AUTOMATIC SPEECH EMOTION RECOGNITION

Alexandros Georgogiannis

Department of Electronic & Computer Engineering
Technical University of Crete, Greece

Submitted on July 15th, 2011 in partial fulfilment of the requirements for the Electronic and Computer Engineering diploma degree.

Advisor: Professor Vassilis Digalakis

Co-advisor: Associate Professor Alexandros Potamianos

Co-advisor: Assistant Professor Michail Lagoudakis

ABSTRACT

Although emotion detection from speech is a relatively new field of research, it has many potential applications. In human-computer or human-human interaction systems, emotion recognition systems could provide users with improved services by being adaptive to their emotions. In virtual worlds, emotion recognition could help simulate more realistic avatar interaction. In this thesis we study automatic emotion recognition from speech signals and build systems capable of detecting the emotional state of a speaker. We examine a set of new cepstral features based on Teager energy operator and compare their performance with existing cepstral representations. We compare the discriminatory capability of sets with prosodic features created from five different feature selection methods and using these sets we evaluate the performance of several classifiers and their combinations. Also we design a system capable of using both cepstral and prosodic features for speech emotion recognition. This system combines decisions made by individual classifiers and shows a significant improvement in classification accuracy.

CHAPTER 1

INTRODUCTION

The work of this thesis focuses on automatic emotion recognition from speech signals. For most people, speech is the most natural and efficient manner of exchanging information. The emotional and physical states of a speaker are known as paralinguistic aspects of speech. Although the emotional state does not alter the linguistic content, it is an important factor in human communication, because it provides feedback information in many applications; below we mention some of those.

In human-robotic interfaces (HRI), robots can be taught to interact with humans and recognise their emotions. Robotic pets, for example, should be able to understand the emotional and health status of their commander and modify their actions accordingly. Such a robot is Robovie [1], which is capable of detecting and moderating tension.

In call-centers, speech emotion recognition helps to detect problems that arise from an unsatisfactory course of interaction. A frustrated customer is typically offered the assistance of human operators or some reconciliation strategy [2].

In intelligent spoken tutoring systems, detecting and adapting to students' emotions is considered to be an important strategy for closing the performance gap between human and computer tutors [3]. Studies in educational psychology point out that students' emotions can impact their performance and learning

In spoken dialogue research, it is beneficial to enable the systems not only to recognize the content encoded in user's response, but also to extract information about the emotional state of the user by analyzing how these responses have been spoken.

Since we tackle the problem of speech emotion recognition as a pattern recognition task, we follow in broad lines the following approach:

- Consider an emotional model (e.g., discrete or continuous)
- Start analyzing one or more of the available speech emotion databases
- Extract a set of features
- Train a classifier in order to make statements on the test data

Each of these steps is actually a point where a decision needs to be made. The first two problems can be regarded as a whole, since there are not many available databases, so the emotional model in most cases will be the one used for the recording of the used database.

When it comes to feature extraction, the methods differ with regard to the features types (e.g., prosodic, spectral, and linguistic) and unit of analysis. Some researchers use features extracted in a frame base, some use entire utterances and some take an intermediate approach, for instance segments between pauses or words. It was proven that many times fusion of features from different levels can lead to improved recognition.

Choosing a classification technique based on literature is also not trivial. So far researchers have used different classification methods. Since everybody is doing emotion recognition in their own way, it should be possible to compare all approaches and to find advantages and disadvantages of each, and later on come with a version that takes the best decisions at each point. However, coming to a conclusion from the previous work is not straightforward. This is mainly because researchers use different databases, many of them unavailable for all researchers, and after using different features and classification techniques they report the results in different manners, so it becomes impossible to know for sure which method was better.

The metrics to rate an emotion recognition system are usually dominated by the accuracy of detecting the right emotion. To summarize, our purpose is to create a speech emotion recognition system that is able to perform emotion recognition from speech, and tackle at least part of the problems that arise in each of the previously described steps. There are many methods, many approaches, many experimental settings, so taking the right decision becomes a very challenging task. We are in search for a good model for emotion

recognition, that can benefit from the information from previous research and therefore is robust and general.

This thesis is organised as follows. The subject and the main problems in the field were presented in introduction. In CHAPTER 2 we give an overview of the related work. In the first part we present the most important emotion models. We proceed with reviewing some popular emotional speech databases and the database we choose to work with. In the third section of the second chapter we present several ways of describing the speech signal. This is done in terms of speech features: prosodic, spectral, voice quality and linguistic. A section describing some of the frequently used machine learning methods used so far in emotion recognition from speech follows.

In CHAPTER 3 we design a system based in Gaussian mixture models capable for emotion recognition using only cepstral features. We introduce a new cepstral representation of speech signals based on the non-linear Teager-energy operator. We compare the performance of the new feature set to the performance of Mel frequency cepstral coefficients and another feature set called T-MFCC.

Moving on, in CHAPTER 4, we design a speech-emotion recognition system based exclusively on statistics of prosodic features. We test many feature-selection algorithms in order to eliminate redundant features. With the feature sets created by feature-selection algorithm, we train and evaluate three popular classifiers: naive Bayes, fuzzy *k*-NN, and linear discriminant analysis. We also investigate the improvement gained by their combinations under fixed combining rules.

In CHAPTER 5, we extend the experiments of CHAPTER 4 by testing ensemble-classification algorithms using prosodic features. Ensemble algorithms use multiple models to obtain better predictive performance than could be obtained from any of the constituent models. Their performance is promising and comparable to the performance of Gaussian mixtures in CHAPTER 3.

CHAPTER 6 is an attempt to combine all the previously designed systems by fusing their decisions. We get a significant improvement in classification accuracy by their combination.

The thesis ends with our conclusions. Chapter 7 is about lessons learned as well as unanswered questions and solutions still to be found. Directions for further research are given within this last chapter.

CHAPTER 2

EMOTION RECOGNITION BASICS

In this chapter we present the main ideas behind the process of recognising emotions. In Section 2.1, we review the main emotion theories and models. In Section 2.2, we present the work developed with regards to speech emotional databases. Section 2.3 presents most of the features used in classification and their variations with respect to emotions. Emotion classification approaches are discussed in Section 2.4.

2.1. EMOTIONS

The human speech communication consists of two channels, the explicit channel and the implicit channel. The first transmits explicit messages (''What was said'') and the second transmits implicit messages about the speakers themselves (''How was it said''). Although scientists invested enormous efforts in understanding the explicit channel through automatic speech recognition (ASR), still much research is needed to reliably decode the implicit channel. The obvious goal of emotion recognition is to assign category labels that identify emotional states. An emotion is generally a mental and physiological state associated with a wide variety of feelings, thoughts, and behavior. There is no agreement on a set of basic emotions, though numerous taxonomies have been proposed. Some categorizations include:

- ''Cognitive'' versus ''non-cognitive'' emotions
- Instinctual emotions (from the amygdala), versus cognitive emotions (from the pre-frontal cortex)
- Basic versus complex: where base emotions lead to more complex ones
- Categorization based on duration: Some emotions occur over a period of seconds (e.g., surprise) where others can last for years (e.g., love)
- Categorization based on activation (arousal), potency(power), and valence (pleasure). It's worth mentioning the difficulty in emotion recognition to distinguish anger and happiness. This pair of emotions differ only in the valence dimension [4].

Some categories do appear in almost every list of ''basic'' emotions – like happiness, sadness, fear, anger, surprise and disgust. It is no doubt that they are key points of reference. It is probably best to describe them as archetypal emotions, which reflect the fact that they are undeniably the obvious examples of emotion. Although the archetypal emotions are important, they cover rather a small part of emotional life. It is a pragmatic problem to find a set of terms that covers a wider range without being unmanageable.

2.2. EMOTIONAL SPEECH DATABASES

A record of emotional speech data is useful for emotional speech analysis. For this purpose, a list of emotional speech data collections is overviewed in [5]. Regarding [5], it is evident that the research on emotional speech recognition is limited to the emotions at which we referenced above as ''basic".

Three kinds of speech are observed. Natural speech is simply spontaneous speech in which all emotions are real. Simulated or acted speech is speech expressed in a professionally deliberated manner. Finally, elicited speech is speech in which the emotions are induced. The elicited speech is neither neutral nor simulated.

Emotion recognition from natural speech, which is the goal in practice, is much more difficult than emotion recognition from acted speech due to the much larger variation of emotional expressions in a natural conversation. Unfortunately, natural speech databases for emotion recognition (e.g. from call centers) are seldom public available due to the privacy of speakers. In addition, the acquisition and labeling of a large size database is very expensive. Among the most popular emotional speech databases, in the sense that they serve as speech corpus for many researches, are the German database of emotional speech [2], danish emotional speech database (DES) [6] and speech under simulated and actual stress database (SUSAS) [7].

TABLE 2.1
Amount of recordings for each emotion from the EmoDB

Neutral	Happiness	Anger	Sadness	Fear	Disgust	Boredom	Total
80	65	125	55	55	40	80	500
16%	13%	25%	11%	11%	8%	16%	100%

2.2.1. The German Emotional Speech Database

The database we chose to work with is the German database of emotional speech. Hereafter, we will refer to it as EmoDB. EmoDB is a recorded database of emotional utterances spoken by actors (i.e., simulated speech utterances). It is developed by the Technical University, Institute for Speech and Communication, Department of Communication Science, Berlin. This database contains recordings, sampled at 16KHz, from 5 actors and 5 actresses, 10 different sentences of 7 kinds of emotions: anger, boredom, disgust, fear, happiness, sadness, and neutral.

Mean recognition rates of 20 participants that were asked to classify the emotional state are shown in Figure 2.1. They were presented with the utterances in random order in front of a computer monitor. They were allowed to listen to each sample only once before they had to decide in which emotional state the speaker had been and how convincing the performance was.

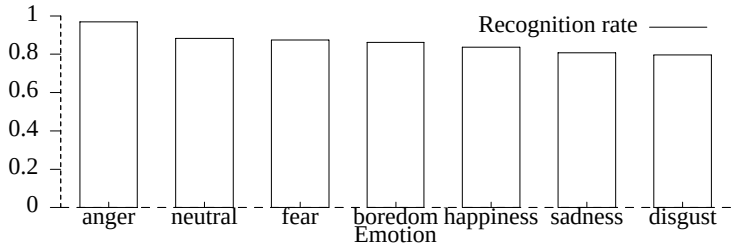

Figure 2.1: Recognition rates by humans

The database consists of totally 500 speech samples. The amount of recordings from each emotion for the Berlin database is shown in Table 2.1. From these 500 speech samples in this database we make a database that correspond to anger, happiness, neutral, disgust, and sadness to come up with 231 files (Table 2.2). We chose these 5 emotions because they appear in almost every list of ‘‘basic’’ emotions. The EmoDB is one of the most often used databases in the context of emotion recognition from speech, and also one of the few for which some results can be compared.

A web interface was developed to present the database of emotional speech. All the available information of the speech database can be accessed via the internet: http://www.expressive-speech.net/emoDB/.

2.3. SPEECH FEATURES

An important task in emotion classification from speech is to get a clear expression of emotions in the feature space. That is, several features should be selected according to their discriminatory capabilities. Since speech signal is time-varying, the analysis should be a time-frequency analysis. Often, in order to analyse signals, we model them as random processes and we assume that are wide sense stationary (WSS) processes, i.e., its mean and autocorrelation, do not vary with a shift in the time origin. In the speech signal,

TABLE 2.2

The number of selected audio files from EmoDB in the emotional states of interest and the total amount of data used.

