Rochester Institute of Technology
RIT Digital Institutional Repository

Theses

11-1-2011

Non-intrusive identification of speech codecs in digital audio signals

Frank Jenner

Follow this and additional works at: https://repository.rit.edu/theses

Recommended Citation
Jenner, Frank, "Non-intrusive identification of speech codecs in digital audio signals" (2011). Thesis. Rochester Institute of Technology. Accessed from

This Thesis is brought to you for free and open access by the RIT Libraries. For more information, please contact repository@rit.edu.

Non-Intrusive Identification of Speech Codecs in Digital Audio Signals

by

Frank Jenner

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of
Master of Science
in Computer Engineering

Supervised by

Assistant Professor Dr. Andres Kwasinski
Department of Computer Engineering
Kate Gleason College of Engineering
Rochester Institute of Technology
Rochester, New York
November 2011

Approved by:

Dr. Andres Kwasinski, Assistant Professor
Thesis Advisor, Department of Computer Engineering

Dr. Shanchieh Yang, Department Head
Committee Member, Department of Computer Engineering

Dr. Juan Cockburn, Associate Professor
Committee Member, Department of Computer Science

Thesis Release Permission Form

Rochester Institute of Technology
Kate Gleason College of Engineering

Title:

Non-Intrusive Identification of Speech Codecs in Digital Audio Signals

I, Frank Jenner, hereby grant permission to the Wallace Memorial Library to reproduce my thesis in whole or part.

Frank Jenner

Date

Dedication

I would like to dedicate this thesis to my grandmother, Rosalin Rubinstein, and my great aunt, Anne Biben. Both educators, they understood and promoted the importance of lifelong learning, and invested heavily in my education from the beginning. I owe my college education to their valuable foresight and contributions.

Acknowledgments

I would like to thank my advisor, Dr. Andres Kwasinski, for his enthusiasm and support with regard to this thesis topic. He has been very optimistic and helpful in helping me to achieve my objectives. I would also like to acknowledge my other committee members, Dr. Yang and Dr. Cockburn, for their added contributions and crucial role in the process of completing my thesis requirements.

Abstract

Non-Intrusive Identification of Speech Codecs in Digital Audio Signals

Frank Jenner

Supervising Professor: Dr. Andres Kwasinski

The use of speech codecs plays a very important role in modern telecommunications networks. Decades of extensive research have yielded voice compression techniques and improvements that have given rise to numerous speech codec standards. However, despite an interesting array of potential applications, very little research has been performed with regard to the ability to distinguish between these codecs in an audio signal. The identification of codecs in speech signals could be used to provide information about call origins, to aid in network diagnostics related to call quality, or to decide how to better enhance the signal prior to performing speech recognition.

The research presented in this paper seeks to provide a novel approach for accurately identifying among several common speech codecs from a speech signal. The developed approach is non-intrusive, requiring no information about the original input signal, nor access to the compressed bitstream. Instead, the identification is performed by analyzing only the reconstructed audio signal. This is a particularly challenging task because all codecs strive to output a signal that is perceptually indistinguishable from the original. As a result, there are only very subtle artifactual differences between speech signals processed with different codecs.

The identification technique developed in this research involves analyzing the input signal to generate a profile that characterizes several features of the input signal. The features include several noise spectra that attempt to isolate artifactual noise components in the signal from the signal components that fit a particular speech model, as well as a histogram of sample amplitudes that attempts to capture quantization patterns in the signal. The profiling procedure is first applied to signals that have been processed with known codecs in order to create a set of training profiles. To then identify the codec in an unknown test signal, the same profiling procedure is applied, and the resulting profile is compared to each of the training profiles to decide which codec is the best match.

Overall, the proposed strategy generates extremely favorable results, with an average of 95% of all test signals' codecs being correctly identified. In addition, the profiling process is shown to require a very small analysis window of less than 4 seconds of signal to achieve these results. Both the identification rate and the small analysis window represent dramatic improvements over previous efforts in speech codec identification.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

In telecommunications, speech codecs are used to increase network capacity while maintaining reasonable voice quality among calls. Although the sound at the receiver tends to represent the original signal well perceptually, the compression schemes used in speech codecs are lossy, resulting in subtle discrepancies between the original signal and the received signal. In this research, we explore these discrepancies among several different codecs in an attempt to find distinguishing features by which to identify particular codecs.

The successful identification of a particular codec in a received speech signal can have many applications. The most obvious use is to extract information about the source of a call. Due to the diversity of codecs between telecommunications networks, the detection of a particular codec may be sufficient to localize the call to a particular network or set of networks. For example, the detection of the SILK codec, commonly used by the Skype client, would probably indicate that the call originated from a VoIP (Voice over Internet Protocol) session. Similarly, the presence of the Adaptive Multi-Rate (AMR) codec might indicate that the traffic was from a GSM cellular network, whereas the presence of the Enhanced Variable Rate Codec (EVRC) would suggest provenance from a CDMA cellular network. This localization could be useful for audio forensics purposes and for targeted content delivery.

As an example, a person navigating a voice-controlled telephone menu system might provide sufficient speech data for the voice menu system to determine which type of network the caller is using, and to play a targeted advertisement as appropriate based on their network. Another important application of the ability to identify codecs is for in-service non-intrusive measurement devices (INMDs). These devices are deployed in communications networks to monitor factors which may degrade speech quality, such as speech level, noise level, echo loss, and speech echo path delay. Because the choice of codec also has a substantial effect on the quality of the

speech, it would be appropriate to incorporate this information into the INMD [21].

In order to be the most useful, the technique for identifying a codec from an audio signal should be non-intrusive. This means that the determination must be made based only upon information available at the output terminal. That is, no access to the channel or codec system (intrusive) is allowed, nor is any control over the input (semi non-intrusive) possible [26]. Although a non-intrusive methodology is the most useful, it is also the most challenging. Access to channel data or input/output relationships would certainly provide a wealth of information of great utility in differentiating between the codecs. However, in the case of our research, where we have elected to use a non-intrusive approach, the identification of codecs must be based solely upon the imperceptible artifacts in the output audio.

Although the final product from this research is a methodology by which codecs can be identified in a non-intrusive manner, the methodology itself is based off of a working knowledge of speech coding and the internal operations of several common codecs. By exploring the techniques used in the codecs amongst which we seek to identify, a better insight and understanding of the artifactual fingerprints in the outputs can be attained. These discrepancies may then be profiled, and those profiles subsequently used for comparison against signals of unknown origin in order to make an identification decision.

Chapter 2

Background

In order to understand how to effectively analyze the speech codecs and their effects on the output signal, it is instructive to first provide a brief overview of the evolution of speech coding, followed by detailed explanations of the concepts behind several common speech coding techniques. The techniques covered in this section form the basis for the codecs examined in this study.

Speech coding has long been an important research area within the telecommunications field, with efforts beginning as early as the 1930's at Bell Telephone Laboratories. During World War II, interest in speech coding began to grow, as it allowed for more efficient representation of voice data over encrypted channels. Throughout the 1940's and 1950's, most speech coding implementations were based on analog speech signals, although primitive digital representations of speech (including PCM and several of its variants) were starting to be developed during the same time. By the end of the 1950's, the underpinnings of the important source-filter model for speech synthesis had been developed. This model was augmented with linear predictive coding (LPC) techniques in the 1960's. In conjunction with the rise of VLSI computer systems and additional research efforts in digital signal processing, these fundamental components formed the basis for a new burst of proposed speech coders in the 1970's and 1980's. Research in the 1980's and 1990's concentrated on codecs that improved the perceptual quality of speech at low bitrates, including the notable Code Excited Linear Prediction (CELP) codec [25]. Finally, research in the 1990's and 2000's concentrated largely on robust speech codecs for mobile wireless technologies and internet voice applications [11].

The primary goal of speech coding is to reduce the channel capacity required to transmit speech, so that more suscribers can be supported concurrently on a single band-limited communications medium. To this end, speech coding exploits the inherent redundancies of speech signals in order to compress the raw speech data

into a lower bandwidth signal suitable for transmission in a telecommunications network [11]. Although most of the demand for speech codecs arises from the need for greater channel capacity in realtime communication, coded speech is equally suitable for applications which require efficient storage of speech data, such as in digital voice recorders, voicemail systems, and toys [23].

The selection of a codec for a particular application depends on many factors, including channel bandwidth, implementation complexity, latency, computational requirements, speech quality, robustness to errors, and licensing terms and costs [12]. In fact, despite the long and constantly-evolving history of speech coding, many of the early digital codecs are still used in contemporary applications because they perform well with regard to one or more of these factors. Because of the diverse selection of speech codecs that have been developed, communications networks tend to be quite heterogeneous in terms of which codecs are used for transmitting voice traffic. This diversification lends itself well to the identification of a particular network based upon the codec detected in the received audio signal. Thus, it becomes of interest to develop a methodology for differentiating between codecs in a received speech signal.

Speech codecs may be broadly categorized into two groups: waveform coders and vocoders. Some literature also makes reference to an additional class of hybrid coders, referring to codecs that borrow techniques from both of the other types. In fact, all of the vocoders in this study would actually fall into the category of hybrid coders, but will nonetheless be referred to herein simply as vocoders in order to better distinguish them from pure waveform coders.

2.1 Waveform Coding Techniques

Waveform codecs are designed to be signal-independent and are therefore suitable not only for speech, but also for musical signals and voice-band data. A waveform codec strives to present the closest replica of the source signal as possible at the output [11]. As a result of the exactness and versatility with which waveform coders compress the signal, the voice quality at the output tends to be exceptional. Quality is evaluated using a subjective 5-point Mean Opinion Score (MOS), as rated by a panel of human listeners. In addition, waveform coders tend to be very computationally simple, making them ideal for applications requiring low processing overhead. Unfortunately, all of these benefits are also countered by very low compression ratios, with output bitrates typically ranging from 16 kbit/s to 64 kbit/s [23].

2.1.1 Companding

One of the first steps in any digital signal processing application is to sample and quantize the input signal. The sampling rate used for most telecommunications applications is 8 kHz (referred to as narrowband), because the majority of the spectral content of speech is focused below 4 kHz. All of the codecs considered in this study operate using narrowband sampling. In addition, the analog-to-digital converter (ADC) typically uses a linear scale for digitizing the amplitude of the sample. This digital signal consisting of a sequence of regularly sampled, linearly quantized values is known as linear pulse code modulation (PCM), or uniform PCM. Uniform PCM in considered to be raw, uncompressed signal data, and is a very prevalent digital representation of audio signals.

The goal of speech coding, of course, is to reduce the amount of data that needs to be transmitted over the channel. Thus, uniform PCM at the full resolution of the ADC is a poor candidate for use in any telecommunication network branch. The naive approach to reduce the amount of data transmitted would be to simply transmit some fixed number of most significant bits of the digitized data for each sample. However, by truncating the least significant bits, the resulting quantization error is very significant for low-amplitude samples. This is undesirable for a couple of reasons: lower amplitude samples occur more frequently than higher amplitude samples in speech signals [24], and the human auditory system is more sensitive to small signal fluctuations during periods of low amplitude sounds than high amplitude sounds [20]. As a result, the high relative quantization error caused by simply truncating each sample can cause a profound degradation in the perceptual quality of the speech.

To counter the unwanted effects of uniform quantization during bit downconversion, the technique of companding has been developed. In this practice, each sample is requantized on a logarithmic or near-logarithmic scale which transforms the high precision samples to span the range available using a desired number of bits. For example, the source signal might be sampled by the ADC with 14-bit precision, and then transformed to 8-bit samples using an intermediate logarithmic scaling function. The logarithmic scaling grants more precision to lower amplitude samples than higher amplitude samples. In effect, the relative quantization error is reduced at the low end, and the SNR becomes near constant across the entire dynamic range [19]. The name companding (sometimes also known as compansion, logarithmic PCM, or non-uniform PCM) arises from the notion that the quantization intervals are compressed at the transmitter, and the audio is recovered at the receiver by re-expanding

the non-uniformly quantized values back into a linear domain using the inverse scaling operation. Figure 2.1 shows a hypothetical example of a companding function that downconverts 8-bit sample values into 5-bit codewords. Note how the quantization step size is very small for low amplitude inputs, and increases with the input amplitude.

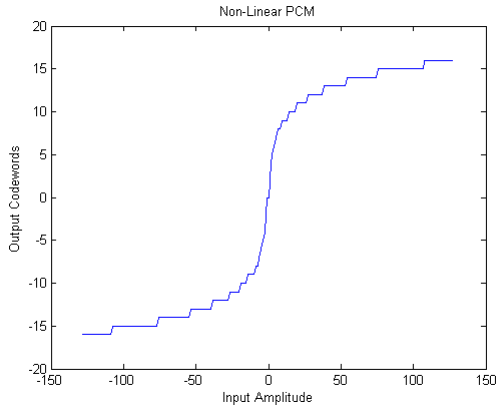

Figure 2.1: Hypothetical logarithmic companding characteristic curve

Companding is exemplified by the codecs specified in the ITU-T G.711 recommendation. This document outlines two codecs – A-law and μ-law – that are based completely upon the companding technique. Both of the G.711 codecs convert samples into 8-bit representations, leading to a 64 kbit/s encoding bitrate for speech sampled at 8000 Hz. A-law takes 13-bit resolution samples as inputs, and is commonly used in Europe, while μ-law takes 14-bit resolution samples as inputs, and is used in the United States and Japan. Although the compression ratio is quite low, these codecs are still favorable because of the extremely low computational complexity (they can, for example, be implemented entirely as a lookup table) and excellent audio quality achieved. The scaling function for A-law is shown in (2.1), and the scaling function for μ law is shown in (2.2), where μ is specified to be 255, and A is 87.6 [24].

$$f(x) = \frac{\ln(1 + \mu \times |x|)}{\ln(1 + \mu)} \mathrm{sgn}(x) \tag{2.1}$$

$$f(x) = \frac{A \times |x|}{1 + \ln(A)} \mathrm{sgn}(x) \tag{2.2}$$

2.1.2 DPCM/ADPCM

Differential PCM (DPCM) is another common waveform coding technique. DPCM takes advantage of the fact that audio signals tend to be highly correlated in the short-term. This assumption means that, given a history of a signal's previous values, the current sample's value may be predicted quite accurately. If this is the case, then the error residual between the predicted value and the actual value will be much smaller than the value of the sample amplitude itself, therefore fewer bits would be necessary to code the residual. Thus, in a DPCM codec, only the error residuals for each sample are quantized and transmitted across the channel, and the residuals are decoded at the receiver and added with the local prediction value in order to recover an approximation of the original sample [11]. A block diagram of a generic DPCM coding scheme is shown in Figure 2.2.

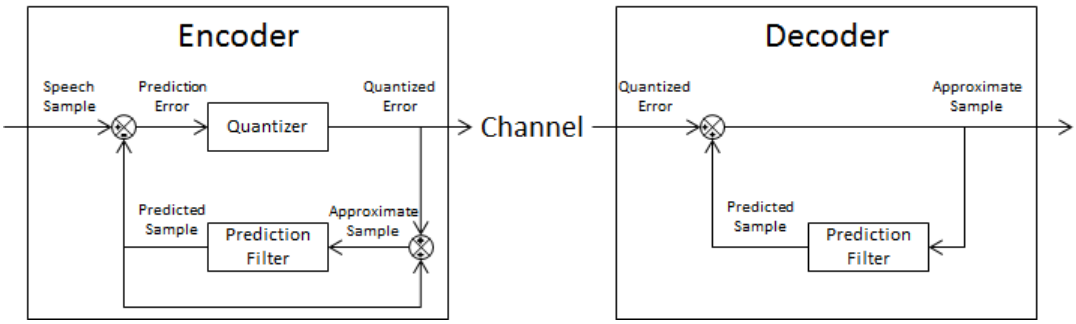

Figure 2.2: Generalized backward-predictive DPCM codec

A popular variant of DPCM is adaptive DPCM (ADPCM). In this scheme, the prediction filter coefficients, and possibly also the quantizer scaling factor, vary with the changing signal characteristics [24]. For example, if the signal is fluctuating very rapidly, then the quantization step size should adapt to become much larger in order to accommodate the larger amplitude swings. Conversely, if the signal is very stable, then the step size may be greatly reduced in order to instead improve accuracy. The G.726 codec is a prominent example of an ADPCM codec, and operates at bit rates of 40, 32, 24, or 16 kbit/s.

DPCM and ADPCM codecs may be either forward predictive, in which the prediction and quantization parameters are explicitly encoded into the channel bitstream, or backwards predictive, in which the parameters are inferred at the receiver based upon previously decoded signal data. The former technique ensures more accurate reconstruction at the cost of higher data overhead. The latter technique (used in

G.726), has better bitrate economy, and also has the desirable byproduct of decaying the effect of transmission errors [11].

