
been evaluated in a predictive synthesis context. Zhang [ZRS15] experimented with
the third degree polynomial representation (a 4d vector) of $f_0$ contours in a clustering
task with a clean read data set of Mandarin tones. The conclusion suggests that the
clustering accuracy is low.

5.1.2 Quantitative Target Approximation (qTA)

Given that qTA model (see Chapter 2.2.2) has been shown to perform well in pro-
ducing curves that closely resemble real tone contours in connected speech, an impor-
tant question to be asked is: do qTA parameters perform well to reflect the similarities
between tones in perception? In other words, we want to make sure that qTA param-
eters have the property where perceptually similar tone contour shapes also have
similar parameter values. In Zhang [ZRS15]'s experiments the results suggest a neg-
avative answer to this question. The poor performance of both polynomial and qTA
features suggests that in order to use coefficients for features, the model from which
the coefficients are derived must preserve the true distances between objects in the
original space, a property that these two models do not seem to possess.

5.1.3 Raw Time-series $f_0$ Vector

Gauthier et al. [GSX07] showed that unsupervised classification using Self-Organizing
Map (Neural Network) yielded a nearly 80% correct result when time-series are rep-
resented with a 30-point raw $f_0$ vector.

In Zhang [ZRS15], in order to find the best method of working with $f_0$ vectors,
several transformations are made from the raw $f_0$ vectors, including un-normalized
and normalized. For each of these, in order to avoid the logarithmic behavior of Hertz
unit, three versions are created, using Hertz, Bark, and Cent scale representations,
giving rise to a total of 6 types of feature vectors. The conversion from Hertz to Bark and Cent are computed as below:

\[ F_{CENT} = 1200 \times \log_2 \frac{F_{HZ}}{F_{REF}} \]  \hspace{1cm} (5.1)  

where the \( F_{REF} \) is the reference frequency, corresponding to the minimum \( f_0 \) in the computation of the pitch track (set to 55Hz).

\[ F_{BK} = 7 \times \log[F_{HZ}/650 + (1 + (F_{HZ}650)^2)^{1/2}] \]  \hspace{1cm} (5.2)

5.1.4 First Derivative Vector (D1)

The discrete first derivative feature (D1) is obtained simply by taking the first derivative of the original signal of the tone contour, and downsampled to 30 point:

\[ D1 = 0.5 \times (F_0(t + 1) - F_0(t - 1)) \]  \hspace{1cm} (5.3)

for all timestamps \( t \).

Intuitively, the D1 feature captures the movement of the pitch trajectory at each timestamp. It also serves as a normalization strategy where the differences in pitch height among different speakers are removed. Otherwise, the D1 does not reduce the dimensionality of the time series, nor does it create new abstract features as a combination of other features. As a first glance, it is not a fundamentally different transformation over the \( f_0 \) feature. However, Gauthier et al. [GSX07] showed a near-perfect performance using the D1 feature in a classification task with Self-Organizing Map. In the same experiment, the \( f_0 \) feature performs around 20% lower than the D1 feature. The authors attributed this success to the fact that D1 is able to capture
the direction of pitch movement at each timestamp, a property that raw $f_0$ vectors do not possess.

5.2 Time-series symbolic representation

There have been a great number of time-series representations proposed in the data mining community, including both real-valued and symbolic representations. Figure 5.1 illustrates the most commonly used representations [LKLC03].

![Figure 5.1: Types of time-series data representations. (Adapted from Lin et al. [LKLC03]).](image)

Lin et al. [LKWL07] points out the limitations of real-valued time-series representations (such as DFT and DWT) in data mining algorithms. For example, in anomaly detection, we cannot meaningfully define the probability of observing any particular set of wavelet coefficients, since the probability of observing any real-valued number is zero. Such limitations have led researchers to consider using a symbolic representation of time series.

There have been many symbolic representations proposed for time-series. However, none of the techniques (that is, before the Symbolic Aggregate approXimation)

---

1Notice in this taxonomy, there is no representation that satisfies the lower bounding requirement under symbolic representation. Later, we will discuss Symbolic Aggregate approXimation (SAX) [LKLC03], the first symbolic representation to satisfy this condition.
allows a distance measure that lower bounds a distance measure defined on the original time series. This constitutes a problem for the generic time-series mining framework discussed above in Chapter 2.5.1, since the approximate solution to problem created in main memory may be arbitrarily dissimilar to the true solution that would have been obtained on the original data. A symbolic approach that allows lower bounding of the true distance would not only satisfy the requirement of the generic framework, but also enable us to use a variety of algorithms and data structures which are only defined for discrete data, including hashing, Markov models, and suffix trees [LKWL07]. Symbolic Aggregation approXmation (abbreviated SAX) [LKLC03] is the first symbolic representation for time series that allows for dimensionality reduction and indexing with a lower-bounding distance measure at the same time. The related MINDIST distance function for SAX is discussed in Chapter 5.4.

The SAX representation transforms the pitch contour into a symbolic representation using Piecewise Aggregate Approximation technique (PAA, Figure 5.2), with a user-designated length ($w=$desired length of the feature vector) and alphabet size ($a$), the latter being used to divide the pitch space of the contour into $a$ equiprobable parts assuming a Gaussian distribution of $f_0$ values (Figure 5.3). Here, the Gaussian distribution is used to obtain the breakpoints for vertical pitch space so that each region (represented by a symbol) is equiprobable (probability of that symbol is given by the integration of the area under the Gaussian curve, as defined by the breakpoints). This ensures that the probability of a segment being assigned any symbol is the same. Figure 5.3 shows an example of SAX transformation of a time-series of length 128. It is discretized by first obtaining a PAA approximation and then using predetermined breakpoints to map the PAA coefficients into SAX symbols.

SAX has been evaluated in various classic time-series mining tasks and shown to work well in numerous applications to mine data from a variety of fields such as
bioinformatics, finance, telemedicine, audio and image signal processing, and network traffic analysis [LKWL07]. In particular, it has been shown to preserve meaningful information from the original data and produce competitive results for classifying and clustering time-series data.

SAX has been less explored in the domain of audio signal $f_0$ pattern mining. In the context of Music Information Retrieval (MIR), Valero-Mas et al. [VMSG15] experimented with SAX representation for the computation of $f_0$ contour similarity from music audio signals in a Query-By-Humming (QBH) task\textsuperscript{2}. However, results suggest that SAX does not perform well for musical time-series data in the context of QBH. The authors attribute this to the fact that SAX does not align well with the particularities that the origin domain of the time series may have — in this case, the discrete musical notes. Thus, in the case of QBH, SAX may be abstracting away key musically-relevant information from the melodic contours required for properly performing the alignment of $f_0$ contour pairs.

Zhang [ZRS15, ZRS14] showed the effectiveness of SAX in mining speech or speech-like $f_0$ contours — tones, where finer details of the $f_0$ contours do not carry crucial information comparing to its global shape. In Zhang et al. [ZRS14], musical $f_0$ contours from Beijing opera singing are converted to SAX representations in order to compare its similarity to linguistic tones of its lyrics. In a manually constructed data set consisted of balanced $f_0$ contour shapes from all four tones, four types of evaluation measures showed that SAX representation faithfully preserves the original clusters. Similarly, Zhang et al. [ZRS15] experimented with the SAX parameters by iterating through different combinations of $w$ and $a$ to find the best correlation with the perceived pitch relationships in pairwise contour analysis.

\textsuperscript{2}In a query by humming task, a user hums a section of the melody of a desired song to search for the song from a database of music recordings.
Figure 5.2: Piecewise Aggregate Approximation. The PAA representation can be visualized as an attempt to model a time series with a linear combination of box basis functions. In this case, a sequence of length 128 is reduced to 8 dimensions (adapted from Lin et al. [LKWL07]).

One strength of SAX is its ability to reduce the dimensionality of the original $f_0$ contour (this is also termed “pitch quantization” in signal processing). This type of simplification is very useful in both addressing the “curse of dimensionality” problem, and to reduce cost and speed up computation. However, there are also many other variants of symbolic representation that can perform pitch quantization. One must choose the most appropriate variant according to the specific needs of the task at hand.

5.3 Time-series normalization

Many literature reviews [LKWL07] in time-series mining assert that time series must be normalized using the z-score transformation normalization strategy so that each contour has a standard deviation of 1 and mean of 0 (in fact this is a general strategy that is also used in mining non-time-series data):
Figure 5.3: Symbolic Aggregate Approximation. With original length $n = 128$, number of segments $w = 8$ and alphabet size $a = 3$, output word $\text{baabccbc}$ (adapted from Lin et al. [LKWL07]).

\[
    z = \frac{x_i - \mu}{\sigma}
\]  

(5.4)

where $\mu$ is the mean of the contour and $\sigma$ is the standard deviation.

However, Zhang et al. [ZRS15] observed that speech tone time-series data has a special property so that the $z$-score transformation would seriously distort the shapes of the tones in the normalized corpus. Essentially this is caused by the presence of many flat or near flat contours (such as in level tones). Since $z$-score transformation expresses each data point in a time series by its relative value to the mean in terms of standard deviation, it would magnify the differences in the values of the flat or near flat contours (since each point in this contour is identical and also identical to the mean), and turn such contours into a significantly non-flat contour.

Since normalization is required for a meaningful comparison of time-series patterns, we need to design an alternative strategy for time-series normalization in this
task. There are many strategies that fulfill this purpose. After trial and error experimentation, one of the best working measure is the Subtract-Mean normalization strategy (Equation 5.5). It effectively retains the original shapes in a normalized corpus of tones.

\[ z = (x_i - \mu) \] (5.5)

This issue also exists in the SAX representation since normalized time-series SAX significantly outperforms un-normalized ones [ZRS15]. Therefore it is a built-in requirement of SAX to first normalize the time-series using z-score transformation. Fortunately, our SAX implementation\(^3\) also has a built-in remedy for this problem: when the standard deviation of a time-series subsequence is less than a pre-set threshold (a very small number), all of its segments will be assigned the same symbol.

5.4 Distance measure

5.4.1 Euclidean distance

The Euclidean distance is one the most widely used and computationally economic distance measures [LKWL07], defined as follows on the \(n\)-dimensional Euclidean space for a time-series of length \(n\):

\[ d(p, q) = d(q, p) = \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2} \] (5.6)

5.4.2 DTW distance

In time-series analysis, dynamic time warping (DTW) is an algorithm for measuring similarity between two temporal sequences which may vary in time or speed

\(^3\)https://github.com/zangsir/saxpy/blob/master/saxpyFast.py
(Figure 5.4). The DTW distance between two time series is computed with dynamic programming, by recursively solving the optimal alignment between two sequences in subproblems, and return the shortest distance between the two (i.e., best alignment) time series. In particular, the optimal path is the path that minimizes the warping cost:

$$DTW(Q,C) = \min \left\{ \sum_{k=1}^{K} (w_k) \right\}$$

where $w_k$ is the matrix element $(i,j)_k$ that also belongs to k-th element of a warping path $W$, a contiguous set of matrix elements that represent a mapping between two time-series objects $Q$ and $C$.

This warping path can be found using standard dynamic programming to evaluate the following recurrence.

$$\gamma(i,j) = d(q_i,c_j) + \min \left\{ \gamma(i-1,j-1), \gamma(i-1,j), \gamma(i,j-1) \right\}$$

where $d(i,j)$ is the distance found in the current cell, and $\gamma(i,j)$ is the cumulative distance of $d(i,j)$ and the minimum cumulative distances from the three adjacent cells.

To make the computation tractable and to prevent pathological warping, many constraints have been proposed to impose upon the possible warping window. One such window is shown in green in Figure 5.4.

In practice, since DTW has a time complexity of $O(n^2)$, where $n$ is the length of the time-series, various lower-bounding techniques are proposed to speed up DTW distance computation in a large database. The LB_Keogh lower bounding technique [KK02], for example, speeds up the computation by first taking an approximated distance between the time-series objects that is both fast to compute and lower bounds the true distance. It would go on to compute the real DTW distance only if this
distance turns out to be smaller than the best-so-far, since otherwise there is no way that the true distance is smaller than the best-so-far. This makes DTW essentially an \( O(n) \) algorithm as we rarely have to do a full DTW calculation. The general approach is illustrated in Figure 5.5. This approach can be used in many applications that require DTW, such as exact motif search (see discussions below) and k-means clustering with DTW, where for each data point one needs to find its closest centroid.

In the meantime, it is worth noting that the DTW distance, similar to Euclidean distance, is not immune to the biases existing in time-series data. For example, Batista et al. [BWK11] has shown that there exists a tendency for a high complexity time-series TS1 (higher degree of micro level variations) to obtain a high similarity score with a low complexity time-series TS3, when in actuality, TS1 is obviously more similar to another high-complexity time-series TS2 (i.e., \( \text{dist}(TS1,TS2) < \text{dist}(TS1,TS3) \)), where \( \text{dist()} \) is the distance function between the two time series). An illustration of

\[ \text{Figure 5.4: Euclidean distance vs. Dynamic Time Warping: example. (Adapted from Rakthanmanon et al. [RCM+12]).} \]
Figure 5.5: An algorithm that uses a lower bounding distance measure to speed up the sequential scan search for the query Q. (Adapted from Ratanamahatana et al. [RK04]).

```
Algorithm Lower_Bounding_Sequential_Scan(Q)
1. best_so_far = infinity;
2. for all sequences in database
3.   LB_dist = lower_bound_distance(C, Q);
4.   if LB_dist < best_so_far
5.     true_dist = DTW(C, Q);
6.   if true_dist < best_so_far
7.     best_so_far = true_dist;
8.     index_of_best_match = i;
9.   endif
10. endif
11. endfor
```

this phenomenon is seen in Figure 5.6 [GSS15b]. Figure 5.6 shows three time series P1, P2, and P3, pitch contours of three melodic phrases. Acoustically, P1 and P2 are the occurrences of the same characteristic base phrase and both are musically dissimilar to P3. However, using the best performing variant of the similarity measure in Gulati et al. [GSS15b], the algorithm obtains a higher similarity score between the pairs (P1, P3) and (P2, P3) compared to the score between the pair (P1, P2). The same bias is found to apply to DTW distance. Therefore, if such problem is present, we must resort to using a complexity-invariant distance (CID) measure as exemplified by Batista et al. [BWK11].

5.4.3 MINDIST distance function

The MINDIST function is a distance measure defined for the SAX representation of the time series, which, crucially, have been proved to lower bound the true distances
Figure 5.6: Pitch contours of three melodic phrases (P1, P2, P3). P1 and P2 are the occurrences of the same characteristic phrase and both are musically dissimilar to P3 (top pink). (Adapted from Gulati et al. [GSS15b]).

of original data [LKLC03]. It returns the minimum distance computed between two strings by building on the PAA representation distance function and substituting the distance computation with a subroutine of dist() function:

$$MINDIST(Q, C) = \sqrt{n \frac{1}{w} \sum_{i=1}^{w} dist(q_i, c_i)^2}$$  \hspace{1cm} (5.9)

The dist() function can be implemented by searching over a lookup table (for details see Lin et al. [LKLC03]). The lower bounding property is an important heuristic for pruning sub-optimal candidate when finding the minimum distance: to avoid expensive computational cost of computing the true distances over a large number of time series, we can first use a SAX approximation to the original time series and compute the MINDIST distance matrix. Since the MINDIST is proved to lower bound the true distance, then we can prune off those whose MINDIST distance is greater than the best-so-far, since there is no way their true distances would be less than the best minimum distance so far.
6.1 Overview

There are many techniques proposed for the data mining of time-series data. These include the choices on time-series representation, distance computation, normalization techniques (temporal and pitch), and time-series data mining and pattern discovery algorithms. However, before we make any particular choices, we need to consider the many features that distinguish one type of time-series data in domain A from another type in domain B, in terms of the local and global characteristics of the time-series data in question, and the meanings they map to\(^1\). In an important meta paper, Keogh et al. [KK02] pointed out:

...the unique structure of time series means that most classic machine learning algorithms do not work well for time series. In particular the high dimensionality, very high feature correlation, and the (typically) large amount of noise that characterize time-series data have been viewed as an interesting research challenge. Most of the contributions focus on providing a new similarity measure as a subroutine to an existing classification or clustering algorithm...

\(^1\)For example, the structure, noise level, periodicity, and complexity of financial, weather, musical, and speech time-series may vary greatly, and the time-series similarity measures meaningful for one domain may not apply to another.
Therefore, the key to successful SPM lies in optimizing the methodological choices for computing similarity in the speech prosody time-series domain. The optimization is achieved by maximizing efficiency (which is crucial in mining massive data sets in terms of the tractability of the algorithm) while preserving the most meaningful information for the prosody domain. In short, the goal of the similarity computation is: how can we efficiently compute meaningful similarity scores and extract similar patterns between pairs of speech prosody (tone) time-series found in a large database?

In this chapter, we propose several dimensions of experimentation that enables us to develop the best combination of core methodologies to efficiently extract meaningful patterns from speech prosody data. First we discuss relevant previous works on mining $f_0$ data in the domain of Music Information Retrieval. Then we carry out time-series data mining experiments to evaluate the set of representations and distance measures considered in the last chapter.

The result of this chapter is a set of fundamental methodologies for computing speech prosody time-series similarity. They are important in their own right, as well as in various time-series mining tasks carried out in the rest of this dissertation.

6.2 Related work

There has been limited amount of previous works on $f_0$ pattern data mining, including MIR $f_0$ melodic pattern mining, and corpus based speech intonation research.

Gulati et al. [GS14] mined a database of melodic contour patterns in Indian Art Music. Due to its astronomical size (over 14 trillion pairs of distance computation), various techniques are used to speed up the computation, including lower bounding

---

In this dissertation we will refer to speech prosody and tone as our target research objects for two reasons. First, despite our focus on tone data, our methods should be generalizable for other speech prosody analysis tasks. Second, we also deal with tone N-grams, which contain data from both local and global speech prosody phenomena.
in DTW distance computation [KK02] and early abandoning techniques for distance computation [RCM+12]. The goal is to discover highly similar melodic time-series patterns that frequently occur in the collection (motifs) in an unsupervised manner, and the result is evaluated using the query-by-content paradigm (in which a seed query pattern is provided and top-K similar patterns are returned). The meaningfulness of the discovered pattern is then assessed by experts of Indian music.

Gulati et al. [GSS15a] experimented with 560 different combinations of procedures and parameter values (including the sampling rate of the melody representation, pitch quantization levels, normalization techniques and distance measures) in a large-scale evaluation for the similarity computation in MIR domain. The results showed that melodic fragment similarity is particularly sensitive to distance measures and normalization techniques, while sampling rate plays a less important role.

Valero-Mas et al. [VMSG15] experimented with the SAX representation for the computation of $f_0$ contour similarity from music audio signals in a Query-By-Humming (QBH) task\textsuperscript{3} in MIR. Results suggest that SAX does not perform well for music time-series data in the context of QBH. The authors attribute this to the fact that the SAX representation loses information important in music melodic retrieval through its dimensionality reduction process.

To the best of our knowledge, the only work that attempted at data mining of speech prosody is Raskinis [RK]'s work on clustering intonation patterns in Lithuanian (although it did not explicitly employ any time-series data mining techniques). While this work examined the use of a number of distance measures (mainly Euclidean vs. DTW), it is an early-stage research and without clear conclusions regarding either the effectiveness of the discussed methods or the significance of the patterns discovered.

\textsuperscript{3}In a query by humming task, a user hums a section of the melody of a desired song to search for the song from a database of music recordings.
6.3 Data sets

In this chapter, we perform time-series data mining experiments using two data sets discussed previously: (1) Xu [Xu99] Read Speech Data set described in Chapter 3.1; (2) CMN (Mandarin Chinese Phonetic Segmentation and Tone (MCPST)) data set by LDC, described in Chapter 3.2. We use the CMN-UVnN30p data set, as listed in Table 7.3.

6.4 Classification of Mandarin tone time-series

6.4.1 Experimental setup

Classification experiments

For classification, we use $k$-nearest neighbor (KNN) and Decision Tree, both of which are widely used in time-series classification\footnote{In practice, KNN is an expensive algorithm that scale more poorly than decision trees.} [LKWL07]. We report only accuracy on the classification experiments considering the balanced nature of the data set. All classification experiments are done with 10-fold cross validation with randomized split of data.

In the KNN classification of the Read speech data set, due to the high time complexity of KNN, we need to consider the optimal split of training and testing data size that reduces the time complexity of the algorithm. In addition, following the convention of using a smaller training size in time-series data mining literature, we carry out the classification experiments using 1600 samples for testing and 320 samples for training (the total size of the data set being 1920 samples of tone contour time-series).
Figure 6.1: KNN classification accuracy depending on \( w, a \).

To optimize SAX parameters, for \( w \), we search from 6 up to \( 2n/3 \) (\( n \) is the length of the time series); for \( a \), we search each value between 6 and 20. It is observed that a lower value for \( a \) results in poor classification results (since the MINDIST distance measure is defined as a sum of pairwise letter distance between two strings). Figure 6.1 shows how classification accuracy varies depending on \( w \) and \( a \). In the Decision Tree classification, we use the standard 90-10 split of the training-testing data.