Anger	Happiness	Neutral	Sadness	Disgust	Total
52	54	62	35	28	231
22%	23%	27%	15%	13%	100%

however, we have to cut the whole signal into blocks to obtain short time stationarity. The process of cutting a signal into blocks assuming wide sense stationarity is called short-term processing of speech signals while the features derived over periods equal to one frame are called short-term features.

Short-term features are estimated on a frame basis, i.e.,

$$f_s(n;m) = s(n)\,w(n-m) \tag{2.1}$$

where $s(n)$ is the speech signal and $w(m-n)$ is a window of length N_w ending at sample m. Acoustic features can be categorized in: prosodic, spectral, and voice quality features.

2.3.1. Prosodic features

Prosody expresses the rhythm, stress, and intonation of speech. Emotional prosody is the expression of feelings using prosodic elements of speech. Pitch, loudness, speaking rate, durations, pause and rhythm are all perceived characteristics of prosody.

Pitch signal

Pitch is an auditory perceptual property that allows the ordering of sounds on a frequency-related scale [8]. Pitch may be quantified as a frequency, but pitch is not a purely objective physical property; it is a subjective psycho-acoustical attribute of sound. Pitch frequency or fundamental frequency, F_0, of the phonation is the vibration rate and pitch period, T_0, is the time elapsed between two successive vocal fold openings.

A pitch detector is an essential component in a variety of speech processing systems and provides necessary information about the nature of the excitation source for speech coding. The pitch contour of an utterance is useful for recognizing speakers, determination of their emotion state, for voice activity detection task, and many other applications.

Various pitch-detection algorithms have been developed: modified autocorrelation method [9], cepstrum method [10], robust algorithm for pitch tracking (RAPT) [11], average magnitude difference function method (AMDF) [12], SIFT [13], etc. Most of them have very high accuracy for voiced pitch estimation, but the error rate considering voicing decision is still quite high [14].

Loudness

Loudness is the quality of a sound that is primarily a psychological correlate of physical strength (amplitude). More formally, it is defined as ‘‘that attribute of auditory sensation in terms of which sounds can be ordered on a scale extending from quiet to loud’’ [15]. Sound energy is perceived as loudness and is related to emotional intensity.

Duration features

Duration features correlate with the speaking style, e.g., speaking rate, duration, pauses. The speech rate is calculated as the inverse duration of the voiced part of speech determined by the presence of pitch pulses [16].

2.3.2. Spectral features

Spectral features are frequency-related features calculated using the speech spectrum. Number of harmonics, formant, mel-frequency cepstral coefficients (MFCC), and linear predictive cepstral coefficients (LPCC) fall in the category of spectral features.

Number of harmonics

The number of harmonics due to non-linear air flow in the vocal tract that produces the speech signal are useful spectral features. Harmonics are multiple integers of fundamental frequency. A method for finding the number of harmonics from the speech signal was proposed by [17], where they introduced the Teager-energy operator.

Formants

The formants are one of the quantitative characteristics of the vocal-tract. In the frequency domain, the location of vocal tract resonances depends upon the shape and the physical dimensions of the vocal tract. Since the resonances tend to ''form'' the overall spectrum, speech scientists refer to them as formants. A simple method to estimate formants relies on linear predictive analysis [18].

Mel-frequency cepstral coefficients

Mel-frequency cepstral coefficients (MFCC) are coefficients that collectively make up the mel-frequency cepstrum. The mel-frequency cepstrum (MFC) is a representation of the short-term power spectrum of a sound, based on a linear cosine transform of a log power spectrum on a nonlinear mel scale of frequency. The mel scale is a perceptual scale of pitches judged by listeners to be equal in distance from one another.

Linear Predictive Analysis and Linear Predictive Cepstral Coefficients

The basic idea behind linear predictive analysis is that a specific speech sample at the current time can be approximated as a linear combination of past speech samples. Through minimizing the sum of squared distances (over a finite interval) between the actual speech samples and linear predicted values a unique set of parameters or predictor coefficients can be determined. In reality the actual predictor coefficients are never used in recognition, since they typically show high variance. The predictor coefficients are transformed to a more robust set of parameters known as linear predictive coding coefficients (LPCC). The principal advantage of cepstral coefficients is that they are generally decorrelated and this allows diagonal covariances to be used in statistical models like hidden Markov models (HMM) and Gaussian mixtures models (GMM).

2.3.3. Voice quality features

Studies on voice quality report that there is a strong correlation between voice-quality features and emotions. Jitter, shimmer, and harmonics to noise ratio (HNR) are voice-quality features extracted from a speech signal. Jitter measures cycle to cycle variation of period length while shimmer measures cycle to cycle variations of peak or average amplitude. HNR measures the degree of periodicity in a sound. Applications involving these features can be found in [19] and [20].

2.3.4. Cues to emotion

The contour of selected short-term acoustic features is affected by emotional states. A short-term feature contour is formed by assigning the feature value computed on a frame basis to all samples belonging to the frame. For example, the energy contour is given by

$$e(n) = E_s(m),\ n = m - N_w + 1, \ldots, m \tag{2.2}$$

where $E_s(m) = 1/N_w \sum_{n=m-1}^{m} | f_s(n;m) |^2$ is the short-time speech energy. The contour trends (i.e., its plateaux, rising or falling slopes) are valuable features for emotion recognition, because they describe the temporal characteristics of an emotion.

If features for an entire speech segment are to be analyzed, statistical functions like mean, median, minimum, maximum, standard deviation, or more seldom third or fourth standardized moments are applied to the extracted contours. Table 2.3 presents a summary of the effects of several emotional states on selected acoustic features [5]. The results are based on few studies, therefore, it is advisable to consider the table as a set of empirical expectations and not a set of established results.

2.4. EMOTION CLASSIFICATION TECHNIQUES

There is a theorem on supervised machine learning called *No free lunch theorem* [21]. It states that in a noise-free scenario, where the loss function is the misclassification rate, if one is interested in off-training-set error, then there are no a priori distinctions between learning algorithms. In other words there is no best classifiers in general, but the choice of classifier depends on the problem at hand; it is important to test many of them before choosing one.

TABLE 2.3
Effects of several emotion states on selected acoustic features.

	Pitch				Intensity		Timing	
	Mean	Range	Variance	Contour	Mean	Range	Speech Rate	Transmission duration
Anger	$>>$	$>$	$>>$		$>>_M$, $>_F$	$>$	$<_M$, $>_F$	$<$
Disgust	$<$	$>_M$, $<_F$			$<$		$<<_M$, $<_F$	
Fear	$>>$	$>$		$\uparrow$	$\geqq$		$<<_M$, $<_F$	$<$
Joy	$>$	$>$	$>$	$\downarrow$	$>$	$>$		$<$
Sadness	$<$	$<$	$<$	$\uparrow$	$<$	$<$	$>_M$, $<_F$	$<$

Explanation of symbols: $>$: increases, $<$: decreases, $=$: no change from neutral, $\uparrow$: inclines, $\downarrow$: declines. The subscripts M and F stands for males and females respectively.

A large variety of machine learning algorithms (classifiers) are used for recognition of several emotional states from speech. The most frequently used are support vector machines (SVM) and artificial neural networks (ANN). A rapidly evolving area in pattern recognition research is the combination of classifiers to build the so-called classifier ensembles. For a number of reasons (ranging from statistical to computational and representational aspects) ensembles tend to outperform single classifiers.

2.5. CONCLUSIONS

After presenting the most recent trends in emotion recognition from speech, it became clear that there is no recipe on how to develop a successful emotion recognition system. For almost all decisions that one can made there are pros and cons. For the future, it would be good if some standards for emotional databases will be considered and followed, which will facilitate the comparison between different results. There is still place for feature set optimization.

CHAPTER 3

EMOTIONAL SPEECH CLASSIFICATION USING
NON LINEAR TEAGER ENERGY BASED FEATURES

In this Chapter, a new set of feature parameters based on the Teager-energy operator (TEO) is introduced. These new feature parameters are motivated by the MFCC and Teager energy based mel frequency cepstrum coefficients (T-MFCC). We call this new features Teager energy mel frequency cepstrum coefficients (TEMFCC). TEMFCC outperform MFCC and T-MFCC in the presence of noise at low SNR values. Also we study how to improve the classification performance of a system by combining the results of different classifiers, each one based on different spectral features.

3.1. INTRODUCTION

Traditional theories of speech production are based on a linearization of pressure and volume velocity relations. Furthermore, these values are assumed constant within a given cross section of the vocal tract, i.e., one dimensional plane wave assumption. The linear model assumption neglects the influence of any non-acoustic motion of the fluid medium.

3.2. TEAGER ENERGY FEATURES

3.2.1. Aeroacoustic Flow in the Vocal Tract

There is an increasing collection of evidence suggesting that non-acoustic fluid motion can significantly influence the sound field. For example, measurements by Teager [22] reveal the presence of separated airflow within the vocal tract. Separated flow occurs when a region of fast moving fluid, a jet, detaches from regions of relatively motionless fluid. When this occurs, viscous forces create a tendency for the fluid to rotate and create vortices. The vortices can convert downstream at speeds much slower (~90% slower) than acoustic propagation speed. Jet flow and vortices thus fall in the category of non-acoustic behavior.

Teager made a hypothesis on the presence of non-acoustic phenomenon in the vocal tract and made extensive measures which showed that the airflow velocity is not uniform across the cross section. These measurements are clearly inconsistent with planar one-dimensional acoustic flow velocity.[17].

3.2.2. Teager Energy Operator

Motivated by the measurements of Teager, Kaiser re-examined the source filter theory in light of aeroacoustic theories, giving further credence to a 'jet-cavity flow' paradigm in the vocal tract. Kaiser hypothesized that the interaction of the jet flow and vortices with the vocal tract cavity is responsible for much of the speech fine structure. He proposed the need for time-frequency analysis methods that can track rapid signal energy changes within a glottal cycle. This led to the definition of TEO based on a definition of energy that accounts for the energy in the system that generated the signal [23].

For continuous-time real signals TEO is defined as

$$\Psi_R(x(t)) = \dot{x}(t)^2 - x(t)\ddot{x}(t) \tag{3.1}$$

and for discrete-time real signals as

$$\Psi_R(x(n)) = x^2(n) - x(n-1)x(n+1) \tag{3.2}$$

TEO has been extended to cover complex signals. As an energy operator, we expect to always give positive values, but this is not always the case for all signals.

In [24] the definition for continuous time complex signals is

$$\Psi_C(x(t)) = \Psi_R(\mathrm{Re}(x(t)) + \Psi_R(\mathrm{Im}(x(t))) \tag{3.3}$$

and for discrete time complex signals is

$$\Psi_C(x(n)) = \Psi_R(\mathrm{Re}(x(n)) + \Psi_R(\mathrm{Im}(x(n))) \tag{3.4}$$

This means that the Teager energy of a complex signal is the sum of the energy of real and imaginary parts of the signal.