2.2 Vocoder Techniques

In contrast with the simple waveform coding techniques, vocoders do not attempt to replicate the input waveform, but instead seek to conform the input signal to fit a known vocal model. Rather than individually encoding each sample using as few bits as possible, vocoders analyze an entire frame (typically a 20 ms segment) of the signal and attempt to find a set of model parameters that can be used to accurately resynthesize that frame of audio. Once an appropriate fit has been established, only the model parameters need to be transmitted across the channel. At the receiver, these parameters are fed into a speech synthesizer that is based upon the same vocal model in order to generate synthetic speech resembling the original signal. This method results in significant bitrate savings and a higher compression ratio. However, vocoders tend to be much more complex systems than waveform coders, and therefore impose substantial computational resource requirements. For example, whereas the G.711 waveform codec typically requires well under 1 million instructions per second (MIPS) to operate in real time, the G.728 hybrid vocoder requires around 30 MIPS [23]. Also, because vocoders are designed specifically for voice signals, they are inadequate for accurately representing sounds which do not fit the voice model, such as musical tones. Impressively, despite the significant compression and limitations of vocoding, many contemporary codecs based on these techniques actually achieve comparable, or even better, speech quality under clean test conditions when compared with waveform codecs [7].

2.2.1 Human Speech and the Source-Filter Model

In order to develop a vocoder, a suitable underlying human voice model must be selected. Developing a model requires an understanding of the voice production mechanisms present in the human body. Speech consists of a sequence of distinct sound segments, called phonemes, which combine to form words. Each phoneme is characterized by the type of excitation source used to generate the sound, and by the spectral envelope through which that excitation source is filtered. For many phonemes, for instance the vowel sounds /æ/, /i/, and /U/, the excitation source consists of energetic pulses of air that form when exhaled air causes the vocal cords

to open and close at the glottis. Speech segments with this type of excitation signal are referred to as "voiced" speech. Alternatively, the excitation source for many other phonemes, such as /sh/ and /f/, is simply noise from air turbulence as it is expelled from the lungs. In this case, where there are no glottal pulses, the speech is said to be "unvoiced". Regardless of the excitation source, the resulting sound waves then pass through the vocal and nasal tracts, where the various acoustical resonances shape the sound further. Thus, the spectral envelope is determined by the geometry of these cavities, which are in turn under control of the speaker in order to produce intelligible language [16].

Figure 2.3 shows an example of the spectrogram for a male speaker reading the sentence "He swung up over the wheel". The periods of voiced speech are clearly distinguished from the unvoiced speech by the regions of light parallel bars. In the time domain, the glottal pulses of voiced speech take the form of high energy, quasiperiodic spikes. In the frequency domain these pulses translate to peaks at the pitch frequency and its harmonics (integer multiples). In speech signals, the strongly resonant harmonics are known as formants. The formants manifest themselves in the spectrogram as parallel bars at multiples of the voiced pitch, with most of the energy concentrated in the lower frequency regions. The energy of the harmonics diminish at higher frequencies in accordance with the spectral envelope imposed by the vocal tract. In contrast, note that the unvoiced sounds, such as the /s/ in "swung", contain much lower energies, and that the spectrum is actually quite flat in these regions.

Although many models have been developed to provide detailed representations of the human speech production mechanisms, unquestionably the most common is the source-filter (or source-system) model [19]. This speech synthesis model is popular due to its simplicity and accuracy. Much like the human speech synthesis mechanisms, the source-filter model is separated into an excitation source and a spectral envelope filter. For voiced speech, the excitation signal is a glottal pulse approximation whose periodicity matches the pitch of the analyzed speech frame. For unvoiced speech, the excitation signal typically comes from a noise source. The excitation signal is then passed through a time-varying filter whose response mimics the spectral envelope observed in the original speech. In this manner, the filter models the vocal tract formants [16]. With the appropriate choice of excitation signal and filter, this model is capable of synthesizing very high quality speech.

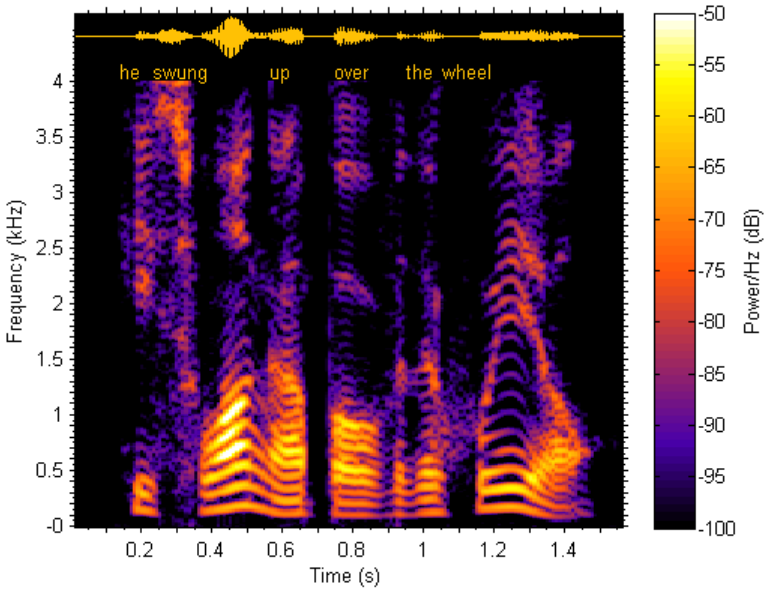

Figure 2.3: Spectrogram for a male speaking a sentence

2.2.2 Linear Predictive Coding

One of the most important implementations of the source-filter model is based on linear predictive coding (LPC). In LPC vocoders, the excitation source is provided by either an impulse train, for voiced speech, or random noise, for unvoiced speech. Although very accurate models for the vocal tract have been developed that take into account many subtle features of speech such as lip radiation and certain theoretical augmentations for fricative and nasal phonemes [19], the vocal tract model can be simplified into a time-varying, all-pole filter [13]. The basic LPC model is shown in Figure 2.4.

The term linear predictive coding arises from the way in which the filter is represented in the time domain. Consider the all-pole filter with impulse response $H(z)$ that relates the desired output speech signal $S(z)$ with the excitation signal $X(z)$, as shown on the left half of (2.3). For convenience, the excitation gain factor K has been absorbed into the filter model. In the time domain, shown on the right, the interpretation is that the current output sample, $s(n)$, may be predicted as a linear combination of the previous p outputs and the current input [13]. A similar type of predictor is used in the ADPCM codecs discussed previously [11].

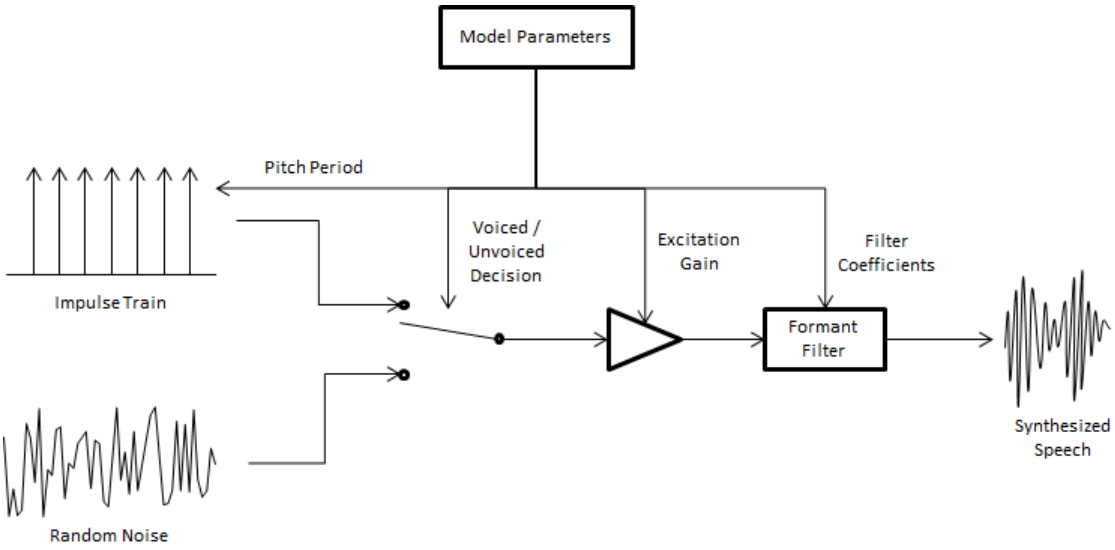

Figure 2.4: LPC model of speech synthesis

$$H(z) = \frac{S(z)}{X(z)} = \frac{K}{1 - \sum_{j=1}^{p} a_j z^{-j}} \overset{\mathcal{Z}^{-1}}{\Longrightarrow} s(n) = Kx(n) + \sum_{j=1}^{p} a_j s(n-j) \tag{2.3}$$

Given that the desired output sequence $s(n)$ is known (indeed, the synthesized speech should be as close as possible to the actual speech signal), the goal is to find the filter coefficients a_j which minimize the impact of the excitation signal in synthesizing the desired speech. This would mean that even a poorly chosen excitation signal should still render intelligible speech due to the appropriate formant filter shaping. Thus, the mean squared error between the desired speech output and the linear combination of past outputs must be minimized. This results in a system of p equations in p unknowns:

$$E\left\{\left[s(n) - \sum_{j=1}^{p} a_j s(n-j)\right] s(n-i)\right\} = 0, \quad \text{for } i = 1, \ldots, p \tag{2.4}$$

Because these equations depend on the expected value function, they are valid only for stationary signals. Speech is not a stationary signal, but it does remain approximately stationary for short time segments. Hence, these equations must be evaluated every frame. Fortunately, through the use of the autocorrelation function, the system can be reduced to a Toeplitz matrix which can be solved in $O(n^2)$ time by

using the recursive Levinson-Durbin algorithm. Alternatively, other methods based on covariance or lattice algorithms may also be used to efficiently solve for the filter coefficients [13].

The order, p, of the short term filter (sometimes also referred to as the LPC filter) is based upon the underlying vocal tract model of lossless tubes, and should generally consist of at least 8 taps. Empirical results have shown that the segmental SNR improves only modestly as the filter order is increased beyond around 12. Similar experiments have been performed with regard to selecting the appropriate frame length. Shorter frames mean that the assumption of stationarity is more likely to be valid for the frame, but there is less information to work with to make accurate predictions. Conversely, longer frame lengths offer plenty of data, but may violate the assumption of stationarity if the speech characteristics are changing rapidly. It was found that frame lengths of around 20 ms (160 samples at 8 kHz) minimize the error encountered during the LPC analysis [11].

As seen in Figure 2.4, the only data that needs to be sent over the channel are the filter coefficients, pitch period, excitation gain, and a voiced/unvoiced flag. These parameters need only be sent once per frame. As a result, LPC vocoders are known for very low bitrate coding, with some codecs requiring 1 kbit/s or less [16]. In general, the LPC filter coefficients, which have a direct effect on the pole locations and hence the frequency response of the filter, have a high dynamic range. This makes them poor candidates for direct quantization, as the quantization error could be very high, and the quantized values could even cause the filter to become unstable. As a result, the LPC coefficients are usually converted into a much more stable representation known as line spectral pairs (LSPs) prior to quantization and transmission [4].

2.2.3 Analysis-by-Synthesis

Although LPC vocoders allow for very high compression ratios, pure LPC codecs are rarely used in mainstream telecommunications. While LPC vocoders do provide intelligible speech at low bit rates, the speech is not of high quality from a MOS standpoint. The speech from LPC vocoders tends to exhibit a "buzzy" quality due to the inadequate representation of the excitation signal by an impulse train [16]. For this reason, they tend to be used only in environments where channel bandwidth is extremely constrained, as in many military applications. In fact, none of the codecs used in this research are purely LPC vocoders. Instead, they rely on hybrid analysis-by-synthesis (AbS) methods that augment the LPC model.

Analysis-by-synthesis approaches attempt to improve the output speech quality by locally synthesizing the speech at the encoder, and then tweaking the model parameters until the error between the input speech and synthesized speech is minimized. Once the optimal set of model parameters has been found, they are transmitted to the receiver, which will decode and synthesize the minimal error speech representation.

In reality, it is computationally infeasible to exhaustively test every possible set of model parameters. Although the source-filter model used in the LPC vocoder is an accurate speech synthesis model, the primitive representation of the excitation signal as either an impulse train or random noise is a major shortcoming of the LPC model with regard to quality. Thus, only the excitation signal is determined through AbS testing, while the spectral filter is borrowed directly from LPC analysis. Furthermore, the excitation signal can be divided into a shape and a gain, where the gain can be calculated independently. The only model parameter that needs to modified during resynthesis, then, is the excitation signal shape [16].

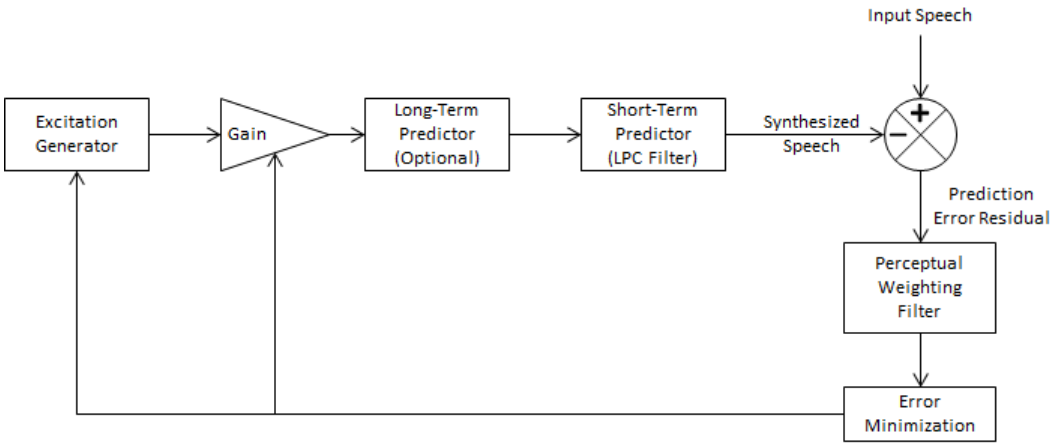

Figure 2.5: Analysis-by-synthesis encoder structure

The closed loop feedback inherent in AbS codecs also takes advantage of an additional perceptual aspect of human speech called frequency masking, or auditory masking. This concept states that signals within a certain frequency range (called a critical band) around a spectral peak will be perceptually masked by the signal at the dominant peak [20]. Similarly, errors within those critical bands will also be masked. Thus, in speech coding, the perceptual quality of the speech will be enhanced if any quantization of model parameters is distributed such that higher quantization step sizes are used for parameters that affect the signal near spectral peaks and lower step

sizes are used around less powerful spectral regions. For AbS coding, the synthesis error is evaluated after being filtered with a perceptual error weighting function that exploits this quality of human speech perception. This practice is known as noise spectral shaping [16].

While the LPC filter in AbS codecs removes short-term redundancies in the speech signal due to the high correlation between samples, some codecs elect to provide even more residual signal whitening by employing prediction between pitch periods. Because voiced speech is quasi-periodic, there is much redundancy between signal segments that are a pitch period apart. Some AbS codecs therefore also implement a long-term predictor that, in concert with the short-term predictor (LPC filter) decreases the overall prediction residual and can therefore further reduce bitrates [16]. A simple diagram of the encoder structure for an AbS encoder is shown in Figure 2.5.

2.2.4 Code Excited Linear Prediction

By far the most common types of AbS codecs are those based upon code-excited linear prediction (CELP). Clearly, if an AbS codec were to transmit the actual excitation sequence for a given frame, there would be little bitrate savings over simply transmitting the original signal. Efficient encoding in AbS codecs therefore requires a concise representation of the excitation signal.

During the analysis process, it turns our that the error residual signal after applying both long-term and short-term prediction to the input speech is nearly Gaussian. In the groundbreaking paper [22], it was demonstrated that using a table of 1024 Gaussian random sequences (known as a codebook) as candidates to represent the residual signal (which becomes the excitation signal for synthesis), it is possible to find at least one candidate which results in a very close replica of the original signal when synthesized. If the same table is used both at the encoder and decoder, then the transmitter need only send the codebook index (in this case, only 10 bits) of the optimal sequence in order to represent the excitation signal. The use of linear predictive filters to remove signal redundancy in conjunction with the use of a codebook to determine the residual signal through AbS techniques is the basis for CELP codecs.

The representation of an arbitrary signal segment into one of a fixed set of segments, such as a codebook, is known as vector quantization. Vector quantization is a very important aspect of contemporary codecs. In the original CELP codec (introduced in 1985), each excitation vector in the codebook was exhaustively tested in the

AbS feedback loop. This approach was obviously very computationally inefficient, operating 125 times slower than realtime even on (what was considered to be at that time) a "supercomputer". However, subsequent developments in CELP research have led to vector quantization techniques and codebook structures that permit fast search algorithms, lending to an explosion of CELP-based codecs that operate in realtime [16]. The G.728, G.729, iLBC, AMR, and SILK codecs examined in this study are all based upon CELP coding techniques.

2.3 Codec Selection

While some of the most common speech coding techniques have just been discussed, there are still many other techniques and practices that have not been covered, even though they may be prevalent in certain applications. Nonetheless, a sufficient background has been presented to understand the underlying principles behind all of the codecs used in this study. The codecs amongst which we have aimed to identify in this research are as follows:

- G.711 μ-law – Companding [30]
- G.726 – ADPCM [27]
- G.728 – Low-latency CELP [28]
- G.729 – Conjugate structure algebraic CELP [31]
- iLBC (Internet Low-Bitrate Codec) – CELP [2]
- AMR (Adaptive Multi-Rate) – CELP [8]
- SILK – CELP [32]

These codecs are selected due to their widespread use in mainstream telecommunications networks, including cellular, VoIP, and PSTN networks. Furthermore, they have reference implementations and/or executables freely available.