Because of the poor scalability of KNN algorithm, it is impractical to use it on larger data sets. Therefore, we perform only Decision Tree classification experiments on our large data set (CMN data set). The size of the CMN data set is on the scale of 100,000 syllables. We randomly sample 10,000 for each classification experiment.
A note on SAX implementation

While the original SAX was implemented in Matlab\(^5\), we have adopted the SAX Python implementation in the Library saxpy\(^6\). Consistent with the definition of symbolic representation, saxpy internally uses strings to represent alphabet letters and outputs literal SAX strings. While this is an intuitive choice that does not affect the correctness of the implementation, we observe that MINDIST distance computation implemented with Python string type is very slow when used with high time-complexity algorithms such as KNN\(^7\). Therefore, inspired by the original Matlab implementation, we have re-implemented saxpy using internal float representation in Python numpy module while taking advantage of numpy’s fast, Matlab-fashion matrix operations to greatly improve the speed of the code. The result is saxpyFast\(^8\) and is estimated to be 4-5 times faster than the original saxpy. This improvement makes running many time-series mining tasks efficient (especially those that depend on MINDIST distance computation), including KNN, clustering, and QBC.

6.4.2 Results

Read Speech Dataset

First we report time-series classification results on the Read data set using K-Nearest Neighbor (KNN) and Decision Trees (J48 as implemented in Weka). These classification results are presented in Table 6.1 and Table 6.2, respectively. First, we observe that the original \(f_0\) (Hertz) representation performs comparably with normalized-Bark

---

\(^5\)https://cs.gmu.edu/~jessica/sax.htm
\(^6\)https://github.com/nphoff/saxpy
\(^7\)With a small data set of 2000 time series, it would take more than one hour to run KNN on a Intel core-i5 laptop with 16GB RAM.
\(^8\)https://github.com/zangsir/saxpy/blob/master/saxpyFast.py
Table 6.1: K-Nearest Neighbor tone classification results. With 10-fold cross validation (CR=compression rate, EU=Euclidean Distance, SAX parameters (w,a)=(20,17), test_size=1600, training_size=320).

<table>
<thead>
<tr>
<th>config</th>
<th>SAX-MINDIST</th>
<th>BK</th>
<th>EU</th>
<th>DTW</th>
<th>D1</th>
<th>qTA</th>
<th>polyN</th>
</tr>
</thead>
<tbody>
<tr>
<td>K</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0.81</td>
<td>1</td>
<td>1</td>
<td>0.92</td>
<td>0.93</td>
<td>0.73</td>
<td>0.66</td>
</tr>
<tr>
<td>3</td>
<td>0.87</td>
<td>1</td>
<td>1</td>
<td>0.92</td>
<td>0.93</td>
<td>0.73</td>
<td>0.66</td>
</tr>
<tr>
<td>5</td>
<td>0.87</td>
<td>1</td>
<td>1</td>
<td>0.92</td>
<td>0.93</td>
<td>0.73</td>
<td>0.66</td>
</tr>
<tr>
<td>7</td>
<td>0.89</td>
<td>1</td>
<td>1</td>
<td>0.92</td>
<td>0.93</td>
<td>0.73</td>
<td>0.66</td>
</tr>
<tr>
<td>dimension</td>
<td>20</td>
<td>30</td>
<td>30</td>
<td>30</td>
<td>30</td>
<td>30</td>
<td>4</td>
</tr>
<tr>
<td>accuracy</td>
<td>0.66</td>
<td>0.66</td>
<td>0.66</td>
<td>0.66</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

and First Derivative (D1) vectors, using Euclidean distance and DTW distance\(^9\). All of these achieved above 90% accuracy and F1 score using \( K = 1 \). The DTW distance showed slight advantage over Euclidean distance. All of the numeric representations performed comparably when \( K \) varies, so only results for \( K = 1 \) are shown. Second, we note that the SAX representation achieved reasonable but lower score (with lower dimensionality, compression rate being 0.66). In particular, it performs worse on \( K = 1 \), and the performance improves significantly when we increase \( K \) to 3, 5, and 7. Third, the qTA and polynomial representations achieved significantly lower classification accuracy, at the advantage of having very low dimensionality (compression rate around 0.1). These trends also showed up in the Decision Tree classification, which has comparable classification accuracy with KNN (with lower cost). Overall, we note that SAX shows slight disadvantage in the time-series classification accuracy, while being able to achieve a 0.66 compression rate for time and space complexity. Taking into consideration the results obtained from the CMN data set of spontaneous speech, we attribute the lower performance of SAX in this experiment to the nature

\(^9\)The difference between Bark and Cent features is small, so we only report results for Bark.
Table 6.2: Decision tree classification results in Read speech data set. CR=compression rate.

<table>
<thead>
<tr>
<th></th>
<th>TSR</th>
<th>SAX</th>
<th>BK</th>
<th>Hertz</th>
<th>D1</th>
<th>qTA</th>
<th>poly</th>
</tr>
</thead>
<tbody>
<tr>
<td>accuracy</td>
<td>0.88</td>
<td>0.93</td>
<td>0.92</td>
<td>0.93</td>
<td>0.83</td>
<td>0.79</td>
<td></td>
</tr>
<tr>
<td>dimension</td>
<td>12</td>
<td>30</td>
<td>30</td>
<td>29</td>
<td>3</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>CR</td>
<td>0.66</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.1</td>
<td>0.13</td>
<td></td>
</tr>
</tbody>
</table>

of the data set: the clean read speech in this data set is easy for a regular classifier to learn so that they already saturated their performance at greater than 90% accuracy. Meanwhile, the advantage of SAX in noise reduction did not show in this clean data.

CMN data set

In this section, our primary goal is to see if SAX can show its advantage in time-series classification tasks using a much larger spontaneous speech data set comparing to the previous small-scale experiments on the Read data set. We have several considerations before we proceed to this experiment. First, we have seen (in both supervised setting above and unsupervised setting below) that qTA and polynomial representations of tones are ineffective for distinguishing tone categories. They are very low dimensional and thus lose too much information. Therefore we do not include them in this round of evaluation. Second, as discussed above, we are only conducting experiments using the Decision Tree algorithm due to the poor scalability of KNN for large data sets.

We report the classification accuracy on the CMN data set in Table 6.3. Here, we are showing results using SAX with parameters $w = 12, a = 7$. First, in this much harder data set, even though absolute classification accuracy is low comparing to the Read data set, we observe that they are still much higher than chance. These results
are in line with the previous works using large data sets of spontaneous speech [Sur07].
Second, we note that the SAX shows its advantage in this classification task, outperforming both original Hertz based time-series and D1 based representation (both normalized across speakers) with much lower dimensions. We conclude that SAX is an effective representation that performs dimensionality reduction and improves classification accuracy comparing to raw time-series. This is also in line with previous works showing the advantage of SAX acting as a noise-reduction algorithm [LKLC03].
Finally, comparing to other classification algorithms such as SVM and Random Forest (which are highly effective algorithms for other types of non-time-series data), we observe that Decision Tree always yields the best classification accuracy in this task. This is consistent with the time-series data mining literature’s assertion of the special properties of time-series data in selecting machine learning algorithms, and that KNN and Decision Tree are among the most competitive classification algorithms for time-series data [LKLC03].

Table 6.3: Decision tree classification results in CMN data set. CR=compression rate.

<table>
<thead>
<tr>
<th>TSR</th>
<th>SAX</th>
<th>Hertz</th>
<th>BK</th>
<th>D1</th>
</tr>
</thead>
<tbody>
<tr>
<td>accuracy(%)</td>
<td>59.97</td>
<td>57.51</td>
<td>58.83</td>
<td>53.95</td>
</tr>
<tr>
<td>dimension</td>
<td>12</td>
<td>30</td>
<td>30</td>
<td>29</td>
</tr>
<tr>
<td>CR</td>
<td>0.4</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
6.5 Clustering of Mandarin tone time-series

6.5.1 Experimental setup

Clustering experiments

For clustering experiments we use the k-means clustering algorithm, where accuracy is computed against true tone labels for each instance. The clustering accuracy is measured by comparing the assigned cluster labels to the true tone labels of each time series. We then construct a confusion matrix showing the true labels against the predicted labels of the cluster assignments, where the predicted label is the most predominant label (i.e., the tone category with the most number of instances among all tones assigned that label) within that cluster. If the predominant label is not unique across clusters, then we assign a low accuracy value of 0.1 to represent the accuracy is undecidable.

Alternative evaluation on the CMN data

This approach of evaluating accuracy based on majority class label is similar to the purity evaluation metric of clustering, as seen in contexts such as information retrieval\(^\text{10}\). However, one shortcoming of this approach is that when multiple clusters have the same majority label, it is impossible for each cluster to have a unique class label. Such scenario is highly likely to occur in larger and more noisy data sets such as the CMN data set. In such cases, we are nevertheless still interested in evaluating whether one method is doing better than others. For this reason, we propose to use a more robust evaluation metric called Mean Average Precision (MAP) within an alternative time-series mining task called Query by Content (QBC). Also related to information retrieval, we formulate the QBC task as retrieving time-series data points of relevant

tones given a specific query of a tone object, and we evaluate the result on how much of the top \( k \) retrieved tones are truly relevant to the query tone time-series object. We will describe the QBC task in Chapter 6.6 and use it to evaluate the time-series representations and distance measures in the CMN corpus. The current clustering task is only applied to the Read speech data set.

6.5.2 Results

The true utility of the SPM framework lies in detecting patterns in an unsupervised setting. Comparing to classification, SAX shows more distinct advantage in the clustering experiments. In the following discussion, we note that we are able to use a bigger compression rate (comparing to the classification experiments on the same data set) for SAX in the clustering experiments (i.e., smaller word size), at \( w = 13 \), which gives a compression rate of approximately 0.43.

Figure 6.2: Average clustering accuracy for 1920 Mandarin tones (%) from 5 iterations. For numeric representations, Euclidean distance is used by default unless otherwise noted.
Figure 6.3: K-means clustering objective function by number of iteration. Intra-cluster distance (y axis) is normalized, distance measure noted except for Euclidean. Only polynomial and qTA (overlapped) showed periodic oscillation without a trend to converge.

The clustering accuracy is summarized in Figure 6.2. We establish a baseline accuracy of 56% with normalized $f_0$ representation, indicating the difficulty of this task (although this is still well above chance level of 25% for four tones). Clustering results suggest that (1) The D1 feature significantly outperforms the $f_0$-based features with Euclidean distance; (2) The DTW distance computed with LB_Keogh heuristic shows its clear advantage with $f_0$ features, although its utility is quite limited in this data set compared to previous works; (3) it is noteworthy that SAX is in a much lower dimension yet performs comparably with D1; (4) polynomial and qTA model coefficient based features perform below chance in this task, indicating distances in the original space are not preserved in the ultra-low dimensional parameter space. To probe into these behaviors, we plot the k-means objective function against the number of iteration in Figure 6.3. In particular, the polynomial and qTA parameters
show periodic oscillation of intra-cluster distances, lacking a trend to converge. SAX shows quick convergence, ranking among the most effective.

Overall, in this unsupervised setting, it is noteworthy that DTW is not showing its advantage in computing time-series similarity for the current tone data set as seen in literature in other domains (see previous discussion in Chapter 5.4). We are yet to further investigate DTW’s utility in the speech prosody domain.

Finally, we plot distance matrixes of tones in this data set (shown in Figure 6.4), which may give us a hint as to why SAX is a more effective representation than the $f_0$ (Hertz) vectors in clustering: In Figure 6.4 we can clearly see that the lower dimension SAX-MINDIST distance reflects the intrinsic structure of the clusters with lower distances along the diagonal, whereas the distances are all somewhat equalized in the $f_0$ distance matrix. Overall, SAX and its low dimensionality property may act as a noise-reduction mechanism for revealing the internal structure of the data, making it more suitable for speech tasks comparing to MIR tasks.

6.6 Query by content

6.6.1 Experimental setup

Query by content

Query by Content (QBC) is a commonly used content retrieval task in many domains and data types, such as text [MRS08], audio [GS14], and image retrieval. In our use case, the advantage of using QBC to evaluate our time-series mining techniques is that it always delivers quantified results regardless of its absolute quality, therefore allowing us to compare different methods even when their accuracy values are on the lower side (as discussed above, clustering will not allow us to do that).
Figure 6.4: SAX-MINDIST (left) and $f_0$-Euclidean (right) Distance matrix of 1920 Mandarin tones sorted by tone category. Tones are ordered by tone categories along the x- and y-axis. Origin at top left corner (i.e., on both axis data points are ordered by 480 instances of tone 1, tone 2, tone 3, and tone 4 when moving away from origin).

QBC works similarly to classic information retrieval (IR) — where the user supplies some query string and the system attempts to find the most relevant documents given the query. It then presents the results in a ranked list. Top ranked results should be the most relevant, and vice versa. In our case, the users information need is expressed by a specific query tone curve (or more generally, a time-series subsequence), and the goal is to find similar time-series objects in the database that are in the same category of the query tone curve. This also has the advantage of allowing us to adopt evaluation techniques typically used in IR, which is a well-researched area where the behaviors of evaluation methods are well understood.

EVALUATION

In IR research, the basic challenge of evaluation stems from the fact that the system produces ranked results. Precision, recall, and the F measure are set-based measures.
They are computed using unordered sets of documents. We need to extend or revise these measures if we are to evaluate the ranked retrieval results that are now standard with search engines. In a ranked retrieval context, appropriate sets of retrieved documents are naturally given by the top \( k \) retrieved documents. For each such set, precision and recall values can be plotted to give a precision-recall curve, such as the one shown in Figure 6.5. Precision-recall curves have a distinctive saw-tooth shape: if the \((k + 1)^{th}\) document retrieved is nonrelevant then recall is the same as for the top \( k \) documents, but precision has dropped. If it is relevant, then both precision and recall increase, and the curve jags up and to the right [MRS08].

Based on this understanding, in recent years, one of the most common IR evaluation measures is the Mean Average Precision (MAP), which provides a single-figure measure of quality across recall levels. Among evaluation measures, MAP has been shown to have especially good discrimination and stability. For a single information need, Average Precision is the average of the precision value obtained for the set of
top $k$ documents existing after each relevant document is retrieved, and this value is then averaged over information needs. That is, if the set of relevant documents for an information need $q_j \in Q$ is \{$d_1, \ldots d_{m_j}$\} and $R_{jk}$ is the set of ranked retrieval results from the top result until you get to document $d_k$, then

$$\text{MAP}(Q) = \frac{1}{|Q|} \sum_{j=1}^{|Q|} \frac{1}{m_j} \sum_{k=1}^{m_j} \text{Precision}(R_{jk})$$

When a relevant document is not retrieved at all, the precision value in the above equation is taken to be 0. For a single information need, the average precision approximates the area under the uninterpolated precision-recall curve, and so the MAP is roughly the average area under the precision-recall curve for a set of queries [MRS08].

For our use case, in the above equation, $|Q|$ is the cardinality of the set of tone queries $q_j$; $m_j$ is the total number of relevant tones, i.e., in the same tone category as the query; $R_{jk}$ is basically the set of top ranked $k$ tones, for which we compute their precision. Effectively, this equation is saying we should compute for all values of $k$ from 1 to $m_j$, the precision of the top ranked $k$ results and then average them. Finally, we repeat this for all queries $q_j$ in a set of queries $Q$, and we yet again average the results.

In our evaluation setup, each time we randomly sample $q$ tones to serve as the set of query tones, and we retrieve top $d$ tones to compute the MAP score.

6.6.2 Results

We present the evaluation results of QBC using MAP scores in the CMN corpus in Table 6.4. The $q$ and $d$ values indicates query and retrieved data sizes (for instance we use a set of 20 queries and for each query we retrieved 100 tones in the first configuration in the table). We can see that across the board, SAX-MIDNIST does better than Euclidean distance in retrieving relevant tones in a ranked retrieval task.
Table 6.4: QBC results MAP scores. Data: CMN corpus, q=query size, d=retrieved size.

<table>
<thead>
<tr>
<th>Evaluation/Iteration</th>
<th>SAX-MINDIST</th>
<th>Euclidean</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAP(q=20,d=100)</td>
<td>0.414</td>
<td>0.348</td>
</tr>
<tr>
<td></td>
<td>0.404</td>
<td>0.429</td>
</tr>
<tr>
<td></td>
<td>0.438</td>
<td>0.390</td>
</tr>
<tr>
<td></td>
<td>0.458</td>
<td>0.362</td>
</tr>
<tr>
<td>MAP(q=50,d=200)</td>
<td>0.412</td>
<td>0.404</td>
</tr>
<tr>
<td></td>
<td>0.439</td>
<td>0.415</td>
</tr>
<tr>
<td></td>
<td>0.452</td>
<td>0.466</td>
</tr>
<tr>
<td></td>
<td>0.396</td>
<td>0.401</td>
</tr>
<tr>
<td></td>
<td>0.438</td>
<td>0.418</td>
</tr>
<tr>
<td></td>
<td>0.499</td>
<td>0.400</td>
</tr>
</tbody>
</table>

This result obtained from the CMN corpus is consistent with the results we observed from the Read speech data set in the experiment using K-means clustering.

6.7 Discussion

In the above experiments, we showed how Speech Prosody Mining (SPM) — specifically data mining of tones — could benefit from time-series mining techniques, such as low-dimension symbolic representation of time-series that can exploit computational gains from the data compression as well as the availability of efficient string matching algorithms [GRP+15]11.

We observed one paradox in our evaluation of SAX in the Read speech data set, between supervised and unsupervised learning: the latter is able to achieve better performance with a greater compression rate (0.4) of SAX, whereas the former performs

11We will explore this latter technique in future research.
relatively worse with a higher compression rate (0.7) while requiring a larger value of k (k=7 is with SAX performs comparably with k=1 for other representations). To extend our discussion on this paradox, we further speculate this difference by comparing the nature of the two algorithms, KNN classification and k-means clustering. It is possible that in SAX-MINDIST space, data points have lower distances to cluster centers (in k-means clustering), but higher distances to its nearest k neighbors within the same tone category (that is, comparing to Euclidean distance in KNN). However, this paradox no long exists in the large spontaneous speech data set using the decision tree classifier, where SAX achieved superior performance in both supervised and unsupervised tone learning tasks.

Meanwhile, a property of SAX is that each segment used to represent the original time series must be of the same length. This might not be an ideal situation in many applications (exemplified in Gulati et al. [GRP+15]) where variable-length segments are desired. The complexity of converting to such an variable-length representation may be greater than the original SAX, as one must design some criteria to decide whether the next frame of audio samples belong to the current segment or it should be the start of a new segment. One intuitive strategy is inactivity detection (i.e., flat period can be represented with a single symbol). Moreover, the utility and implications of symbolic representations (equal- or variable-length) for tones and intonation is also of great theoretical interest to phonology\textsuperscript{12}. These research directions are outside the scope of the current work and we will leave them for future research.

\textsuperscript{12}Personal communication with scholars in phonology.
Part IV

Analyzing Tone N-grams
Overview

One of the fundamental problems in tone research is the variability of tone contour shapes in spontaneous speech production. Given a particular tone category and a template contour, we usually observe many different realizations of that template in real speech data. However, the specific mechanisms that give rise to these variabilities are not completely well understood.

In Chapter 2.2.1, we have seen that the tonal context is a strong factor in explaining the variability of tones. Motivated by this phenomenon, in this chapter, we focus on the analysis of tone N-grams — a sequence of N tones (where \( N = 1, 2, 3 \)) that occur consecutively in natural speech\(^{13}\).

In order to efficiently mine tone N-gram data in a large corpus, we need to first develop computational methods targeted at analyzing speech prosody domain time-series patterns. In Chapter 7, we investigate the problem of finding previously unknown patterns (motif discovery) in this domain: how can we modify existing motif discovery algorithm to prune spurious motifs and retrieve real meaningful tone N-gram motifs? In Chapter 8, we look at the predictability of the tone contour shapes: how well can we predict the highly variable tone N-gram contour shape types of real spontaneous speech, using features from a variety of domains with linguistic knowledge? In both chapters we will develop creative solutions to investigate these problems using motif discovery, network (graph) analysis, machine learning, and information retrieval.

---

\(^{13}\)The N-gram modeling is a well known technique in language modeling (LM) in text-domain natural language processing (NLP) with a wide varieties of applications. Here, we borrow the concept to capture the contextual factor of tones.
Chapter 7

From Contours to Categories: Motif Discovery of Tone N-grams

Motif discovery is a well-researched task in time-series data mining [MKZ+09, CKL03]. It aims to find previously unknown patterns that repeat across different parts of the time-series subsequences in a long time-series. These patterns are highly similar to each other (within a range $R$ of each other according to a specified distance measure) and therefore are candidates of patterns that are significant, meaningful and worth further investigation.