3.2.3. Existing Work

TEO is frequently used in the development of feature representations based on non-linear transformations. [25] developed a system for detection of human stress and emotions based on TEO and LFPC.[26] also developed a set of features derived from TEO for stress classification. [27] proposed a feature set called Teager energy cepstrum coefficients (TECC) that use TEO and a constant Q-gammatone filter-bank.

The key advantage of TEO that lead to their extended use in stress and emotion classification, is their potential to reflect the nonlinear airflow structure of speech production under stressful conditions.

3.3. MFCC AND T-MFCC FEATURE PARAMETERS

TEMFCC are motivated by MFCC and T-MFCC. MFCC and T-MFCC mimic the human perception process that responds with better frequency resolution to lower frequency range and relatively low frequency resolution in high frequency range.

MFCC

The computation of MFCC requires the following steps:

1) The speech waveform is first windowed with an analysis window $w(n)$, and the STFT, $S(\hat{n},\omega_k)$ is computed:

$$S(\hat{n},\omega_k) = \sum_{m=-\infty}^{\infty} s(m)\, w(\hat{n}-m)\, e^{-j\omega_k m} \tag{3.5}$$

where $\omega_k = \dfrac{2\pi}{N} k$ with N the DFT length.

2) The magnitude of $S(\hat{n},\omega_k)$ is then weighted by a mel filter bank $V_l(\omega_k)$. This filter bank is composed by a series of filter frequency responses whose center frequencies and bandwidths roughly match those of the auditory critical band filters. That is the so-called mel frequency wrapping. An example of such a filter is shown in Figure 3.1.

3) The next step in determining the mel-cepstrum is to compute the energy in the STFT weighted by each mel-scale filter frequency response. The result energies are given for each speech frame at time n and for *lth* mel-scale filter, $V_l(\omega_k)$, $l = 1, \ldots, L$, as

$$MC(\hat{n},l) = \frac{1}{A_l} \sum_{k=L_l}^{U_l} |\, V_l(\omega_k) S(\hat{n},\omega_k) \,|^2$$

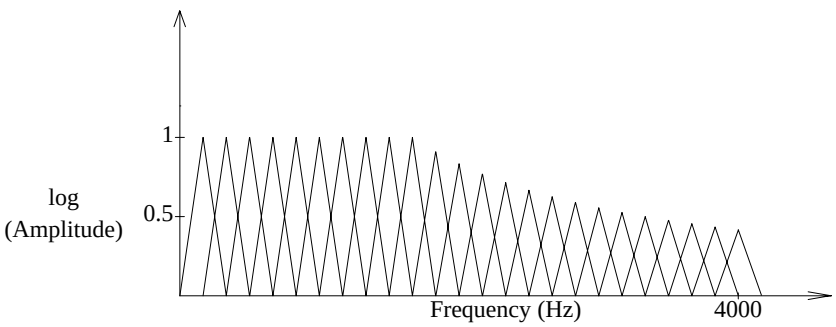

Figure 3.1: Triangular mel-scale filter bank with 24 filters following the mel-scale in the range [0-400] Hz

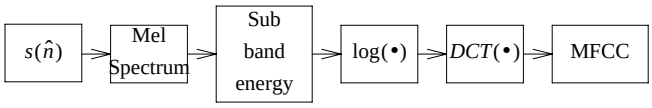

Figure 3.2: Flowchart diagram for the computation of MFCC. $s(\hat{n})$ is a frame from the pre-processed speech signal $s(n)$.

where L is the total number of filters, and L_l, U_l denote the lower and upper frequency indices respectively over which each filter is nonzero, and

$$A_l = \sum_{k=L_l}^{U_l} |V_l(\omega_k)|^2$$

is a normalizing factor for the *lth* mel-filter. This factor is needed so that for a perfectly flat input Fourier spectrum will produce a flat mel-cepstrum.

4) For each frame, a discrete cosine transform of the log of the magnitude of the filter outputs is computed to form the function $MFCC(\hat{n},k)$, i.e.:

$$\underset{k=1,2,\ldots,N_c}{MFCC(\hat{n},k)} = \frac{1}{R}\sum_{l=1}^{L}\log(MC(\hat{n},l))\cos\left[\frac{k(l-0.5)}{L}\pi\right]$$

Figure 3.2 is a flowchart diagram describing the computation of MFCC.

T-MFCC

T-MFCC feature parameters were developed for Language Identification (LID). Language identification refers to the task of identifying an unknown language from the test utterances. The computation of T-MFCC requires the following steps:

1) The speech signal $s(n)$ is first passed through a pre-processing stage, which includes frame blocking, hamming windowing with an analysis window $w(n)$, and pre-emphasis, to give the pre-processed speech signal $s(\hat{n})$.

2) Next we calculate the Teager energy of $s(\hat{n})$:

$$\Psi_R(s(\hat{n})) = s^2(\hat{n}) - s(\hat{n}-1)s(\hat{n}+1)$$

3) The magnitude spectrum of T-MFCC is computed and wraped to Mel frequency scale followed by log compression and DCT computation:

$$\underset{k=1,2,\ldots,N_c}{T\text{-}MFCC}(\hat{n},k) = \sum_{l=1}^{L}\log(\Psi_1(l))\cos\left[\frac{k(l-0.5)}{L}\pi\right]$$

where $\Psi_1(l)$ is the filterbank output of $F\{\Psi_R(s(\hat{n}))\}$ and $T\text{-}MFCC(\hat{n},k)$ is the k^{th} T-MFCC.

Figure 3.3 is a flowchart diagram describing the computation of T-MFCC.

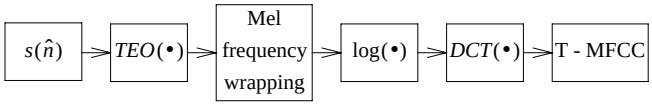

Figure 3.3: Flowchart diagram for the computation of T-MFCC. $s(\hat{n})$ is a frame from the pre-processed speech signal $s(n)$.

3.4. TEMFCC FEATURE PARAMETERS

The first step in the computation of TEMFCC is the same as in the computation of MFCC and T-MFCC. After the first step we come up the signal $S(\hat{n},\omega)$, described in Equation (3.5).

In the next step, the TEO of $S(\hat{n},\omega_k)$, $\Psi_C(S(\hat{n},\omega_k))$, is computed. Because $S(\hat{n},\omega_k)$ is complex we use Equation (3.4) for the computation of TEO. The magnitude of $\Psi_C(S(\hat{n},\omega_k))$, $|\Psi_C(S(\hat{n},\omega_k))|$, is then weighted (multiplied in frequency domain) by a filter bank $V_l(\omega_k)$ following the mel-scale (see Figure 3.1).

Then we compute the energy in $\Psi_C(S(\hat{n},\omega_k))$ weighted by each mel-scale filter frequency response. The resulting energies are given for each speech frame at time n and for *lth* mel-scale filter, $V_l(\omega_k)$, with $(l = 1, \ldots, L)$ as

$$e(\hat{n},l) = \sum_{k=L_l}^{U_l} |V_l(\omega_k)\Psi_C(S(\hat{n},\omega_k))| \tag{3.6}$$

where L is the total number of filters, and L_l, U_l denote the lower and upper frequency indices respectively over which each filter is non zero.

At the last step, the discrete cosine transform (DCT) of the log magnitude of the filter outputs for each frame is computed to form the $TEMFCC(\hat{n},k)$, i.e.,

$$\underset{k=1,\ldots,N_c}{TEMFCC}(\hat{n},k) = \frac{1}{R}\sum_{l=1}^{R}\log(e(\hat{n},l))\cos\left[\frac{k(l-0.5)}{L}\right] \tag{3.7}$$

The first 12 $TEMFCC(\hat{n},k)$, $k = 1, \ldots, 12$, coefficients are used in the feature vector. The first and second order differentials could be appended. Figure 3.4 shows a flowchart diagram for the computation of TEMFCC.

3.5. DIFFERENCES AMONG TEMFCC, MFCC AND T-MFCC

As mentioned earlier, TEMFCC are motivated by MFCC and T-MFCC. The main differences among them are in the energy measure used in Equation 3.6. TEMFCC use Teager energy of STFT of the input signal, $\Psi_C(S(\hat{n},\omega_k))$, MFCC use the STFT of the input signal, $S(\hat{n},\omega_k)$, and T-MFCC use Teager energy in time domain to determine the spectrum.

3.6. EXPERIMENTAL RESULTS

In this section, we explore the robustness of the proposed features and their combination with other spectral features. We artificially inject two types of noise to the speech signal and then compute their recognition accuracy through the system depicted in Figure 3.5. Also, we compare their performance to that of MFCC and T-MFCC.

3.6.1. Speech Data Corpus

We have created the ''EmoDB+Noise'' database by adding pink and white noise to the test set of EmoDB database (Figure 3.6).

3.6.2. Feature Extraction

In detail, the classification of speech into emotional classes is frame based. Every speech signal is divided into frames, windowed with an analysis window $w(n-m)$, $m = 1, \ldots, M$. The analysis frame (window) duration is 30ms and the frame increment 15ms. Every frame is then represented by three feature vectors:

- MFCC with 12 coefficients
- TEMFCC with 12 coefficients
- T-MFCC with 12 coefficients

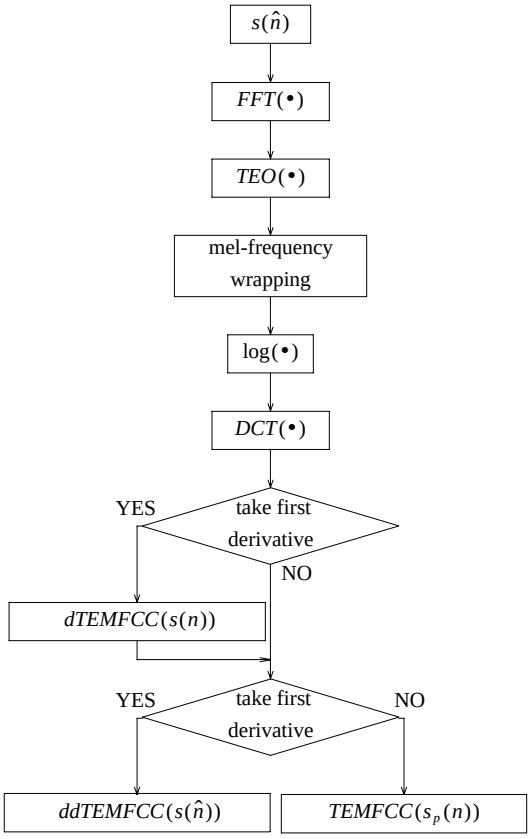

Figure 3.4: Flowchart diagram describing the computation of TEMFCC.

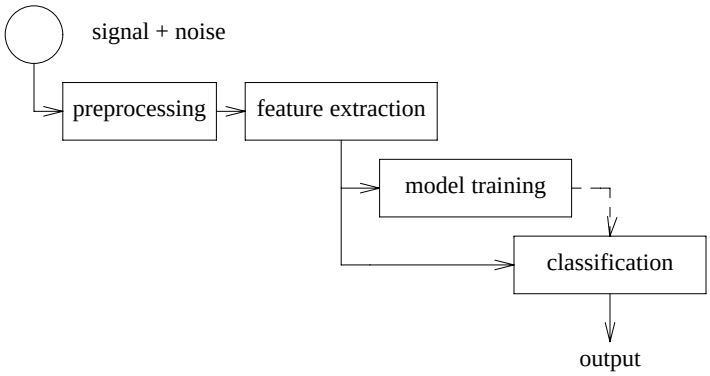

Figure 3.5: Generic block diagram of the classification system used to explore the robustness of new feature set.

excluding the zeroth cepstrum coefficient c_0 and augmented with their 1^{st} and 2^{nd} time derivatives. Therefore, feature vectors belong to space R^{36}.