Chapter 3

Related Work

The identification of speech codecs in audio signals is a rather niche objective that has seen limited research. This section attempts to describe some of the motivating works for our research, as well as to present the results of some pursuits with very similar goals.

3.1 Imaging Fingerprints

The primary motivating work for this research actually stems from image processing. The work in [26] demonstrates the identification of particular digital camera makes and models based upon the content of the output image. This identification was extended further to include the detection of tampered images by applying the same identification techniques to localized regions of the image, and determining whether there are inconsistencies in the identification results. This research therefore demonstrates a non-intrusive methodology for determining system information based upon output signals that are perceptually similar. Although the models and identification features used in the image processing research diverge significantly from those used in speech coding, the research has nevertheless been the inspiration for our selection of research topic.

3.2 Alley's Speech Codec Identification

Despite the rich history of speech coding, relatively little effort has been put into the determination of speech codecs in audio signals. Admittedly, the applications of such an ability are rather limited. Nonetheless, at least two other studies have examined the prospect of identifying speech codecs.

The work by Alley in [1] demonstrates a methodology specifically oriented at the

identification of speech codecs in a telephony channel. In this early work (dated 1993), an adaptive least mean squares filter is placed across a communications channel, and several of its statistics are measured. In particular, the variance of the maximum filter coefficient, the input signal power, and the probability distribution histogram of the error output of the filter are used as features for a multilayer perceptron (MLP) neural network classifier.

The results of this work are reproduced in Table 3.1. Although the figures appear to be quite favorable, there are a few notable shortcomings of this research. Most obviously, the codec selection among which the signal may be identified is extremely limited, consisting of only two simple waveform coders (and with the ADPCM bitrate unspecified). Of course, it would be desirable to be able to distinguish between a much more diverse set of codecs than those presented in this study. Also, the use of a neural network is a fairly heavyweight approach for a problem that has been resolved with comparable success using a much simpler identification strategy.

Learning Rate ϵ	Channel Type	% Correctly Identified
0.025	Linear	95
	A-law	86
	ADPCM	91
0.05	Linear	97
	A-law	86
	ADPCM	92
0.075	Linear	91
	A-law	83
	ADPCM	97
0.10	Linear	94
	A-law	83
	ADPCM	87

Table 3.1: Codec identification results from Alley

3.3 Scholz' Speech Codec Identification

The most important supporting work for our research is presented by Scholz in [21]. This work, in fact, embodies the exact goals that we are trying to achieve in our research. Specifically, they are able to successfully identify between a diverse selection of popular codecs in a completely non-intrusive manner.

The identification scheme used in Scholz' research is based around the harmonic-plus-noise decomposition developed in the multiband excitation (MBE) voice model

[10]. The MBE vocoder is an attempt to synthesize high quality speech at low bitrates by using an excitation signal that consists of both a harmonic component and a noise component. The input speech is analyzed in the short-time frequency domain. For each frame, once the pitch has been determined, the spectrum is divided into bands representing each harmonic. Based on the power and shape of each band, the band is declared as either voiced or unvoiced. This allows for a much more accurate representation of speech compared to having a single voiced/unvoiced decision for an entire frame of audio. The synthesis model parameters therefore consist of the voiced/unvoiced decisions for each frequency band, the magnitudes and phases of those bands declared as voiced, and the fundamental frequency. At the receiver, the spectra of the voiced bands are synthesized by using shifted copies of the analysis window spectrum (they call this the harmonic spectrum) that are scaled in accordance with their magnitude measurements. The unvoiced bands are replaced by spectra of band-limited random noise.

In Scholz' work, a set of training audio samples from various codecs are first pre-processed to find frames that are considered to be voiced speech. These voiced frames then continue to be analyzed using the harmonic-plus-noise decomposition. This results in an artificial harmonic spectrum that contains a purely voiced representation of the input speech, and a difference spectrum that highlights the discrepancies between the input speech spectrum and the harmonic spectrum. It is hypothesized that the shape of this difference spectrum (called the noise spectrum) is largely defined by the codec used on the input speech.

Based upon this premise, the noise spectra for the voiced frames of numerous speech samples are aggregated to construct a training profile of the noise spectrum of each codec of interest. In order to identify a codec from a speech sample, then, the unknown test sample must undergo the same processing steps, and its profile should be compared to the training profiles of known codecs. The codec whose profile fits the best (where the fit is determined via a normalized cross-correlation between the training profile and the test profile) is determined to be the source codec.

The major results of their research are reprinted in Table 3.2. Overall, the accuracies are very impressive. However, the approach does have a few problems. First, many ADPCM samples are being incorrectly identified as G.711 codecs. More importantly, although not evident from the results shown, this identification process requires analyzing a long speech sample. The table shown contains the results obtained

using 2560 voiced frames per test sample. For their claimed framing parameters, signal voicing characteristics, and sampling rate, this corresponds to approximately two minutes of audio. They also present results obtained using only 640 voiced test frames for analysis (approximately 30 seconds of audio), although the accuracies suffer significantly for that test. Lastly, while the results show that codecs can be identified successfully, there is no indication that the bitrate or other codec settings could be determined.

			Classified As						
			G.726	AMR	EFR	G.723.1	G.729	HR	G.711
Source Codec	G.726	16 kbit/s	**100.00%**	-	-	-	-	-	-
		24 kbit/s	**100.00%**	-	-	-	-	-	-
		32 kbit/s	**85.71%**	-	-	-	-	-	14.29%
		40 kbit/s	**24.00%**	-	-	-	8.00%	-	68.00%
	AMR	4.75 kbit/s	-	**93.33%**	-	-	3.33%	-	3.33%
		5.9 kbit/s	-	**88.89%**	-	-	4.76%	-	6.35%
		10.2 kbit/s	-	**80.90%**	-	-	5.62%	-	13.48%
	EFR		-	-	**100.00%**	-	-	-	-
	G.723.1	6.3 kbit/s	-	-	20.00%	**80.00%**	-	-	-
	HR		-	-	-	-	-	**100.00%**	-
	G.711	μ-Law	-	-	-	-	11.63%	-	**88.37%**

Table 3.2: Codec identification results from Scholz

3.4 PinDr0p

The work in [3] deals with codec identification for the purpose of determining telephone call provenance. The resulting system is given the name "PinDr0p". The objective of this work is not directly to classify codecs, but to detect the classes of networks (for example, VoIP, cellular, or PSTN) through which the audio signal has traversed. They generalize that VoIP networks use G.711, iLBC, Speex, or G.729 codecs, that PSTN uses the G.711 codec, that and cellular networks use the GSM-FR codec.

In order to differentiate between the VoIP codecs, they develop a technique for detecting packet losses, taking into account possible packet loss concealment techniques, and then characterize how each VoIP codec's output is affected by lost packets. To detect the presence of the G.711 codec, and subsequently distinguish the remaining cellular codecs from the PSTN codec, they use noise estimation factors such as the noise spectral range, noise spectral deviation, and spectral clarity as profiling features for characterization. Factors from the ITU-T P.563 single-ended speech quality

measurements are also included in the characterization. All of these features are used in conjunction with a multi-label classifier. Ultimately, the system is able to detect a path traversal signature that denotes the sequence of networks (for example, PSTN→Mobile, or Mobile→PSTN→VoIP) through which a call was determined to traverse. Overall, call provenance was correctly determined with between 90% and 100% accuracy, depending on the number of training sets used. Take note, however, that the identification of codecs is neither a primary outcome of this work, nor are any results presented that indicate the success of such identification techniques. Instead, the detection of a codec is just one of many features that is used to fingerprint a call and determine its provenance. Thus, this work mainly demonstrates an application for codec determination, rather than focusing on the actual codec identification methodology.

Chapter 4

Identification Methodology

The previous chapters served to describe the motivation and applications for our research, to explain the fundamental concepts behind contemporary speech coding technologies, and to present the current state of affairs with regard to research into the identification of codecs. It should be evident that, while some research has already performed in this area, there are still many shortcomings to be addressed. In this section we begin to introduce our novel approach for highly accurate speech codec identification. Our contributions will demonstrate improved accuracies over previous work while requiring very little input audio for analysis.

4.1 Noise Spectrum

One of the prominent concepts used in our codec identification strategy is the notion of collecting noise spectrum information from the input signal. This aspect is borrowed from Scholz' work in [21], with some important modifications. Thus, a closer examination of Scholz' strategy is merited.

In Scholz work, each speech signal is broken down into overlapping frames in order to perform short-time spectral analysis. Observing the signal in the frequency domain is very important for speech coding because many of the vocoder techniques involve the compression of synthesis filter coefficients into the data codewords. Clearly, these filter parameters have a direct effect on the frequency spectrum of the decoded signal. Because different codecs may use slightly different filter models or quantize the filter parameters differently, these variations will manifest themselves as differences in the shape of the frequency spectra of those codecs' output signals. As a result, some of the characteristics that may be helpful in distinguishing between different codecs are most prominent in the frequency domain.

In addition to the aspects of the vocoders that have obvious links to frequency

spectrum characteristics, speech signals themselves can be understood more clearly in the frequency domain. Figure 2.3 demonstrated that the frequency spectrum of voiced speech contains strong peaks at the pitch frequency and several of its harmonics. As a result, the overall shape of the spectrum in voiced regions is dictated almost entirely by the pitch of the speaker and the formant structure of the phoneme being spoken. The spectrum will be continuously changing throughout the course of the speaker's conversation as the phonemes and pitch inflection change. Furthermore, the spectrum may change even more dramatically from person to person, as different speakers may exhibit widely varying pitch characteristics or formant structures.

The spectrum of a speech signal is dominated mostly by the actual speech content, and is therefore influenced only very weakly by the effect of the codec that was used on the signal. This makes the challenge of picking out spectral discrepancies between different codecs especially formidable. It becomes necessary to find a means by which to separate the components of the signal that stem from the speech itself and the components that are artifacts from the codec processing. Unfortunately, because the speech content changes significantly between phonemes and speakers, this is not a trivial task. A processing technique must be devised that analyzes a speech signal and generates one or more output parameters that are largely independent of the actual speech content. Ideally, such a technique should result in outputs that are very similar when processing speech segments that have been coded with the same codec, even if those segments contain entirely different sentences or are voiced by different speakers. Conversely, the technique should result in outputs that differ greatly when processing speech segments that have been coded with different codecs, even if the source speech is identical.

In Scholz work, the separation of the speech component from the codec artifacts was attempted through the use of a harmonic/noise decomposition. This technique is the basis for the multiband excitation (MBE) vocoder introduced in [10]. Although the MBE vocoder has been eclipsed by advances in CELP coding methods, the MBE vocoder is very effective for preserving a relatively high speech quality in the class of low bitrate speech coders. Most of the codecs that we have described previously rely upon modeling the excitation signal and vocal tract filter to represent the speech. These codecs make a binary voiced/unvoiced decision for an entire frame of speech and select model parameters based upon that initial decision. In reality, however, a speech signal may be more complex than can be modeled with such a system, and may contain both periodic components and additional noise components. The MBE

vocoder attempts to replicate both components in a given frame of speech.

The MBE vocoder first attempts to pick out a pitch for each frame. Once the pitch has been determined, the short-time frequency spectrum of the frame is divided into multiple bands for each harmonic of the pitch. Bands that contain periodic energy will necessarily take the shape of the spectrum of the windowing function that was used for the FFT, while bands that contain noise-like energy will have a more irregular shape. The shape of each band is used to make a voiced/unvoiced decision for each individual harmonic band of the frame, thereby leading to potentially more accurate representations of the speech signal than a single frame-wise voiced/unvoiced decision. The speech signal spectrum can be approximated by shifted and scaled copies of the analysis window spectrum for each of the voiced bands, and sub-bands of a random noise spectrum for each of the unvoiced bands. The transmitter therefore only needs to send the magnitudes and phases of each of the voiced harmonic bands to the receiver. The receiver can then reconstruct the spectrum and perform an inverse FFT to reconstruct the speech signal.

While Scholz' work is not concerned with the distinction between voiced and unvoiced harmonic bands or how to efficiently represent the synthesis parameters in the channel, it does borrow the idea of creating a purely harmonic spectrum by shifting and scaling copies of the analysis window spectrum. In attempting to match the actual audio spectrum with this harmonic spectrum, any interesting "noise" spectral features can then be isolated by subtracting the two spectra. An example of this harmonic/noise decomposition is shown in Figure 4.1. The top subfigure shows the magnitude spectrum of a voiced frame from a speech signal. Notice that the spectrum contains numerous peaks at regularly spaced intervals. These peaks represent the harmonics of the pitch frequency of the speech signal. Stronger peaks indicate more resonant formants in the vocal tract. The center subfigure shows the harmonic spectrum representation of the signal. This harmonic spectrum is created by centering a copy of the analysis window spectrum at the bandcenter of each of the harmonics. The magnitude of each of these copied spectra is scaled so as to minimize the error between the generated harmonic spectrum and the original spectrum. Thus, the harmonic spectrum very closely resembles the original spectrum, but consists only of a small number of periodic components. Thus, when the harmonic spectrum is subtracted from the original spectrum, only small discrepancies are evident in the resulting spectrum. This spectrum will be referred to as the noise spectrum. The

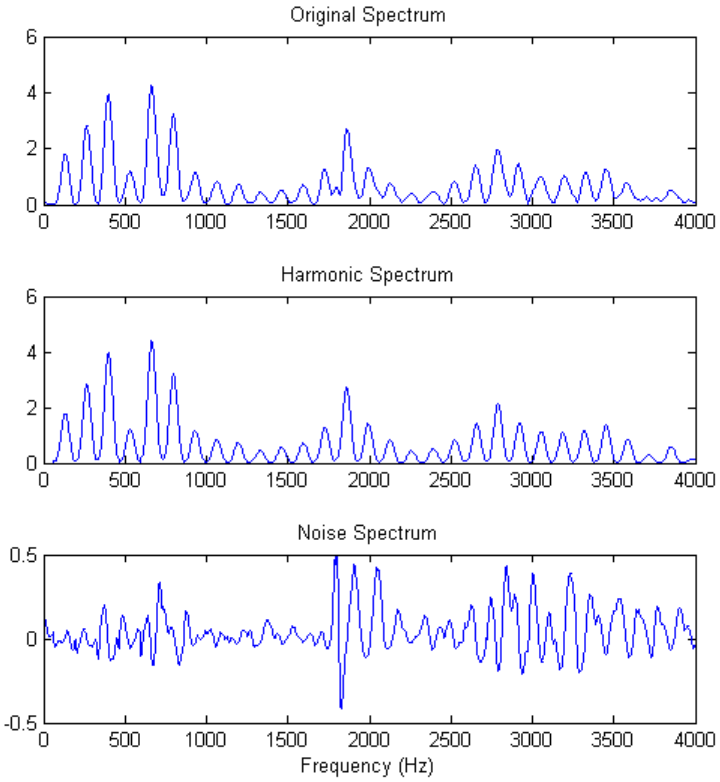

Figure 4.1: Decomposition of speech signal spectra into harmonic and noise components

noise spectrum is of interest because it is largely independent of periodic speech characteristics such as pitch and formant structure, meaning that it will be more sensitive to other signal characteristics, such as those imparted by codec processing.

To demonstrate this point, Figure 4.2 illustrates the noise spectra that are generated from the same input signal after being independently processed with a few different codecs. In these examples, each noise spectrum is actually the aggregate (via an arithmetic mean) of the noise spectra generated from numerous frames of voiced speech. In this way, each noise spectrum smooths out to accentuate only the spectral features that are persistent over a broad sampling of the speech signal. It is important to realize that, although the overall shape of the noise spectrum is similar among each codec shown in the figure, there are nevertheless distinguishing features in each spectrum that arise only from the difference in input codec. It was demonstrated in [21] that these subtle discrepancies were sufficient to distinguish between

the presence of different codecs in the analyzed signal.

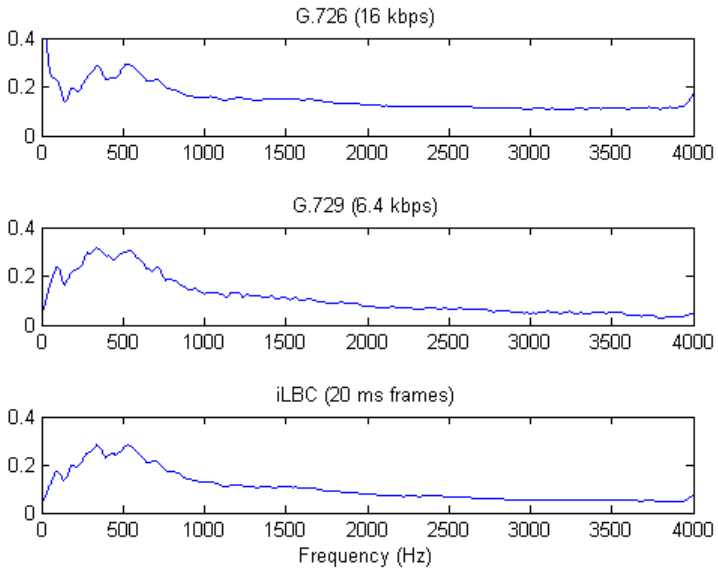

Figure 4.2: Example codec noise spectra from harmonic/noise decomposition

In our research, a very similar technique to Scholz' is used in order to extract unique noise spectra for different codecs. A novel contribution of our effort, however, is the use of existing codecs to analyze the signal and compute a noise spectrum. Similarly to the way in which the harmonic/noise decomposition conformed the signal to conform to a particular model (in that case, a small number of purely periodic bands), so too do codecs conform the signal to fit some underlying vocal model. The output of most of the codecs that we have looked at comes directly from synthesis through a vocal model. Thus, the output signal contains only signal attributes that can be expressed by that particular underlying model, regardless of whether the original signal may have possessed additional qualities that cannot be represented with that model. Note that the converse is not necessarily true: if an input signal may be modeled exactly by a codec (for example, if it were already processed by the same codec), it does not necessarily mean that the output signal will be identical to the input signal. In other words, most codecs do not exhibit idempotence. Nonetheless, if the spectrum from the output of the codec is subtracted from the spectrum of the original signal, we arrive at a difference spectrum that is relatively independent of the speaker characteristics.