In the context of speech prosody data mining, we perform motif and pattern discovery as a main step in the analysis and understanding of the speech prosody time-series subsequences in a corpus. Here, we state the most fundamental hypothesis of the motif discovery in SPM: Meaningful events in speech prosody tend to occur in a repeated pattern more than expected by chance. Therefore, by discovering previously unknown patterns that are highly similar to each other across the speech prosody time-series corpus — within and between speakers — the motif discovery allows us to zoom in on interesting patterns within the data that merits further assessment using linguistic knowledge.

There are two types of motif discovery tasks in the context of SPM, and in particular, for the study of the property of tones. The first is motif discovery on a database of pre-extracted time-series subsequences as input. The time-series subsequences correspond to tone N-grams, where $N = 1, 2, 3$. This allows us to discover only the
time-series subsequence motifs that are N-syllable long and whose boundaries align with the syllable boundaries. We denote this tone \textit{N-grams motif discovery}. Second, we can perform motif discovery taking a long time-series as input. In this mode, the long time-series may be an entire recording of newscast speech from our corpus. It can contain a single speaker or multiple speakers (in which case the pitch values are normalized by speaker). In this case, the motif discovery algorithm extracts time-series subsequences on the fly with a user-specified window length $m$ and a hop size $h$ of 1 sample (i.e., each time the sliding window moves forward one sample, generating adjacent subsequences overlapping $m - 1$ samples). Therefore, in this mode, we can only discover motifs of length $m$ samples, instead of the integer multiple of syllables (i.e., $N$ syllables, or N-grams). If we use downsampled equal-length representation of syllables, we can still get motifs with window length equal to $N$ syllables; however, there is no guarantee that their boundaries align with syllable boundaries. Therefore we denote this mode as \textit{any-length motif discovery}.

Any-length motif discovery is more exhaustive, but due to its massive scale and time complexity, it is hard to experiment with and control the outcome. Therefore, in this chapter, we focus on the tone N-gram time-series motif discovery using the MK algorithm for exact discovery of time-series motifs [MKZ+09].

7.1 Motif discovery: problem definition and algorithms

Motif discovery is the discovery of previously unknown but repeated, highly similar patterns in a time-series database. There are two basic types of motif discovery algorithms: exact algorithms [MKZ+09] vs. probabilistic algorithms [CKL03]. Exact algorithms are guaranteed to find all motifs as specified by the user with no uncertainty. In contrast, probabilistic algorithms are guaranteed to find motifs with a probability
close to 1, for example, 0.95. While exact algorithms are often more desirable, probabilistic algorithms remain useful for very large data sets of time-series databases.

7.1.1 Problem definition

Here, we give a formal definition of the concepts used in motif discovery for clarity [LKLC03], and we comment on the necessity and utility of each definition in plain English.

**Definition 1.** Time Series: A time series $T = t_0, ..., t_m$ is an ordered set of $m$ real-valued variables.

**Definition 2.** Subsequence: Given a time series $T$ of length $m$, a subsequence $C$ of $T$ is a sampling of length $n < m$ of contiguous position from $T$, that is, $C = t_p, ..., t_{p+n-1}$ for $1 \leq p \leq m - n + 1$. This definition is equivalent with the way we have described subsequences above.

**Definition 3.** Match: Given a positive real number $R$ (called range) and a time series $T$ containing a subsequence $C$ beginning at position $p$ and a subsequence $M$ beginning at $q$, if $D(C, M) \leq R$, then $M$ is called a matching subsequence of $C$. This states that given a parameter of distance $R$, a match is defined as all subsequences within distance of $R$ from a seed pattern (motif). In other words, $R$ is a similarity threshold for the grouping of motifs.

**Definition 4.** K-Motifs: Given a time series $T$, a subsequence length $n$ and a range $R$, the most significant motif in $T$ (called thereafter 1-Motif) is the subsequence $C_1$ that has the highest count of non-trivial matches (ties are broken by choosing the motif whose matches have the lower variance). The $K^{th}$ most significant motif in $T$ (called thereafter K-Motif) is the subsequence $C_K$ that has the highest count of non-trivial matches, and satisfies $D(C_K, C_i) > 2R$, for all $1 \leq i < K$. This is used to rank
the significance of the motifs based on the frequency of the occurrence of a motif in the data.

7.1.2 General algorithm

The general obvious algorithm of finding the 1-Motif is to use brute force distance computation and constantly update the 1-Motif candidate — i.e., the motif with the largest count of time-series data points. Figure 7.1 shows the high level code for this algorithm. A generalization to the K-Motif discovery is obvious and omitted here. Note that the motif discovery algorithm takes three inputs: the time-series data (stored as one long sequence or subsequences); $n$, the length of the desired motif; and $R$, the similarity/distance threshold for motifs. This algorithm is an $O(m^2)$ algorithm where $m$ is the length of the entire time-series sequence (i.e., when stored as one long sequence).

7.1.3 MK exact discovery of time-series motifs

The MK exact motif discovery [MKZ+09] is the first tractable algorithm for the exact discovery of time-series motifs. Fundamentally, it leverages the power of pruning the search space by projecting higher dimension time-series data points onto the one dimensional space. By iteratively selecting reference points and computing a ranking of distances from all other time-series objects to the reference points in 1-dimensional space, the algorithm is a vast improvement over brute force algorithm in terms of time complexity. It is an exact algorithm, guaranteed to find the motifs with a probability of 1. Figure 7.2 shows the MK algorithm.
7.1.4 Probabilistic discovery of time-series motifs

The probabilistic motif discovery algorithm [CKL03] is guaranteed to find motifs with a probability close to 1. Inspired by algorithms for DNA motif mining from the bioinformatics community, this algorithm leverages the symbolic representation of time-series to randomly hash partially masked time-series symbols into buckets in order to speed up processing time. The results of the hashing are logged into a collision table marking the indexes of time-series subsequences. This process is called random projection. After a large number of times of running random projection, the cell in the collision table with a higher value indicates the two masked subsequences have been hashed to the same bucket many times and therefore are likely to be a match (motif). This heuristic greatly speeds up the high time complexity required by
**Algorithm** MK Motif Discovery

**Procedure** \( \{L_1, L_2\} = \text{MK}_\text{Motif}(D, R) \)

1. \( \text{best-so-far} = \text{INF} \)
2. for \( i = 1 \) to \( R \)
3. \( \text{ref} = \) a randomly chosen time series \( D_i \) from \( D \)
4. for \( j = 1 \) to \( m \)
5. \( \text{Dist}_{rj} = d(\text{ref}, D_j) \)
6. if \( \text{Dist}_{rj} < \text{best-so-far} \)
7. \( \text{best-so-far} = \text{Dist}_{rj}, \ L_r = i, L_j = j \)
8. \( S = \text{standard deviation}(\text{Dist}) \)
9. find an ordering \( Z \) of the indices to the reference time series in \( \text{ref} \) such that \( S_{Z_1} \geq S_{Z_{i+1}} \)
10. find an ordering \( I \) of the indices to the time series in \( D \) such that \( \text{Dist}_{Z(I,0)} \leq \text{Dist}_{Z((I-1),0)} \)
11. \( \text{offset} = 0, \text{abandon} = \text{false} \)
12. while \( \text{abandon} = \text{false} \)
13. \( \text{offset} = \text{offset} + 1, \text{abandon} = \text{true} \)
14. for \( j = 1 \) to \( m \)
15. \( \text{reject} = \text{false} \)
16. for \( i = 1 \) to \( R \)
17. \( \text{lower bound} = |\text{Dist}_{Z(I,0)} - \text{Dist}_{Z((I-1),0)}| \)
18. if \( \text{lower bound} > \text{best-so-far} \)
19. \( \text{reject} = \text{true}, \text{break} \)
20. else if \( i = 1 \)
21. \( \text{abandon} = \text{false} \)
22. if \( \text{reject} = \text{false} \)
23. if \( d(D_{Z(I,0)}, D_{Z(I,0)+\text{offset}}) < \text{best-so-far} \)
24. \( \text{best-so-far} = d(D_{Z(I,0)}, D_{Z(I,0)+\text{offset}}) \)
25. \( L_r = I(j), L_j = I(j+\text{offset}) \)

---

**Figure 7.2:** MK algorithm for motif discovery of 1-Motif. (Adapted from Mueen et al. [MKZ+09]).
the brute force algorithm. In practice, probabilistic discovery is not exact since its probability of finding true motif is less than 1. However, it is still useful in mining truly massive data sets where the exact algorithm is too slow.

Considering the size of our data set is not astronomical, we choose MK to be our base algorithm for motif discovery.

7.1.5 DTW motif discovery

Both MK and probabilistic algorithms described above depend on Euclidean distance computation at its base. Despite its efficiency, in some domains, Euclidean distance is insufficient to capture the meaningful similarity between time-series objects, where Dynamic Time Warping (DTW) distance is necessary. One example is Gulati et al. [GS14]. However, in Part III of this dissertation, we have shown that this is not the case in the speech prosody domain.

7.2 Research questions

MK is an algorithm for efficient discovery of time-series motifs. Running MK database version off-the-shelf, however, only gives us top motif pairs that looks suspicious (Figure 7.3) and uninteresting. This indicates the challenges we face for the discovery of meaningful patterns from speech prosody data.

One of the primary goals in this dissertation is to identify the unique challenges of time-series data mining in the speech prosody domain and develop appropriate methods to address those challenges. In the N-grams motif discovery task, we define our research questions as follows:

1. What are the unique properties of speech prosody time-series data that pose challenges for motif discovery in this domain?
Figure 7.3: Top five motif pairs (spurious) from the CMN corpus. On the top of each plot: their ranking, distance and tone categories noted.
2. What novel techniques or modifications of existing techniques can solve these problems in order to find meaningful motifs?

3. How can we analyze the tone N-gram patterns to extract meaningful insights and knowledge?

In this chapter, we will address the first two questions to develop effective methods for speech prosody motif discovery of tone N-grams. The final question requires more domain expertise and its solution can vary significantly depending on the specific type of research questions being asked. Therefore, we defer the analysis and extraction of meaningful knowledge from tone N-gram patterns to the next chapter, where we will develop new methods well-suited to this problem.

7.3 Properties of speech prosody time-series motifs

Time-series data exists in many domains. This encompasses a variety of data sources where time-series data can come from. For instance, time-series data from a sensor that records the behavior of insects or from EKG machine recording signals from the heart can have very different properties than data from the domain of music information retrieval (MIR) or speech prosody mining (SPM). In the time-series mining community, the last decade has seen the introduction of hundreds of techniques designed to efficiently measure the similarity between time series with invariance to properties such as (or the various combinations of) the distortions of warping [KWX+09], uniform scaling [YKM+07], offset [FRM94a], phase [KWX+09], and many others.

In this chapter, we discuss two particular types of properties of the speech prosody time-series data: rarity of events and complexity. These two properties are key contributing factors that distinguish the mining of speech prosody time-series data from the type of time-series data that the majority of time-series data mining researchers
are concerned with (such as weather, sensor, etc. data). Investigating the unique properties of the speech prosody time-series data is an important step in both understanding the characteristics of this domain and performing effective data mining and motif discovery.

### 7.3.1 Rarity of events

In time-series data mining community, motif discovery is most commonly defined as finding the most similar pair of time-series subsequences across the database. In fact, the default version of the MK motif discovery algorithm only returns a top pair of motifs that has the lowest distance between them under some distance measure while ignoring all other possibly interesting but lower ranked motifs. Another motif discovery tool, the Matrix Profile\(^1\) algorithm, considers the top three motif pairs and the top discord (discord is the subsequences that have the largest distance to its nearest neighbor in the database, therefore can be used for anomaly detection).\(^2\)

In sum, these algorithms are typically developed in conjunction with data sets from domains where meaningful and significant events in the entire time-series database is rare. An intuitive example is a pattern that signifies an abnormal heart beat in EKG time-series data.

Does this property apply to speech prosody time-series data? Intuitively, it does not, but it is also an empirical question. On the one hand, we note that in the particular case of tone N-grams, the most typical task considered is a classification task: each tone N-gram can be classified into some combination of tone categories. In that sense, each instance of the subsequence can be partitioned into a divided space

---

\(^1\) [http://www.cs.ucr.edu/~eamonn/MatrixProfile.html](http://www.cs.ucr.edu/~eamonn/MatrixProfile.html)

\(^2\) To be fair, Matrix Profile also gives the user the option to discard uninteresting motifs among the top three. However, once you found three interesting enough motifs, there is no straightforward way to search for more.
of classes, and therefore is a meaningful and significant event (as opposed to a data set like EKG signal of heart beats, where we are the most interested in discovering the motifs symptomatic of rare cardiac events). On the other hand, nonetheless, motif discovery — the discovery of previously unknown patterns — can still be of utility in uncovering interesting patterns, based on our basic assumption that significant and interesting speech prosodic events tend to repeat more than expected. However, what we are interested is not to discover a few pairs of motifs; we want to uncover all motif clusters in a database of tone N-grams given a distance threshold (range) for intra-cluster distances.

7.3.2 Time-series Complexity

If we contrast the top ranked spurious motifs from Figure 7.3 to a number of lower ranked real motifs (Figure 7.4) from the same data set, we observe an important fact about the top ranked motifs: they tend to be extremely simple motifs that are almost straight lines.

In fact, the impact of the complexity of time-series data on the outcomes of data mining algorithms was a topic largely omitted until recently [BWK11, GSS15b]. In the research of the current dissertation, we independently discovered that this is an important issue to be considered in the discovery of meaningful time-series motifs in the speech prosody data.

The complexity of time-series data refers to the intuitive definition that some time-series objects have more complex shapes (more peaks and valleys) than others with simpler shapes. In Figure 7.5, we show several shapes in the artificial geometric figures data set from Batista et al. [BWK11] and their corresponding “time” series objects with different levels of complexity. Here, we illustrate the impact of complexity on the distances between objects: the distance between pairs of complex time-series objects
Figure 7.4: Five real motif pairs from the CMN corpus. On the top of each plot: their ranking, distance and tone categories noted.
is frequently greater than the distance between pairs of simple ones. In fact, complex time series are commonly deemed by distance measures (e.g., Euclidean distance) as more similar to simple time series than to other complex time series they look like. This phenomenon is shown when considering the distance matrix of the same data set in Table 7.1. As we can clearly see, whereas shape 24 is clearly more similar to 32 than to 4, the pair 24 and 4 has a lower distance than the pair 24 and 32.

Different levels of complexities of speech prosody N-gram subsequences can cause problems in many time-series mining tasks. To foreshadow the discussion in the next chapters, motif discovery algorithms tend to find very simple motif pairs or clusters as top ranked motifs as they have the lowest intra-motif distance. However, the problem is that these very simple motifs tend to be not real pitch estimations of the speech from the audio but are artifacts of the linear interpolation of the unvoiced segments as a result of preprocessing. These (quasi-) linear motifs are therefore artificial and uninteresting, but they dominate all the top ranked motifs returned by the algorithm.
Table 7.1: Euclidean distance matrix for the geometric figures data set.
(Adapted from Batista et al. [BWK11]).

<table>
<thead>
<tr>
<th></th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>10</th>
<th>12</th>
<th>24</th>
<th>32</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>1.000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>1.318</td>
<td>1.068</td>
<td>1.097</td>
<td>1.217</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td>1.088</td>
<td>1.181</td>
<td>1.199</td>
<td>1.263</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td>1.103</td>
<td>1.195</td>
<td>1.135</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1.198</td>
<td>1.199</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1.186</td>
<td></td>
<td></td>
</tr>
<tr>
<td>24</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1.200</td>
<td></td>
</tr>
<tr>
<td>32</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1.200</td>
</tr>
</tbody>
</table>

Figure 7.6 and 7.7 show the contrast of a simple (artificial) vs. a complex (real) motif cluster (in addition to the motif pairs previously shown) in the tone bi-grams subsequence data set from the CMN corpus. In the rest of this chapter, I will consider the problem of finding real motifs from a database of time-series subsequences through the use of complexity measures and other features in machine learning.

7.4 MK motif discovery algorithm setup and hyper-parameter tuning

We take the original implementation of the MK algorithm\(^3\) and modify/recompile the program as necessary. The original MK algorithm is implemented in C++ and takes only long time-series as the input as it is hardwired to extract time-series subsequences with a hop size of 1 sample throughout the entire long time-series. We therefore modified the code in order to perform N-grams motif discovery taking a pre-extracted database of subsequences as input.

\(^3\)Acquired online at http://alumni.cs.ucr.edu/~mueen/MK/.

115
Figure 7.6: A top-ranked simple (linear) spurious and artificial motif cluster from the CMN corpus.

Figure 7.7: A real motif cluster from the CMN corpus.
In our modified version of the MK algorithm (database version), there are three parameters to be set: $X$, the range of the motif cluster (i.e., all subsequences within distance $X$ of the motif pair would be included in the motif cluster); $k$, the number of motif clusters the user would like to find; and $R$, the number of reference points in the MK motif discovery algorithm.

The MK documentation has provided a guideline for parameter setting: for $X$, they suggest to start with 2 and increase to 3; for $k$, it should be up to the user’s needs; for $R$, they showed that [MKZ+09] the exact value of $R$ does not affect the time performance of the algorithm as long as it is greater than 5, and they strongly suggest $R = 10$. In the experiments, we follow their suggestions in general but also experiment with different sets of parameter settings.

7.4.1 Experiments on the value of $X$

Running the MK with $X = 2$ is in general a good starting point, as the authors of the MK algorithm have suggested. Nonetheless, we have several observations regarding running MK with different values of $X$. First, we observe that the value of $X$ is negatively correlated with the number of motif clusters found. A lower value of $X$ will in most cases yield more numerous motif clusters from the same data set. Second, the value of $X$ also positively correlate with the size of the motif clusters found. Taken these two points together, we see that using smaller $X$ will cause MK to discover more numerous but small motif clusters, whereas a bigger $X$ will lead to fewer but bigger clusters. If we go back to the original definition of $X$ as a range parameter, this behavior is predictable because a smaller range will produce many highly similar (low intra-cluster distance) and compact motif clusters, and vice versa. Therefore, we can think of the fewer yet bigger sized clusters found with a large $X$ as lumping together
many smaller clusters from the smaller $X$, as many small clusters may merge into one
big cluster as the range $X$ increases.

The implication of this observation is twofold. First, it shows the tradeoff between
finding more motifs and the size of each motif cluster. Second, it is also a tradeoff
between intra-cluster distances and inter-cluster distances. The question is, is it more
sensible to find more numerous tight clusters or fewer but more tolerant clusters?
Considering the stochastic and analog nature of the speech production, we tend to
prefer the latter type. That is, considering $f_0$ patterns produced by the human vocal
fold will always contain some stochastic noise, it makes more sense to loosen the
distance threshold of range $X$, rather than requiring motifs to be the near-exact
copies of each other. Moderate variances should be allowed in a meaningful motif
cluster of $f_0$ patterns. On the other hand, as $X$ increases we can have a big cluster
that contains many motifs that are too dissimilar to each other. Therefore, we still
need to find a balance point given the tradeoff.

The third observation is that when $X$ is fixed, the number of motifs ($num_{\text{motif}}$)
found also correlates negatively with the length of time-series subsequences ($len_{\text{ts}}$).
This effect is demonstrated in Table 7.2 (note the contrast in the number of motifs
discovered when $X=2$ or 3 and $len_{\text{TS}}$ varies from 100 to 200). Intuitively, this is
because the distances from a data set tend to increase when $len_{\text{TS}}$ becomes larger,
therefore under the threshold of the same range $X$, less number of subsequences make
the cut into the motif cluster (and note that MK algorithm did not normalize for the
different values of $len_{\text{TS}}$). To confirm this suspicion, we have computed the pairwise
distances of 50,000 randomly sampled subsequence pairs from each data set of bigram
100p, 200p, and 300p. Figures 7.8, 7.9, and 7.10 show the distance distribution of these
three data sets (x axis is Euclidean distance). From these plots we confirm that the
distances increase when $len_{\text{TS}}$ increases. Therefore, we conjecture that when we
increase $X$, we are able to find more motifs. To leverage this effect, we compute that $X = 2$ translates into excluding 99.9% of the subsequence pairs when $len_{TS}$ equals 100 (as they have a distance greater than this threshold), and we similarly experiment with a 0.1% in the cumulative distance distribution for $len_{TS}$ equals 200 and 300, using $X = 3$ and 4 respectively. Our hope is that this will yield more motifs from these data sets.
Figure 7.10: Distance distribution of 50,000 randomly sampled pairwise distances in the B100p data set.

Table 7.2: Number of motif clusters discovered varies with \textit{len TS} and X.