One of the key advantages of using differential parameters, such as delta cepstrum or delta-delta cepstrum, is that the differencing operation removes the effect of simple linear filtering on the parameter values, thereby making them less sensitive to channel shaping effects that might occur in a speech communication system.

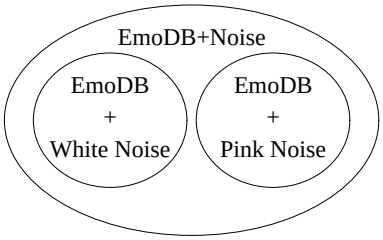

Figure 3.6: The ‘‘EmoDB+Noise’’ database consists of two sub-databases; ‘‘EmoDB+White Noise’’ and ‘‘EmoDB+Pink Noise’’ databases where samples were distorted with white and pink noise respectively at several SNR levels.

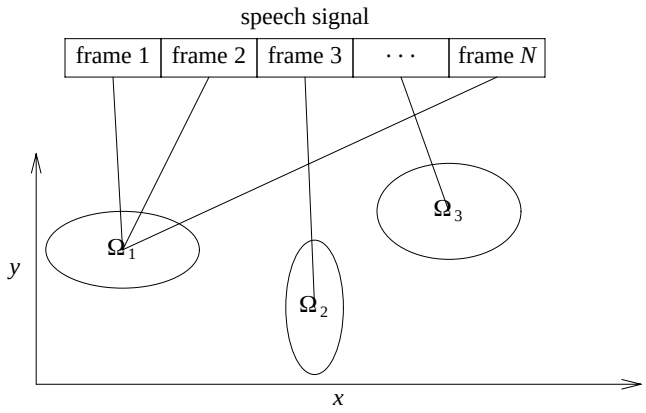

Figure 3.7: The classification of signals is frame based. The emotional class where the signal belongs is the class where the majority of its frames belongs to. In this Figure the final decision is Ω_1 class.

These frames are then classified into emotional states according to the maximum a posteriori probability (MAP) rule and the emotional class where the test signal belongs is the class where the majority of its frames belongs to. For example in Figure 3.7 the majority of frames belongs to Ω_1 class, so the final decision for the test signal is class Ω_1.

3.7. MODEL TRAINING

Gaussian mixtures models (GMM) were used to estimate the probability density function (pdf) of feature vectors in each emotional state. One problem we are faced when using GMM for classification is how to choose the number of mixture components *M*. The CLUSTER software package has been used to automatically estimate model parameters from feature vectors representing speech frames. CLUSTER is an unsupervised algorithm for GMM that is based on the expectation maximization algorithm (EM) and the minimum description length (MDL) criterion. This process is essentially similar to conventional clustering except that it allows cluster parameters to be accurately estimated even when the clusters overlap significantly. For more details see the original paper [29]*.

3.8. ANALYSIS OF RESULTS

In summary, we have applied TEMFCC, MFCC, and T-MFCC features to the ‘‘EmoDB+Noise’’ database. The recognition rate is calculated by 5-fold cross-validation, where 80% of data where used for training and 20% for validation (testing). When using the *k*-fold method, the training dataset is randomly partitioned into *k* groups. The learning algorithm is then trained *k* times, using all of the training set data points except those in the k^{th} group. The form of the algorithm is as follows:

---

* also refer to https://engineering.purdue.edu/~bouman/software/cluster/ for an implementation of the algorithm in C and full description of theory

TABLE 3.1
Recognition rates of MFCC, TEMFCC, and T-MFCC for various SNR levels under white noise.

	White Noise		
SNR_{dB}	MFCC	TEMFCC	T-MFCC
5	0.18	0.32	0.18
15	0.36	0.43	0.26
25	0.53	0.59	0.42
35	0.74	0.71	0.59
45	0.74	0.73	0.62
55	0.76	0.72	0.63

TABLE 3.2
Recognition rates of MFCC, TEMFCC, and T-MFCC for various SNR levels under pink noise.

	Pink Noise		
SNR_{dB}	MFCC	TEMFCC	T-MFCC
5	0.26	0.29	0.26
15	0.45	0.49	0.44
25	0.70	0.70	0.57
35	0.80	0.78	0.63
45	0.79	0.77	0.67
55	0.82	0.77	0.65

- Divide the training set into *k* partitions.
- For each *k*:
  - Make T the dataset that contains all training data points except those in the k^{th} group.
  - Train the algorithm using T as the training set.
  - Test the trained algorithm, using the k^{th} set as the test set. Record the number of errors.
- Report the mean error over all *k* test sets.

3.8.1. TEMFCC vs MFCC vs T-MFCC

Tables 3.1 and 3.2 presents the recognition results. In case of white noise and SNR values lower than 30dB, TEMFCC have the best performance. As SNR values increase, MFCC better performance. T-MFCC have the overall lowest recognition rate in case of white noise. In case of pink noise the results are slightly different, i.e., TEMFCC have the best performance for values lower than 25dB, while MFCC perform better than TEMFCC and T-MFCC as SNR values increase. Again, T-MFCC have the overall lowest recognition rate in the range [0dB, 55dB].

The improvement by using TEMFCC over MFCC, for white noise, in the range [5dB,30dB] varies from ~43% to ~10%. The improvement by using TEMFCC over MFCC, for pink noise, in the range [5dB,20dB] varies from ~10% to ~8%. A graphical representation of tables 3.1 and 3.2 is shown in Figure 3.8.

3.8.2. Mean and Max Fixed Combining Rules

One possible way to improve the classification performance of a system is to combine the results of individuals classifiers under fixed combining rules. The fixed combining rules make use of the fact that some classifiers are able to output not just labels or numbers, but also confidences.

Consider a set of different classifiers e_k, $k = 1, \ldots, K$, and a set of classes ω_j, $j = 1, \ldots, J$. In our case, the confidence $P_k^j(x)$ of sample x with respect to class ω_j is defined as $P_k^j(x) = P_k(\omega_j/x)$, where the subscript k refers to classifier e_k. Every classifier k is able to output a decision for the test sample x in the

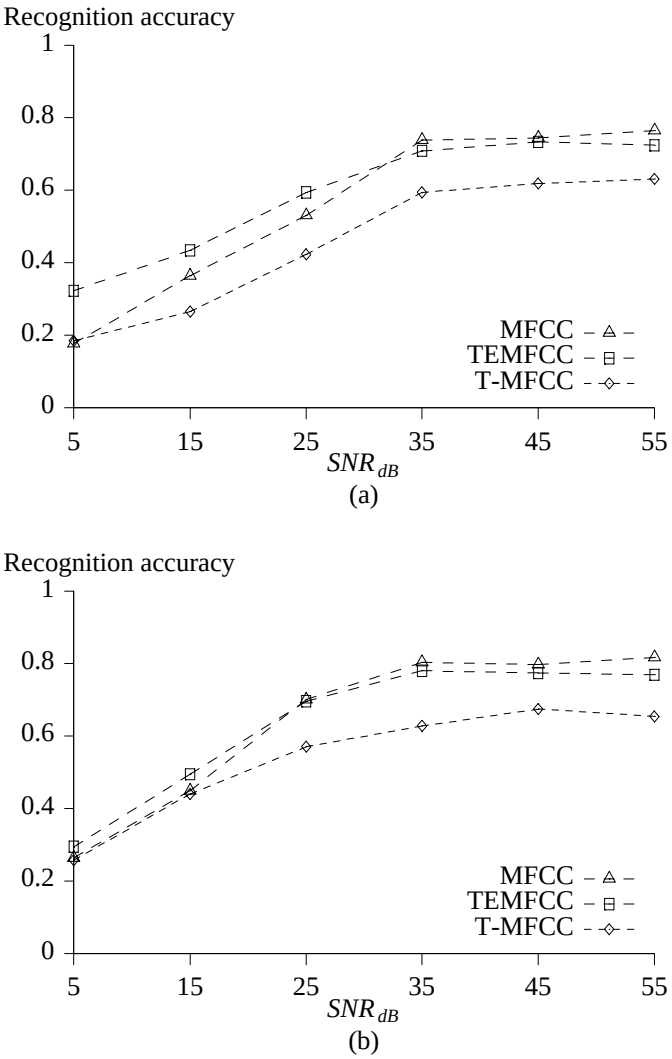

Figure 3.8: Graphical comparison of performance between MFCC and TEMFCC with the presence of (a) white and (b) pink noise for SNR values in the range [5,55].

form $P_k^j(x), j = 1, \ldots, J$. The following average value can be used as a new estimation of combined classifier E_{mean}:

$$P_{E_{mean}}(\omega_j/x) = \frac{1}{K}\sum_{k=1}^{K} P_k(\omega_j/x), \quad j = 1, \ldots, J \tag{3.8}$$

The final decision made by E_{mean} is:

$$E_{mean}(x) = j, \quad \text{for which} \quad P_{E_{mean}}(\omega_j/x) = \underset{i=\{1,\ldots,J\}}{\operatorname{argmax}} P_{E_{mean}}(\omega_i/x) \tag{3.9}$$

In case of max rule the following probabilities must be computed in order to take the final decision:

$$P_{E_{\max}}(\omega_j/x) = \underset{\{k=1,\ldots,K\}}{\operatorname{argmax}} P_k(\omega_j/x), \quad j = 1, \ldots, J \tag{3.10}$$

Then the final decision of combined classifiers $E_{\max}$ has the form:

$$E_{\max}(x) = j, \text{for which} \quad P_{E_{\max}}(\omega_j/x) = \underset{\{i=1,\ldots,J\}}{\operatorname{argmax}} P_{E_{\max}}(\omega_i/x) \tag{3.11}$$

TABLE 3.3
Feature vectors used to train each classifiers.

e_1	e_2	e_3
TEMFCC	MFCC	T-MFCC

3.8.3. Combining TEMFCC,MFCC, and T-MFCC under Mean and Max Combining Rules

We combine the results of classifiers e_1, e_2, and e_3 under the fixed combining rules (3.9), (3.11), in different noise environments. For this purpose we designed the system depicted in Figure 3.9.

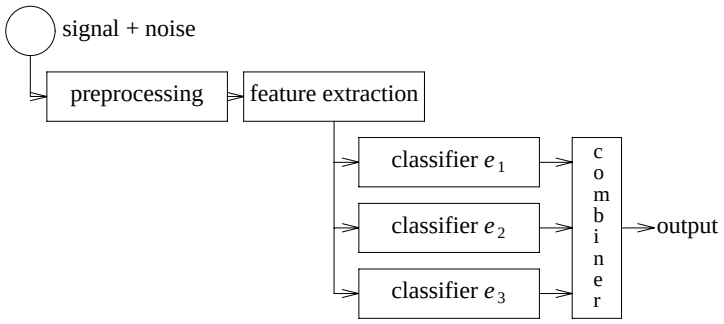

Figure 3.9: Generic block diagram used to combine three different classifiers

The first environment is white additive gaussian noise and the second pink additive noise. Every classifier is based on GMM and is trained with different feature vector. Table 3.3 shows the feature vector used to train each classifier. The following cases of combinations remain the same for both white and pink noise environments:

1^{st} case: e_1, e_2, and e_3

2^{nd} case: e_1 and e_2

3^{rd} case: e_1 and e_3

Table 3.4 shows the above combinations.