As we will demonstrate later, the standalone use of an existing codec as a model

against which to compute a noise spectrum is not nearly as effective as the use of the harmonic/noise decomposition. However, because the codecs are off-the-shelf components, it becomes fairly easy to design an identification system based upon their use.

4.2 Histogram

While the use of a noise spectrum can reveal some interesting signal characteristics that transcends speakers, all of the analysis therein is performed in the frequency domain. Although it is true that speech signals might be more easily represented and analyzed in the frequency domain, there is also useful information about the signal that is most easily accessible in the time domain.

Speech compression always involves some sort of quantization. At some stage in the codec, it is necessary to limit the precision of the model parameters or data codewords in order to gain bitrate economy before transmitting the compressed signal over the channel. In the case of most vocoders, the quantization is likely to take place on the representation of filter coefficients and pitch information. In the case of most waveform codecs, the quantization is likely to take place directly on the amplitudes of individual samples or on the differential codeword between samples. The effect of such quantization is readily observable in the output signal from the codec.

As an example, consider the ITU-T G.711 μ-law codec. Recall that this codec is based upon the principle of companding. In this codec, the amplitude of each sample is logarithmically compressed into an 8-bit value that is used for transmission across the channel. As a result, there are only 256 possible values that can be decoded at the receiver. Even though the receiver re-expands the values into amplitudes in a much wider 14-bit linear PCM space (corresponding to 16,384 possible values), only 256 of those values will ever be used.

It should be clear at this point that examining the amplitudes of the samples for an unknown signal may provide some insight about the codec with which the signal had been processed. A convenient way to observe the distribution of output amplitudes is to create a histogram of the sample values. As examples, several such histograms are shown in Figure 4.3. Each of these histograms has been generated from the same source signal as processed by the codec labeled in the subplot title (note that the x-axis has been scaled such that the individual histogram bars can be seen clearly). From this figure, it is obvious that there are significant differences between

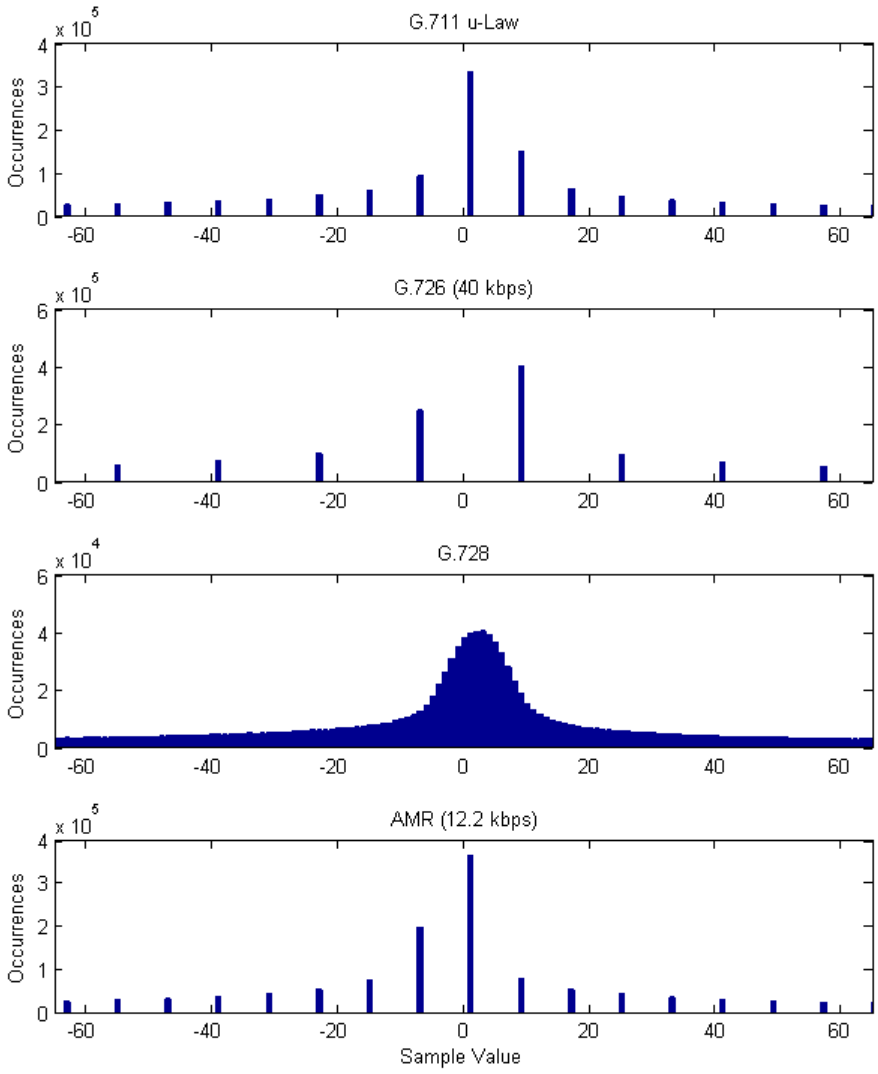

Figure 4.3: Sample value histograms for several codecs

the histograms for each codec. Furthermore, note that even the AMR codec, which is based primarily upon CELP vocoding techniques, imparts very distinct sample amplitude quantization, demonstrating that this approach is applicable to waveform codecs as well as to vocoders.

4.3 Profiling

We have now introduced two interesting ways in which a speech signal may be processed to accentuate some of the signal characteristics imparted by the source codec. Herein, the term "feature" will be used to refer to one of these items that highlights the codec artifacts in a given signal: any aggregated noise spectrum vector, or the sample amplitude histogram vector. While computing these features certainly reveals differences between each of the codecs, the task still remains of using this information to actually identify the codec present in a signal of unknown origin. The overall strategy employed for the identification procedure consists of using a set of these features to create profiles that characterize audio signals that have been processed with each of the codecs that we are interested in detecting. Thus, a "profile" is simply the set of features computed from a signal. The profiles that have been constructed from signals with known codecs will be referred to as "training" profiles. Each training profile will be created by analyzing a diverse selection of speech signals processed with the same known codec. Once the training profiles have been created, codecs from unknown test signals may be identified by following the same profiling procedure and then comparing the resulting profile, which we will call a "test" profile, to each of the training profiles. Of course, the training profile that matches the test profile most closely will determine which codec is declared to be present in the newly analyzed signal.

Clearly, the profiling procedure is the cornerstone of the identification strategy at large, and will therefore be covered in great detail. Indeed, this is also the area in which our research provides most of its novel contributions. Before continuing to describe the profiling procedure, it is important to point out the specific aspects by which this research diverges from and improves upon previous efforts. As we cover the profiling procedure in more depth, each of these contributions will be explained in greater detail:

1. Instead of performing harmonic/noise decomposition, off-the-shelf codecs are used as a basis for generating noise spectra.

2. Rather than using a single feature to characterize a signal, this research analyzes several features of the signal to generate multidimensional profiles that are used to characterize the signal.

3. Time domain information in the form of the sample amplitude histograms are

included in the features used to characterize the signal.

In this research, a profile refers to a collection of features that have been measured from the analysis of a particular input signal. Specifically, the features comprising the profile include several noise spectra as well as the sample amplitude histogram of the signal. Because we have demonstrated that each of these features takes unique shape for input signals processed with different codecs, it follows that corresponding features in profiles extracted from input signals processed with different codecs will also be distinguishable. Also, like the individual features that comprise the profile, corresponding profile features for signals that have been processed with the same codec will be similar, even if the speech content is different. Thus, a profile is simply a set of features that capture some of the intrinsic codec characteristics in the analyzed signal.

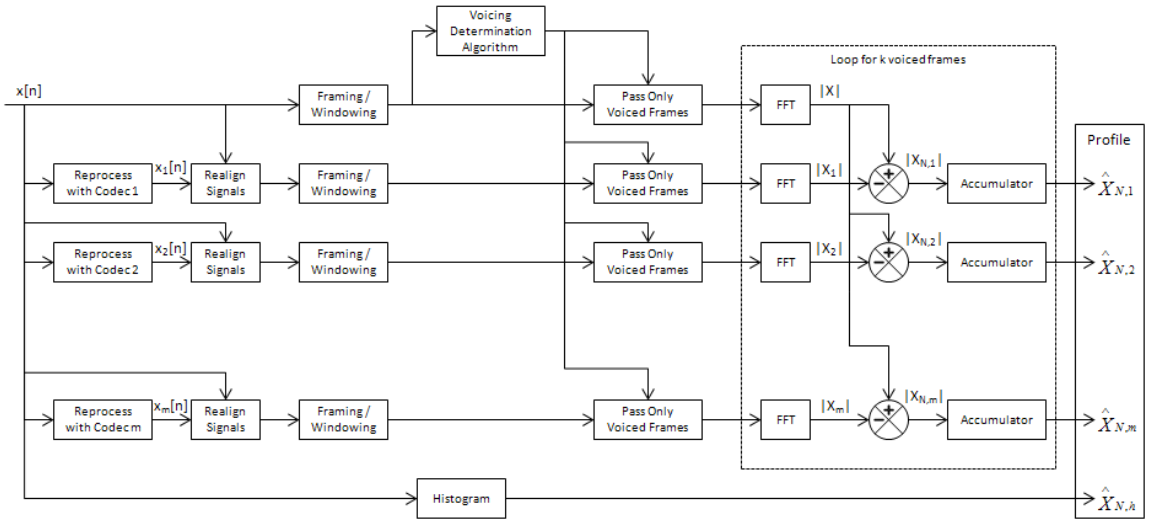

Figure 4.4: Signal profiling procedure

Figure 4.4 outlines the overall strategy used to create the profile for a given input signal, $x[n]$. The first step in the approach is to reprocess the signal with each of several codecs. The signal will be encoded and decoded by each of the m codecs to generate signals $x_1[n], \ldots, x_m[n]$. The role of these codecs is to form the comparison signals with which m noise spectra will be computed. Essentially, each codec plays the analogous role to that of the harmonic spectrum in the harmonic/noise decomposition. Recall that each of these codecs is only capable of synthesizing signals that are consistent with its underlying speech synthesis model. Each of the m codecs will have slightly different models and implementation mechanics that cause them to generate

different output signals even though they are all being sourced from the same input signal.

Some (or all) of the m codecs may exhibit a coding lag that causes the output signal to be delayed from the input signal. For example, a codec may require aggregating some of the signal history and computing internal variables based on past sample values before it can effectively begin to analyze the input signal or synthesize the output signal. It is therefore likely that the codec input and output may be misaligned. In order to fairly compare each of the m codec outputs against the original signal, all of the codec output signals should be realigned with the input signal. To determine how much each of the codec outputs needs to be shifted, a cross-correlation function can be used. The cross-correlation function essentially takes a pair of sequences, in this case x and y, and returns a sequence $R_{xy}[p]$ that represents how closely the two sequences are statistically correlated for a given lag, p. The more closely correlated the two inputs are, the higher the cross-correlation value will be. For example, for two signals that are already aligned, $R_{xy}[p]$ will be maximized for $p = 0$, and two signals that lead or lag one another by 100 samples will be maximized when $p = \pm 100$. The cross-correlation function is shown in Equation 4.1. In reality, this equation is an approximation of the cross correlation sequence for signals of finite length. Both sequences are assumed to be of equal length, N, or that the shorter sequence is otherwise zero-padded to meet the length, N, of the longer sequence.

$$R_{xy}[p] = \begin{cases} \sum_{n=0}^{N-p-1} x[n+p]y[n] & , p \geq 0 \\ R[-p] & , p < 0 \end{cases} \tag{4.1}$$

To realign the codec output signals, $x_i[n], i \in \{1, \ldots, m\}$, with the input signal, $x[n]$, the lag that maximizes the cross correlation function must be found. Because speech codecs typically introduce a latency of less than 100 ms, it is sufficient to restrict the lag search to under one thousand samples. The resulting lag value can then be used as an offset by which to shift signal x_i. All m codec output signals should be aligned with the input signal in this manner.

Next, each of the audio streams is broken down into frames. This framing is mainly for the purpose of the subsequent short-time frequency analysis that will be performed during the computation of the noise spectra. In our research, the length of the frames is selected to be 256 samples, corresponding to 32 ms of audio at an 8 kHz sampling rate. Recall that speech signals are generally regarded to be stationary

for short segments of 10 to 40 ms. Our frame length leans toward the long end of the range in order to incorporate more signal data into the analysis for each frame. This relatively long frame length comes at the cost of decreased time localization and increased risk of violating the assumption of stationarity. However, some of this concern is offset by the fact that a 50% (128 sample) frame overlap is also used, which allows for greater temporal resolution.

The next step in the procedure is to perform voicing determination on the input signal. This step involves the use of a voicing determination algorithm (VDA) to classify each frame of the input speech as either voiced, unvoiced, or silence. This step is critical because most of the remainder of our analysis will take place only on voiced frames. Recall that most of the information in speech signals is carried in voiced speech segments. In fact, in many codecs, unvoiced frames are simply replaced with filtered random noise. Thus, the majority of effort in speech coding goes into the representation of voiced speech. Because codecs perform so much processing on the voiced frames, it is the voiced segments of the codec's output signal in which most of the characteristic differences between codec artifacts may be expected to be observed. As a result, in this research, unvoiced and silence frames are discarded from all of the analysis except for the histogram generation. Because all of the output signals from the codecs are now aligned with the input signal and have frame boundaries at the same locations, the VDA analysis from the input signal can be used to prune unvoiced and silence frames from the input signal as well as all m signals from the codec processing.

Once all of the unvoiced and silence frames are filtered out, each frame can be transformed into the frequency domain via a Fast Fourier Transform (FFT) for spectral analysis. Prior to transformation, each frame is preprocessed with a Hamming window. This windowing operation will attenuate the side lobes in the spectral leakage that result from performing a short-time analysis. Furthermore, the windowed signal is zero padded to 4096 samples. This padding will allow for greater resolution in the frequency domain when the FFT is applied. It is important to note that zero padding does not increase the amount of information available in the FFT representation of the signal. In fact, as long as the Nyquist criteria was satisfied during the initial sampling of the signal, the FFT contains all of the necessary information to perfectly reconstruct the original signal, regardless of any padding that is applied. The padding does, however, increase the number of frequency bins present in the FFT

by implicitly interpolating the spectrum. This interpolation is very helpful for distinguishing closely spaced peaks and mimicking a continuous spectrum representation. Figure 4.5 demonstrates how zero padding a signal prior to transformation can result in a more detailed spectrum. The signal analyzed in the figure consists of 256 samples of superimposed sinusoids at 220 Hz and 260 Hz, with an 8 kHz sampling rate. Notice that the two peaks cannot be distinguished in the spectrum of the unpadded signal, but become much more defined as the padding is increased. Recall that the purpose of the FFT in this research will ultimately be to create a noise spectrum for the frame of voiced speech. However, since the frame length is only 256 samples, and the signal is real-valued, the positive frequency spectrum without zero-padding would be only 128 points (for real signals, the negative frequency spectrum is identical to the positive frequency spectrum, so it may be disregarded). Because the noise spectrum must be directly compared against noise spectra generated from other signals, it is essential that they be detailed enough to capture fine spectral features of the signal, hence the zero padding.

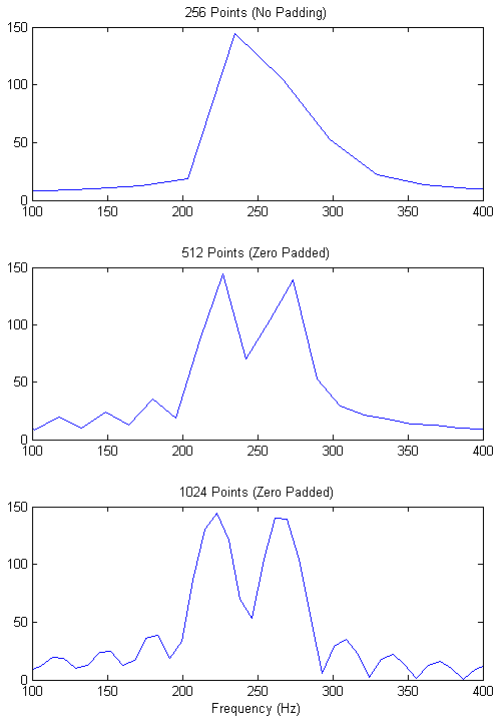

Figure 4.5: Effect of zero padding on FFT magnitude spectrum

Note that the FFT was chosen for this research due to its simplicity and ubiquity. However, several other techniques exist for collecting frequency domain information from the signal. For example, the technique introduced in [17] provides a high resolution frequency analysis tool that could have been used to examine the spectrum of the signal using much finer bin sizes to bring out greater spectral detail. This technique, however, was dismissed from use in our research because the provided implementation proved to be too computationally slow to feasibly analyze all of the signals used for training and testing. Furthermore, it was demonstrated in [21] that the FFT was sufficient to generate distinguishable noise spectra. Another alternative frequency analysis technique that has become popular in modern signal processing is the wavelet transform. This transform results in a representation that is localized in both time and frequency, which means that the spectra have a much greater temporal resolution than the coarsely-framed short-time Fourier Transform (STFT) [9]. However, this temporal resolution is of little benefit, as speech signals tend to be mostly stationary for our selected frame size, making the FFT approach again sufficient for our analysis.