<table>
<thead>
<tr>
<th>X</th>
<th>No.motif clusters</th>
<th>Len(TS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.6</td>
<td>90</td>
<td>200</td>
</tr>
<tr>
<td>2</td>
<td>48</td>
<td>200</td>
</tr>
<tr>
<td>2.2</td>
<td>44</td>
<td>200</td>
</tr>
<tr>
<td>2.5</td>
<td>33</td>
<td>200</td>
</tr>
<tr>
<td>3</td>
<td>22</td>
<td>200</td>
</tr>
<tr>
<td>2</td>
<td>90</td>
<td>100</td>
</tr>
<tr>
<td>3</td>
<td>33</td>
<td>100</td>
</tr>
</tbody>
</table>
However, it turns out that this factor tends to be quite irrelevant in the presence of the effect of $X$’s negative correlation with $\text{num\_motif}$. This effect is again showing in Table 7.2: as $X$ increases, we see that the $\text{num\_motif}$ decreases, despite the distance effect discussed in the last paragraph. We conclude that the distance effect with $\text{len\_TS}$ is quite diminishing in the presence of the tradeoff associated with $X$ and inter- / intra- cluster distances and in turn with the $\text{num\_motif}$.

Given these observations, we now move on to propose methods for finding the appropriate value of $X$ that satisfies several conditions. First, our objective is to find maximally distinct motif clusters (inter-cluster distance is big). Second, given the same level of distinctiveness, we want to find more motifs. Alternatively, as a balance point given the tradeoff, we want to find motif clusters with high ratio of inter- to intra- cluster distances.

We first propose a Motif Cluster Quality Measure (MCQM) based on the first two conditions above:

$$MCQM = ICD \times \log_{10}\text{num\_motifs}$$

(7.1)

where the ICD is the mean Inter-Cluster Distances given all motif clusters found at a fixed value of $X$.

Next, having computed the MCQM and the inter- to intra- cluster mean distance ratio, we combine the two measures:

$$MCQM\_ratio = MCQM \times (1/100 + IIC\_ratio/10)$$

(7.2)

where $IIC\_ratio$ is the Inter- to Intra-Cluster mean distance ratio given a fixed value of $X$. The $MCQM\_ratio$ measure is intended to combine the three conditions outlined above when evaluating the optimal value of $X$, while giving different weights
Figure 7.11: Experiments on the value of $X$ and motif cluster quality measures in the B100p data set.

to these considerations. Figure 7.11, 7.12, 7.13 show the evolution of inter-cluster distance (mean), number of clusters, inter-to-intra cluster distance ratio, MCQM, and MCQM_ratio measure given the different values of $X$ in the bigram-100p, 200p, and 300p data sets (values are normalized to be plotted in the similar range). We can clearly see the tradeoff between these different parameters, and how the MCQM_ratio measure combines them into a single decision. We therefore pick $X = 2.2, 2, 2$ for len_TS of 100, 200, and 300 respectively. In addition, the decision can be automatically made given any other values of len_TS of unseen data sets.

7.4.2 Varying the value of $k$

When we follow MK’s suggestion and fix the value of $X$ at 2, we first set $k$ to 200 in order to exhaustively retrieve the maximum possible number of motifs. The result depends on the specifics of the data set, the length of time-series subsequences in the data set, the value of $X$, and the size of the data set. In general, in a data set...
Figure 7.12: Experiments on the value of X and motif cluster quality measures in the B200p data set.

Figure 7.13: Experiments on the value of X and motif cluster quality measures in the B300p data set.
that has about 100,000 subsequences (or the same order of magnitude), the number of distinctive motifs that the MK algorithm discovers has an upper limit of about 100. If running MK only once, then at least 30% of those are linear motifs (spurious), which should be pruned off. In the following sections, we will focus on the details of pruning these spurious motifs in conjunction with running MK iteratively while varying $k$ from 30 to 100.

7.5 Distinguishing spurious and real motifs

In this section, we analyze the properties and characteristics of motif clusters from the output of the MK algorithm on databases of tone N-gram subsequences. We then propose several measures to both quantitatively describe these characteristics, and utilize them to reach our end goal of filtering out spurious motifs and finding real motifs.

7.5.1 Annotating motif classes

Our first task in distinguishing spurious from real motifs is to define what constitutes a spurious motif versus a real motif. In order to have a better understanding of the output of the MK motif discovery algorithm, we run the algorithm with different configurations and have a human annotator annotating the motif classes. Using the B100p data set, we observe that when $k = 30$, all motif clusters are strictly linear, looking similar to the one in Figure 7.6. We therefore use $k = 200$ and $X = 2$, in order to exhaustively find all motif clusters. The result shows that in this data set, given the subsequence length (200) and size ($\sim80,000$) of the data set, the algorithm would exhaust and stop returning distinctive motif clusters at around 110 — i.e., it only finds 110 distinctive motif clusters given the constraints of the range parameter
$X = 2$. This configuration therefore gives us an overview of the range of motif clusters in terms of its goodness and naturalness.

We have shown the two classes of motifs previously: a spurious motif (also known as a linear motif, Figure 7.6) and a real motif class (also known as a non-linear motif, Figure 7.7). However, through annotating the 110 motif clusters found in the B100p data set, we have found another class of motif: the ambiguous motifs. We also refer to them as q-linear (quasi-linear) motifs. Figure 7.14 shows an example of q-linear motif cluster. As the name suggests, q-linear motifs are mostly linear (i.e., consider a linear segment that occupies at least more than half of the motif length), and it is ambiguous to annotate them as whether a real motif or an artificial (bad) motif. Therefore, in a real-world application scenario, we would hesitate to prune them out completely and we also do not want to say they are real motifs of interest. Our strategy is to first annotate the motif cluster results using the three-class distinction. This distinction can be maintained in real applications as well: when presenting to the user, we rank them as potentially of interest in descending order: non-linear, q-linear, linear. In the following sections we also present further analysis as evidence to justify this ranking.

In Figure 7.15, we show the distribution and correlation between motif cluster ranking and classes T100p corpus, where the bars correspond to the first, second, third, and fourth quarter of the data according to motif ranking. Interestingly, the annotated linearity classes are distributed according to the rank of the motif clusters: top ranked motif clusters tend to be linear; mid-ranked clusters tend to be q-linear; low-ranked clusters tend to be linear. There are some transitional regions where classes maybe mixed; however, the trend is consistent across different data sets (Figure 7.16, 7.17, 7.18).

In the following sections our primary goal is to develop quantitative measures in order to automatically learn to classify these classes. For this task, we have annotated
Table 7.3: Motif class datasets annotated. L=linear, QL=q-linear, NL=nonlinear, X=2, k=200 for number of motif clusters found by MK algorithm. A bad cluster is a cluster where members have dissimilar shapes visually so that they should not be a motif — this is a motif range problem and usually with X=2 there are less than 5 bad motif clusters ranked lowest in discovered motifs. This can be resolved by experimenting with smaller value of X.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Size</th>
<th>TS length</th>
<th>#motif clusters</th>
<th>L</th>
<th>QL</th>
<th>NL</th>
<th>bad</th>
</tr>
</thead>
<tbody>
<tr>
<td>B100p</td>
<td>88440</td>
<td>100</td>
<td>107</td>
<td>43</td>
<td>19</td>
<td>40</td>
<td>5</td>
</tr>
<tr>
<td>B200p</td>
<td>85819</td>
<td>200</td>
<td>74</td>
<td>34</td>
<td>10</td>
<td>24</td>
<td>6</td>
</tr>
<tr>
<td>T200p</td>
<td>77588</td>
<td>200</td>
<td>48</td>
<td>10</td>
<td>5</td>
<td>31</td>
<td>2</td>
</tr>
<tr>
<td>T300p</td>
<td>76586</td>
<td>300</td>
<td>39</td>
<td>10</td>
<td>6</td>
<td>22</td>
<td>1</td>
</tr>
<tr>
<td>Pooled</td>
<td>n/a</td>
<td>n/a</td>
<td>268</td>
<td>97</td>
<td>40</td>
<td>117</td>
<td>14</td>
</tr>
</tbody>
</table>

four data sets and pool them together as the pooled data set. Table 7.3 shows the details of these data sets.

7.5.2 BWK complexity measure

There are many complexity measures for time-series, such as Kolmogorov complexity, many variants of entropy, the number of zero crossings, etc. Batista et al. [BWK11] considers the desirable properties any such complexity measure should have: (1) it should have low time and space complexity; (2) it should have few parameters, ideally none; (3) it should be intuitive and interpretable. Given these considerations, we adopt an intuitive measure of the complexity of time-series proposed by Batista et al. [BWK11] (here on referred as BWK complexity measure). It has O(1) space and O(n) time complexity, is completely parameter-free, and has a natural interpretation. It is empirically tested to be one of the best complexity measures, even though it has not been claimed to be “optimal”.

126
Figure 7.14: A \(q\)(uasi)-linear motif cluster from the B100p data set.

Figure 7.15: Stacked bar chart showing the distribution and correlation between motif cluster ranking and classes in the T200p data set.
Figure 7.16: Stacked bar chart showing the distribution and correlation between motif cluster ranking and classes in the T300p data set.

Figure 7.17: Stacked bar chart showing the distribution and correlation between motif cluster ranking and classes in the B200p data set.
Figure 7.18: Stacked bar chart showing the distribution and correlation between motif cluster ranking and classes in the B100p data set.

The complexity measure is simply defined as the squared sum of the differences between adjacent points in the entire time-series:

$$CE(Q) = \sqrt{\sum_{i=1}^{N-1} q_i^2 + q_{i+1}^2}$$

(7.3)

where $Q$ is a time series $q_1, q_2, ..., q_N$. Intuitively, this approach is based on the physical intuition that if we could stretch a time series until it becomes a straight line, a complex time series would result in a longer line than a simple time series [BWK11].

Despite being evaluated on many time-series data sets as an effective measure of complexity [BWK11, GSS15b], the BWK complexity measure has shortcomings to be used in the current tone N-gram motif mining task. First, we note that the z-score
normalization (which centers the time-series subsequence at mean of 0 and standard deviation of 1) must be applied in order for the measure to work properly. This is due to the fact that the BWK method only measures the squared distance between a sample and its neighbors while ignoring the direction of the difference. Therefore we could easily make up a counter-example where the pair of time-series objects obviously have different complexity but are assigned the same scores. Figure 7.19 shows such an example — and this problem is fixed when z-score normalization is applied (Figure 7.20). However, as discussed in earlier chapters, in tone $f_0$ data, it is beneficial to avoid z-score normalization but preferable to simply apply a subtract-mean normalization due to the existence of quasi-flat contours with low values of standard deviation. Moreover, we could imagine a scenario where the two subsequences in Figure 7.19 are in fact post-normalization. In that case, we have no way of correctly detecting their complexity ranks.

There is a second problem of BWK that cannot be easily fixed. Intuitively, we want a complexity measure that is invariant to the length of the time-series subsequence, or invariance to sampling rate. For example, if we have two versions of the same subsequence with length 100 and 200, we want a measure that can assign the same complexity score to both\textsuperscript{4}. That can be easily achieved by normalizing the complexity score by a scaler/factor in proportion to the length — if the scores behave in a linear way in the first place. However, the BWK does not behave linearly in such a fashion. Concretely, if we take a subsequence and compute the BWK score of it as well as the BWK score of the first half of the subsequence, we expect the latter to have half the BWK score of the former. However, that does not hold for BWK if the

\textsuperscript{4}In the same database usually the subsequences are all of the same length, which is why this has not caused a problem when evaluated within a data set. However, our goal is to train a classifier that can work across different data sets of arbitrary length of subsequences, thus the invariance to sampling rate becomes important.
Figure 7.19: Example of BWK complexity on z-normalized time-series pair (inconsistent with intuition).

Figure 7.21 shows such an example from the CMN corpus where a “stretched” version (blue) of the same subsequence (red) having a much lower BWK score due to its smaller slope. Figure 7.22 shows a generalized example of two linear time-series subsequences of the same complexity (visually) but are assigned drastically different BWK scores due to their slope difference.

Due to these shortcomings, we propose a novel complexity score in the next section that has the desired property of invariance to sampling rate.

7.5.3 Polynomial regression based complexity measure: LSSE

In this section, we propose a novel complexity measure for time-series subsequences in order to achieve the invariance to sampling rate. This algorithm is based on the intuition that a less complex time series will have lower sum of squared error value when we fit a linear regression line to the data points. We call this algorithm the Lowest Sum of Squared Error (LSSE) algorithm.

The LSSE algorithm works as follows:
Figure 7.20: Example of BWK complexity on z-normalized time-series pair (consistent with intuition).

Figure 7.21: Time-series subsequences with different slopes. Blue has a smaller slope than red, therefore it has a smaller BWK score (0.250) despite it being twice as long than the red (0.359).

Figure 7.22: Time-series subsequences with different slopes: generalization. BWK(blue)=0.6, BWK(red)=0.12.
(1) We fit a linear regression line to the time series, where $y_i$ is the $i^{th}$ data point and $x_i$ is the index $i$ of the data point. Therefore, we can approximate $y$ (written as hypothesis $h(x)$) as a linear function of $x$, where $\theta$ is a matrix of weights:

$$h(x) = \sum_{i=0}^{n} \theta_i x_i = \theta^T x$$  \hspace{1cm} (7.4)

And we define the cost function as:

$$J(\theta) = \frac{1}{2} \sum_{i=0}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$$  \hspace{1cm} (7.5)

We then use stochastic gradient descent to minimize the cost function. In the current experiments we use implementation available in `numpy.polyfit` in Python's `numpy` library.

(2) We compute the sum of squared errors (SSE) for all data points $y_i$, for $i = 1$ to $N$, where $N$ is the length of a time-series subsequence:

$$SSE = \sqrt{\sum_{i=0}^{N} (y_{pred}^{(i)} - y^{(i)})^2}$$  \hspace{1cm} (7.6)

(3) We find the factor $k$ to normalize the time-series by its length $N$. The factor is computed as the maximum value that is smaller than 10 when $N$ is iteratively divided by 10 (the iteration stops when the value is less than 10). The LSSE complexity measure is then given by:

$$LSSE = SSE/k$$  \hspace{1cm} (7.7)

Initial experimentation shows this method is invariant to sampling rate, and it performs favorably in motif classification tasks comparing to BWK. We will present a systematic evaluation against BWK in the evaluation section.
7.5.4 Tone Label Consistency (TLC) score

In previous chapters, we have indicated that linear motifs are in general spurious and less reliable / interesting than non-linear motifs because they are not true representations of pitch estimation but the result of interpolation. Also, we distinguished these two classes from a third “ambiguous” class, which has a substantial linear segment that accounts for at least 50% of the length of the time series. How can we decide the goodness or interestingness of such motifs? Should they be presented to the user as motif cluster candidates for further investigation? Obviously, the answer to this question will depend on the specific research task and data set, but in the current stage, the first thing we can do is to utilize the ground truth of tone N-gram labels within a motif cluster and evaluate whether they have high consistency within the cluster\(^5\).

The Tone Label Consistency (TLC) score takes an array of tone N-gram labels from a motif cluster and measures how consistent the labels are. Concretely, it computes the pairwise similarity of tone N-gram labels and aggregate them. Given a motif cluster \(C\) with \(m\) time-series subsequences, the label of the \(i^{th}\) member of the cluster has a label in its \(k^{th}\) position of the N-gram label as:

\[
L_{ki}, i \in \{1, 2, ..., m\}; k \in \{1, 2, ..., N\}
\]

(7.8)

And the \(TLC_{raw}\) score is defined as:

\[
TLC_{raw}(C) = \sum_{k=1}^{N} \sum_{\text{all } i,j\in\{1,2,...,m\}} I\bullet L_{ki}=L_{kj}
\]

(7.9)

\(^5\)In this chapter, we are in general only concerned with finding real motif clusters instead of asking the question of what is a meaningful or interesting motif cluster. Therefore, in this case, we are not saying that a motif cluster of tone N-grams with highly consistent tone labels are necessarily more interesting to look into than those without. Rather, we are using this intuitive metric of tone label consistency to provide justifications (or falsifications) for our division of classes and their reliability as a good potential motif cluster to look into.
where $I$ is an indicator function as follows:

$$
\begin{align*}
0 & \quad \text{if } L_{ki} \neq L_{kj} \\
1 & \quad \text{if } L_{ki} = L_{kj}
\end{align*}
$$

Finally, we further normalize the $TLC_{\text{raw}}$ score so that its values range between 0 and 1. In order to achieve that we divide the $TLC_{\text{raw}}$ score by the maximum possible score value considering all pairs of matching labels in all $N$ positions:

$$
TLC(C) = \frac{TLC_{\text{raw}}}{\binom{m}{2} \times N}
$$

### 7.5.5 Correlation between TLC and Complexity score

Having proposed two features to quantify the characteristics of the motif classes, we report some initial analysis in order to better understand the distribution of complexity (using LSSE$^6$) and TLC scores among the three classes. All analyses are done using a ground-truth annotated, pooled data set of tone N-grams (see Table 7.3 for an overview of data sets), including B100p, B200p, T200p, and T300p motif cluster data sets.

First, we show that the complexity score is negatively correlated with motif cluster ranks: higher ranked motif clusters have low complexity scores. We can see this is indeed the case across different data sets (Figures 7.23, 7.24, 7.25, 7.26,).

Second, we plot the complexity score distribution among the three classes. As seen in Figure 7.27, the three classes are relatively well separated by the LSSE complexity metric. Figure 7.28 shows the boxplot of the complexity scores.

---

6The results obtained from using BWK complexity measure is in general consistent with those obtained from LSSE if we analyze a single data set. However, in the current analysis we analyze a pooled data set of different time-series subsequence lengths, therefore using LSSE gives us more consistent results due to its invariance to sampling rate.
Figure 7.23: Correlation between rank and complexity in the T300p data set.

Figure 7.24: Correlation between rank and complexity in the T200p data set.
Figure 7.25: Correlation between rank and complexity in the B200p data set.

Figure 7.26: Correlation between rank and complexity in the B100p data set.
Figure 7.27: Distribution of LSSE complexity scores among the three motif cluster classes.

Figure 7.28: Boxplot of LSSE complexity scores among the three motif cluster classes.
As discussed previously, TLC is an intuitive and simple indicator of the quality of the motif clusters: we hypothesize that a random, spurious cluster is of lower quality than a real motif cluster, and the TLC score should be able to capture that. Here, we take our three annotated classes, linear, q-linear, and non-linear, and we conjecture that the quality of motif clusters from these three clusters should increase linearly. Our plot of the distribution of TLC scores among the three classes in Figure 7.29 supports this conjecture, even though the classes are much less well separated by TLC than the LSSE complexity score. This result also provides support to our division of the three classes and their rank of reliability as a good motif cluster. However, we also note that the correlation between motif cluster rank and the TLC score is weaker than those of complexity scores. We show this correlation in Figures 7.30, 7.31, 7.32, 7.33.

Finally, we note that the LSSE complexity score is correlated with the TLC score (Figure 7.34). This correlation again provides justification for the division of the three
Figure 7.30: Correlation between rank and TLC in the B200p data set.

Figure 7.31: Correlation between rank and TLC in the B100p data set.

Figure 7.32: Correlation between rank and TLC in the T200p data set.
Figure 7.33: Correlation between rank and TLC in the t300p data set.

classes and the use of complexity and TLC scores in quantifying the characteristics of motif clusters.

7.6 LEARNING TO CLASSIFY MOTIFS

In this section, we describe a machine learning module we use to perform classification on motif clusters. In the last few sections, we have proposed several features to characterize the motif clusters obtained from running the MK motif discovery algorithm on a database of tone N-gram time-series subsequences. Now we proceed to design a series of classification tasks in order to learn the motif classes. In the next section, we plug these classifiers into a framework for refining motif cluster ranks in order to find top ranked true motifs, our final goal for the tone N-gram motif discovery task.

The features we use include complexity features (computed for each member of a motif cluster, then averaged to derive the average complexity of the cluster) and the TLC feature (computed per motif cluster), which measures how consistent a
motif cluster’s tone labels are. For complexity features, we have two varieties: BWK and LSSE. To foreshadow the experimental results reported in evaluation, these two features capture different aspects of the complexities of time-series and can be used in conjunction or individually along with TLC. For TLC, we follow our proposed measure as described in Chapter 7.5.4. These features and their acronyms to be used later in the reports are summarized in Table 7.4. We will defer the discussion of how to combine these features under different machine learning algorithm frameworks to the evaluation section.

We propose two frameworks to perform the classification tasks. In the first framework, we simply perform a three-class classification to distinguish the linear, q-linear, and non-linear classes. We call this a 3-class task in the Classification Framework (CF).