TABLE 3.4
Combinations between classifiers.

Case	Combination
1^{st}	e_1 combined with e_2 and e_3
2^{nd}	e_1 combined with e_2
3^{rd}	e_1 combined with e_3

3.8.4. Results of Combinations

In both environments, the recognition accuracy of mean and max combiners is the same. The results of combinations are shown in Tables 3.5 and 3.6. Figures 3.10(a), 3.10(b), 3.10(c), and Figures 3.11(a), 3.11(b), 3.11(c) are graphical representations of Tables 3.5 and 3.6 respectively.

White noise

In the range [5dB, 15dB] e_1 has the best performance. At 25dB performances of e_1, (e_1,e_2) and (e_1,e_2,e_3) is the same and also are the greatest. For values grater than 30dB, e_2 has the best performance. For SNR values 35dB and 55dB, e_2 has the greater recognition accuracy. At 55dB combination (e_1,e_2) has the greater accuracy.

TABLE 3.5
Recognition rates for the combinations of classifiers in case of white noise.

White noise						
SNR_{dB}	mean $(e_1,e_2,e_{3)}$	max $(e_1,e_2,e_{3)}$	mean (e_1,e_2)	max (e_1,e_2)	mean (e_1,e_3)	max (e_1,e_3)
5	0.22	0.22	0.24	0.24	0.28	0.28
15	0.37	0.37	0.39	0.39	0.41	0.41
25	0.59	0.59	0.59	0.59	0.57	0.57
35	0.72	0.72	0.72	0.72	0.71	0.71
45	0.73	0.73	0.77	0.77	0.72	0.72
55	0.74	0.74	0.75	0.75	0.71	0.71

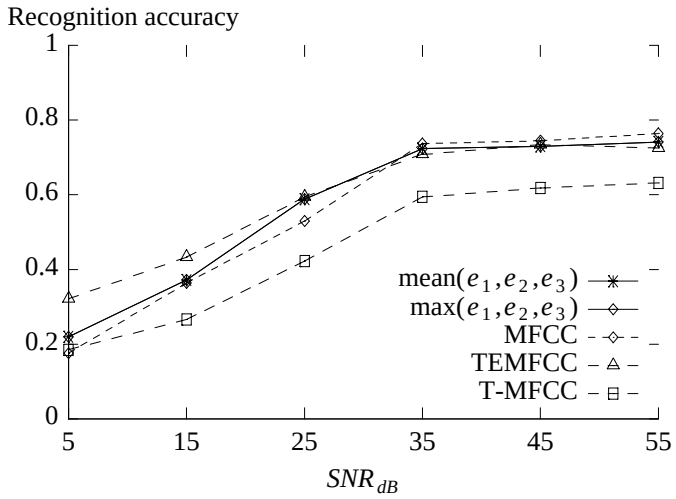

Figure 3.10(a): Recognition rates for the combinations of classifiers e_1,e_2,e_3in the presence of white noise.

Pink noise

For the case of pink noise the results are slightly different. In the range [5dB, 15dB] the combination (e_1,e_3) has the best performance. For 25dB the performance of e_1 and e_2 is the best. For values grater than 30dB e_1 has the best performance. It's worth mentioning that in the range [45dB,55dB] combination of (e_1,e_2) is greater than (e_1,e_2,e_3). e_3 's performance decreases the recognition accuracy of (e_1,e_2,e_3) because e_3 has the lowest recognition rate among all individual classifiers.

An important requirement for combiners comes out from the analysis of the above results: individual classifiers should not be strongly correlated in their misclassification. That is, classifiers should not ''agree'' with each other when they misclassify a sample, or at least should not assign the same incorrect class to a sample. If this requirement holds, combiners perform better than the best individual classifier. This happens in white noise, for the combination (e_1,e_3) and SNR value equal to 45dB. This requirement can be satisfied to a certain extend by[30]:

(1) using different feature vector representations for the samples

(2) using different classification principles for each individual classifier.

Using different representations (feature sets) leads, in many cases, to a reduction in the correlation between the outputs of individual classifiers, since there is almost always less correlation between the input vectors using different representations than when using the same set of features. Different classifiers usually use different assumptions about the structure of the data and the stochastic model that generates it. This leads to a different estimate of the a posteriori probabilities especially around the Bayes decision boundaries.

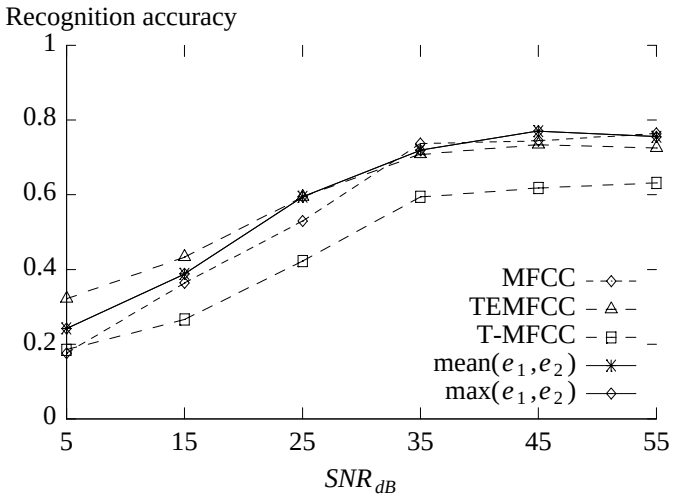

Figure 3.10(b): Recognition rates for the combinations of classifiers e_1, e_2 in the presence of white noise.

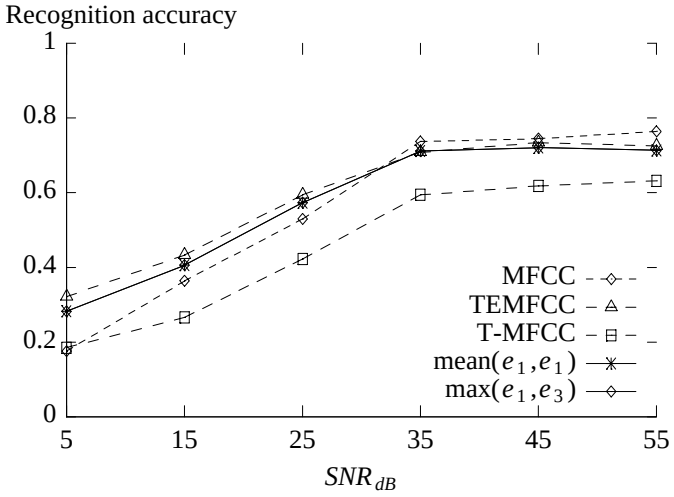

Figure 3.10(c): Recognition rates for the combinations of classifiers e_1, e_3 in the presence of white noise.

TABLE 3.6
Recognition rates for the combinations of classifiers in case of pink noise.

Pink noise						
SNR_{dB}	mean (e_1,e_2,e_3)	max (e_1,e_2,e_3)	mean (e_1,e_2)	max (e_1,e_2)	mean (e_1,e_3)	max (e_1,e_3)
5	0.25	0.25	0.25	0.25	0.26	0.26
15	0.47	0.47	0.44	0.44	0.49	0.49
25	0.68	0.68	0.69	0.69	0.63	0.63
35	0.73	0.73	0.72	0.72	0.69	0.69
45	0.76	0.76	0.77	0.77	0.71	0.71
55	0.72	0.72	0.75	0.75	0.70	0.70

3.9. CONCLUSIONS

In this Chapter we addressed the implementation of an automatic emotion-state recognition system capable of working in noise environments, using cepstral features extracted from an audio signal. The experiments were carried out using the EmoDB speech corpus. A new feature representation based on TEO was intro-

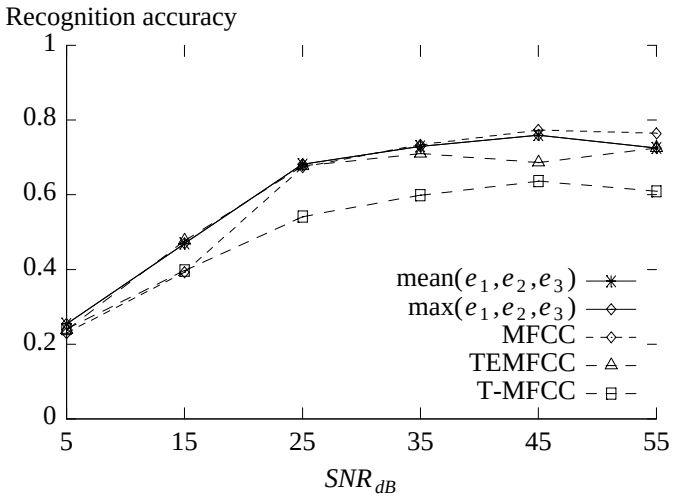

Figure 3.11(a): Recognition rates for the combinations of classifiers e_1, e_2, e_3 in the presence of pink noise.

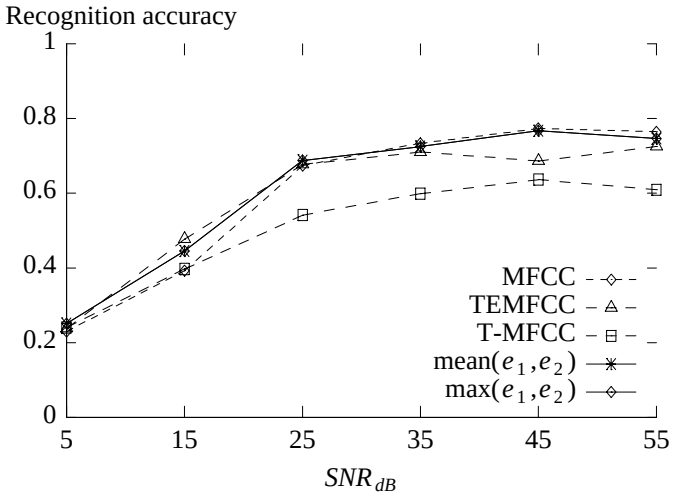

Figure 3.11(b): Recognition rates for the combinations of classifiers e_1, e_2 in the presence of pink noise.

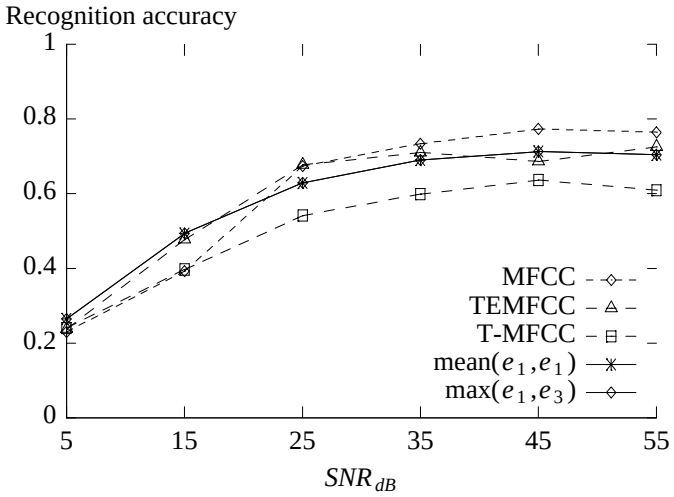

Figure 3.11(c): Recognition rates for the combinations of classifiers e_1, e_3 in the presence of pink noise.

duced and combined with existing features (MFCC and T-MFCC) made possible the enhancement of the overall performance of system in the presence of white and pink noise. Also, combinations between different spectral features showed that we can improve the recognition accuracy of a classification. A necessary requirement for the combined classifiers is that they must be uncorrelated in their misclassification. The requirement for ‘‘disagreement independence’’ among classifiers is fulfilled in the experiments of CHAPTER 7.