From the FFT, only the positive magnitude spectrum of each frame is of interest. Because the analyzed signal is real, the negative frequency spectrum is simply a reflection of the positive frequency spectrum, and is of no additional value. As indicated in Figure 4.4, the FFT is performed on voiced frames both from the original signal, and from each of the m codec outputs. The magnitude spectra, $|X_1|, \ldots, |X_m|$, from each of the m codecs is subtracted from the magnitude spectrum of the same voiced frame from the original signal, $|X|$. Thus, each of these spectra is generated in the same manner as the harmonic/noise decomposition from Figure 4.1, but with the spectrum from each of the m codecs taking the place of the harmonic spectrum. This yields the m noise spectra, $|X_{N,1}|, \ldots, |X_{N,m}|$. Each spectrum, however is generated from only a single frame of voiced speech, and may not be representative of the overall speech signal. Thus, the noise spectrum is additively accumulated over numerous voiced frames to create an average noise spectrum that is more representative of a longer segment of the speech signal. For research purposes, the length of the signal over which the noise spectra are computed is artificially limited to k voiced frames in order to observe the effect of different analysis lengths on the accuracy of the identification. The resulting m aggregate noise spectra, $\hat{X}_{N,1}, \ldots, \hat{X}_{N,m}$, are recorded and used as features in the profile.

In conjunction with the other signal processing operations for analyzing the signal,

the profiling strategy also records all sample amplitude values of the input signal until the k voiced frames have been analyzed. This data is used to construct a histogram. Note that any quantization of amplitude values imparted by the codec will most likely be independent of the voicing state of the speech, so it is appropriate to bypass the voiced frame filtering to which the other signal processing operations are subject. In this research, when constructing the amplitude histogram, the range of amplitude values is artificially restricted and centered about zero. The purpose of this is twofold. First, the amplitude data is concentrated most heavily around low amplitudes, as observed in Figure 4.3. Any histogram discrepancies will therefore be most pronounced in this region. Secondly, maintaining a fixed number of histogram bins makes it much easier to perform comparisons between histograms across different profiles, because every profile's histogram will contain the same number of data points. Out of the 16-bit space available for each sample, we record histogram samples only for amplitudes from -4000 to 4000 in our approach. In addition, restricting the range of histogram bins also potentially saves a lot of memory (for example, maintaining a bin for every possible amplitude in the 16-bit PCM space would require 512 kB per histogram). In the profile, the histogram has been denoted by $X_{N,h}$, where $h = m+1$.

With the addition of the histogram, the construction of a signal profile is complete. In total, then, the profile consists of m noise spectra (each of which contains the magnitudes of the 2048 bins of the positive frequency spectra as accumulated over k voiced frames), and the sample amplitude histogram (consisting of the 8001 bins centered around zero). This set of $m + 1$ features is sufficient to highlight characteristics of the input signal that are largely dependent upon the original source codec.

4.4 Identification

With the profiling procedure established, it is then necessary to formulate a meaningful way to utilize the signal profiles to make an identification decision. The overall approach here consists of first creating a set of training profiles from speech signals that have been processed with known codecs. Then, for any new signals of unknown origin, the profiling procedure is applied again, and the resulting profile is compared against each of the training profiles. The codec whose training profile yields the best match will be declared to be the codec present in the new signal.

The challenge here, of course, is to determine a quantitative manner by which to compare a set of signal profiles. For this research, the profile of the unknown

signal is compared to each of the training profiles of the codecs that are desired to be detected, one at a time. Each profile comparison, in turn, consists of several comparisons between corresponding features in each the two profiles. The comparison between corresponding profile features is performed by means of a normalized cross correlation function, as shown in Equation 4.2. Similarly to the generalized cross-correlation function shown in Equation 4.1, the normalized cross correlation returns a value that indicates how closely two sequences match up with one another. The major differences are that this version computes the cross correlation fixed at zero lag, and normalizes the result such that the output ranges from -1 to 1, where 1 indicates an exact match between the two sequences (and -1 indicates that the two sequences are additive inverses of one another).

$$\rho_{i,j} = \frac{\left\langle (\hat{X}_{N,j})_i, \hat{X}_{N,j} \right\rangle}{\|(\hat{X}_{N,j})_i\| \|\hat{X}_{N,j}\|} \tag{4.2}$$

This normalized rendition of the cross correlation function is useful for a couple of reasons. Recall that each feature consists of a fixed number of data points, and those points have all been measured in such a way that they are already lined up with one another. For example, the 2048 points in every noise spectrum in every profile always correspond to the same set of frequency bins ranging from 0 Hz to 4 kHz, and the 8001 points in each of the histograms always correspond to the same set of sample amplitudes ranging from -4000 to 4000. Thus, when assessing the similarity of a feature between two profiles, it is correct to assume zero lag between the sequences. The normalization aspect means that only the shape of a feature, not the scaling, is taken into account when comparing between profiles. This is useful because the scaling of the feature is determined largely by the length of the signal being analyzed. The relative independence from the analysis length provided by the normalization makes it is possible to compare profiles from signals of differing lengths. The normalized cross correlation function is consequently a very convenient means for comparing the corresponding features between profiles.

Though the normalized cross correlation function is utilized in our research to compare each of the features, including the histograms, it is worthwhile to note that there are several alternative techniques developed specifically for comparing histograms and probability distributions. Such techniques are usually oriented towards image processing applications and include metrics such as earth mover's distance [15] and diffusion distance [14]. However, these techniques were not used in our research because they

involve unnecessary dimensionality and complexity. Our results will demonstrate that the use of the normalized cross correlation for comparing the histograms proves to be an effective technique for this particular application.

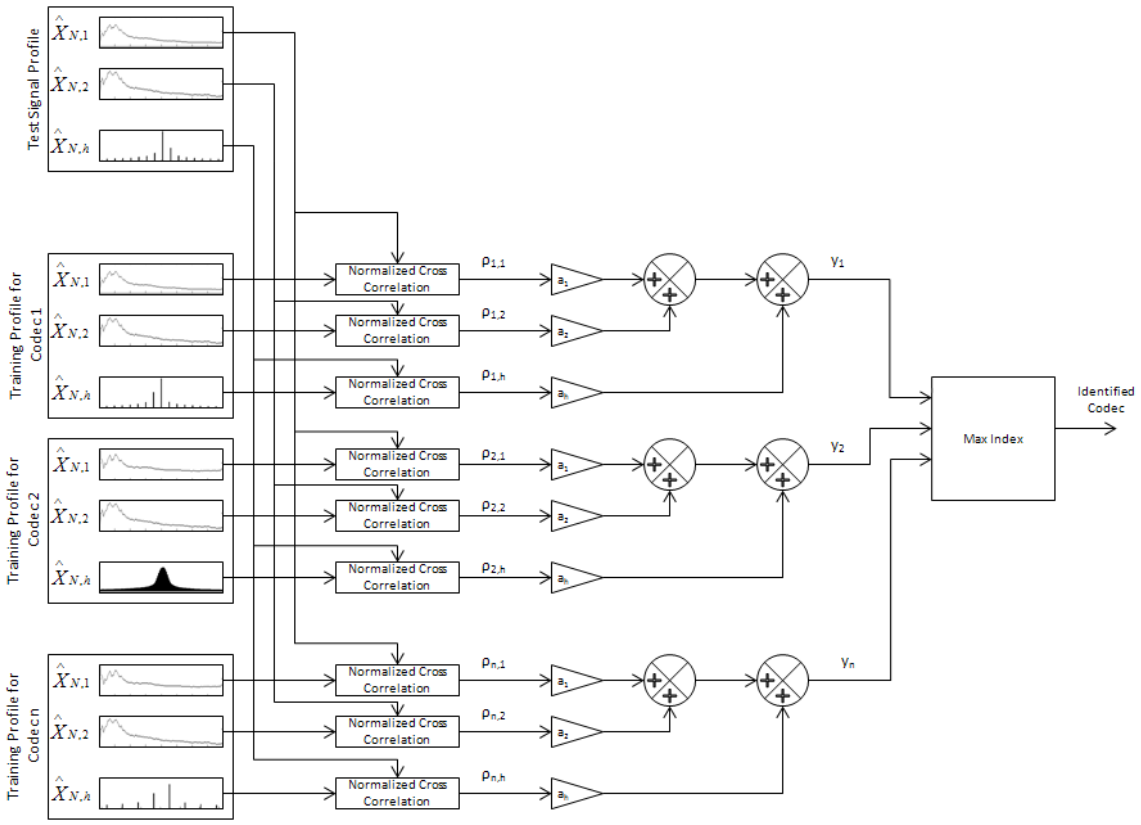

Figure 4.6: Identification procedure for a new signal

Although we have demonstrated how we compare the corresponding features between profiles, this still does not account for how to compare the profiles themselves, which consist of $m + 1$ features (the m noise spectra plus the histogram). The approach used in this research is to simply take a weighted sum of the normalized cross correlation values of all of the features between two profiles, as illustrated in Figure 4.6. For making an identification decision, a test profile from a new signal must be compared against the profiles of each of the n codecs amongst which the signal is to be identified. This is expressed in Equation 4.3, where y_i represent the overall fit between the test signal profile and the i-th codec's training profile. Each $\rho_{i,j}$ is the value from the normalized cross correlation of the j-th feature ($j \in \{1, \ldots, m, h\}$, where features $1, \ldots, m$ are the m noise spectra, and feature $h = m + 1$ is the histogram)

between the test signal's profile and the i-th codec's profile.

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} \rho_{1,1} & \rho_{1,2} & \cdots & \rho_{1,m} & \rho_{1,h} \\ \rho_{2,1} & \rho_{2,2} & \cdots & \rho_{2,m} & \rho_{2,h} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \rho_{n,1} & \rho_{n,2} & \cdots & \rho_{n,m} & \rho_{n,h} \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_m \\ a_h \end{bmatrix} \tag{4.3}$$

The role of the values a_j will be examined in greater depth later in Section 5.3. For now, it is sufficient to note that these values specify the weight that each feature should carry in determining the overall fit between two profiles. In general, some features are more effective than others in distinguishing between codecs, so these features will carry more weight. Nonetheless, it is important that several features are used, because many of the features provide strengths where other features are weaker.

Finally, it should be evident that the training profile that matches the test signal's profile the closest using the comparison methodology presented will yield the greatest weighted sum of feature comparisons. That is, the final decision for the codec present in the input signal is given by the profile that maximizes y_i, as expressed in Equation 4.4.

$$\text{Codec Index} = \arg\max_i y_i : y_i \in \{y_1, y_2, \ldots, y_n\} \tag{4.4}$$

Chapter 5

Testing

The previous chapter introduced the general codec identification strategy devised in this research effort. Where possible, most of the parameter values, including the number and types of codecs, feature weights, speech samples, and other factors were generalized in order to decouple the overall approach from the implementation details. This chapter attempts to fill those voids and explain the details of how the strategy was applied during the collection of test results for our research.

5.1 Generating Training Profiles

Because this research revolves around the processing of speech signals, it is essential to have a large database of speech signals with which to experiment. The Texas Instruments / MIT (TIMIT) speech corpus is used in our research for that purpose. The TIMIT corpus contains thousands of clean speech files spanning hundreds of speakers, and is used primarily for research purposes in fields such as speech and speaker recognition. The speech files contain English sentences from male and female speakers from each of 8 different dialect regions throughout the United States [29]. The distribution of these speakers is shown in Table 5.1.

Dialect Region	Male Speakers	Female Speakers	Total
New England	31 (63%)	18 (27%)	49 (8%)
Northern	71 (70%)	31 (30%)	102 (16%)
North Midland	79 (67%)	23 (23%)	102 (16%)
South Midland	69 (69%)	31 (31%)	100 (16%)
Southern	62 (63%)	36 (37%)	98 (16%)
New York City	30 (65%)	16 (35%)	46 (7%)
Western	74 (74%)	26 (26%)	100 (16%)
Army Brat (moved around)	22 (67%)	11 (33%)	33 (5%)
Total	438 (70%)	192 (30%)	630 (100%)

Table 5.1: Distribution of speakers in TIMIT speech corpus

At a high level, the results of this research are gathered in two major steps. The first step is to create a set of training profiles from each of the codecs of interest, and the second is to generate numerous test signals from various codecs and compare their profiles against the training profiles. In order to perform these two steps, a different set of speech files must be selected for the construction of each set of profiles. There should be no overlap between the speakers or sentences used in the training stage and those used in the testing stage. Clearly, if there are shared speakers or sentences between both sets, then the content of the test set will be unfairly representative of the content of the training set, and the results will be uncharacteristically favorable. Fortunately, the TIMIT speech corpus has already been divided into training and testing partitions in order to account for such use cases. Thus, in our research, we make use of the existing partitions, and use only the speech files from the training partition to construct our training profiles, and only the speech files from the testing partition to construct the test profiles.

Each file in the TIMIT corpus contains the audio for a single sentence from a single speaker. The corpus consists of three types of sentences, as shown in Table 5.2. The "SA" sentences are sentences that have been specially crafted to bring out the distinguishing qualities between the different dialect regions. However, because the "SA" sentences are shared across all speakers, they have been excluded from our training so as to remove bias toward those particular sentences. The remainder of the sentences are denoted as either "SX" or "SI". The "SX" sentences were designed at MIT to be phonetically-compact. That is, the speech exercises very few differing phonemes. In contrast, the "SI" sentences, selected at TI, are phonetically diverse and exercise a wide assortment of phonemes [29]. Both of these types of sentences are included in the training stage.

Sentence Type	# Sentences	# Speakers	Total	# Sentences/Speaker
Dialect (SA)	2	630	1260	2
Compact (SX)	450	7	3150	5
Diverse (SI)	1890	1	1890	3
Total	438	192	630	10

Table 5.2: Distribution of sentence types in TIMIT speech corpus

Before any profiles can be created, it is first necessary to generate the set of signals that will be used as inputs to the profiling procedure. In this research, we have elected to use 100 sentences for generating every training profile. This number was selected

mostly due to time constraints, as our implementation of the profiling procedure is rather naive (performance is not a goal of this research) and requires considerable time to construct the training profiles. It is, however, important to realize that the number of sentences also directly corresponds to the number of voiced frames processed during the profiling procedure. Because the noise spectra from every voiced frame are additively accumulated, the overall shape of the resulting overall noise spectrum is essentially averaged over all those frames. Thus, there is a tradeoff regarding the amount of speech used to generate a profile: using too much will oversmooth fine-grained details in the spectrum, while using too little will result in a noisy spectrum that may not be representative of other signals from the same codec.

The 100 speech files used for training were chosen via a uniform random selection from among all of the "SX" and "SI" sentences in the training partition of the TIMIT corpus. For consistency, all 100 files are processed by each of several different codecs at different settings. By using the same set of inputs files to profile each codec and setting, it ensures that no one profile is misrepresented due to differences in the random selection. For example, if the files were randomly chosen for each codec independently, it might be possible for one codec to be profiled using all male speakers as inputs while another is profiled from all female speaker inputs. It is worth noting that, in a practical application, it would be advisable to select training speech signals to be consistent with the expected demographics of the intended deployment scenario. This would ensure that the training profiles will be more representative of the profiles from the actual speech signals that will be encountered, and ultimately lead to better accuracy.

To construct a training profile for one of these codecs, each of the 100 randomly selected speech signals is first downsampled to a narrowband sampling rate (the TIMIT speech files are natively sampled at 16 kHz), and then passed through the encoder and decoder for the selected codec. These processed signals are then profiled using the strategy discussed previously. The noise spectra and histogram data are allowed to accumulate over all 100 speech files. This is functionally equivalent to concatening all of the speech files into one long signal, and then performing the profiling process on that signal. Once all of the files have been analyzed, the resulting features (the noise spectra and the histogram) are saved as the training profile for that codec. This process is repeated for all codecs and settings desired to match against. The complete list of codecs and settings profiled in this research is listed in Table 5.3.