In the second framework (Iterative Pruning Framework, or IPF), we first perform a 2-class classification task (referred to as 2C1) to classify the motif clusters into

![Figure 7.34: Correlation between LSSE complexity and TLC in pooled data set.](image)
either linear (spurious) or others class. Then in a second step, we perform another 2-class classification (2C2) to classify the remaining data into either qlinear or non-linear class. The rationale for this framework is two fold. First, we observe that in certain data sets (ex: Figure 7.35), when using BWK complexity measure, the linear class can be best separated from the other classes whereas there is not a good separation between the q-linear and non-linear classes. We therefore would like to explore if adding a second feature (TLC) would help improve the classification of q-linear and non-linear classes. Second, and more importantly, we found that an iterative approach to prune spurious motifs first and re-run MK algorithm on the pruned data set would result in more real motifs being discovered from running the MK algorithm. In that case, we first perform an iterative pruning step until the MK algorithm stops outputting spurious (linear) motif clusters, according to the 2C1 classifier. Then we perform the 2C2 classification in order to distinguish q-linear (ambiguous) from non-linear (real) motifs. We will further discuss CF and IPF in the next chapter.
Table 7.4: Features used in classification.

<table>
<thead>
<tr>
<th>Features</th>
<th>Acronym</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average Complexity BWK</td>
<td>AC_B</td>
</tr>
<tr>
<td>Average Complexity LSSE</td>
<td>AC_L</td>
</tr>
<tr>
<td>Tone Label Consistency</td>
<td>TLC</td>
</tr>
</tbody>
</table>

Table 7.5: Classification tasks overview.

<table>
<thead>
<tr>
<th>Framework</th>
<th>Task</th>
<th>Acronym</th>
</tr>
</thead>
<tbody>
<tr>
<td>CF</td>
<td>3-class classification</td>
<td>3C</td>
</tr>
<tr>
<td>IPF</td>
<td>2-class (linear,others) classification-1</td>
<td>2C1</td>
</tr>
<tr>
<td>IPF</td>
<td>2-class (qlinear,nonlinear) classification-2</td>
<td>2C2</td>
</tr>
</tbody>
</table>

7.7 Developing algorithms for refining motif cluster ranks

We have seen that despite being an efficient algorithm for motif discovery, the MK algorithm cannot be used as an off-the-shelf algorithm for the motif discovery of tone N-grams due to its tendency to retrieve extremely simple motif as top ranked motif clusters. Our goal for this chapter is therefore to develop methods for refining the motif cluster ranks to automatically identify real motif clusters to be returned as top ranked motifs.

Having annotated training data sets with a three-class division and developed complexity measures (BWK, LSSE) and a cluster-quality metric (TLC), we now proceed to put them together and outline the variation of algorithms for refining the ranks for motif clusters in CF and IPF frameworks.
Figure 7.36: Iterative Pruning Framework (IPF).
The input to the following algorithms will be the output of ranked motif clusters after running MK algorithm once on a data set of tone N-grams (initial setting $X = 2, k = 200$). The output will be a ranked array of motif clusters. We propose five variations of our rank refinement algorithms in Chapter 7.8 (assuming we’ve already trained the classifiers listed in Table 7.5).

7.8 Rank refinement algorithms

These algorithms assume constant hyper-parameter values over different iterations given a data set.

1. Classification Framework: 3-class classification

   (a) Perform 3-class classification on the input motif clusters and return the re-ranked clusters, where classes are ranked in the order of non-linear, q-linear, and linear. Two clusters with the same class label are ranked by their original ranks in the input.

   (b) We denote this method by “3C”.

2. Iterative Pruning Framework: Exhaustive pruning. This algorithm is a generative IPF algorithm outlined in the block diagram of Figure 7.36. This algorithm is also outlined in pseudocode in Algorithm 17.

   (a) Perform 2C1 on the input motif clusters and prune linear motif clusters.

   (b) Update the tone N-gram database.

   (c) Run MK algorithm again on the updated input data.

---

7The pseudocode for this algorithm is written as taking the original tone N-gram database as input, whereas all verbose descriptions of algorithms in this chapter assumes the output of one iteration of MK algorithm as the input.
(d) Repeat the last three steps until there are no motif clusters classified as linear from the output of the MK algorithm.

(e) Perform 2C2 on the remaining data and return re-ranked clusters, where classes are ranked in the order of non-linear, q-linear, and linear. Two clusters with the same class label are ranked by their original ranks in the input.

(f) We denote this method by “IPF”.


(a) Perform 2C1 on the input motif clusters and prune linear motif clusters.

(b) Update the tone N-gram database.

(c) Run MK algorithm again on the updated input data.

(d) Perform 3C on the remaining data and return re-ranked clusters, where classes are ranked in the order of non-linear, q-linear, and linear. Two clusters with the same class label are ranked by their original ranks in the input.

(e) We denote this method by “IPF-1”.

7.9 Evaluation of motif discovery rank refinement

7.9.1 Data sets

The data sets used in N-grams motif discovery vary in several aspects. First, they vary in the value of $N$: $N = 1$ for unigram, $N = 2, 3$ for bigram and trigrams, respectively. Second, they vary in the length of the downsampled time-series subsequences. We denote this length by $dl$ (downsample length). The value of $dl$ represents the
**Algorithm 1** Iterative Pruning General Algorithm (Exhaustive)

1: procedure IterPrune(subseq_database)
2:    spurious ← True
3:    all_inds_rm ← initialize dynamic array
4: repeat
5:    motif_clusters ← run MK_DB algorithm on subseq_database
6:    for each motif_cluster in motif_clusters do
7:        motif_class ← predict/classify class of the motif cluster
8:        if motif_class == linear then
9:            indexes ← get the indexes of the motif cluster
10:           all_inds_rm+ = indexes
11:        if sizeof(all_inds_rm)!=0 then
12:           update subseq_database
13:        else
14:           spurious ← False
15:    until spurious == False
16: newRank ← 2C2_classify (motif_clusters)
17: return newRank

sampling rate of the downsampled pitch time-series. Its choice is constrained on the one hand by the faithfulness to the original sampling rate of pitch estimation (high dimensional), and on the other hand, by the efficiency of an economic representation (low dimensional). The choice of $dl$ also depends on the value of $N$. In previous chapters, we have used $dl = 30$ for experiments time-series data mining. In the current experiments, we also explore more values of $dl$: 100, 200, 300, and 400 for bi-grams and trigrams. Finally, it is noted that the number of time-series subsequences after applying downsampling will depend on the value of $dl$ as well: since we are not performing upsampling, any time-series subsequences with a length less than $dl$ will be eliminated from the data set.
7.9.2 Experiment setup

We propose a Motif Retrieval Task (MRT) as the evaluation for the N-grams motif discovery. The task is analogous to the Query By Content (QBC) task in Chapter 6.6 of the dissertation, conceptualized from a information retrieval perspective. In the Motif Retrieval Task (MRT), our input is the database of time-series subsequences (in the current context, a database of tone N-grams). The goal of the task is to retrieve the real motifs from the database as the top ranked motifs. This is analogous to retrieving the most relevant documents as top ranked documents in Information Retrieval. Since the user may be interested to find a variable number of top $K$ interesting motifs, we cannot specify a value for $K$ a priori. Therefore we use the Mean Average Precision (MAP) score as our evaluation metric again.

We first run classification experiments on the three classification tasks proposed in Table 7.5. In these experiments, our first goal is to evaluate the effectiveness of using complexity and TLC features. Second, we evaluate the competing complexity measures BWK and LSSE. Third, we experiment with several different classification algorithms, including Support Vector Machine (SVM), Decision Tree and Random Forest. Fourth, we observe that the 2C2 data sets have imbalances in number of training examples, with the q-linear class being the minority class. This can lead to false impression of high accuracy scores while introducing more false negatives for the q-linear class. We therefore run additional experiments on a corrected data set (pooled) in order to train a more effective classifier evaluated on F1 score. The corrected data set is obtained through upsampling the minority class (sample twice for each training example). This turns out to be an effective strategy for obtaining higher F1 scores for the 2C2 task. Finally, to select the best classifier in practice, we test classifiers extensively on an unseen test set with 90 motif clusters with different com-
binations of classification strategies, including probability threshold, feature scaling, and SVM kernel.

We perform classification experiments individually on our four training sets using 10-fold cross-validation, as well as the pooled data set. Finally, we pick the best performing classifier for the pooled data set to use as our classifier module for the rank refinement algorithms.

We run experiments using the three variants of the rank refinement algorithms outlined in Chapter 7.8.

7.9.3 Results: Training classifiers

First, we present the results of classification experiments in both CF and IPF. All results reported are from experiments conducted using 10-fold cross-validation.

We report the first round experiments using accuracy scores as the evaluation metric. The results for the four annotated data sets as well as the pooled data set are reported in Tables 7.6, 7.7, 7.8, 7.9, and 7.10. We use the pooled data set to select our classifiers.

First, we note that the LSSE complexity measure consistently outperforms BWK in all classification tasks by a large margin. We therefore adopt LSSE as our complexity measure for subsequence experiments. Second, we observe that the addition of the TLC feature often results in a boost in classification accuracy while accompanying BWK feature. However, it did not help improve accuracy significantly in most cases when using LSSE as the complexity feature. We also observe that by incorporating TLC, we narrow the confidence interval of the accuracy visibly. In our final task of rank refinement, we evaluate both of the best performing one-feature and two-feature classifiers obtained here in order to make a selection based on its performance in
Table 7.6: Classification accuracy B200p (SVM/Decision Tree).

<table>
<thead>
<tr>
<th>Features/task</th>
<th>2C1</th>
<th>2C2</th>
<th>3C</th>
</tr>
</thead>
<tbody>
<tr>
<td>BWK</td>
<td>0.61/0.65</td>
<td>0.7/0.75</td>
<td>0.56/0.62</td>
</tr>
<tr>
<td>BWK+TLC</td>
<td>0.69/0.74</td>
<td>0.7/0.82</td>
<td>0.62/0.65</td>
</tr>
<tr>
<td>LSSE</td>
<td><strong>0.82/0.82</strong></td>
<td>0.93/1.0</td>
<td>0.81/0.83</td>
</tr>
<tr>
<td>LSSE+TLC</td>
<td>0.82/0.81</td>
<td>0.93/1.0</td>
<td>0.8/0.78</td>
</tr>
</tbody>
</table>

Table 7.7: Classification accuracy B100p (SVM/Decision Tree).

<table>
<thead>
<tr>
<th>Features/task</th>
<th>2C1</th>
<th>2C2</th>
<th>3C</th>
</tr>
</thead>
<tbody>
<tr>
<td>BWK</td>
<td>0.71/0.72</td>
<td>0.68/0.55</td>
<td>0.65/0.53</td>
</tr>
<tr>
<td>BWK+TLC</td>
<td><strong>0.78/0.69</strong></td>
<td>0.68/0.58</td>
<td>0.64/0.58</td>
</tr>
<tr>
<td>LSSE</td>
<td>0.92/0.88</td>
<td><strong>0.83/0.78</strong></td>
<td><strong>0.82/0.75</strong></td>
</tr>
<tr>
<td>LSSE+TLC</td>
<td><strong>0.93/0.89</strong></td>
<td>0.83/0.78</td>
<td>0.81/0.73</td>
</tr>
</tbody>
</table>

the downstream task. Third, regarding classifiers, we note that SVM outperforms Decision Trees in most cases in this experiment.

Next, we report classification results using the upsampled pooled 2C2 data set in Table 7.11. Upsampling turns out to be an effective strategy in this case, as the best performing F1 score is improved from 0.8 (original data set using SVM) to 0.95 (upsampled data set using Decision Trees). It is noteworthy that Decision Tree based classifiers (Decision Trees and Random Forest) are the most effective when dealing with this data set, boosting both F1 score and accuracy in training and validation sets.
Table 7.8: Classification accuracy T300p (SVM/Decision Tree).

<table>
<thead>
<tr>
<th>Features/task</th>
<th>2C1</th>
<th>2C2</th>
<th>3C</th>
</tr>
</thead>
<tbody>
<tr>
<td>BWK</td>
<td>0.73/0.82</td>
<td>0.82/0.82</td>
<td>0.59/0.66</td>
</tr>
<tr>
<td>BWK+TLC</td>
<td>0.73/0.84</td>
<td>0.82/0.78</td>
<td>0.59/0.72</td>
</tr>
<tr>
<td>LSSE</td>
<td>0.88/0.87</td>
<td>1.0/1.0</td>
<td>0.89/0.85</td>
</tr>
<tr>
<td>LSSE+TLC</td>
<td>0.8/0.82</td>
<td>0.97/1.0</td>
<td>0.84/0.79</td>
</tr>
</tbody>
</table>

Table 7.9: Classification accuracy T200p (SVM/Decision Tree).

<table>
<thead>
<tr>
<th>Features/task</th>
<th>2C1</th>
<th>2C2</th>
<th>3C</th>
</tr>
</thead>
<tbody>
<tr>
<td>BWK</td>
<td>0.78/0.8</td>
<td>0.88/0.79</td>
<td>0.68/0.7</td>
</tr>
<tr>
<td>BWK+TLC</td>
<td>0.78/0.77</td>
<td>0.88/0.88</td>
<td>0.68/0.7</td>
</tr>
<tr>
<td>LSSE</td>
<td>0.96/0.94</td>
<td>0.82/0.93</td>
<td>0.86/0.9</td>
</tr>
<tr>
<td>LSSE+TLC</td>
<td>0.96/0.92</td>
<td>0.85/0.93</td>
<td>0.88/0.9</td>
</tr>
</tbody>
</table>

Table 7.10: Classification accuracy pooled data set (SVM/Decision Tree).

<table>
<thead>
<tr>
<th>Features/task</th>
<th>2C1</th>
<th>2C2</th>
<th>3C</th>
</tr>
</thead>
<tbody>
<tr>
<td>BWK</td>
<td>0.76/0.72</td>
<td>0.74/0.59</td>
<td>0.69/0.54</td>
</tr>
<tr>
<td>BWK+TLC</td>
<td>0.76/0.72</td>
<td>0.74/0.66</td>
<td>0.7/0.57</td>
</tr>
<tr>
<td>LSSE</td>
<td>0.87/0.84</td>
<td>0.91/0.87</td>
<td>0.82/0.76</td>
</tr>
<tr>
<td>LSSE+TLC</td>
<td>0.88/0.86</td>
<td>0.91/0.86</td>
<td>0.82/0.8</td>
</tr>
</tbody>
</table>
However, we observe that this strategy alone is still not able to perform well in practice, as the performance of our best classifier degrades considerably when evaluated on an unseen test set consisting of 90 motif clusters discovered from the B100p data set using the MK algorithm. We therefore performed additional experiments using classifier optimization techniques such as probability threshold (altering decision boundary for probabilistic classifiers such as SVM and Random Forest), feature scaling (min-max scaling and z-score scaling), and the kernel trick in the case of SVM (RBF vs linear kernel with `class_weights` argument, which has built-in weight adjustment for imbalanced classes). This result is summarized in Table 7.12. Note that the results reported in this table are obtained from using SVM classifier. Even though Decision Tree classifier achieved best results on training and validation set (as noted above), it is not a probabilistic classifier and is difficult to calibrate further when found to be ineffective in the testing set evaluation. SVM (and Random Forest to a certain extent), in contrast, allows multiple optimization techniques in order to achieve higher F1 score in testing time.

From Table 7.12, we select the best F1 classifier to be the SVM with RBF kernel using 2 features, thresholding at a probability of 0.3 for the positive (q-linear) class. In general, we have found three sets of optimization techniques that are able to achieve more or less equivalent improvement over the baseline of default setting of SVM (RBF kernel with 0.5 threshold). These are: (1) RBF kernel with threshold of 0.3; (2) Linear kernel with class weights adjustments for imbalanced classes (otherwise default); (3) RBF kernel with min-max feature scaling (otherwise default). The fourth variation of using z-score scaling of features resulted in the highest recall score but lower F1 score. This would lead to the best looking results when evaluated at top k motif (similar to the precision@10 evaluation metric for information retrieval) since there will be a

---

8The “test set” in this context is more like a downstream application testing.
large number of motifs classified as q-linear and ranked lower, which also tend to be simple motifs (if not q-linear). However, given the use case of this application and our final objective of MAP score, we select the best F1 classifier, which is the most sensible metric to optimize at the data set level.

7.9.4 Results: Rank refinement

We report the evaluation results for the MK motif discovery rank refinement algorithms proposed in Chapter 7.8 along with a baseline MK algorithm where we simply run MK algorithm and take the result as is.

The results are presented in Figure 7.37. First of all, we immediately see the benefit of rank refinement in improving the ranking — even the simplest scheme of three-way classification (linear, q-linear, non-linear) increases the MAP score considerably over the baseline of MK algorithm and often achieves almost the best score among all methods. Second, we expected the iterative methods (IPF, IPF-1) to have benefits of finding extra real (non-linear) motifs. However, to our surprise, this benefit only shows when the algorithm exhaustively removes all linear motifs it finds (IPF), whereas

Table 7.11: Classification accuracy pooled 2C2 data set. Results reported using SVM/Decision Tree/Random Forest in each cell, Orig=original data, UpS=Upsampled data, 1f=one feature (LSSE), 2f=two features(LSSE,TLC).

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Orig-1f</td>
<td>0.82/0.77/0.78</td>
<td>0.8/0.75/0.75</td>
<td><strong>0.80/0.75/0.75</strong></td>
<td>0.91/0.87/0.89</td>
</tr>
<tr>
<td>Orig-2f</td>
<td>0.87/0.72/0.81</td>
<td>0.76/0.78/0.73</td>
<td>0.79/0.72/0.76</td>
<td>0.91/0.86/0.92</td>
</tr>
<tr>
<td>UpS-1f</td>
<td>0.86/0.90/0.88</td>
<td>0.81/1.0/0.95</td>
<td>0.83/0.95/0.89</td>
<td>0.87/0.95/0.91</td>
</tr>
<tr>
<td>UpS-2f</td>
<td>0.86/0.94/0.89</td>
<td>0.84/1.0/0.98</td>
<td>0.85/0.94/0.89</td>
<td>0.88/0.95/0.95</td>
</tr>
</tbody>
</table>
Table 7.12: Classification results using different parameter settings. The pooled 2C2 data set is used as training/validation sets, and B100p data set as the test set, using the exhaustive IPF. Results obtained using SVM classifier and upsampled training set (except when using linear kernel with class weights (CW) option in the scikit learn library for Python, which has built-in weight adjustments to account for class imbalance in training set). Complexity computed using the z-score normalized version of motif clusters. T=classification threshold for probabilistic classifier (SVM) for the positive (QL) class, #f=number of features in classifier. First two rows indicate baseline results.

<table>
<thead>
<tr>
<th>T</th>
<th>Feature scaling</th>
<th>SVM kernel</th>
<th>#f</th>
<th>p/r/F1(validation)</th>
<th>p/r/F1(test)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5</td>
<td>No scaling</td>
<td>RBF</td>
<td>1</td>
<td>0.88/0.84/0.86</td>
<td>1/0.65/0.79</td>
</tr>
<tr>
<td>0.5</td>
<td>No scaling</td>
<td>RBF</td>
<td>2</td>
<td>0.87/0.88/0.87</td>
<td>1/0.65/0.7</td>
</tr>
<tr>
<td>0.5</td>
<td>minmax</td>
<td>RBF</td>
<td>1</td>
<td>0.83/0.91/0.86</td>
<td>0.96/0.74/0.84</td>
</tr>
<tr>
<td>0.5</td>
<td>minmax</td>
<td>RBF</td>
<td>2</td>
<td>0.85/0.98/0.91</td>
<td>0.93/0.71/0.81</td>
</tr>
<tr>
<td>0.5</td>
<td>zscore</td>
<td>RBF</td>
<td>1</td>
<td>0.89/0.87/0.87</td>
<td>0.67/0.85/0.75</td>
</tr>
<tr>
<td>0.5</td>
<td>zscore</td>
<td>RBF</td>
<td>2</td>
<td>0.87/0.88/0.87</td>
<td>0.68/0.86/0.76</td>
</tr>
<tr>
<td>0.3</td>
<td>No scaling</td>
<td>RBF</td>
<td>1</td>
<td>0.88/0.85/0.86</td>
<td>0.96/0.74/0.84</td>
</tr>
<tr>
<td>0.3</td>
<td>No scaling</td>
<td>RBF</td>
<td>2</td>
<td>0.87/0.88/0.87</td>
<td>0.96/0.77/0.86</td>
</tr>
<tr>
<td>0.5</td>
<td>No scaling</td>
<td>Linear(CW)</td>
<td>1</td>
<td>0.77/0.97/0.86</td>
<td>0.96/0.74/0.84</td>
</tr>
<tr>
<td>0.5</td>
<td>No scaling</td>
<td>Linear(CW)</td>
<td>2</td>
<td>0.8/0.98/0.87</td>
<td>0.93/0.74/0.83</td>
</tr>
</tbody>
</table>
Figure 7.37: Motif discovery MAP scores using different methods proposed. Data sets used: B100p, B200p, T100p, T200p.

the results are worse than non-iterative methods in general if we only run MK one additional time (IPF-1). Third, we note that the (often) small gain by running IPF exhaustively is also weighed down by the excessive time it takes to run many iterations of MK algorithm (typically less than 7 iterations before it prunes all linear motifs from the database). Therefore, there is a tradeoff between efficiency (time complexity) and MAP score.
7.9.5 Conclusion

In this chapter, we have developed novel time-series complexity measures and methods for pruning and classifying tone N-gram time-series motifs produced by the MK algorithm. The goal is to refine the ranked retrieval results to achieve the best MAP score and retrieve meaningful true motif clusters. Our proposed methods greatly improve the baseline generated by MK algorithm and are well suited for motif discovery in the speech prosody domain. We observe a time-accuracy tradeoff between the non-iterative and iterative pruning frameworks.