CHAPTER 4

EMOTION SPEECH CLASSIFICATION AND FEATURE SELECTION USING PROSODIC, ENERGY, AND SPECTRAL FEATURES

In Chapter 2, we presented many features for speech emotion recognition. However, there is no agreement on a fixed set of features although many of them are common in the majority of researches and perform remarkably well. In this study, we compare the discriminatory capability of sets with prosodic, energy and spectral features created from five different feature selection methods and using these sets we evaluate the performance of several classifiers and their combinations.

4.1. FEATURE EXTRACTION

The primary features extracted from each speech sample are the pitch, speech rate, energy, and first three formant frequencies. Pitch, speech rate and formants frequencies belong to prosodic and spectral features respectively. The pitch contour is derived by applying the robust algorithm for pitch tracking (RAPT) described in [11]. The RAPT fundamental frequency estimation algorithm uses the normalized cross correlation function and dynamic programming. The speech rate is calculated as the inverse duration of the voiced part of speech determined by the presence of speech samples [16]. For estimating the 3 formant contours we use the method proposed in [31]. The formant frequencies are computed by peak-picking the linear predictive coding (LPC) spectrum. To get accurate estimates of the formant frequencies, one needs to choose the LPC order properly depending on the sampling frequency. In our case, we had a sampling frequency F_s of 16000kHz and the LPC order chosen was 16. To estimate the energy contour, a simple short-term energy function has been used. All the aforementioned feature, were extracted from speech frames of duration 30ms.

The trends of contours from the aforementioned features, i.e., falling and rising slopes, plateaux at minima and maxima, contain valuable information about the emotional states. These trends can be computed using the first derivative of the contour according to the algorithm shown in Figure 4.1. In order to compute the above sets for a function $g(n)$, the first derivative of the smoothed contour, $\hat{g}(n)$, is computed from the finite difference $\Delta\hat{g}(n)=\hat{g}(n+1)-\hat{g}(n)$. The smoothing process can be done by using the moving average algorithm. Let S_f, S_r be the subset of domain points at the rising and falling slopes of the contour respectively, and S_{mi}, S_{ma} the subset of domain points of plateaux at minima and maxima.

u_1 = 10% $max(\Delta\hat{g}(n))$
u_2 = 50% $max(\hat{g}(n))$

```
if (|Δĝ(n)| ≦ u1 ) {
  if ( ĝ(n) < u2 ) { n∈ Smi }
  else { n∈ Sma }
}
else if ( Δĝ(n) < −u1 ) { n∈ Sf }
else if ( Δĝ(n) > u1 ) { n∈ Sr }
```

Figure 4.1: Algorithm for detecting plateaus and slopes of a smoothed contour $\hat{g}(n)$.

In this algorithm, u_1 is a threshold that allows the distinction between plateaus and slopes. The distinction between plateaus at minima and maxima is done using threshold u_2.

4.1.1. Fundamental Frequency Features

We obtain the following statistics from the pitch contour of speech samples. The pitch contour is smoothed using a moving average filter with window width 5.

[1-6] mean, median, min, max, range, interquartile range of pitch values

[7-10] mean, median, min, max value of rising slopes

[11-14] mean, median, min, max value of falling slopes

[15-18] mean, median, min, max value of plateaus at minima

[19-22] mean, median, min, max value of plateaus at maxima

[23-26] mean, median, min, max duration of rising slopes

[27-30] mean, median, min, max duration of falling slopes

[31-34] mean, median, min, max duration of plateaus at minima

[35-38] mean, median, min, max duration of plateaus at maxima

[39] speech rate

4.1.2. Energy Features

Energy features are statistical properties of the energy contour. We calculate the following energy features.

[40-45] mean, median, min, max, range, interquartile range of energy values

[46-49] mean, median, min, max value of rising slopes

[50-53] mean, median, min, max value of falling slopes

[54-57] mean, median, min, max value of plateaus at minima

[58-61] mean, median, min, max value of plateaus at maxima

[62-65] mean, median, min, max duration of rising slopes

[66-69] mean, median, min, max duration of falling slopes

[70-73] mean, median, min, max duration of plateaus at minima

[74-77] mean, median, min, max duration of plateaus at maxima

4.1.3. Spectral Features

The set of spectral features contains statistical properties of the first three formant frequencies.

[78-83] mean, median, min, max, range and interquartile range of first formant

[84-89] mean, median, min, max, range and interquartile range of second formant

[90-95] mean, median, min, max, range and interquartile range of third formant

From now on all features will be referenced by their corresponding indices.

4.2. CLASSIFICATION APPROACH

In contrast with the frame-based classification method followed in Chapter 3, here we follow an utterance based classification approach. Remember that when we decide in a frame basis, the final label for the utterance is taken to be the majority of its' frame labels.

To make it more clear, suppose that $s(n)$ is the speech signal. In Chapter 3, $s(n)$ was partitioned in short frames $s_f(n;m) = s(n)\,w(m-n)$, where $w(n-m)$ is a window of length N_w ending at sample m [32]. Each frame was represented by a feature vector $\vec{x}_i$ with the cepstral coefficients. The classifier was trained using $\vec{x}_i$'s, where i is the frame index, and the final decision of classifier is about frame $s_f(n)$. In order to make the final decision about the class label of $s(n)$ we look at the class labels of its frames, $s_f(n)$ and decide using simple majority vote.

On the other hand, in this Chapter each frame is represented by a real number x_i, e.g., the short-time frame energy or fundamental frequency value for that frame, and the classifier is trained using statistics taken over all N frames x_i, $i = 1, \ldots, N$. It's input is of the form:

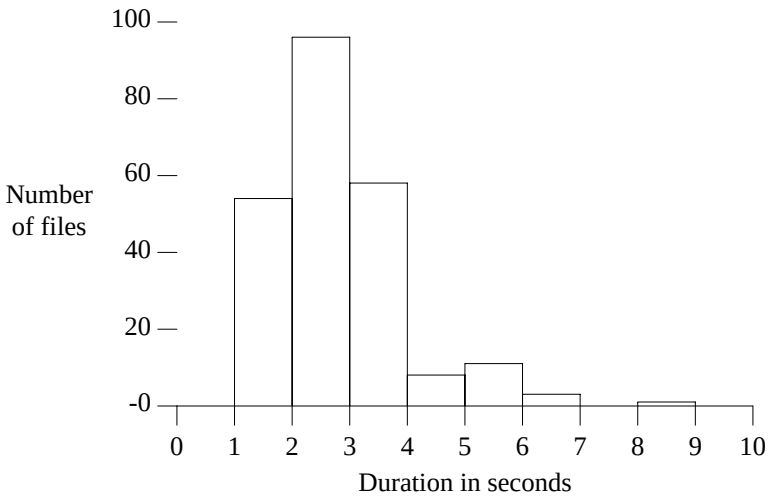

Figure 4.2: Histogram for the duration of samples in out database.

$$f(X),\ f:R^N \rightarrow R,\quad X = \begin{bmatrix} x_1 \\ . \\ . \\ . \\ x_N \end{bmatrix}$$

where f is a statistic, like mean or variance, and X a vector composed with the feature values x_i that represents the frames of speech signal $s(n)$.

The number of frames N is a varying parameter. In our case, N is equal to the total number of frames in speech signal. Our choice was influenced by the fact that mean duration of speech signals in our database is 2.8s. We assumed that 2.8s is a typical duration for utterances and during this period the emotional state of a speaker remains the same. In Figure 4.2 we show histogram of the durations of speech samples in our database. If the speech utterances have larger duration, one should partition the speech signal in smaller parts that represent utterances and decide about their class labels. In that case, the final decision for a phrase $p = \{u_1 \cup \cdots \cup u_N\}$ is a simple majority rule of the utterance labels u_j, $j = 1, \ldots, N$ that is decomposed, where N is the number of utterances.

4.3. ANALYSIS OF SINGLE FEATURES

The classification performance of each feature in isolation is rated according to the probability of correct classification achieved by the classifier in use. We test five classifiers; Naive Bayes, fuzzy *k*-NN, Linear Discriminant classifier, and their combinations under the mean and max combining rules. Mean and max combiners have a generic block diagram as this shown in Figure 3.5. The decision logic of each rule is implemented in the combiner block.
All the above classifiers are able to output information in the measurement level; every classifier e assigns to each class C_i from the set $\{C_1, \ldots, C_M\}$, where M is the total number of classes, a value to measure the degree that sample x comes from that class. In other words they are able to output probabilities of the form $P(C_i/x)$.

Naive Bayes classifier is a simple probabilistic classifier based on applying Bayes's theorem (from Bayesian statistics) with strong (naive) independence assumptions. In case of naive bayes classifier, we apply kernel density estimation to estimate the pdf of each feature using gaussian kernels. The bandwidth h of kernel density estimation for naive Bayes was chosen to be $h = (4\hat{\sigma}^5/3N)^{1/5}$, where N is the number of samples and $\hat{\sigma}$ the sample standard deviation of the samples. The main difference between *k*-NN and fuzzy *k*-NN is that fuzzy *k*-NN assigns class membership to a sample vector x rather than assigning the vector to a particular class. Also the assigned memberships of x are influence by the inverse of the distances from the

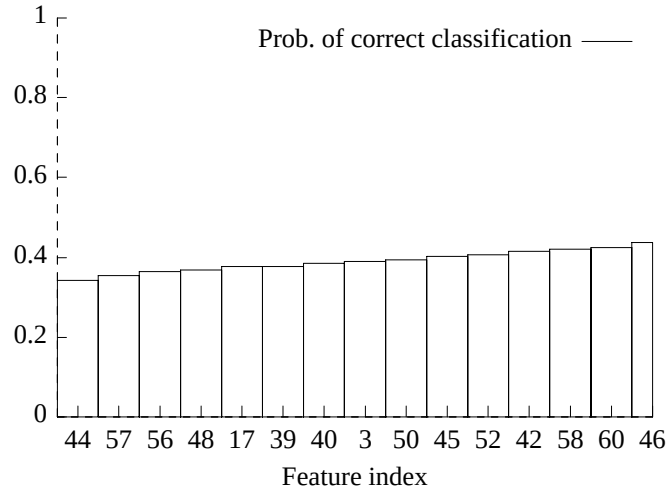

Figure 4.3: Single feature evaluation using as criterion the probability of correct classification fuzzy 3-NN classifier achieves. Sorting is in ascending order.

nearest neighbors and their class memberships [33]. Linear discriminant classifier fits multivariate gaussian distributions $p(x \mid C_k)$ with diagonal covariance matrix estimate and classifies each sample according to Bayes posterior probabilities $P(C_k \mid x) = \frac{p(x \mid C_k)p(C_k)}{p(x)}$.