Codec	Setting
G.711	μ-Law
G.726	40 kbit/s 32 kbit/s 24 kbit/s 16 kbit/s
G.728	Default
G.729	11.8 kbit/s 8 kbit/s 6.4 kbit/s
iLBC	15.2 kbit/s 13.33 kbit/s
AMR	12.2 kbit/s 10.2 kbit/s 7.95 kbit/s 7.4 kbit/s 6.7 kbit/s 5.9 kbit/s 5.15 kbit/s 4.75 kbit/s
SILK	VBR, Default quality

Table 5.3: Source codec selection for training profiles

5.2 Testing

Once the 20 training profiles (one for each of the codecs in Table 5.3 have been created, the testing is performed. The testing procedure consists first of constructing a set of test signals. Whereas the speech files used to construct the training files were selected from the training partition of the TIMIT corpus, the speech files for testing are selected from the testing partition of the corpus. As the testing partition is simply a subset of the whole database, the distribution of speakers in the test partition is very similar to that of the overall set, as outlined in Table 5.4 [18]. Unlike the training profiles, which have been constructed from speech files from numerous speakers, each of the testing signals will be generated from files from a single speaker. This is because speech signals in telecommunications applications will typically have only one speaker. It is probably unrealistic to expect most speech signals to consist of multiple speakers, and would lead to unfairly favorable results since the test signal would contain greater speaker diversity and be more representative of the training signals.

For our research, all 168 of the speakers in the test partition of the database are used in testing. To create the test signals, each speaker's speech files are concatenated

Dialect Region	Male Speakers	Female Speakers	Total
New England	7 (64%)	4 (36%)	11 (7%)
Northern	18 (69%)	8 (31%)	26 (15%)
North Midland	23 (88%)	3 (12%)	26 (15%)
South Midland	16 (50%)	16 (50%)	32 (19%)
Southern	17 (61%)	11 (39%)	28 (17%)
New York City	8 (72%)	3 (27%)	11 (7%)
Western	15 (65%)	8 (35%)	23 (14%)
Army Brat (moved around)	8 (73%)	3 (27%)	11 (7%)
Total	112 (67%)	56 (33%)	168 (100%)

Table 5.4: Distribution of speakers in TIMIT test partition

together (in a random order) to form a long speech signal for that speaker. Like the construction of the training profiles, these signals are downsampled and processed by each of the 20 codecs to form 168 test inputs for each codec. These signals are each profiled as usual, except that the profile is captured at several intervals of the signal, as shown in Figure 5.1. This allows for examining the effect of the analysis length (how much of the signal is profiled) on the overall identification accuracy.

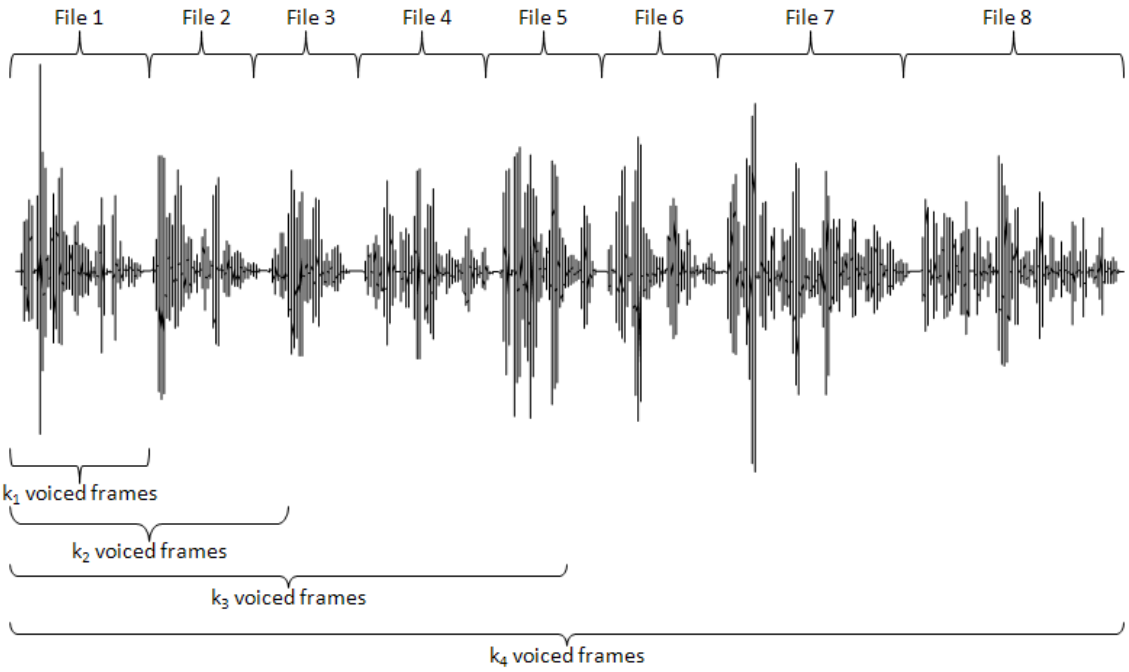

Figure 5.1: Concatenation of speech files and varying analysis lengths for testing

With all of the profiles constructed, the identification decision procedure is applied.

Each test signal's profile is compared to every training profile to determine which codec is the best match, as per Equations 4.3 and 4.4. The identified codec is recorded for each of the 168 input signals from each of the 20 codec settings. The results are stored in a confusion matrix, as will be presented in Section 6.

5.3 Feature Weights

One very important aspect that has not yet been discussed is the selection of the m codecs used in the profiling procedure (refer back to Figure 4.4), and the related notion of the weights of each of the $m + 1$ features from Equation 4.3. These aspects can have a profound effect on the overall accuracy, and therefore require some additional explanation.

As mentioned previously, the role of each codec is to remove the aspects of the input signal that adhere to the underlying model of that codec. When subtracted from the original signal (forming the noise spectrum), the spectral aspects that are mostly independent of the speaker and speech content are left behind. Since the resulting spectrum is no longer dominated by the strong harmonic components, it is able to better unmask more subtle signal characteristics such as those that are imparted by the source codec.

Because the codecs have slightly different models and quality settings, each one will have a unique effect on the the shape of the noise spectrum that it generates. Some of these noise spectra may be very distinguishable between signals from certain source codecs while other noise spectra may be very distinguishable between signals from a different set of codecs. It may also be possible that some noise spectra are good at differentiating all codecs while some perform very poorly overall.

The plot in Figure 5.2 was developed to provide a quantitative view of these concerns. This plot is rather difficult to understand at first, and will require some explanation. It is important to note that this is not a confusion matrix, and that it should not be expected to reveal any sort of pattern along the diagonal (realize, in fact, that there are more columns than rows, so there is no true diagonal anyway). This plot is generated by performing the profiling procedure for the test signals using all 20 of the source codecs as the m codecs for generating noise spectra. The identification procedure, however, has been modified to evaluate only a single feature, where each of the columns in Figure 5.2 correspond to the feature that was isolated for performing the identification. The rows indicate the codec used on each of the test signals applied

as inputs. The shading of each cell represents the percentage of the test signals from the given source codec that were correctly identified when only the feature in the given column was used for identification. As an example, consider the column labeled "G.729 (11.8 kbit/s)". The data in this column tells us that, when G.729 at 11.8 kbit/s is the only codec used for generating a noise spectrum in the profiling procedure (and no histogram is used, either), around 70% of input signals containing G.726 at 24 kbit/s will be correctly identified, around 60% of the input signals containing G.729 at 8 kbit/s will be correctly identified, less than 20% of signals containing AMR at 6.7 kbit/s will be correctly identified, etc.

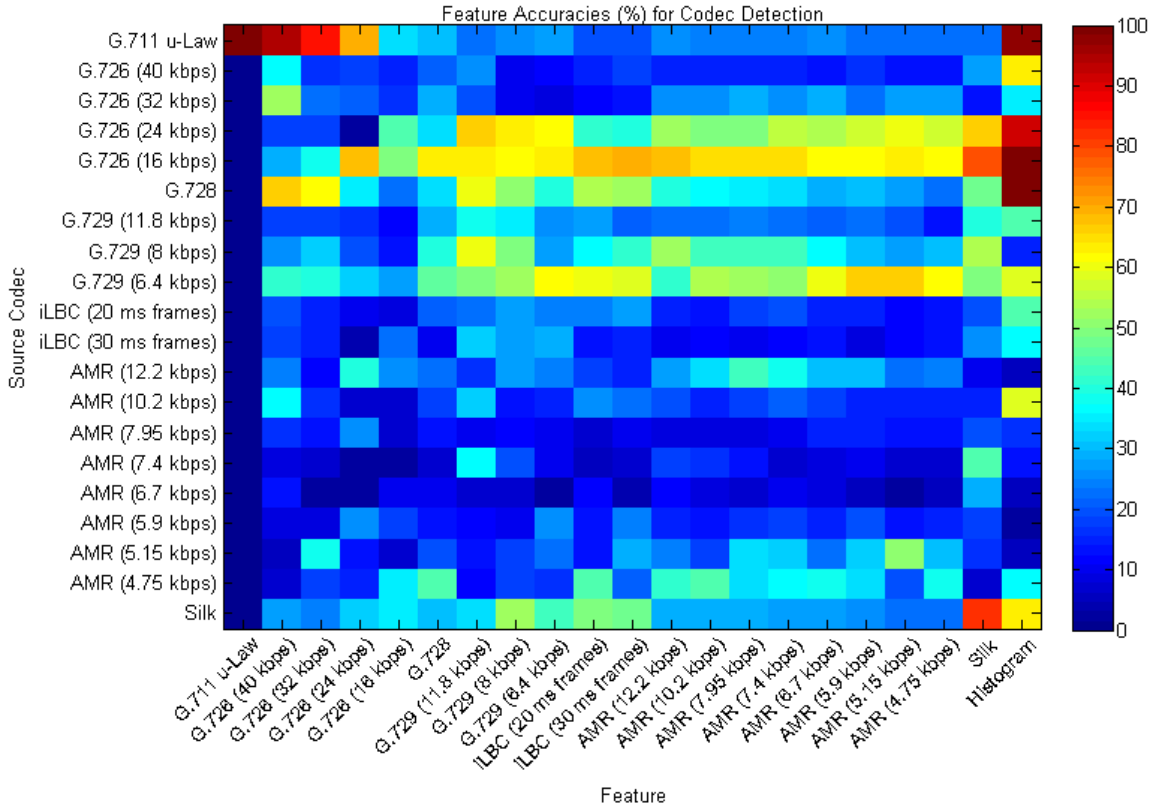

Figure 5.2: Effectiveness of each feature in identifying each codec

This figure is very useful, as it reveals a plethora of information about how effective each codec used for noise spectrum generation is at detecting each codec in the input signal (the same information is also presented for the histogram). The ideal scenario would be for the entire plot to be red, as this would indicate that the use of any one profile feature could accurately distinguish between any of the source codecs. Instead,

however, we see that certain profile features are strong at detecting some codecs and weak at detecting others. In particular, there are a few very interesting observations that can be made from this plot. Consider the use of G.711 μ-Law as a profile feature. The plot demonstrates that it can identify the presence of G.711 μ-Law in the input signal 100% of the time, but is completely ineffective at identifying any other codec. The reason for this anomaly is that the G.711 codecs are idempotent. As a result, the noise spectrum will be exactly flat when the input signal comes from the same codec as the profiling codec. This flat noise spectrum is very distinguishable from the noise spectra generated by other codecs, making it very easy to identify accurately. However, the G.711 codecs do not provide very much signal compression or degradation, so the differences in noise spectra when any other codec is present at the input will be very subtle and difficult to accurately distinguish. In contrast to G.711, notice that the other codecs (which do not exhibit idempotence) do not have strong responses when the codec in the source signal is profiled using the same codec as the only feature.

Unfortunately, the plot also reveals that no one feature is particularly accurate at identifying every type of codec. It therefore becomes necessary to carefully select a set of codecs which provide reasonable overall accuracy spanning the whole set of source codecs. An easy way to visualize the average effectiveness of each feature is to flatten the plot by taking the mean of each column. This results in a more readily comprehensible plot, as shown in Figure 5.3. This plot shows the overall accuracy of each profile feature assuming a uniform distribution of input signals from all 20 codecs.

This new plot offers a few new interesting conclusions. First, of all, the histogram turns out to be much more effective than any other feature. G.711 μ-Law, while being very precise for detecting μ-Law signals, is of almost no use when considering a wide range of possible input signals. The remainder of the codecs tend to be relatively close to one another in overall accuracy. Another interesting aspect to note is that no feature provides greater than 50% accuracy when used standalone, with most features only yielding around 20% accuracy. This is clearly far less effective than the standalone noise spectrum feature from the harmonic/noise decomposition in [21]. However, as will be demonstrated in Section 6, the combination of several of these features in concert will lead to much improved results.

The m codecs used for the profiling procedure in this research are selected based upon the observations from Figure 5.3. It is obvious that using only a single feature

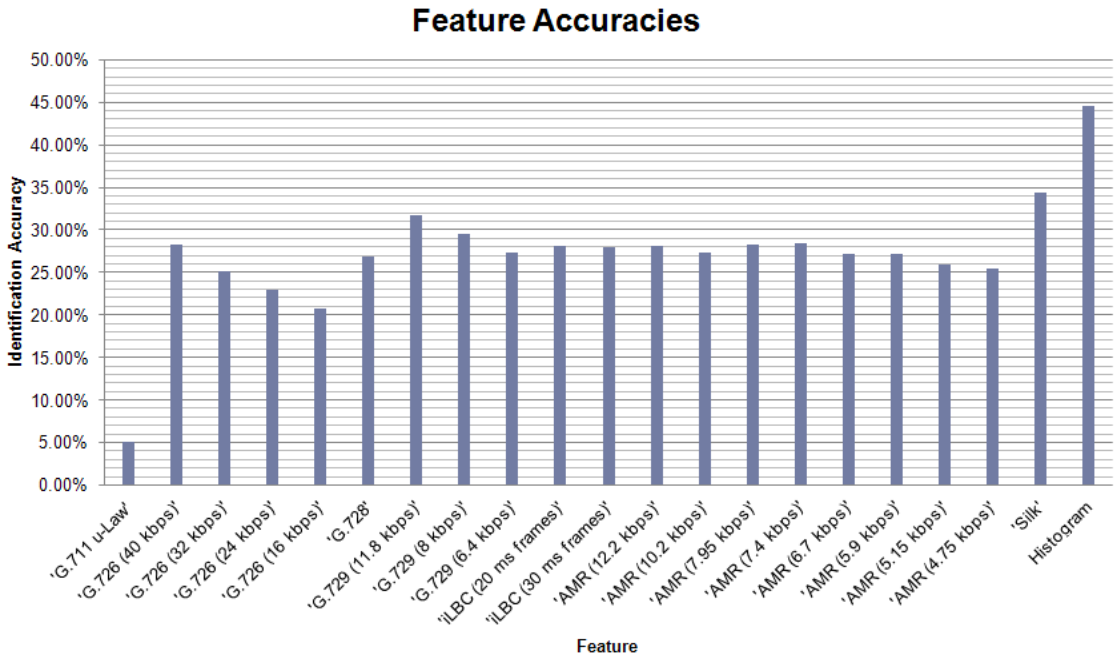

Figure 5.3: Overall identification accuracy of each feature

is insufficient to attain favorable accuracies. Nonetheless, Figure 5.2 hints that using every feature available does not necessarily add significant value while adding substantial processing overhead (there are several features that are very inaccurate for detecting any codec). To remain in the middle of this tradeoff, we have elected to use 7 of the highest ranking features for the profiling procedure in this research. These selected features are outlined in Table 5.5.

Profile Features	
G.726 Noise Spectrum	40 kbit/s
G.728 Noise Spectrum	Default
G.729 Noise Spectrum	11.8 kbit/s
iLBC Noise Spectrum	15.2 kbit/s
AMR Noise Spectrum	7.4 kbit/s
SILK Noise Spectrum	VBR, Default quality
Histogram of Sample Amplitudes	

Table 5.5: Features selected for use in signal profiles

Lastly, the coefficients from Equation 4.3 must be determined. These weights determine how much influence each feature should have when calculating how closely

two profiles match. The values of the coefficients are based very loosely around the overall effectiveness of the features as shown in Figure 5.3. Note that the time domain histogram is by far the most accurate feature within our selection. It is also the only time domain feature present in the profile. The remainder of the features are noise spectra that are based only upon spectral characteristics from the frequency domain. With these points in mind, it was reasoned that the time domain feature should have half of the weight, while the noise spectra will be uniformly weighted among the other half, as expressed in Equation 5.1.

$$\begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_m \\ a_h \end{bmatrix} = \begin{bmatrix} 1/2m \\ 1/2m \\ \vdots \\ 1/2m \\ 1/2 \end{bmatrix} \tag{5.1}$$

Chapter 6

Results and Analysis

The profiling and identification methodologies developed in this research have now been explained in full. All relevant details to the testing procedure have been covered in order to give a full understanding of how the results of this research were gathered. This chapter now presents the actual results collected from several testing scenarios.

6.1 Identification Accuracy

The major results of this work concentrate on the accuracy achieved in identifying the codecs. Table 6.1 shows the results of how all 168 input signals from each codec setting were identified when analyzing the signals for $k = 160$ voiced frames. The labels on the top and left axes have been represented as numbers, rather than codec names, in order to present the table compactly. For both axes, labels 1 through 20 refer to the codecs/settings in Table 5.3 from top to bottom, respectively. For example, the cell in row 11 column 8 reveals that 12 out of the 168 input samples that were processed with iLBC at 13.33 kbit/s were incorrectly identified as G.729 at 8 kbit/s.

Unlike Figure 5.2, Table 6.1 is actually a confusion matrix. Thus, favorable results should manifest themselves as high values along the diagonal. For some of the codecs, a 100% identification rate was observed, as indicated by all 168 input signals appearing on the diagonal. Unfortunately, many codecs exhibit significant confusion. Observe, for instance, that only 3 out of the 168 signals from the AMR codec at 6.7 kbit/s were correctly identified as such (an accuracy of less than 1.8%).