Overall, the motif discovery enables us to efficiently retrieve highly similar patterns (motif clusters) within a speech prosody database. Meanwhile, in order to extract linguistic meaning and improve our understanding of tone N-grams patterns, we need a linguistically informed conceptual and methodological framework. In the next chapter, we target the very heart of the variability problem in lexical tone production and analyze tone N-gram patterns in conjunction with cross domain linguistic features.
8.1 Research questions

In the last chapter, we looked at how we can use motif discovery to discover similar contour patterns from tone N-gram data. This is a mapping from contour shapes to the patterns of tone N-gram categories\(^1\). In this chapter, we look at the reverse of the motif discovery problem: given a tone N-gram category, can we predict what shapes the tone N-gram will take in spontaneous speech, given a set of linguistic features in discourse, syntax, morphology, and phonology?

The motivation of this problem directly comes from the central question in tone and prosody research: how can we better understand the surface variability problem of tone production? Given a certain tone or tone N-gram category, to what degree can we predict the contour shape profile it will take, given a variety of linguistic features from both sound and text domain?

In order to formalize this task, we need to first have a method to derive tone contour shape classes so that we can perform machine learning experiments targeting these classes. In this chapter, we first describe a method for automatically deriving such contour shape classes using network pattern analysis and we discuss its advantage

\(^1\)This conceptual mapping is in the sense that in motif discovery, we first observe the contour shapes, and then discover patterns that can be analyzed in terms of tone N-gram categories. In this case, we ask: what attributes or categories make up this motif cluster?
over traditional unsupervised methods such as k-means clustering. Then, we proceed
to investigate the problem of predicting contour shape classes using machine learning.

8.2 Deriving tone N-gram contour shape classes through network analysis

8.2.1 Methodology

We adapt a method proposed by Gulati et al. [GSIS16] to automatically derive tone
contour shape classes given a particular tone N-gram using network analysis. A net-
work, similar to a graph, is a mathematical data structure consisting of nodes that
are connected by edges. Each edge can be either directed/undirected or weighted/un-
weighted depending application.

On a high level, this method filters a fully connected network representing a pair-
wise distance matrix of all time-series objects of a tone N-gram category. After the
filtering step, only those nodes that have a similarity score beyond a threshold will
remain connected. It then leverages network community detection algorithms to opti-
mize the community structure, effectively deriving clusters where each pair of node
inside each cluster are highly similar. Therefore, we can use the clusters as our contour
shape classes.

An advantage of this method over traditional clustering algorithms like K-means
clustering is that, it automatically identifies and excludes outliers from entering main
clusters because an object needs to not only have a low distance to the cluster centroid
but also must have low distances to all objects in the cluster. Under this algorithm,
the outlier objects will form their own clusters naturally and we can filter them out
by thresholding on the cardinality of the cluster.
Another advantage over K-means is that, unlike K-means, we do not need to determine the number of cluster parameter before hand, since the algorithm naturally picks clustering structures by network filtering and community detection.

Finally, given our goal of deriving only contour shape classes (not any hard categories), this is a completely unsupervised process. Since this problem is purely a distance-based problem\(^2\) with a built-in optimization for distance-based filtering and community structure/modularity, there is no need to evaluate the classes in order to optimize any ground truth (extrinsic evaluation)\(^3\).

### 8.2.2 Network Construction

We use the Python package networkx\(^4\) to implement networks and graphs and their operations. We first construct a fully connected tone N-gram pattern network where each node is a tone N-gram pattern time-series object and the edge between any two nodes is weighted by the distance between the two nodes. For simplicity we use Euclidean distance in this step. We then derive an undirected, weighted and fully connected network of tone N-gram patterns for each N-gram category (e.g., all data from tone trigram 1-3-4 is used to constructed a network). We have excluded tone N-grams categories where the data is too sparse (especially those involving neutral tones).

---

\(^2\)This is in the sense that all the data points in a data set already belong to the same tone N-gram category. We only need to derive contour shape classes that are close-knit by their distances.

\(^3\)This point can be made more clear by comparing this task with a tone recognition clustering task. In a tone recognition task, our objective is to cluster according to tone categories, not contour shapes. If two tones are similar in shape but actually belong to different categories, we still want them to be clustered into different clusters. But in this case, they already belong to the same tone N-gram category, so as long as their contour shapes are similar enough beyond an optimized threshold, they should be in the same cluster, and we do not need to optimize the clustering any further according to additional ground truths.

\(^4\)https://networkx.github.io
8.2.3 Network filtering

In this step, we take a fully connected network $G$ of a given tone N-gram category and use a principled method to remove edges from the network. Our goal is to find an appropriate threshold so that all edges whose weights (distance between two time-series objects) are greater than the threshold will be cut. Specifically, we decide the threshold value by a six-step process: (1) after observing the weight distribution of the network edges and trial and error, we search for the appropriate threshold in the set of values $T \in \{1.0, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0, 4.5\}$ for bigrams and trigrams, $T \in \{0.2, 0.4, 0.6, 0.8\}$ for unigrams; (2) we iterate over this set of values and each time apply a threshold to the network; (3) after we applied the threshold we convert the network to a unweighted network $G'$ that is no longer fully connected (only those nodes that have a distance below the threshold will remain connected); (4) we produce a randomized network $G_r$ by randomly swapping edges from $G'$ $k$ times while keeping the degree of the nodes constant, where $k$ is equal to the number of edges in $G'$. This can be seen as producing a maximally random network given the degree distribution of the current network; (5) we compute the difference in Clustering Coefficient (CC) of both $G'$ and $G_r$; (6) after repeating this for all values in $T$, we pick the threshold that has the largest difference of CCs.

The Clustering Coefficient (CC) of an undirected graph is a measure of the number of triangles in a graph. Intuitively, it measures how well connected a network is on average, ranging from not connected to fully connected. In other words, it expresses how saturated the network is — how many of the possible connections are actually expressed. The CC is based on a local clustering coefficient $C_i$ for each node $i$:

$$C_i = \frac{\text{number of triangles connected to node } i}{\text{number of triples centered around node } i}$$
where a triple centered around node $i$ is a set of two edges connected to node $i$. The clustering coefficient for the whole graph is the average of the local values $C_i$:

$$C = \frac{1}{n} \sum_{i=1}^{n} C_i$$

where $n$ is the number of nodes in the network. By definition, the value of $C$ is in the range of $[0,1]$. The clustering coefficient of a graph is closely related to the transitivity of a graph, as both measure the relative frequency of triangles.

In this method, we are comparing the CC of the thresholded vs. randomized network. Here, we provide some intuitions as to why this method works. What we are really getting at is how much the communities cluster together. Consider that if we have a fully connected network, then there is no communities, and the CC of this network and a randomized one (which is basically the same as this one) is identical. Then we start to threshold and cutting edges, and for a thresholded network we also create a randomized version. We then compare the CC of these two networks. What happens when they are maximally different? First, the CC of the thresholded network is not the same as a randomized version of that network. Secondly, on the one hand, a healthy community structure requires a higher CC, since a sparsely connected network does not have much community to begin with. On the other hand, we do not want the CC to approach a fully connected network, therefore we want to place an upper limit on the CC. Ultimately, we are after a balance point between these two forces, and the maximal difference between the two CCs is that point.

8.2.4 Community detection

We use Python package community\(^5\) to perform community detection. Specifically, we use the Louvain algorithm proposed in Blondel et al. [BGLL08], a widely used

---

\(^5\)http://perso.crans.org/aynaud/communities/#
algorithm due to its effectiveness and efficiency. This method is based on the optimization of modularity of the network as the algorithm progresses. Modularity is a scale value between -1 and 1 that measures the density of edges inside communities to edges outside communities. Optimizing this value theoretically results in the best possible grouping of the nodes of a given network. In the Louvain Method of community detection, first small communities are formed by optimizing modularity locally on all nodes, then each small community is recursively grouped into higher level communities.

Figure 8.1 shows a sample evolution of the clustering coefficients of a filtered network and a corresponding randomized network, where red vertical line is superimposed to indicate the maximum difference between the CCs. In this case, we choose the value of $T = 3.0$. Figures 8.2 shows an example community structure of the derived community partitions. The plot shows a small numbers of tightly connected bigger communities and some outlier small communities. Figure 8.3 shows a zoomed-in version of a bigger community. Figure 8.4 shows a sample well-behaved community structure (with no outlier communities) in the bigram tone 0-1 data set.

### 8.2.5 Outlier Community Filtering

We propose an extra step of outlier community filtering before deriving our final contour shape classes. In this step, we use a heuristic threshold of $t = 10$ to filter out any communities (clusters) with a size less than $t$. After we have derived the remaining classes for each tone N-gram category, we map the nodes in the network to the actual time-series data points (tone N-grams) and their metadata attributes and serialize a contour shape class `.csv` file for each tone N-gram category.
8.2.6 Evaluation of tone contour shape classes

Previously, we have shown why it is unnecessary to perform extrinsic evaluation on the contour shape classes derived from this method. To make sure the result makes sense intrinsically, we perform two types of evaluations. First, we vary the value of the threshold $T$ outside of the range we tested in the network filtering stage and observe the outcome community structure. This is to test the possibility that a smaller distance threshold might yield tighter clusters. However, we observe that when using smaller threshold, the community structure of the output classes becomes unstable — often producing a plethora of small communities. This shows that our method produces sensible and centralized community structures.

Second, we tested the upper bound of a potential classification task we are proposing next: how well can we classify the tone N-grams into different contour
Figure 8.2: Sample network community structure (Full). This shows a small numbers of tightly connected bigger communities and some outlier small communities. Number shows index of individual time-series object.
shape classes using syntactic, morphological, phonological and other features? Here, we test the classes using the compliment feature of that task: how well can tone N-grams be classified into these shape classes using their time-series $f_0$ values? This is similar to a tone classification task except that the classes are contour shape classes. Since the contour shape classes are supposed to capture the different types of shapes in the tone data, we expect this intrinsic task to obtain high accuracy. Indeed, we are able to classify tone N-gram time-series points into these classes with around 90% accuracy. This indicates an upper bound on how well these data can be predicted to belong to one of the shape classes.

8.3 Machine learning of tone N-gram contour shapes

Our goal in this chapter is to understand how features from various linguistic domains play a role in predicting the contour shape type (class) of a particular tone N-gram.
Figure 8.4: Sample network community structure with no outliers. Number shows index of individual time-series object.
category in its realization in spontaneous speech, a key issue at the center of intonation and tone research. There has been a long line of research documenting the interaction between syntax/semantics and prosody, discourse and prosody, etc. in linguistic literature [Li09, Bur13]. In tone research, Wang et al. [WX11b] showed how discourse domain features (and “communicative functions”) such as focus, topic, information structure affect the realization of tone contours. However, to the best of our knowledge, there is no previous work that uses machine learning to systematically investigate the interaction between linguistic features and prosodic realizations of tone N-grams from a large data set of spontaneous speech.

We can also view this problem from an alternative angle — namely, an information theory perspective on prosody. Speakers produce a large amount of variability in their tone production. Similarly, they also face numerous choices in the use of linguistic forms in their speech, including morphological, referential, syntactic, and phonetic, to name a few. The information theory view of speech production postulates that these choices are not random. Rather, speakers choose to use specific linguistic forms in order to maximize the rate of information transmission [SW63]. In the prosody domain, the evidence of this optimization process can be observed from the statistical correlations between various linguistic attributes (prosodic or non-prosodic) and the tone/intonation contour shape profiles. In this experiment, we therefore take machine learning approaches to find these correlations.

8.3.1 Feature engineering

We aim at investigating a number of linguistic features from the domain of syntax, phonology, morphology, discourse, semantics, etc. In selecting specific features for this task, we are also constrained by the availability and reliability of the annotated corpora and natural language processing (NLP) technologies. The main CMN corpus we
use has ground truth segmentations and annotations of tones and words in both pinyin and Chinese character forms. Given the maturity of open source Chinese-language NLP tools to extract features in syntactic and discourse domains, we therefore are able to extract relatively reliable features in these domains. There are some features, such as topic and focus, that currently do not have reliable automatic methods for detection, and we therefore have to exclude them from this study. Previous works discussed in Chapter 2.2.1 have shown the effect of these features in controlled experiments of read speech, but not large scale spontaneous speech, which would also require significant resources to annotate.

**Syntactic features**

We extract the part-of-speech (POS) tags for all syllables in a tone N-gram. In addition, we also extract the dependency function \([CM14]\) of all syllables in the tone N-gram. Therefore there are \(2 \times N\) syntactic features where \(N\) is the number of syllables included in the tone N-gram data under consideration. The original tag set used in
CoreNLP\textsuperscript{6} comes from Penn Chinese Treebank\textsuperscript{7} and is too fine grained. To avoid data sparseness, we collapsed several categories. For both POS tags and dependency edge function categories, we compute their distributions using the original tag set and we collapse any categories that appear less than 5 times in the data (often into a “others” category, but see details below). For POS tags we mapped the original 33 tags onto 5 categories. For dependency functions, we collapsed all tags with a subcategory separated by a colon (e.g., “advmod:loc”, “advmod:rcomp”, mapped to “advmod” etc.). Then, during classification we further preprocess the data by removing any values (factor levels) that appear less than 5 times, in order to avoid the “zero variance” warning in the \texttt{caret} package of R.

**DISCOURSE/SEMANTIC FEATURES**

We extract two discourse features for a tone N-gram data point: (1) whether the tone N-gram includes a named entity; (2) whether the tone N-gram includes a singleton (as opposed to being part of a coreference chain in the discourse). Discourse features such as information structure have been postulated to have an effect on the prosody domain [Bur13]. In particular, given information may encode prosodic features different from new information. This could also apply to named entities vs. non-named entities. Named entities points to definite, specific objects in the real world. Whether the token is a singleton or part of a coreference chain can be correlated with information structure. That is, a singleton may signify new information in discourse, while a non-singleton is part of a coreference chain with potential antecedent or anaphor, pointing to potentially a different information structure. Both can have distinguishing effects on the mental representations and the production of speech prosody.

\textsuperscript{6}As will be described next, coreNLP is the tool we use to extract features from text.
\textsuperscript{7}http://repository.upenn.edu/cgi/viewcontent.cgi?article=1039&context=ircs_reports
Morphological features

In Mandarin Chinese, each word usually consists of one or \( m \) syllables (where each syllable is one Chinese character, the smallest semantic unit). The value of \( m \) is usually two to four. Building on the intuition that the first syllable is usually spoken with higher prominence (e.g., neutral tone, which does not carry stress, only occurs on word-final positions), we extract morphological features for each syllable in the given tone N-gram: whether they cross word boundary or not. There are \( N \) features in this category in total.

Phonological features

A basic representation of phonological features is the identity of phonemes in each syllable of the N-gram. However, due to the sparseness of this feature representation, we have designed 7 binary features to encode the phonological properties of the syllables in the tone N-gram: (1) whether the syllable includes a nasal; (2) whether the syllable includes a diphthong; (3) whether the syllable includes a high vowel; (4) whether the syllable includes a low vowel; (5) whether the syllable includes a front vowel; (6) whether the syllable includes a back vowel; (7) whether the syllable includes a round vowel. In addition, we add two contextual tone features: the tone identity of the previous and following syllables of the tone N-gram in question.

Other features

We add two pitch features to the feature set: the beginning and ending pitch of the tone N-gram. This is based on the notion in previous works of Mandarin tone modeling (Parallel ENcoding and Target Approximation model, or PENTA) [XLPoL15] that in speech production, the actual realized tone shape of a given tone category highly
depends on the starting point of the pitch contour and its distance to the actual pitch
target of the current tone, which affects its course of trajectory when it approximates
the target \[\text{PoXT09}\]. An additional feature to be included is the position of the
current tone N-gram within the context of the current sentence as a percentage. It
is a known effect that pitch tends to downdrift in speech production as sentence
progresses \[\text{WX11a}\]. Therefore, we also want to account for the effect of sentence
position.

8.3.2 Feature Extraction Methodology

Bag of features

In this task, we note that the unit of feature extraction is not as straightforward as
it would be in classic NLP tasks. That is, instead of a typical syntactic constituent
(word, phrase, sentences) as the feature extraction unit, here our target is tone N-
gram, a sequence of \(N\) syllables that may or may not be a syntactic constituent.
As described above, in many features we have adopted a “Bag of features” approach
(similar to the speech coreference resolution work in Roesiger et al. \[\text{RR15}\]) where
each feature describes whether the N-gram contains a certain target value in any
position. For some other features, we simply use a set of \(N\) features applied to each
syllable in the N-gram in question.

Feature extraction

We use Stanford CoreNLP \[\text{MSB}^{+14}\] to extract syntactic, morphological and dis-
course domain features, a state-of-the-art open source NLP software for Chinese NLP.
The CoreNLP suite provides several “annotators” to be used in the current feature
extraction, including POS tagging, dependency parsing, named entity recognition
(NER), and coreference resolution for Chinese\(^8\). Since coreference is only available in XML format, we configured CoreNLP to output both CONLL\(^9\) and XML formats and merged the coreference information as an additional column in the final CONLL output format. The complete text transcripts of the CMN corpus audio is processed in batch through the CoreNLP.

In previous steps, we have described serializing the tone N-gram network object to contour shape class `csv` files, where each row is the downsampled time-series sub-sequence of a tone N-gram and its metadata attributes followed by its class assigned by the network community detection process. Each `csv` file contains data points corresponding to a particular tone N-gram, such as 1-3-4. In order to extract features from text domain, we implement a mapping from the `csv` files to `.phons` files and text files. Both these files are included in the CMN corpus. The `.phons` file contains a segmentation and phonetic transcription (using specifically defined symbols) of the audio at the segment level (onset and nuclear/coda) along with the tone information. The text file is a tokenized version of the transcription of the audio newscast speech. We use `.phons` files to extract phonological features and text files to extract text domain features using CoreNLP. Since the `.phons` file is tokenized by phonetic segments and the text file is tokenized based on an automatic Chinese text segmentation process, we also created a mapping between these two, as well as the mapping to sentence IDs in the CONLL file. In the end, the feature extraction takes place by leveraging these mappings among several resources and the result is a training set for each tone N-gram with their class labels appended.

\(^8\)https://nlp.stanford.edu/projects/chinese-nlp.shtml
\(^9\)http://conll.cemantix.org/2011/
8.3.3 Predicting tone N-gram contour shape classes

Feature selection

Our goal is to see how well we can predict the contour shape type that a tone N-gram will take on in spontaneous speech production, given linguistic features in syntax, morphology, phonology and other domains. Alternatively, we can also say that we aim to better understand how factors and information contents from these other domains affect the production of tones. From an information theoretical point of view, even though tone production resides in the prosodic domain, the flow of cross-linguistic-domain information content is much more fine grained.

From a machine learning perspective, feature selection helps us see which features are the most effective in producing a better classification accuracy, and which features are redundant and contain noise that might hurt the performance of a classifier. Alternatively, it enables us to rank linguistic features by the importance of their contribution in tone contour production. Feature selection is an active area of research in machine learning and data mining, with many dedicated packages across tools and programming languages such as Caret\textsuperscript{10}, randomForest\textsuperscript{11} in R, Weka, and scikit learn in Python.

In the previous step, we have generated feature files for each of the tone unigram, bigram and trigram category, giving rise to a total of $5 + 4^2 + 4^3 = 85$ data sets for the prediction task. To perform feature selection, we randomly selected five trigram data sets to carry out feature ablation experiments and feature importance ranking analysis in order to gain a deeper understanding of the set of features we are using. After this step, we use the subset of most effective features to run our classification algorithm on all data sets.

\textsuperscript{10}caret.r-forge.r-project.org
\textsuperscript{11}https://cran.r-project.org/web/packages/randomForest/randomForest.pdf
Table 8.2: Feature ablation experimental results in data 1-3-4. Asterisk (*) indicates no pos and no pos/func also applied in that condition.