Let e_n, $n = 1, \ldots, N$, be the set of classifiers, C_i, $i = 1, \ldots, M$, the set of classes, and E the combiner. Also let $P_n(C_i/x)$ be the output of classifier n. Mean combiner calculates the following probabilities $P_E(C_i/x) = 1/N \sum_{n=1}^{N} P_n(C_i/x)$ and takes his final decision $E(x)$ as $E(x) = \text{argmax}_{i=1,\ldots,M} P_E(C_i/x)$. Another alternative is to use the median rule. Median combiner calculates the median values of $P_n(C_i/x)$, $i = 1,\ldots,M$, $P_E(C_i \mid x) = median\{P_1(C_i \mid x), P_2(C_i \mid x), \ldots, P_N(C_i \mid x)\}$, and takes the decision according to $E(x) = \text{argmax}_{i=1,\ldots,M} P_E(C_i/x)$.

The probability of correct classification rate was estimated using 10 fold cross-validation. The first 15 features with the highest recognition rates in case of fuzzy 3-NN classifier are (we reference them with their indices): 44, 57, 56, 48, 17, 39, 40, 3, 50, 45, 52, 42, 58, 60,and 46. In Figure 4.3, the features are sorted in ascending order according to the probability of correct classification.

In the case of the Naive Bayes classifier, the first 15 features with highest recognition rates are: 45, 46, 52, 39, 56, 9, 42, 48, 58, 61, 31,44, 60, 17, and 3. In Figure 4.4, the features are sorted in ascending order according to the probability of correct classification.

In the case of the linear discriminant classifier, the group of fifteen features with highest recognition accuracies are: 47, 56, 43,49, 31, 45, 52, 59, 48, 17, 46, 50, 60, 3, and 44. In Figure 4.5, the features are sorted in ascending order according to the probability of correct classification.

For the mean and median combining rules the best 15 features are 48, 51, 44, 39, 50, 3, 40, 56, 17, 45, 42, 52, 60, 58, 46 and 39, 49, 40, 50, 56, 3, 44, 45, 42, 31, 48, 46, 58, 60, 17 respectively. In Figures 4.6-4.7, the features are sorted in ascending order according to the probability of correct classification.

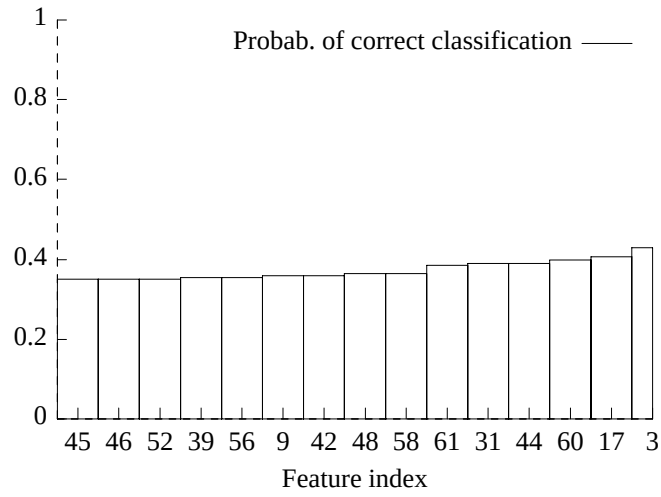

Figure 4.4: Single feature evaluation using as criterion the probability of correct classification Naive Bayes classifier achieves, when the pdf of each feature is modeled using normal kernel distribution. Sorting is in ascending order.

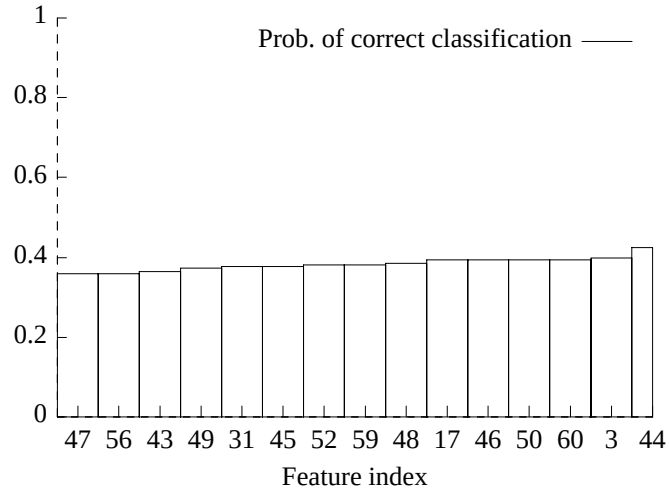

Figure 4.5: Single feature evaluation using as criterion the probability of correct classification a linear discriminant (LDC) classifier achieves. Sorting is in ascending order.

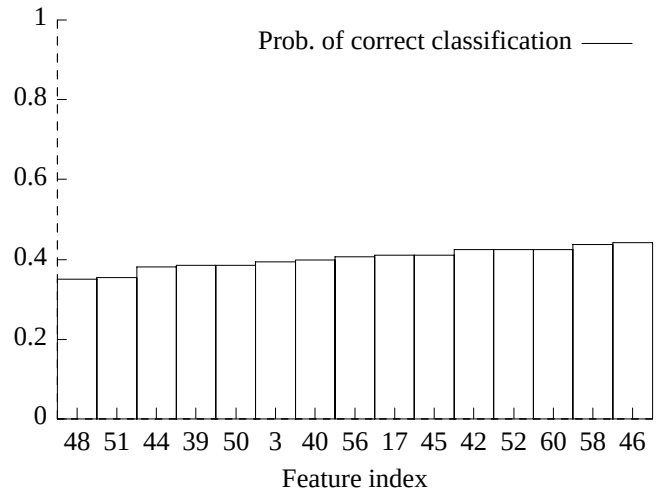

Figure 4.6: Single feature evaluation using as criterion the probability of correct classification the mean combiner of fuzzy *k*-NN, naive Bayes and LDC classifiers acheives. Sorting is in ascending order.

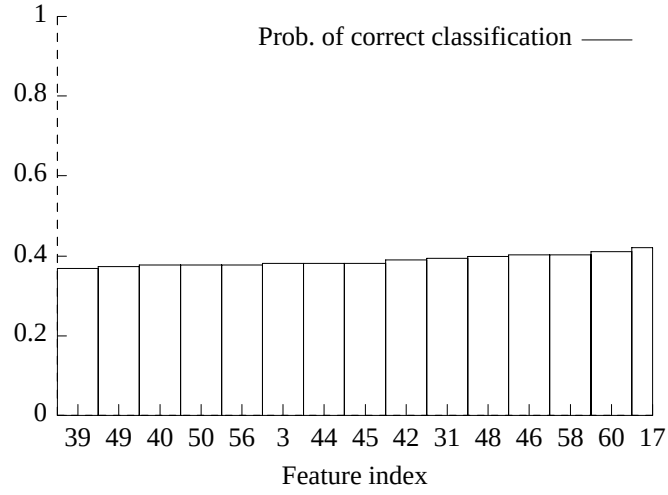

Figure 4.7: Single feature evaluation using as criterion the probability of correct classification the median combiner of fuzzy *k*-NN, naive Bayes and LDC classifiers acheives. Sorting is in ascending order.

Features with indices 3, 17, 44, 45, 46, 48, 52, 56, and 60 are common in the list of 15 best features for all classifiers, both individual and their combinations. Features 3 and 15 refer to statistics of the pitch contour while the remaining to statistics of the energy contour.

4.4. AUTOMATIC FEATURE SELECTION

The feature vector as described in Section 4.1 contains a lot of features, many of them probably redundant. The purpose of feature selection techniques is to select a subset of relevant features for building robust classifiers. By removing the most irrelevant and redundant features from the data, feature-selection helps improve the performance of learning models by:

- eliminating the effect of the curse of dimensionality.
- enhancing generalization capability.
- speeding up learning process.
- improving model interpretability.

Our aim is to reduce the original dimensionality of samples from R^{95} to R^{15}, i.e., select 15 features out from 95. For this purpose we examine several feature selection algorithms and train classifiers to test their recognition accuracy using the reduced feature sets.

There are three approaches for feature selection algorithms: filters, wrappers and hybrid. Filter approaches use general characteristics of data to select a subset of features according to a reasonable criterion that is independent of the problem. Wrapper approaches use estimated accuracy of a classifier to obtain feature subsets. Hybrid approaches try to utilize different evaluation criteria of two approaches in different search stages. We focus on filter-type feature-selection algorithms which have better generalization properties and can be computed easily and efficiently. The feature selection methods we examine are:

- Fischer's discriminant ratio (FDR)
- Relief-F
- three algorithms based on mutual information, but using different criteria:
  - mutual information based feature selection (MIFS)
  - maximum-relevance-minimum-redundancy (MRMR)
  - conditional mutual information maximization (CMIM)

In the following subsections we make a brief review of the examined feature-selection algorithms.

4.4.1. Fisher's Discriminant Ratio

FDR [34] is commonly employed to quantify the discriminatory powers of individual features between two equal probable classes and is independent of the type of class distribution.

Let m_1, m_2 and σ_1^2, σ_2^2 be the respective mean and variances values associated with a feature in the one dimensional, two-classes problem. The FDR for this feature is defined as:

$$FDR = \frac{(m_1 - m_2)^2}{\sigma_1^2 + \sigma_2^2}$$

The higher is the score, the more important becomes the feature. FDR can be extended easily for multi class problems. One possibility is

$$FDR_l = \sum_{i}^{M} \sum_{i \neq j}^{M} \frac{(m_i - m_j)^2}{\sigma_i^2 + \sigma_j^2}$$

where the subscripts *i, j* refer to the mean and variance corresponding to the feature *l* under investigation for the classes *i, j* respectively.

4.4.2. Relief-F

Relief-F [35] is a multi-class generalization of Relief [36] and assigns scores to features based on how well they separate training samples from their nearest neighbours belonging to their class and to opposite classes.

The algorithm constructs iteratively a weight vector for each feature, which is initially equal to zero. The number of iterations *m* is a user-defined parameter. At each iteration, selects one sample, adds to the weight the difference between that sample and the samples from the opposite classes in the *k* nearest neighbours and subtracts the difference between that sample and its nearest neighbours from the same class. A user defined parameters in Relief-F algorithm is the number *k* of nearest samples to examine. A good choice is to start with $k = 10$ and investigate the stability and reliability of Relief ranks and weights for various values of *k*.

4.4.3. Feature Selection with Mutual Information

A common approach in feature selection is to use the mutual information between the features and the class label. Let *X* (feature), and *Y* (label), be two random variables. Mutual information $I(X;Y)$, measures the

amount of information shared by X and Y. It can be shown that the Bayes error of predicting Y from X is lower-bounded by Fano's inequality [37], and upper-bounded by half the conditional entropy:

$$\frac{H(Y)-I(X;Y)-1}{\log(|Y|)} \leqq P(g(X)\neq Y) \leqq \frac{1}{2}H(Y|X) \tag{4.1}$$

where X, Y are vectors and $H(X)$, $H(X|Y)$, $I(X;Y)$ denote the entropy, conditional entropy and mutual information respectively, and $g(x)$ denotes the decision of the classifier.