However, notice that there are several regions of the table where there appears to be a rectangular bounding box about the diagonal within which most of the results are clustered. A closer look at Table 5.3 reveals that these regions are actually formed where there are different settings for the same codec. For example, the region from

Source Codec \ Classified As	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	**168**	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
2	-	**166**	2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
3	-	72	**96**	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
4	-	99	13	**56**	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
5	-	-	-	-	**168**	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
6	-	-	-	-	-	**168**	-	-	-	-	-	-	-	-	-	-	-	-	-	-
7	-	-	-	-	-	-	**114**	46	2	-	1	-	-	-	-	-	-	-	-	-
8	-	-	-	-	-	-	19	**120**	29	-	-	-	-	-	-	-	-	-	-	-
9	-	-	-	-	-	-	14	69	**85**	-	-	-	-	-	-	-	-	-	-	-
10	-	-	-	-	-	-	8	11	1	**45**	103	-	-	-	-	-	-	-	-	-
11	-	-	-	-	-	-	10	12	2	28	**116**	-	-	-	-	-	-	-	-	-
12	-	-	-	-	-	-	-	-	-	-	-	**48**	57	34	11	-	3	2	13	-
13	-	-	-	-	-	-	-	-	-	-	-	16	**86**	30	17	-	2	4	13	-
14	-	-	-	-	-	-	-	-	-	-	-	10	41	**44**	15	2	7	10	39	-
15	-	-	-	-	-	-	-	-	-	-	-	6	39	25	**46**	1	6	9	36	-
16	-	-	-	-	-	-	-	-	-	-	-	4	40	29	17	**3**	16	17	42	-
17	-	-	-	-	-	-	-	-	-	-	-	1	35	31	16	5	**15**	14	51	-
18	-	-	-	-	-	-	-	-	-	-	-	-	9	29	6	4	7	**34**	79	-
19	-	-	-	-	-	-	-	-	-	-	-	-	5	31	-	3	9	32	**88**	-
20	-	-	-	-	-	-	5	26	5	-	-	-	-	-	-	-	-	-	-	**132**

Table 6.1: Raw test results for $k = 160$

rows 12 through 19 consists of source signals that were all processed with the AMR codec (but at different bitrates). Notice that, even though these figures do not lie on the diagonal, they do fall within the region between columns 12 and 19. This means that all of the input samples processed with AMR were correctly identified as AMR, even though the bitrate may be incorrect.

For the applications discussed in this research, the correct identification of the bit rate is not nearly as important as the correct identification of the codec itself. In fact, for some codecs such as AMR (recall that this acronym stands for "Adaptive Multi-Rate"), the bit rate may frequently change based upon factors such as channel conditions. Thus, it is oftentimes more useful to be able to simply determine the codec in use, rather than extracting the specific settings that have been employed, as shown in Table 6.2. This table contains the same data as Table 6.1, but condenses the fine-grained classification of bitrate settings into fewer, broader groupings. Also, because the number of columns has been reduced significantly, the labels have been made more verbose and the cell values converted to percentages for better readability. Cells that represent correct identifications have been highlighted in bold face text.

Here, the power of the identification strategy is very clear. When grouped more broadly by codec, the results are extremely favorable. Most of the source codecs were correctly identified for 100% of the input signals applied. Even those codecs that were more elusive were still correctly identified for the vast majority of input signals.

			Classified As						
			G.711	G.726	G.728	G.729	iLBC	AMR	Silk
Source Codec	G.711	μ-law	**100.00%**	-	-	-	-	-	-
	G.726	40 kbit/s	-	**100.00%**	-	-	-	-	-
		32 kbit/s	-	**100.00%**	-	-	-	-	-
		24 kbit/s	-	**100.00%**	-	-	-	-	-
		16 kbit/s	-	**100.00%**	-	-	-	-	-
	G.728	16 kbit/s	-	-	**100.00%**	-	-	-	-
	G.729	11.8 kbit/s	-	-	-	**96.43%**	0.60%	-	2.98%
		8 kbit/s	-	-	-	**100.00%**	-	-	-
		6.4 kbit/s	-	-	-	**100.00%**	-	-	-
	iLBC	15.2 kbit/s	-	-	-	11.90%	**88.10%**	-	-
		13.33 kbit/s	-	-	-	14.29%	**85.71%**	-	-
	AMR	12.2 kbit/s	-	-	-	-	-	**100.00%**	-
		10.2 kbit/s	-	-	-	-	-	**100.00%**	-
		7.95 kbit/s	-	-	-	-	-	**100.00%**	-
		7.4 kbit/s	-	-	-	-	-	**100.00%**	-
		6.7 kbit/s	-	-	-	-	-	**100.00%**	-
		5.9 kbit/s	-	-	-	-	-	**100.00%**	-
		5.15 kbit/s	-	-	-	-	-	**100.00%**	-
		4.75 kbit/s	-	-	-	-	-	**100.00%**	-
	Silk	VBR	-	-	-	21.43%	-	-	**78.57%**

Table 6.2: Identification accuracy results for $k = 160$

The average accuracy of these results across all input codecs is 94.9%. Note that this average is computed with uniform weight given to each codec group (rather than to each individual codec setting) so as to avoid any bias caused by differing numbers of settings among codecs.

Overall, the accuracy is very high. However, it is evident that there is still some confusion between a few of the codecs. Specifically, G.729, iLBC, and SILK seem to be incorrectly identified as one another frequently. The most straightforward explanation for this behavior is that they are all CELP based codecs and therefore have very similar underlying speech synthesis models. This means that they would be likely to have the most similar outputs from among the codec selection. Because these three are also the most recently developed codecs in the set (if you consider that the G.729 Annex C+ used in this research was released several years after the initial specification), it is likely that the modern enhancements are capable of reproducing speech very faithfully to the original signal. Furthermore, all of these codecs have sample values that continuously span the output space (similarly the the third subfigure in Figure 4.3), making their histograms less distinguishable from one another. This is an important consideration because the histogram feature carries half of the total weight in the comparison between signal profiles.

6.2 Effect of Analysis Length

The results shown in the previous section were gathered using 160 voiced frames of audio from the input signal. We opted to show the full results for this nominal value because it roughly corresponds to a single sentence worth of speech. However, it may also be of interest to examine how the choice of the number of voiced frames affects the overall accuracy of the identification. The number of voiced frames has a direct and obvious effect on the overall analysis length. For most situations in which codec identification would be employed, it would be desirable to attain not only high accuracies, but to do so with minimal audio input.

In this research, the length of the analyzed signal is defined by the number of voiced frames. Recall that speech content is dynamic, and that different speech signals may contain different proportions of voiced and unvoiced segments, as well as silence. Using the voiced frame count as the metric for the analysis length gives a fair basis for evaluating the identification strategy in a manner that is mostly independent of the proportions of different voicing states in the signal. By using the number of voiced frames, rather than time, it ensures that the same amount of useful data is extracted from different input signals. For example, it would be unfair to compare the performance of the identification strategy on a 3 second signal that is mostly silence versus a 3 second signal filled with intelligible speech. The signal containing mostly silence would obviously have much less meaningful data to analyze, and would therefore result in a lower identification accuracy than a signal of the same length that is rich in voiced speech content.

It is not intuitive to think of signal lengths in terms of the number of voiced frames. Unfortunately, because different signals have different proportions of voiced speech content, there is no direct conversion from voiced frames to seconds. Furthermore, there is no single well-defined threshold or strategy for distinguishing between voiced and unvoiced speech, so different voicing determination algorithms (VDAs) could differ significantly in their voicing classifications for the same input signal. Despite these aspects, the proportion of voiced frames is fairly consistent over time for continuous speech (as with the sentences from the TIMIT corpus), so it is possible to roughly correlate the actual signal length (in time), to the number of voiced frames. In Figure 6.1, the average length signal length over all 168 test signals is plotted in relation to the number of voiced frames. From this plot, it is clear that there is a strong linear relationship of just under one second of audio per 40 voiced frames. Indirectly, we

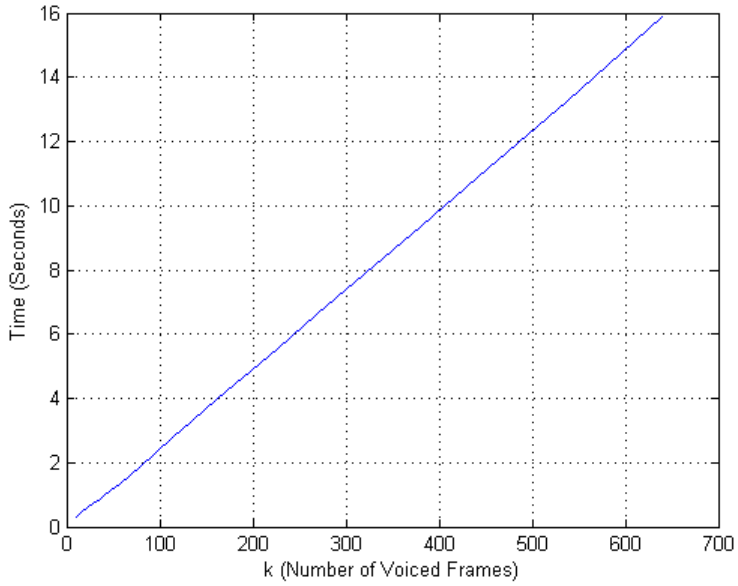

Figure 6.1: Relationship between average test signal length and number of voiced frames

can also calculate that our voicing determination algorithm classifies approximately 65% of all frames as voiced.

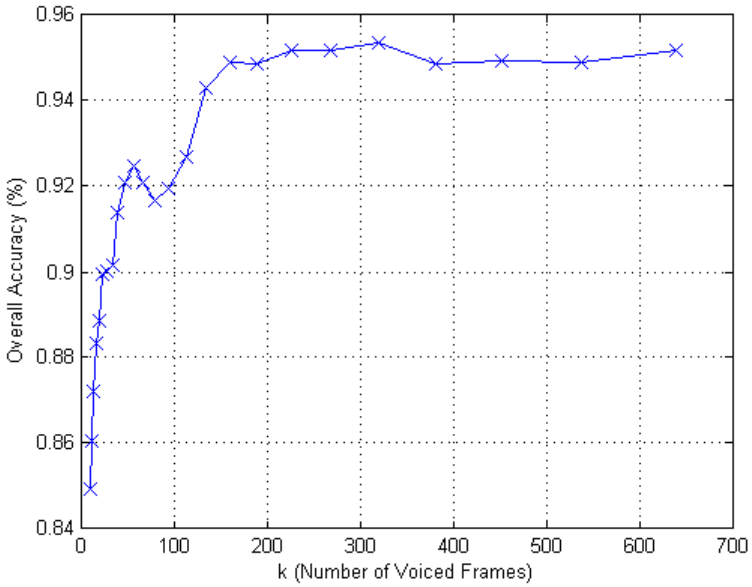

Figure 6.2: Effect of analysis length on overall identification accuracy

Figure 6.2 shows the relationship between the analysis length (in terms of voiced frames) and the overall identification accuracy. Again, the overall accuracy shown at every point was computed by taking the average of the average accuracy of each codec grouping. As indicated previously in Figure 5.1, each point represents the identification strategy as applied to the leading k voiced frames of every test signals. Therefore, the points do not represent independent tests with different analysis lengths, but instead show the effect of different analysis lengths on the exact same input set. This ensures that there are no unfair discrepancies caused by an otherwise random selection of input sets. Furthermore, it would be dramatically more time consuming to perform independent tests at each point.

The results in Figure 6.2 demonstrate that the accuracy increases rapidly for very short analysis lengths, but then asymptotically approaches an accuracy of around 95%. Notice that the first data point, $k = 10$, actually still yields a fairly high accuracy of just under 85%. This high accuracy can be attributed to the fact that, despite having accumulated the noise spectra over so few frames, there are always 7 features on which to match. Even though some features may not yet have sufficiently aggregated to resemble the corresponding feature in the training profile, it is likely that the combination of all of the features together will be sufficient to make the profile distinguishable. As more voiced frames are taken into consideration, each of the profile features incorporates more signal data and should start to better resemble the features from the training profile (which each consist of voiced frames aggregated over 100 sentences). Thus, the accuracy improves with increasing analysis length. However, once the analysis length approaches around 320 voiced frames, the addition of more signal data appears to yield no further value. At this point, the profile features have probably been smoothed out enough (due to the accumulation) that continuing to aggregate data has very little effect on the overall shape of the feature. The reason that the accuracy seems to have a ceiling at only around 95% is simply because a few of the codecs (as mentioned previously) cannot be identified well with the histogram having so much weight. With the current feature weights, the overall accuracy, which is held back by these codecs, simply cannot improve with an increase in analysis length alone.

6.3 Comparison With Previous Work

Recall that the most relevant research against which we can compare our results is Scholz work in [21]. That work presents many of the fundamental strategies that were also used in our research, including the generation of noise spectra, and the quantitative comparison of such by using a normalized cross-correlation function. It also uses a broad selection of contemporary codecs, many of which overlap with those used in our research. For these reasons, Scholz' work is the most applicable for comparing the performance of the respective identification strategies.

The major results from [21] were already presented in Table 3.2. Those results demonstrated an overall accuracy of 88.9% (taken as the average of every codec's average identification rate). Although the accuracy is agreeable, their strategy does require a fairly long source signal in order to achieve those figures. The results in that table were collected by analyzing $k = 2560$ frames of voiced speech. Scholz research used a 256 sample frame size with 50% overlap and narrowband sampling rate, and reported a 33.52% proportion of voiced frames over their input signal data set. This corresponds to just over two minutes worth of signal analyzed per identification. Note that the proportion of voiced frames varies considerably from the 65% in our research. This is due primarily to different VDAs, but may also be partially attributed to the fact that half of their input set was in the German language, which probably contains a greater proportion of unvoiced phonemes.

Because TIMIT does not provide enough audio per speaker to generate 2560 frames of voiced speech, we were unable to perform a comparable test in our research. However, [21] did also publish results for $k = 640$. These results are reproduced in Table 6.3. Clearly, the reduced analysis length, which now corresponds to around 30 seconds of audio, had a significant impact on the accuracy of the identification. The average accuracy from this set of results is only 77.1%.

In our research we were also able to perform a set of tests using $k = 640$ voiced frames for analysis, the results of which are shown in Table 6.4. Here, the average accuracy is 95.1%. This is clearly a significant improvement over Scholz' results for the same number of voiced frames. Figure 6.3 shows a direct comparison of the accuracies for those codecs which are common between Scholz' work and ours.

Note that, despite using the same number of voiced frames, this is not necessarily a fair comparison. Both sets of research use different input speech signals, different VDAs, and different source codecs. Unfortunately, without knowing the details of

			Classified As						
			G.726	AMR	EFR	G.723.1	G.729	HR	G.711
Source Codec	G.726	16 kbit/s	**100.00%**	-	-	-	-	-	-
		24 kbit/s	**95.35%**	-	3.49%	1.16%	-	-	-
		32 kbit/s	**74.12%**	-	0.32%	-	3.19%	-	22.36%
		40 kbit/s	**16.82%**	-	0.47%	-	8.88%	-	73.83%
	AMR	4.75 kbit/s	-	**92.41%**	-	-	3.63%	-	3.96%
		5.9 kbit/s	-	**88.44%**	-	-	2.51%	-	9.05%
		10.2 kbit/s	-	**82.01%**	-	-	5.02%	-	12.97%
	EFR		7.06%	28.24%	**51.76%**	7.06%	-	-	5.88%
	G.723.1	6.3 kbit/s	-	-	19.32%	**77.27%**	-	3.41%	-
	HR		-	-	-	15.93%	-	**84.07**	-
	G.711	μ-Law	-	-	-	-	9.64%	-	**90.36%**

Table 6.3: Scholz results for $k = 640$

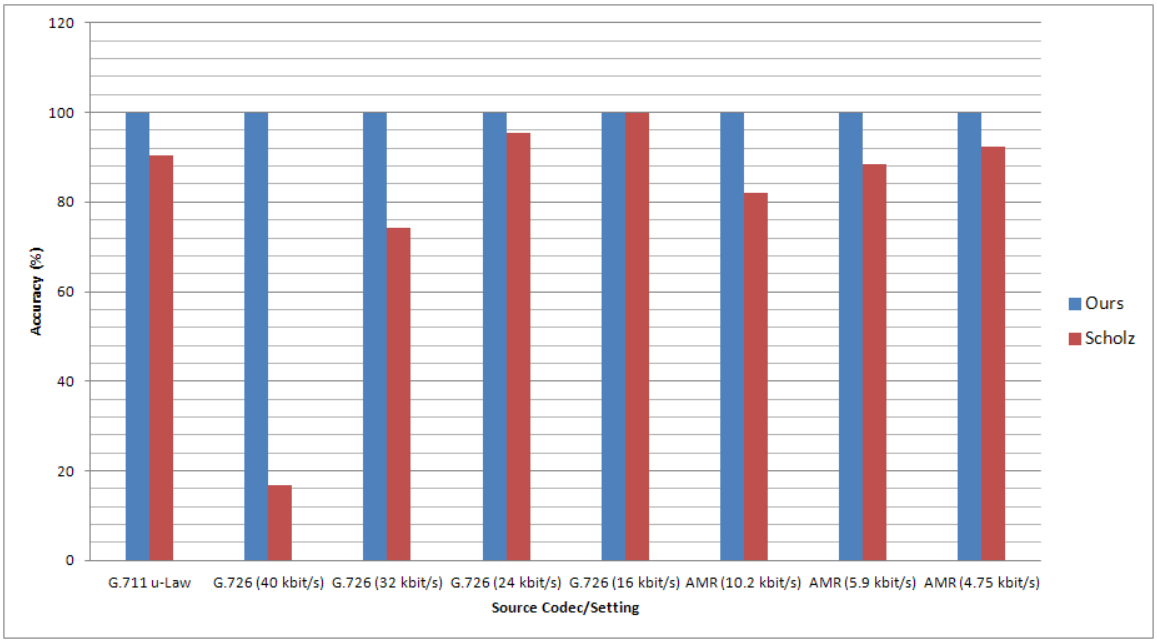

Figure 6.3: Comparison between identification strategies for $k = 640$

many of these aspects from the Scholz work, it is impossible to determine exactly how one identification strategy would perform relative to the other in a perfectly. Nonetheless, the fact that our research has demonstrated comparable accuracy to Scholz' best results, while requiring less than one second of audio for analysis is compelling evidence to suggest that our strategy is a dramatic improvement upon existing work.