<table>
<thead>
<tr>
<th>Data</th>
<th>Condition</th>
<th>Accuracy(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-3-4</td>
<td>all</td>
<td>57.82</td>
</tr>
<tr>
<td>1-3-4</td>
<td>no pos</td>
<td>57.82</td>
</tr>
<tr>
<td>1-3-4</td>
<td>no pos/func</td>
<td>*61.88</td>
</tr>
<tr>
<td>1-3-4</td>
<td>*no entity/singleton</td>
<td>61.59</td>
</tr>
<tr>
<td>1-3-4</td>
<td>*no entity</td>
<td>61.01</td>
</tr>
<tr>
<td>1-3-4</td>
<td>*no singleton</td>
<td><strong>62.32</strong></td>
</tr>
<tr>
<td>1-3-4</td>
<td>*no tok bound</td>
<td>60.86</td>
</tr>
<tr>
<td>1-3-4</td>
<td>*no phons</td>
<td>60.00</td>
</tr>
<tr>
<td>1-3-4</td>
<td>*no sent pos</td>
<td>62.17</td>
</tr>
<tr>
<td>1-3-4</td>
<td>*no prev/next tone</td>
<td>60.14</td>
</tr>
</tbody>
</table>

Concretely, we first perform feature ablation experiments by iteratively removing subsets of our features. For a trigram data set, there are in total 37 features as indicated in Table 8.1. We know from linguistic knowledge that these features are grouped into categories (syntactic, phonological, etc.). Therefore, it would make sense to select and remove feature subsets by category rather than to test features individually, which is what a feature selection algorithm that lacks domain knowledge would do.

All feature ablation experiments are done using the Support Vector Machine (SVM).

**Feature selection results**

We present results from feature selection experiments using feature ablation and SVM coefficients ranking.

1. **Feature ablations via SVM on randomly selected trigram data sets.**

We present results on feature ablation experiments on the five trigram data sets.
Table 8.3: Feature ablation experimental results in data 3-1-4. Asterisk (*) indicates no pos and no pos/func also applied in that condition.

<table>
<thead>
<tr>
<th>Data</th>
<th>Condition</th>
<th>Accuracy(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3-1-4</td>
<td>all</td>
<td>52.09</td>
</tr>
<tr>
<td>3-1-4</td>
<td>no pos</td>
<td>52.24</td>
</tr>
<tr>
<td>3-1-4</td>
<td>no pos/func</td>
<td><strong>57.31</strong></td>
</tr>
<tr>
<td>3-1-4</td>
<td>*no entity/singleton</td>
<td>57.74</td>
</tr>
<tr>
<td>3-1-4</td>
<td>*no entity</td>
<td><strong>58.32</strong></td>
</tr>
<tr>
<td>3-1-4</td>
<td>*no singleton</td>
<td>56.87</td>
</tr>
<tr>
<td>3-1-4</td>
<td>*no tok bound</td>
<td>55.42</td>
</tr>
<tr>
<td>3-1-4</td>
<td>*no phons</td>
<td>52.82</td>
</tr>
<tr>
<td>3-1-4</td>
<td>*no sent pos</td>
<td>55.57</td>
</tr>
<tr>
<td>3-1-4</td>
<td>*no prev/next tone</td>
<td>54.56</td>
</tr>
</tbody>
</table>

Table 8.4: Feature ablation experimental results in data 3-2-2. Asterisk (*) indicates no pos and no pos/func also applied in that condition.

<table>
<thead>
<tr>
<th>Data</th>
<th>Condition</th>
<th>Accuracy(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3-2-2</td>
<td>all</td>
<td>60.86</td>
</tr>
<tr>
<td>3-2-2</td>
<td>no pos</td>
<td>60.11</td>
</tr>
<tr>
<td>3-2-2</td>
<td>no pos/func</td>
<td><strong>66.10</strong></td>
</tr>
<tr>
<td>3-2-2</td>
<td>*no entity/singleton</td>
<td>65.17</td>
</tr>
<tr>
<td>3-2-2</td>
<td>*no entity</td>
<td>65.54</td>
</tr>
<tr>
<td>3-2-2</td>
<td>*no singleton</td>
<td>65.36</td>
</tr>
<tr>
<td>3-2-2</td>
<td>*no tok bound</td>
<td>65.54</td>
</tr>
<tr>
<td>3-2-2</td>
<td>*no phons</td>
<td>63.48</td>
</tr>
<tr>
<td>3-2-2</td>
<td>*no sent pos</td>
<td>64.04</td>
</tr>
<tr>
<td>3-2-2</td>
<td>*no prev/next tone</td>
<td>63.30</td>
</tr>
</tbody>
</table>
Table 8.5: Feature ablation experimental results in data 3-4-2. Asterisk (*) indicates no pos and no pos/func also applied in that condition.

<table>
<thead>
<tr>
<th>Data</th>
<th>Condition</th>
<th>Accuracy(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3-4-2</td>
<td>all</td>
<td>58.33</td>
</tr>
<tr>
<td>3-4-2</td>
<td>no pos</td>
<td>58.17</td>
</tr>
<tr>
<td>3-4-2</td>
<td>no pos/func</td>
<td>60.95</td>
</tr>
<tr>
<td>3-4-2</td>
<td>*no entity/singleton</td>
<td>60.95</td>
</tr>
<tr>
<td>3-4-2</td>
<td>*no entity</td>
<td>60.78</td>
</tr>
<tr>
<td>3-4-2</td>
<td>*no singleton</td>
<td>62.25</td>
</tr>
<tr>
<td>3-4-2</td>
<td>*no tok bound</td>
<td>60.78</td>
</tr>
<tr>
<td>3-4-2</td>
<td>*no phons</td>
<td>59.31</td>
</tr>
<tr>
<td>3-4-2</td>
<td>*no sent pos</td>
<td>61.11</td>
</tr>
<tr>
<td>3-4-2</td>
<td>*no prev/next tone</td>
<td>59.31</td>
</tr>
</tbody>
</table>

in Tables 8.2, 8.3, 8.4, 8.5, 8.6. From these results, we also summarize the feature strength in Table 8.7, where we group all features into three sets of weak, medium, and good features. First, we observe that the weakest features are syntactic features POS tags and dependency functions, as removing these features always results in a significant boost in classification accuracy. This is true before and after we collapsed some categories to combat data sparseness problems. Second, the medium features, including discourse features (entity and coreference-singleton) and sentence position, have a mixed effect on the classifier performance across different data sets. They could either boost or degrade the performance in an insignificant way. In the result tables, we have highlighted the results obtained by all features but the POS tag and dependency functions, as well as the best result obtained on any subset of the features. It is noteworthy that the former usually reaches near the best or is the best result. Otherwise, the best results are often obtained by removing either the entity feature or the singleton feature (but not both).
Table 8.6: Feature ablation experimental results in data 4-2-1. Asterisk (*) indicates no pos and no pos/func also applied in that condition.

<table>
<thead>
<tr>
<th>Data</th>
<th>Condition</th>
<th>Accuracy(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>4-2-1</td>
<td>all</td>
<td>58.94</td>
</tr>
<tr>
<td>4-2-1</td>
<td>no pos</td>
<td>59.16</td>
</tr>
<tr>
<td>4-2-1</td>
<td>no pos/func</td>
<td><strong>63.69</strong></td>
</tr>
<tr>
<td>4-2-1</td>
<td>*no entity/singleton</td>
<td>64.13</td>
</tr>
<tr>
<td>4-2-1</td>
<td>*no entity</td>
<td><strong>64.24</strong></td>
</tr>
<tr>
<td>4-2-1</td>
<td>*no singleton</td>
<td>64.02</td>
</tr>
<tr>
<td>4-2-1</td>
<td>*no tok bound</td>
<td>63.02</td>
</tr>
<tr>
<td>4-2-1</td>
<td>*no phons</td>
<td>62.03</td>
</tr>
<tr>
<td>4-2-1</td>
<td>*no sent pos</td>
<td>62.80</td>
</tr>
<tr>
<td>4-2-1</td>
<td>*no prev/next tone</td>
<td>62.58</td>
</tr>
</tbody>
</table>

Third, the good features are the most effective features, including phonological features, tone context, token boundary features (morphological), and start and ending pitch of the contours. The phonological features are closely related to speech prosody production even though it has not been investigated in this way before. Both morphological and start/ending pitch features have groundings in their roles in tone production (as discussed above) and are therefore expected to contribute positively to the prediction accuracy. Overall, we note that all classification results are significantly above chance (to see the number of classes in each data set, visit Figures 8.7, 8.10, 8.13).

(2) Support Vector Machine (SVM) linear kernel feature weight coefficients ranking on all unigram, bigram, and trigram data sets. In SVM, a decision hyperplane can be defined by an intercept term $b$ and a decision hyperplane normal vector $\vec{w}$ which is perpendicular to the hyperplane. This vector is commonly referred to in the machine learning literature as the weight vector [MRS08]. While
Table 8.7: Feature strength from ablation experiments.

<table>
<thead>
<tr>
<th>POS tag</th>
<th>sent position</th>
<th>phonological start pitch</th>
</tr>
</thead>
<tbody>
<tr>
<td>dep func</td>
<td>is_entity</td>
<td>end pitch</td>
</tr>
<tr>
<td>is_singleton</td>
<td>tok_bound</td>
<td></td>
</tr>
</tbody>
</table>

the direction of the weight vector gives the predicted class when taking a dot product with a input feature vector, Guyon et al. [GWBV02] showed that in a linear SVM, the weight vector of a given classifier can be used as feature ranking coefficients, and those inputs with the largest weights correspond to the most informative features.

Following Guyon et al. [GWBV02], we therefore extracted weight vectors (coefficients) associated with all features in the SVM classification experiments. The number of features in unigram, bigram, and trigrams are available in Table 8.1. We then visualize the feature weights for all data sets in each of the N-grams in order to have an estimate of the distribution of feature importance in all of our experiments. For each data set in a given N-gram, we trained a linear SVM classifier and obtained feature weight vectors for each of the output shape classes. We take the absolute values of feature weight vectors and normalize them to be comparable across data sets. We then aggregated feature weights for all output classes and then across all data sets in a given N-gram.

For unigram data sets, Figure 8.5 shows the distribution of feature weights (importance) for all features in boxplot. We note that the starting and ending pitch, as expected from previous experiments, are the most important features with weights in a much higher range than others. We therefore present an extra boxplot in order
Figure 8.5: Feature weights for all unigram data sets.
Figure 8.6: Feature weights for unigram data sets excluding starting pitch and ending pitch.
Figure 8.6: Histogram of number of output shape classes in unigram data sets.

Figure 8.7: Histogram of number of output shape classes in unigram data sets.

to show the ranking of other less important features in more detail in a more appropriate value range. This is shown in Figure 8.6. In addition, we give an overview of the distribution of number of output shape classes for unigram data sets (Figure 8.7). We then present the same set of three plots for bigram (Figures 8.8, 8.9, 8.10) and trigram (Figures 8.11, 8.12, 8.13) data sets.

Comparing to the previous ablation experiments, the result from the feature weights analysis gives us a more detailed and complete picture of the feature importance. Overall, this set of result is largely consistent with the results obtained in the ablation experiments using a much smaller subset of the data. First, we emphasize that this is an exhaustive result obtained by using all unigram, bigram, and trigram data sets available to us, whereas in the feature ablation experiments, we only utilized a subset of five trigram data sets.

Second, in the ablation experiments, we removed features in batch according to their linguistic domains; that is, for instance, for phonological features, we removed
Figure 8.8: Feature weights for all bigram data sets.
Figure 8.9: Feature weights for bigram data sets excluding sentence position and ending pitch.
all phonological features as a group and observed its impact on the classifier’s performance. In that case, we are assuming as a prior knowledge that phonological features or any other feature groups based on linguistic knowledge worked as a whole, but there is no evidence that this is empirically so on the individual feature level unless we try to remove individual features one by one. However, in the current experiments of feature weights, we see empirical evidence of feature behavior on the individual feature level that is consistent with linguistic domains. Concretely, we see that individual features that belong to a linguistic domain, such as phonological features, syntactic features, etc., do indeed exhibit group behavior. This is an unexpected byproduct of the machine learning model: the machine learning model itself, even though without knowledge of linguistic domains whatsoever, has naturally learned to assign similar weights to features that belong to the same linguistic domain. This provides empirical evidence of the behavior of feature groups based on linguistic knowledge, and it corroborates our analysis about feature importance based on linguistic domain grouping.

Third, we observe that the feature (weight) importance distribution patterns are similar across unigram, bigram, and trigram data sets. More specifically, we observe consistently across different N-grams that the top level important features include starting/ending pitch and the position of the N-gram within the sentence (sent_position). Next on the ranked list are phonological features and morphological token boundary features. On the lower side of the ranked list are syntactic (pos_tag, dependency functions) and contextual features (previous and next tone). Overall, this ranking is similar to the ranking produced by ablation experiments, with small inconsistencies that are most obvious is the contextual features, possibly due to the change in the scale of the evaluation. Now we can be confident of the ranking of feature importance as listed in Table 8.7 with only minor modifications. The final ranking is shown in Table 8.8.
Figure 8.10: Histogram of number of output shape classes in bigram data sets.

Figure 8.11: Feature weights for all trigram data sets.
Figure 8.12: Feature weights for trigram data sets excluding starting pitch and ending pitch.

Figure 8.13: Histogram of number of output shape classes in trigram data sets.
Table 8.8: Feature strength from feature weight analysis using all unigram, bigram and trigram data (un-ordered within each column).

<table>
<thead>
<tr>
<th></th>
<th>Weak</th>
<th>Medium</th>
<th>Strong</th>
</tr>
</thead>
<tbody>
<tr>
<td>POS tag</td>
<td>phonological</td>
<td>start position</td>
<td>sent position</td>
</tr>
<tr>
<td>dep func context</td>
<td>is_entity</td>
<td>start pitch</td>
<td>end pitch</td>
</tr>
<tr>
<td>context</td>
<td>is_singleton</td>
<td></td>
<td></td>
</tr>
<tr>
<td>tok_bound</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

8.3.4 Evaluation

Experimental setup

We carry out classification experiments on all tone unigrams, bigrams and trigram data sets. We use SVM RBF kernel in our experiments with the feature set described in the last chapter. We evaluate the classifier performance directly on accuracy. As described above, even though each data set has a different number of classes, they are in general balanced in their sizes (a result naturally given by the network outlier filtering stage). Therefore, the accuracy is a sufficient metric for the evaluation.

Results

We present the classification accuracies on all unigram (Figure 8.14), bigram (Figure 8.15) and trigram (Figure 8.16) data sets. In these plots, “accuracies” (as in the legend) are computed with all strong and medium ranked features as described in Table 8.8. As discussed previously, this usually brings accuracy scores close to the maximum but

---

12 Excluding N-gram data sets that yielded very small data (i.e., less than 10 data points). These are low likelihood events in tone N-grams such as consecutive sequences of tone 0 or tone 3.
at times excluding entity or singleton features might give a slight gain in classifier performance. This is indeed what we see across all data sets, which supports our characterization of the feature importance in the previous chapters. In the same plots, we also give the random baseline (chance) that is computed by $1/num_{\_classes}$ in that data set. We note that in all data sets, regardless of unigram, bigram or trigram, we obtain significantly better results than this baseline. This strongly supports the hypothesis that a variety of syntactic, morphological, phonological and contextual features are able to predict the realization of a particular category of tone N-grams in spontaneous speech production.

It is worth pointing out that the performances between unigram, trigram and bigram classifiers could not be directly compared unless we normalize by the number of classes in each data set. For instance, comparing to trigram data, there are less number of bigram combinations (thus data sets) but each data set is of a larger size than trigram. Therefore, we typically see lower baseline values in bigrams than trigrams from the two accuracy plots. The same trend goes for bigram and unigram. Nevertheless, we observe from Table 8.9 that while bigram has a much lower baseline than trigram, the mean accuracy of the two data sets are actually comparable (if you take only two decimal points they are the same). This means that the predicting power of the classifier is stronger in bigram data sets than trigram, as suggested in the far right column (bigram classification accuracy is on average 3.8 times the baseline while trigram is 2.9). Similarly, the unigram has the strongest predicting powers given its lower baseline and highest classification accuracy. This may suggest that the longer the window of N-grams, the stronger an effect of other factors come into play (longer range prosodic events).
Figure 8.14: Unigram data sets classification accuracy.

Table 8.9: Comparison of unigram, bigram and trigram classifier performance.

<table>
<thead>
<tr>
<th>data sets</th>
<th>mean accuracy</th>
<th>mean baseline</th>
<th>ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>trigram</td>
<td>0.641</td>
<td>0.222</td>
<td>2.9</td>
</tr>
<tr>
<td>bigram</td>
<td>0.637</td>
<td>0.172</td>
<td>3.8</td>
</tr>
<tr>
<td>unigram</td>
<td>0.724</td>
<td>0.140</td>
<td>5.17</td>
</tr>
</tbody>
</table>
Figure 8.15: Bigram data sets classification accuracy.
Figure 8.16: Trigram data sets classification accuracy. X-axis indicates tone trigram categories, i.e., 111, 112, ... , 444.


**Discussion**

We have shown that features from syntactic, morphological, phonological and other domains can successfully predict the contour shape class a tone N-gram will take, without supplying any information about the contour $f_0$ trajectory in the feature set (i.e., as in $f_0$ based tone recognition). Furthermore, there are many prosodic and discourse phenomena we haven’t taken into account due to lack of annotated data, including focus, topic, information structure, etc. These factors have been shown in previous works to correlate with the realization of $f_0$ contour shapes in tone production. Considering these factors, the current evidence points to strong contribution of non-prosodic predictors to the realization of prosodic events. Meanwhile, we’ve shown a negative correlation between $N$ (as in N-gram) and the prediction power, suggesting stronger unaccounted factors at play when $N$ increases.

We may view this from an information theory point of view of speech production. As an example, two different sequences of three phonemes in a tone trigram technically convey the same number of bits (modulo phonotactic constraints, etc.). However, even if one takes two different CVC syllables with a phonologically valid shape in language L and similar phoneme frequencies, they never have the same information content. Some words are more likely than others, and if this is combined with contextual constraints (such as previous words, syntax, information structure, and intonation), suddenly a good amount of informational inequalities holds. This puts pressure on speakers to optimize phonation length, tenseness, stress, etc., as a function of predictability to adjust their pronunciation based on factors from all linguistic levels. If we accept this premise, then the opposite should apply too: if we have anno-
tations corresponding to these constraints, they should have predictive power for the properties of the phonation.\footnote{Personal communication with Prof Amir Zeldes for the discussion on this point.}

Even though the methods and experiments we proposed here are novel in their own rights, in fact, this idea of information theoretic view of speech production has long existed in the previous works that studied the computational modeling of speech production and perception [Jur02, HDX+15, LJ07, Lev08, QJ12, BBG+09]. More formally, Shannon’s information theory [SW63] characterized an efficient communication system as one under which the rate of information transmission is maximized (i.e., information transmission per unit time is maximized). In the case of language communication, maximal efficiency can be achieved when the transmission rate of linguistic signals is relatively uniform and close to the channel capacity. Based on Shannon’s noisy channel theorem [SW63] (as stated above), recent works showed that an ideal code should keep the average amount of information conveyed per word constant across discourses [QJ12]. One simple example is Zipf’s observation that more frequent words tend to have fewer syllables, in which case the entropy per letter (or per phoneme for speech) across words in the mental lexicon could be relatively constant, matching the pattern predicted by the noisy channel theorem.

A number of studies in recent years have tested to what extent language and language use exhibit properties that are predicted by this theorem (see Qian et al. [QJ12] for an overview). For instance, Hu et al. [HDX+15] discussed the effect of information structure on speech and discourse production with respect to referential form, morphology, syntax (word order), and prosody (intonation). The central idea is that, given numerous ways meanings can be mapped onto linguistic forms, speakers adjust the various aspects of linguistic forms in speech and discourse production based on the information structure. In the prosody domain, for example, speakers
modulate prosody based on the information status of their words, using acoustic reduction (i.e. shorter, unaccented, and less intelligible pronunciations) for previously-mentioned words or entities. Similarly, listeners are faster to interpret references to given information if the word is unaccented [HDX+15].

Aylett et al. [AT04] extended the theory of constant information content across words (described above) to the speech domain. The central premise is that in speech production, the most efficient way of ensuring robust information transmission in noisy environments is to have smooth signal redundancy\(^1\) (which can be thought of in terms of a smooth distribution of the probability of recognition) throughout an utterance. They propose that an inverse relationship between syllable duration and predictability arising from lexical, syntactic, semantic and pragmatic factors is to be expected, since it provides an efficient way of ensuring that elements with low levels of language redundancy are produced for a longer period of time and perhaps with more salient acoustic characteristics, and will thus be likely to be recognized.

In this context, the current work on predicting tone contour shapes has meaningful implications for information theory based accounts of speech and discourse production. It can be considered as providing another facet of evidence in support of such accounts. Previous works targeting information theory and information structure in prosody domain have largely looked at acoustic correlates directly, such as accent and duration, all of which may in turn have an impact on the shape of tone contours in speech production. Therefore, looking at tone contour shapes can be thought of as a different level of manifestation of such phenomena, an amalgamation of single dimensioned acoustic correlates such as duration and intensity. It is also a level that is most difficult to quantify and measure in the traditional linguistic/phonetic investigations.

\(^{14}\)Here redundancy is used to mean predictability. More redundant words are more predictable.
Overall, we can outline three main contributions of the current investigation in the context of information theory accounts of speech production. First, the strong predicting power of our machine learning models provides good evidence for the information theory accounts of speech prosody (tone and intonation) production as discussed above. This evidence is striking considering that the information theory account, probabilistically linking prosody production to other linguistic domains, is a built-in (implicit) prior assumption of our experiments when we set up our model with feature engineering. Second, we used network analysis and machine learning from the SPM toolkit to enable our analysis — i.e., these computational modeling methods have made it possible for us to quantify speech prosody tone/intonation contour shapes in an effective and meaningful way. Third, we leverage the power of big data in speech prosody so that large-scale statistical patterns emerge.

One possible future direction of research is to build models to test specific predictions of information theory account of speech production. In that case, we will need to build formal models and experiments to test specific ways information content inequalities impact dimensions of speech prosody production and its direction of impact. In doing so, we combine methods from the SPM toolkit with computational psycholinguistic approaches seen in previous works to leverage the power of bigger data in the computational modeling of cognitive processes of linguistic production and comprehension.
Part V

Applications
Chapter 9

Applications

9.1 Computing speech prosody time-series similarity for other tone languages

Throughout this dissertation, we have developed various methods and tasks aimed at the understanding and analysis of Mandarin tones. However, as we have discussed since the beginning chapters, these analytical methods are also generic and extensible in their application to other tone languages and other speech prosody phenomena/tasks of interest.

In this chapter, we extend our methodology on using symbolic representation to perform the unsupervised learning of tones on other tone languages. Specifically, we will carry out experiments on Thai tones.

9.1.1 Background on Thai tones

The Thai tone system is slightly more complex than Mandarin. Thai has 5 lexical tones, Falling (F), Rising (R), High (H), Mid (M) and Low (L). The first two are contour tones, as they show distinct directional changes in their pitch trajectory; the Falling tone shows a rise followed by a fall, while the Rising shows a fall and then a rise. Though the other three are traditionally called level tones, we note that these tones are not characterized the best by “flat” pitch values, but actually have more complicated trajectories of their own, though less extreme in inflection than the
contour tones [RW15]. Figure 9.1 illustrates the pitch trajectories of Thai tones. The data collection used in this experiment is described in Chapter 3.2.

9.1.2 Experimental setup

The Thai tone data set is preprocessed in a number of pitch preprocessing steps as described in Chapter 4.2: pitch estimation; pitch normalization and log (Bark scale) transformation; finally, pitch downsampling into 30-point pitch vectors for each tone contour. We use the 1-5 numeric scale to denote the five tone categories as ground truth labels.

We adopt the same Query By Content (QBC) experimental setup as discussed in Chapter 6.6. In this setup, each time we randomly sample a group of $k$ queries and for each query $q$, we retrieve $r$ ranked tones according to their similarity (depending on the specific similarity measure used) to the query. Among these $r$ tones we compute
their Mean average precision (MAP) score on the top $n$ results, where $n$ is the number of total relevant tones that belong to the same tone category as $q$. Finally we average MAP scores for all $k$ queries $q_1, ..., q_k$ and we return the final average MAP score for this iteration.

9.1.3 Results

Table 9.1 shows the results for the different values of $k$ and $r$ at optimal SAX parameters $w = 18, a = 8$. Comparing this table with the Mandarin results in Chapter 6.6.2, we see that the result is in general a bit lower due to the larger number of tone categories in Thai, which increases the chance of making a mistake in QBC. Moreover, we have noted above that the quality of this data set (as well as it being child-directed speech) also contributes to lowering the MAP score for QBC. Nonetheless, we see that the results are reasonable compared to the Mandarin and compared to chance level. In particular, at $k = 20, r = 100$, we found that SAX consistently outperforms Euclidean distance on $f_0$ numeric representation. At $k = 50, r = 200$, a larger data set, we see more mixed results with SAX outperforming Euclidean distance on average (with comparable but slightly larger standard deviation on the average MAP score results). Overall, this result is consistent with results obtained from Mandarin: given optimal parameter settings, SAX achieves better MAP scores in the QBC task possibly due to its dimensionality and noise reduction mechanisms. As expected, there should be no reason that this does not generalize to another tone language such as Thai. It remains to be seen if SAX’s advantage generalizes to other types of prosody tasks where similarity computation between speech prosody time-series is crucial.
Table 9.1: QBC results MAP scores for Thai. Data: Thai data set, \(k=\)query size, \(r=\)retrieved size.

<table>
<thead>
<tr>
<th>evaluation/iteration</th>
<th>EUCLIDEAN</th>
<th>SAX-MINDIST</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAP((k = 20, r = 100))</td>
<td>0.306</td>
<td>0.349</td>
</tr>
<tr>
<td></td>
<td>0.317</td>
<td>0.364</td>
</tr>
<tr>
<td></td>
<td>0.258</td>
<td>0.265</td>
</tr>
<tr>
<td></td>
<td>0.311</td>
<td>0.315</td>
</tr>
<tr>
<td></td>
<td>0.284</td>
<td>0.354</td>
</tr>
<tr>
<td>mean</td>
<td>0.295</td>
<td>0.329</td>
</tr>
<tr>
<td>std</td>
<td>0.021</td>
<td>0.03</td>
</tr>
<tr>
<td>MAP((k = 50, r = 200))</td>
<td>0.329</td>
<td>0.345</td>
</tr>
<tr>
<td></td>
<td>0.301</td>
<td>0.268</td>
</tr>
<tr>
<td></td>
<td>0.274</td>
<td>0.279</td>
</tr>
<tr>
<td></td>
<td>0.281</td>
<td>0.298</td>
</tr>
<tr>
<td></td>
<td>0.285</td>
<td>0.295</td>
</tr>
<tr>
<td>mean</td>
<td>0.294</td>
<td>0.297</td>
</tr>
<tr>
<td>std</td>
<td>0.02</td>
<td>0.026</td>
</tr>
</tbody>
</table>

9.2 Speech Prosody Query and Retrieval (SPQR) tool

A direct application of SPM is a Speech Prosody Query and Retrieval (SPQR) tool that can assist phonologists, phoneticians, and speech prosody researchers in their routine tasks that might otherwise be done manually. The basic functionality of this tool is for the user to query a speech prosody corpus using a seed pattern (to be defined using a meaningful prosodic unit, such as a syllabic tone, tone N-gram, or an intonation phrase) and retrieve the top \(k\) similar patterns in the corpus. The seed pattern can be selected using examples extracted from the corpus, or using a user-supplied source (numeric, symbolic data points, or audio files). The researcher can further assess the meaningfulness of the patterns discovered by means of intrinsic (i.e., within the phonology/phonetics domain) or extrinsic evaluation (i.e., combined with
annotations in other domains such as syntax, semantics, and information structure in discourse). Extended functionality of the SPQR tool will showcase the motif discovery and network community detection/machine learning of speech prosody time-series data. The application can be implemented with a GUI web interface and use pre-computed time-series similarity indexing for faster retrieval.

Here, we show a demo application of SPQR with its basic Q(uery) and R(etrieval) functionality implemented as a web application running on a server. The user can access this application via a web browser and interact with the unigram tone time-series data set in the CMN corpus (or any corpus that is imported by the user/administrator). Currently the goal of the application is for users to be able to perform a query on a corpus of tones using a set of seed patterns (which can be a set of tones) and see the top K retrieved items that are similar to the query. Crucially, the user is able to select different parameters, including the time-series representation ($f_0$, SAX, qTA, Bark, D1), the distance measure (Euclidean, DTW, MIN-DIST), the SAX parameters, and specify the number of query tones and retrieved tones. In this way, the user is able to explore and interact with the speech prosody database with the ease of experimenting with different time-series retrieval methods, which may lead to further steps in their investigation after inspecting the results. The application enables researchers to do this efficiently without having to implement relevant algorithms from scratch. Meanwhile, it can provide an efficient speech prosody research tool for non-technically oriented users.

The SPQR application is implemented as a Python-CGI based web application running on a server, using SQLite as the local database to store corpora and pre-computed similarity indices. We follow the static form-submit architecture, in which no running services are used: Python scripts are exposed via a Web server (e.g., Apache), and calling them from a browser accesses the DB to serialize HTML for the
client. This means that no service needs to constantly run and “listen” for an event while the application is being used, thereby minimizing server workload. Utilizing this framework, we are able to re-use the Python code base from earlier chapters in this dissertation where we implemented algorithms for time-series representation and similarity computation (as well as QBC).

Figure 9.2 shows the example configuration interface of the SPQR web tool. Using this interface, the user is able to configure the parameters of the query algorithm such as time-series representations and distance measures. Currently we are implementing more features to allow greater configurability for the users, such as the ability to import new corpora, query using audio files and user supplied pitch files, and other parameters related to the query and retrieval algorithm.

Once the user selected appropriate configurations of query parameters, at the click of the “QBC” button, the application will randomly sample $k$ tone time-series objects.
Figure 9.3: SPQR web tool QBC results page (partial) screenshot.
from the database and perform the Query by Content task in the backend. This set up is identical to the QBC experiments we carried out earlier.

After the computation is finished, the result is displayed. Figure 9.3 shows an example result page of the QBC on a random set of 50 queries. On the top of the page we display the results of the current QBC experiment (i.e., results from retrieving tones using the query), including the configuration parameters of the experiment, the random seed used, the average MAP score, etc. On the bottom half we show interactive features of the query and retrieval results: First we display the query time series\(^1\), and then we show the top \( K \) \((K = 10\) in this case\)) results \(^2\) from the retrieved set of tones. These visualizations give users an intuitive understanding of the retrieved results under the current parameter settings. It also provides information such as the tone label of each query and retrieved result, the distance of the result from the query, etc.

Moreover, the user is able to click on the “audio” hyperlink next to each visualization of the results, thereby listening to the tone time-series in question. This amalgamation of top query results from audio, visual, meta (tone label), and mathematical (distance) perspectives pools together relevant information from various sources, which facilitates a thorough understanding and analysis of the top similar items to the query in the database under the current experimental parameters.

As discussed above, the SQPR tool will continue to evolve by adapting to different speech prosody research tasks and by including a number of more advanced tools such as motif discovery and network analysis/machine learning. All subsequent development also depends on further use cases in the speech prosody domain with input from and collaborations with speech researchers.

---

1 There are 50 queries in this experiment and here we only show a randomly selected one.
2 Due to the visibility issues, we only showed two out of the ten top results in the screenshot since showing all ten would make the text illegible.
9.3 A general framework for SPM

9.3.1 SPM: Goals

We have outlined the goal of Speech Prosody Mining (SPM) in the beginning of this dissertation: “... to discover novel and robust knowledge by developing new methods to address the challenges posed by mining this [speech prosody] data. ... this approach harnesses the power in the big data in speech prosody and contributes a set of novel technologies for researchers in speech technology, speech prosody, and linguistics”.

Throughout this dissertation, we have employed a variety of techniques and knowledge sources from linguistics, machine learning, time-series mining, speech prosody modeling, network analysis, etc., to address various tasks and challenges in the understanding and analysis of linguistic tone patterns (chiefly in Mandarin) from a large corpus. However, we have yet to speculate on why we want to mine speech prosody big data, and how we achieve the goal of a better understanding and analysis of prosody phenomena. Concretely, we ask: what can data mining tell us about prosody that traditional manual analysis of small data sets cannot? In this chapter, by linking perspectives from phonetics and phonological theory, we use one of the examples developed in this dissertation to put this in perspective.

Here, we quote a passage from a paper entitled “Speech melody as articulatorily implemented communicative function” [Xu05], discussing the two different theories on phonological encoding and phonetic implementation:

... In many experimental approaches, a different division is often assumed, namely, there is a direct link between communicative functions and surface acoustic forms. Thus the quest is typically to find the acoustic correlates of certain communicative functions, such as focus, stress, newness, questions, etc. (e.g., Cooper et al., 1985; Cooper and Sorenson, 1981; Fry,
Such approaches have met criticisms from phonologists, who argue that prosodic meanings are not directly mapped onto acoustic correlates (Ladd, 1996; Liberman and Pierrehumbert, 1984). Instead, as they argued, intonational meanings should be first mapped onto phonological structures, which should in turn be linked to surface acoustic forms through phonetic implementation rules ... these two general approaches are both only partially right ...

Given this paradox, this paper argues that communicative functions are indeed encoded in phonological structures and thus their sum cannot be directly observed in surface $f_0$ forms. That is, we cannot directly predict the exact realization of the $f_0$ contours given a set of underlying parameters in various linguistic domains.

Meanwhile, in this dissertation, we have shown that using techniques from network analysis and machine learning, we can predict the contour shape class of a prosodic category (such as a tone N-gram) using features from syntactic, morphological, discourse, phonological and other domains, even though we are not predicting its exact shape$^3$. In doing so, we are finding a new middle ground between the two stances outlined above by being able to predict fuzzy classes of surface acoustic forms while in harmony with the phonological mapping theory. This is possible precisely because of the power of big data: if we have a large enough amount of observations of the surface $f_0$ realizations of a particular prosodic category, then we can leverage our SPM toolkit and analyze the relationships between those observations in order to derive plausible (however coarse and fuzzy) categories — namely, pattern discovery via network analysis and machine learning. The rationale behind this approach lies deep

$^3$To be exact, we are not directly talking about encoding communicative functions in their sense — nonetheless, the idea is similar, that is, to observe $f_0$ correlates of other linguistic functions.
in the theory of inferential statistics such as the Central Limit Theorem: when our sample size is sufficiently large, we can approach the ground truth hidden in phonological encodings via surface realizations. Even though such phonological theories are not the main focus of this dissertation, we use this point to illustrate the rationale and goal of SPM: with bigger data, we have greater ability and power to make inferences about the observations on many problems. The SPM framework allows us to efficiently and meaningfully leverage that power. As a result, SPM has the potential to contribute to research in a variety of related domains including phonetics, phonology, phonology-syntax/semantics interface, speech prosody, as well as computational modeling of psycholinguistic processes (in Chapter 8.3.4, we showed how the SPM toolkit can be leveraged to shed light on the information theory based account of speech production). In the next section, we sketch a generalization of the SPM framework by outlining a series of methodologies and research topics, and we discuss the potential future directions for research and application.

9.3.2 SPM: Methods and Tasks

In this section, we outline a number of important SPM methods and tasks based on the results of this dissertation.

Similarity Computation for Speech Prosody Time-Series

We have shown that using time-series data mining techniques, we can improve the similarity computation for the speech prosody domain. This includes representations from both speech prosody modeling and time-series data mining:

1. Time-series representation

   (a) Parametric model representations
(b) Non-parametric representations
(c) Time-series representations (real-valued)
(d) Time-series symbolic representations

2. Distance measure

(a) Euclidean distance
(b) Dynamic Time Warping
(c) MIN-DIST(SAX)

3. Evaluation

(a) Classification
(b) Clustering
(c) Query by Content (QBC)
(d) Mean Average Precision (MAP)

The best similarity computation methods can be evaluated on a variety of data mining tasks such as classification, clustering, and query by content (QBC).

**Pattern discovery and category mapping**

We move to a higher level and analyze the relationships between time-series objects in the speech prosody database. In general, this includes two directions:

1. From contour shapes to categories: how can we discover similar contour shape patterns from the database and analyze the categorical compositions of these shape clusters?
2. From categories to contour shapes: given a particular prosodic category, how can we index/group its surface observations into contour shape classes? Can we predict if a prosodic category will take on a particular shape class?

In a general framework for SPM, we propose a three-step evaluation plan for analyzing and exploring the discovered patterns/classes/clusters, etc. Here is the process:

1. Evaluation: How effective and efficient is the pattern discovery algorithm in the speech prosody domain?

2. Analysis: analyze and characterize the patterns and their statistical properties beyond the evaluation step. This includes methods such as descriptive/inferential statistics, graph/network analysis, detection of communities, etc.

3. Assessment: Use pattern discovery to derive knowledge and guide research. This includes previously known or unknown phenomena. For example, using machine learning to investigate the computational modeling of psycholinguistic processes of tone production.

Correlations: linguistic domain interface

One of the most interesting aspects of SPM is finding correlations not only between processes within the prosodic domain and the sound (phonology/phonetics) domain, but also outside of the sound domain into syntactic, semantic, discourse, etc. In this dissertation, we demonstrated how we can predict the contour shape classes of tone N-grams by interfacing with other linguistic domains and constructing a multi-stage pipeline using network analysis and machine learning.
Time-series data mining tasks

We outlined several classic time-series mining tasks in Chapter 2.5.1. Classification, clustering and query by content are the most relevant to SPM. Another classic time-series data mining task is the motif discovery in speech prosody. In the network analysis module, we also demonstrated the use of anomaly detection and outlier pruning. These tasks are not only useful in their own right in solving speech prosody related problems; they are also useful in evaluating various methods, such as distance measures and time-series representations.

Assisting linguistic/prosody research

Current investigations on the phonology of intonation and tones (or pitch accent) typically employ data-driven approaches by building research on top of manual annotations of a large amount of speech prosody data (for example, Moren et al. [MZ06], Zsiga et al. [ZZ13] and many others). Meanwhile, researchers are also limited by the amount of resources invested in the expensive endeavor of manual annotations. Given this paradox, we believe that this type of data driven approach in phonology-phonetics interface can benefit from tools that can efficiently index, query, classify, cluster, summarize, and discover meaningful prosodic patterns from a large speech prosody corpus. Previously we have demonstrated the potential use of SPQR (for query and retrieval) in the process of investigating a large prosody database, which may be followed by further steps of automated or manual analysis. The utility of pattern discovery is highly task dependent and is therefore subject to the researchers’ goals and toolkits in developing task-specific methods, such as those demonstrated in the Part IV of this dissertation.
9.4 Future directions

The current work proposes a variety of tasks and technical solutions in SPM within a particular focus on Mandarin tones. While we have demonstrated the utility of the SPM framework to extract meaningful patterns from speech prosody data, there are many research questions and tasks that remain for future works. We propose several lines of future works that are outside of the scope of the current work.

1. **SPM with other speech prosody tasks.** In the current work, we exclusively focus on Mandarin tones (with a small generalization to Thai tones). Meanwhile, there are many speech prosody tasks with a different nature than tone tasks that merit their specific investigations. Pitch movement analysis is only one dimension of speech prosody. We envision extending our SPM toolbox to other dimensions of prosody, such as rhythm and accent. Even within tone and intonation, each problem merits its own consideration in terms of methodologies for computational analysis, depending on the nature of the problem. The tone contour shape prediction problem, for instance, has a completely different structure and set of methods from the motif discovery problem. Therefore, we will continue to add to the SPM toolbox through further exploration of a diverse range of prosody domain problems.

2. **Other predictors of tone shapes.** Despite our success in predicting tone contour shape classes, we haven’t completely accounted for the variability of tones in spontaneous speech. Concretely, communicative functions such as topic and focus have been shown to contribute greatly to tone shapes. However, due to the lack of high-precision method to detect such phenomena automatically, the study of these predictors will rely on manually annotated corpora, which
is costly. Therefore we can also investigate how SPM can contribute to the development of such methods.

3. **Computational modeling of psycho-linguistic processes in speech production.** Previously, we have shown how our experiments on predicting tone contour class shapes uncovered a new middle ground for the phonologically and phonetically (acoustic) encoded models of tone production. We also showed its implications for the information theory based views of speech production. These successes demonstrate how computational methods can improve our understanding of mental processes of speech and language. In the meantime, these are not the main goals of the current dissertation, and we have yet to build models to describe specific mechanisms of speakers’ optimization of information transmission in tone and intonation production.

4. **Improving tone recognition.** While the primary goal of this dissertation is the analysis and understanding of tones, the current work does not directly address the task of improving tone recognition algorithms (supervised learning) by using knowledge derived from data mining. This transfer will likely be worthy of a line of effort independent of the current work (but will inevitably also build on this work).

5. **Extending to other languages.** Can SPM be used to investigate other tone or tone-like phenomena in other languages? What about intonations and pitch-accents? What can we learn from mining different tone languages and non-tone languages, with regard to computational methods and linguistic theory? These are examples of how SPM can help answer cross-language and typological questions in linguistic research.
6. **Collaborations targeting domain-motivated research questions.** SPM is ultimately targeted at advancing the understanding, analysis, and state-of-the-art technologies related to speech prosody. It will likely evolve through applications to well-motivated problems in specific prosody-related domains, as well as collaborations with colleagues from relevant disciplines. There are many classic problems that are traditionally solved through manual analyses on smaller data sets but can benefit from the insights of bigger data. By working on these problems, we can begin to develop and evaluate SPM with a deeper understanding of its ever evolving goals, methods, and utilities. Most importantly, linguistic domain knowledge and statistical tools will complement each other to shed light on old and new challenges in speech prosody research.
Bibliography


220


[ZRS15] Shuo Zhang, Rafael Caro Repetto, and Xavier Serra. Predicting pairwise pitch contour relations based on linguistic tone information in Beijing