			Classified As						
			G.711	G.726	G.728	G.729	iLBC	AMR	Silk
Source Codec	G.711	μ-law	**100.00%**	-	-	-	-	-	-
	G.726	40 kbit/s	-	**100.00%**	-	-	-	-	-
		32 kbit/s	-	**100.00%**	-	-	-	-	-
		24 kbit/s	-	**100.00%**	-	-	-	-	-
		16 kbit/s	-	**100.00%**	-	-	-	-	-
	G.728	16 kbit/s	-	-	**100.00%**	-	-	-	-
	G.729	11.8 kbit/s	-	-	-	**98.81%**	-	-	1.19%
		8 kbit/s	-	-	-	**100.00%**	-	-	-
		6.4 kbit/s	-	-	-	**100.00%**	-	-	-
	iLBC	15.2 kbit/s	-	-	-	7.14%	**92.86%**	-	-
		13.33 kbit/s	-	-	-	10.71%	**89.29%**	-	-
	AMR	12.2 kbit/s	-	-	-	-	-	**100.00%**	-
		10.2 kbit/s	-	-	-	-	-	**100.00%**	-
		7.95 kbit/s	-	-	-	-	-	**100.00%**	-
		7.4 kbit/s	-	-	-	-	-	**100.00%**	-
		6.7 kbit/s	-	-	-	-	-	**100.00%**	-
		5.9 kbit/s	-	-	-	-	-	**100.00%**	-
		5.15 kbit/s	-	-	-	-	-	**100.00%**	-
		4.75 kbit/s	-	-	-	-	-	**100.00%**	-
	Silk	VBR	-	-	-	24.40%	-	-	**75.60%**

Table 6.4: Our results for $k = 640$

Chapter 7

Future Work

Despite the very favorable results that arose from this research, there is still potential room for improvement and additional research. There are several extensions that could be made with regard to the profiling and identification strategies proposed in this research. Moreover, there are plenty of related research topics that could stem from the techniques developed in this research.

One area of our identification strategy proved to be particularly difficult, and is still partially unresolved. The feature weights discussed in Section 5.3 were ultimately selected based largely on the results of repeated empirical testing. This trial-and-error process demonstrated that the feature weights could potentially have a significant impact on the overall identification accuracy. For example, it was clear in the results for this research that the identification of a few of the codecs was impeded by the strong weighting of the histogram feature. One extension of the research could be to devise a way to select an optimal set of features and weights for a given expected distribution of source codecs. This would allow for even higher accuracies, and possibly also shorter analysis lengths.

Another goal for additional research would be to expand the selection of identifiable source codecs. It would be very desirable, for example, to consider many newer codecs, and perhaps wideband codecs, in the research. However, as more source codecs are considered, the additional training profiles would necessarily cause the profiles to become more similar to one another. This would lead to more ambiguity during the identification process, and ultimately to worse results. This was exhibited, for example, with the different codec bitrates shown in Table 6.1. Here, the differences between the profiles from the same codec using different settings were too subtle to be accurately distinguished using our feature set. Thus, there is still plenty of room for improving the overall strategy.

The prominent contributions of our research included recognizing several features

that may be used for gathering signal information that highlights codec-imposed noise. Our identification strategy then made use of these features to construct training profiles and compare the test profiles from unknown signals against these training profiles. However, an alternative approach would be to use the technique of supervised learning [6]. In this approach, each signal from a known codec can be used to train a machine learning system, such as a support vector machine (SVM), based upon the features of that signal. Provided a sufficiently large training set of signals, the SVM can then take subsequent signals with unknown codecs and directly classify them based upon the statistical clustering of the feature values from each class (codec) in the training set. Thus, the problem of codec identification would be an examplary application of machine learning, and merits further research in using such an approach.

The signals used in this research originated from clean recordings. However, in practical applications, this may not be a reasonable assumption. Real calls might contain background noise from the environment of the caller, interference in the transmission medium, and other possible unwanted signal components. This noise would probably not work well with our proposed strategy because the computation of the noise spectral is most effective at eliminating speech components while preserving the noise component. The subtle artifactual noise from the codec would probably masked by the more dominant noise from the original signal. An even bigger, but related, challenge would be to cope with the effect of cascaded codecs. In a real telecommunications network, it is common for a signal to be transcoded several times from one endpoint to another as it passes through different carriers. It would be an interesting research pursuit to attempt to identify not one codec from the received signal, but the entire transcoding sequence. It is worth noting that the work in [3] offers a step in this direction, but at a higher level (for example, broadly determining that a call traversed from a cellular network to a VoIP network).

Because there has previously been limited research in this specific area, almost no research has been done on the applications of this type of work. Thus, a major branch of future work could concentrate on utilizing this codec identifcation strategy to aid in determining call provenance, detecting tampered audio streams, improving speech recognition in compressed speech signals [5], or a myriad of other practical applications.

Chapter 8

Conclusions

Overall, this research was very challenging. It is exceptionally difficult to be able to separate perceptually similar audio signals based solely upon the subtle artifacts imparted by the codec with which they had initially been processed. Being able to accurately identify these codecs using nothing more than the observed signal is an impressive feat that we were able to achieve with the methods developed in this research. The methods are based upon a firm understanding of a diverse selection of speech coding techniques, as well as influences from the limited amount of existing research in this area.

Although our research did build upon a very similar study, we introduced several novel aspects which contributed to the dramatic improvements in results. The contributions included using a multidimensional approach to the profiling process, whereby several features are captured from the input signal. Among these features, we introduced the use of a histogram of sample values, which proved to be the most effective feature in distinguishing the source codecs used in our research by incorporating important time domain information into the profile. In addition, rather than synthesizing a harmonic spectrum and performing difficult harmonic/noise decomposition, we generated each noise spectrum using existing off-the-shelf codecs. This makes the overall system design much simpler and allows us to generate several unique noise spectra for each input signal. Of course, with the advent of multiple features in the profile, we also had to devise a meaningful way in which to compare and evaluate all of the features between a pair of profiles. This led us to the use of the weighted sum for the identification decision.

Using the set of codecs and speech samples available to us, we evaluated our strategy by performing testing using every codec setting on every speaker as inputs. Our results were extremely favorable, demonstrating average accuracies of around 95% for speech signals as short as 4 seconds in length. Both the accuracy and the

required signal analysis length represent dramatic improvements over previous work. Furthermore, we collected sufficient data to show a definite trend with regard to the effect of the analysis length on the overall accuracy. This plot is useful because it tells us that there is no merit in analyzing more than a few seconds worth of signal, but that the accuracy decreases rapidly for shorter windows.

Although the results were satisfactory, there are still many extensions of this research that might be explored. Such endeavors might seek to increase the accuracy, distinguish between more codecs, or deal with noisy signals. There are also plenty of extensions with regards to applications of such research, for example, in audio forensics.

Bibliography

[1] D. M. Alley. Automatic Identification of Voice Band Telephony Coding Schemes Using Neural Networks. *Electronics Letters*, 29(13):1156–1157, 1993.

[2] S. C. Andersen, A. Duric, H. Astrom, R. Hagen, W. B. Kleijn, and J. Linden. Internet Low Bit Rate Codec (iLBC). RFC 3951, Internet Engineering Task Force, December 2004.

[3] Vijay A Balasubramaniyan, Aamir Poonawalla, Mustaque Ahamad, Michael T Hunter, and Patrick Traynor. PinDr0p : Using Single-Ended Audio Features To Determine Call Provenance Categories and Subject Descriptors. In *Proceedings of the 17th ACM Conference on Computer and Communications Security*, pages 109–120, New York, 2010. Georgia Institute of Technology, ACM.

[4] J. Benesty, J. Chen, and Y. Huang. Linear Prediction. In Jacob Benesty, M. Mohan Sondhi, and Yiteng (Arden) Huang, editors, *Springer Handbook of Speech Processing*, volume 46, pages 121–134. Springer, June 2008.

[5] L. Besacier, C. Bergamini, D. Vaufreydaz, and E. Castelli. The effect of speech and audio compression on speech recognition performance. In *Multimedia Signal Processing, 2001 IEEE Fourth Workshop on*, pages 301 –306, 2001.

[6] Francesco Camastra and Alessandro Vinciarelli. Machine learning. In *Machine Learning for Audio, Image and Video Analysis*, Advanced Information and Knowledge Processing, pages 83–89. Springer London, 2008.

[7] J. H. Chen and J. Thyssen. Analysis-by-Synthesis Speech Coding. In Jacob Benesty, M. Mohan Sondhi, and Yiteng (Arden) Huang, editors, *Springer Handbook of Speech Processing*, pages 351–392. Springer, 2008.

[8] ETSI. Digital Cellular Telecommunications System (Phase 2+); Adaptive Multi-Rate (AMR) Speech Transcoding (GSM 06.90 version 7.2.1 Release 1998). 1998.

[9] Robert X Gao, Ruqiang Yan, Robert X. Gao, and Ruqiang Yan. From fourier transform to wavelet transform: A historical perspective. In *Wavelets*, pages 17–32. Springer US, 2011.

[10] D. W. Griffith and J. S. Lim. Multiband excitation vocoder. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 36(8):1223–1235, 1988.

[11] Lajos Hanzo, F. Clare Somerville, and Jason Woodard. *Voice and Audio Compression for Wireless Communications*. IEEE Press, Chippenham, 2nd edition, 2007.

[12] W. B. Kleijn. Principles of Speech Coding. In Jacob Benesty, M. Mohan Sondhi, and Yiteng (Arden) Huang, editors, *Springer Handbook of Speech Processing*, pages 283–305. Springer, 2008.

[13] A. M. Kondoz. *Digital Speech: Coding for Low Bit Rate Communication Systems*. John Wiley & Sons Ltd., West Sussex, 2nd edition, 2004.

[14] Haibin Ling and K. Okada. Diffusion distance for histogram comparison. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 1, pages 246 – 253, June 2006.

[15] Haibin Ling and K. Okada. An efficient earth mover's distance algorithm for robust histogram comparison. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 29(5):840 –853, May 2007.

[16] A. V. McCree. Low-Bit-Rate Speech Coding. In Jacob Benesty, M. Mohan Sondhi, and Yiteng (Arden) Huang, editors, *Springer Handbook of Speech Processing*, volume 72, pages 97–105. Springer, January 2008.

[17] Andrew J. Noga. A Short-Segment Fourier Transform Methodology. Technical Report March, Air Force Research Laboratory, Rome, NY, 2009.

[18] National Institute of Standards and Technology. *TIMIT Suggested Training/Test Subdivision*, 1990.

[19] Lawrence Rabiner and Ronald Schafer. *Digital Processing of Speech Signals*. Prentice-Hall International, Inc., London, 1978.

[20] David Salomon. Audio Compression. In *Data Compression: The Complete Reference*, chapter Audio Compression, pages 719–850. Springer, 2006.

[21] K. Scholz, L. Leutelt, and U. Heute. Speech-codec detection by spectral harmonic-plus-noise decomposition. In *Conference Record of the Thirty-Eighth Asilomar Conference on Signals, Systems and Computers, 2004.*, pages 2295–2299. Ieee, 2004.

[22] Manfred R. Schroeder and Bishnu S. Atal. Code-Excited Linear Prediction (CELP): High-Quality Speech at Very Low Bit Rates. In *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP '85*, pages 937–940, 1985.

[23] Priyabrata Sinha. Speech Compression Overview. In *Speech Processing in Embedded Systems*, pages 93–100. Springer, 2010.

[24] Priyabrata Sinha. Waveform Coders. In *Speech Processing in Embedded Systems*, pages 101–112. Springer, 2010.

[25] A. S. Spanias. Speech Coding: A Tutorial Review. *Proceedings of the IEEE*, 82(10):1541–1582, 1994.

[26] Ashwin Swaminathan. *Multimedia Forensic Analysis Via Intrinsic and Extrinsic Fingerprints*. Doctor of philosophy, University of Maryland, 2008.

[27] The International Telegraph and Telephone Consultative Committee. *Recommendation G.726: 40, 32, 24, 16 kbit/s Adaptive Differential Pulse Code Modulation (ADPCM)*, 1990.

[28] The International Telegraph and Telephone Consultative Committee. *Recommendation G.728: Coding of Speech at 16 kbit/s Using Low-Delay Code Excited Linear Prediction*, 1992.

[29] DARPA TIMIT. Acoustic-phonetic continuous speech corpus cd-rom. In *Document NISTIR 4930, NIST Speech Disk 1-1.1.*

[30] International Telecommunication Union. *Recommendation G.711: Pulse Code Modulation (PCM) of Voice Frequencies*, 1993.

[31] International Telecommunication Union. *Recommendation G.729: Coding of Speech at 8 kbit/s Using Conjugate-Structure Algebraic-Code-Excited Linear Prediction (CS-ACELP)*, 2007.

[32] K. Vos, S. Jensen, and K. Soerensen. SILK Speech Codec. Technical report, Internet Engineering Task Force, 2010.

Appendix A

Contents of DVD-ROM

The implementation of the identification strategy introduced in this research was performed in the MATLAB environment on a Windows PC. The DVD-ROM included with this document contains the source code for the MATLAB functions and scripts that were used during this research. Although several additional functions were developed during the research phase, they were ultimately unused during the final testing, and are not included on this disc. The MATLAB workspaces containing the training profiles and the final results are included. The disc also includes the speech samples from TIMIT that were used for creating the training profiles and performing the testing. Lastly, the DVD-ROM contains the binary executables for the codecs that were used in this research (note that these have been compiled for Windows).

A more detailed description of each notable file or directory on the DVD-ROM follows:

Item	Description
/Codecs/G711.exe	Encoder and decoder application for ITU-T G.711 codec
/Codecs/G726.exe	Encoder and decoder application for ITU-T G.726 codec
/Codecs/G728.exe	Encoder and decoder application for ITU-T G.728 codec
/Codecs/G729_Decoder.exe	Decoder application for ITU-T G.729 codec
/Codecs/G729_Encoder.exe	Encoder application for ITU-T G.729 codec
/Codecs/iLBC.exe	Encoder and decoder application for iLBC codec
/Codecs/AMR_Encoder.exe	Encoder application for AMR codec
/Codecs/AMR_Decoder.exe	Decoder application for AMR codec
/Codecs/Silk_Decoder.exe	Decoder application for SILK codec
/Codecs/Silk_Encoder.exe	Encoder application for SILK codec
/MATLAB/training_profiles.mat	Workspace containing codec training profiles generated in this research
/MATLAB/testing_results.mat	Workspace containing the test results collected in this research
/MATLAB/align_and_trim.m	Function to align two similar signals and crop to the overlapping region
/MATLAB/data_align.m	Function that determines the time shift between similar signals
/MATLAB/find_leaf_dirs.m	Function that recursively finds terminal directories in a filesystem tree
/MATLAB/make_reprocess_template.m	Function that generates an aggregate noise spectrum
/MATLAB/process_AMR.m	Wrapper around the AMR codec application
/MATLAB/process_G711.m	Wrapper around the G.711 codec application
/MATLAB/process_G726.m	Wrapper around the G.726 codec application
/MATLAB/process_G728.m	Wrapper around the G.728 codec application
/MATLAB/process_G729.m	Wrapper around the G.729 codec application
/MATLAB/process_iLBC.m	Wrapper around the iLBC codec application
/MATLAB/process_Silk.m	Wrapper around the SILK codec application
/MATLAB/raw2array.m	Function to read a raw speech file and return the downsampled signal values
/MATLAB/recurse_dirs.m	Function to recursively find all files in a filesystem tree
/MATLAB/run_build_training_profiles.m	Script to create codec training profiles from TIMIT training files
/MATLAB/run_identification_test.m	Script to perform the identification process on TIMIT test files
/MATLAB/Voicebox and Utilities	Collection of various speech processing functions (3rd party)
/MATLAB/Voicebox and Utilities/pda	Pitch detection algorithm used for voicing determination (3rd party)
/TIMIT/TRAIN	TIMIT training speech files with header information removed
/TIMIT/TEST	TIMIT testing speech files with header information removed

Table A.1: Description of contents included on DVD-ROM