To understand what $I(X;Y)$ actually means, we must first understand entropy. Qualitatively, entropy is a measure of uncertainty – the higher the entropy, the more uncertain one is about a random variable [38]. Information theory measures information content in bits. One bit of information is enough to answer a yes/no question about which one has no idea, such as the flip of a fair coin. If the possible answers to a question are v_i, $i=1,\ldots,n$, and have probabilities $P(v_i)$, then the entropy of the actual answer V is given by:

$$H(V)=-\sum_{i=1}^{n}P(v_i)\log_2 P(v_i), \quad V=\begin{bmatrix} v_1 \\ . \\ . \\ . \\ v_n \end{bmatrix} \tag{4.2}$$

In (4.2) the term $\log_2 P(v_i)$ denotes the amount of information associated with answer v_i and the average information content of the various answers is weighted by the probabilities of the answers. For example, for the case of tossing a fair coin where the possible answers are two, $v_1=$ ''head'' and $v_2=$ ''tail'', we get

$$H(V)=-\frac{1}{2}\log_2\frac{1}{2}-\frac{1}{2}\log_2\frac{1}{2}=1, \quad V=\begin{bmatrix} v_1 \\ v_2 \end{bmatrix}$$

which means that the content of information for the answer is 1 bit. If the coin is loaded to give 99% head and 1% tail we get $H(0.99,0.01)=0.08$ bit. As the probability of head goes to 1 the amount of information of a possible answer goes to 0.

The mutual information of two random variables, X,Y, is a quantity that measures the mutual dependence of the two variables. For discrete random variables is defined as

$$I(X;Y)=\sum_{i=1}^{N}\sum_{j=1}^{M}P_{XY}(x_i,y_j)\log\left[\frac{p(x_i,y_i)}{P_X(x_i)P_Y(y_i)}\right] \tag{4.3}$$

where P_{XY} is the join pdf of X and Y, and P_X, P_Y the marginal pdf's of X, Y respectively. Mutual information is related to entropy with the following equalities

$$I(X;Y)=H(X)-H(X|Y) \tag{4.4a}$$

$$=H(Y)-H(Y|X) \tag{4.4b}$$

Conditional entropy, $H(Y|X)$, is a measure of what X does not say about Y. Thus the right side of (4.4b) can be read as ''the amount of uncertainty in Y which is removed by knowing X''.

Returning to (4.1), we see that the first inequality states that for any function $g(X)$ of the inputs, the probability of error is lower bounded by an expression dependent on the mutual information. As the mutual information grows, the bound is minimized;whether or not the bound can reached depends on the ability of our classifier, i.e., the function $g(X)$. Our task is therefore to select k features from a pool of n, $\{X_1,X_2,\ldots,X_n\}$, such that their joint mutual information $I(X_{1,\ldots,n};Y)$ is maximized.

In order to know if we should include a feature, we must be able to compute the mutual informations $I(X_{1,\ldots,n};Y)$. The computation of $I(X_{1,\ldots,n};Y)$ is expensive and difficult. We could assume feature independence between X_i's and rank the features in descending order according to a criterion $J_n(X_n;Y)$. In general, it is widely recognised that a good set of features should not only be individually relevant, but also

should not be redundant with respect to each other;features should not be highly correlated. Among several criteria, we choose to apply the following:

MIFS Battiti [39] proposed the mutual information based feature selection criterion:

$$J_{MIFS} = I(X_n;Y) - \beta \sum_{k=1}^{n-1} I(X_n;X_k)$$

where β is an adjustable parameters, which must be set experimentally. Using $\beta = 0$ is equivalent to selecting features independently, while a larger value will place more emphasis on reducing inter-feature dependencies.

MRMR Peng [40] proposed the maximum-relevance-minimum-redundancy. Two different criteria are used to combine relevance and redundancy and lead to the selection of a new feature:

$$J_{MRMR} = I(X_n;Y) - \frac{1}{n-1} \sum_{k=1}^{n-1} I(X_n;X_k) \ ,$$

known also as mutual information difference criterion (MID) and

$$J_{MRMR} = \frac{I(X_n;Y)}{\frac{1}{n-1} \sum_{k=1}^{n-1} I(X_n;X_k)} \ ,$$

known also as mutual information quotient (MIQ). It can be seen that MRMR when using the MID criterion is a special case of MIFS with $\beta = 1/(n-1)$. We used the MIQ criterion.

CMIM Conditional mutual information maximization is probably the most well-known and was proposed by Fleuret [41]:

$$\begin{aligned} J_{CMIM} &= \min_k [I(X_n;Y \mid X_k)] \\ &= I(X_n;Y) - \max_k [I(X_n;X_k) - I(X_n;X_k \mid Y)] \end{aligned}$$

CMIM examines the information between a feature and the target, conditioned on each current feature.

4.5. EXPERIMENTAL RESULTS

A total of 95 features have been calculated from 231 speech utterances from EmoDB, to form the dataset *D* (see Talbe 2.2). We test the algorithms Fisher's score, Relief-F, MIFS, MRMR, and CMIM on *D* with the aim of selecting 15 features out of 95. With the feature sets selected by the aforementioned algorithms, we train and compare the performance of five classifiers: fuzzy *k*-NN, naive Bayes, LDC classifier and their combinations under mean and median combining rules, on the task of speech classification in five emotional states (see Table 2.2).

The probability of correct classification was estimated using 10 fold crossvalidation, where 10% of data were used for testing and 90% for training. The features selected by each algorithm are shown in Table 4.1. Each column in that table represents the feature set created by the algorithm whose name is written at the start of the column.

All algorithms revisited here except from FDR criterion, need extra user-defined parameters and tuning. Relif-F needs to be supplied as input with the number *k* of nearest samples to search for. After experimenting with values of *k* in the range $5 \leqq k \leqq 20$ we found that $k = 5$ leads to a feature set with the highest recognition accuracy for all classifiers. Feature-selection algorithms based on mutual information deal with continues variable in different ways. A possible solution is to discretize the continuous variables and that is the method we followed in order to deal with the continuous case. We discretize a continuous variable *X* using a method based on quantiles of the values in *X*. Quantile-based transformation has the advantages of stability and independence of transformation of input values. The number of quantization levels we choose for the algorithms MIFS, MRMR, and CMIM is 5.

TABLE 4.1
15 best features selected by the examined algorithms

FDR	Relief	MIFS	MRMR	CMIM
63	57	45	45	45
82	39	17	77	17
33	45	77	9	78
74	49	63	17	85
94	40	85	61	79
72	46	36	79	51
80	50	71	43	43
15	58	67	44	31
75	42	75	71	90
87	43	33	31	93
81	48	25	92	77
29	59	74	38	89
37	60	29	65	20
91	51	92	3	87
93	47	94	51	26

Indices in table are references to features presented in Section 4.1

TABLE 4.2
Recognition rates of classifiers in combination with feature selection algorithms.

	FDR	Relief	MIFS	MRMR	CMIM
fuzzy *k*-NN	0.32	0.39	0.33	0.44	0.48
Naive Bayes	0.25	0.38	0.47	0.63	0.6
LDC	0.25	0.39	0.5	0.50	0.58
mean combiner	0.32	0.37	0.46	0.59	0.61
median combiner	0.31	0.38	0.50	0.61	0.60

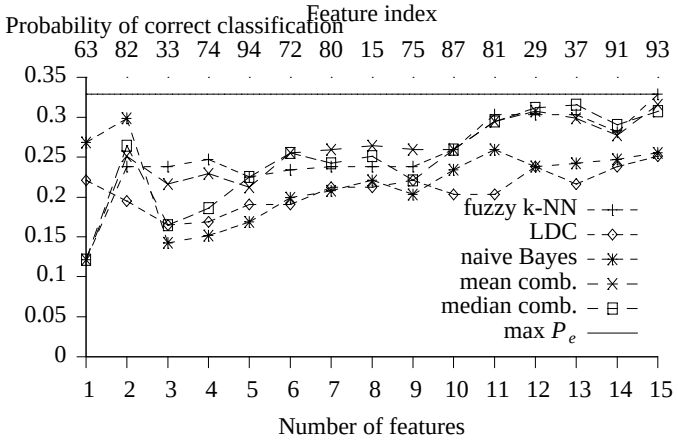

Figure 4.8: Selecting 15 features with FDR method. The progression of recognition accuracy as wee add new features from the selected ones.

The recognition rates of classifiers using the feature vector selected by each algorithm are shown in Table 4.2. In Figures 4.8-4.12 we show the progression of recognition accuracies of classifiers with respect to a feature selection algorithm, as we add new features to the set from the selected ones.

Looking at the tables and Figure of Section 4.5, we can make the following observations:

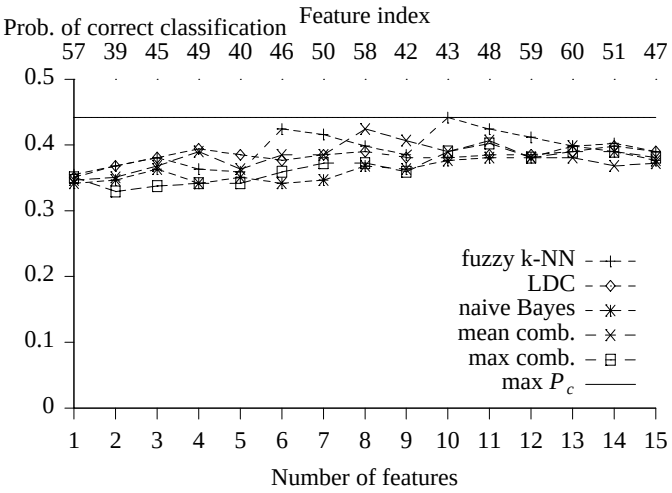

Figure 4.9: Selecting 15 features with Relief-F method. The progression of recognition accuracy as we add new features from the selected ones. Mean combiner outperforms the other classifiers at every step.

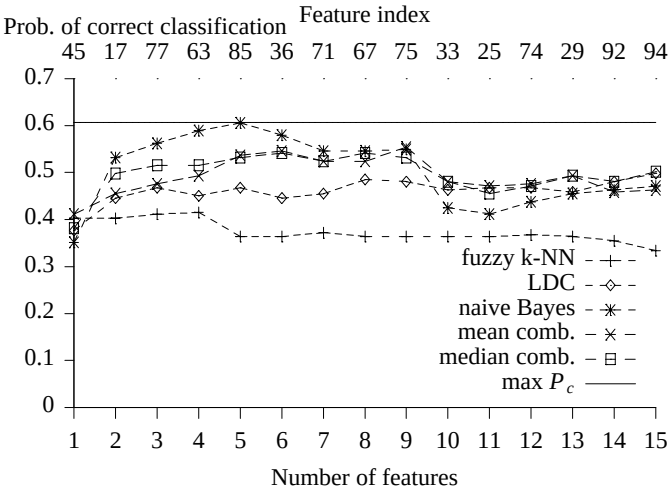

Figure 4.10: Selecting 15 features with MIFS method. LDC classifier has the highest recognition rate with naive Bayes having the lowest.

(1) FDR features lead to the lowest classification rates among the feature sets selected by the algorithms. We have to mention that mean and median combiners don't outperform the individual classifiers. One possible explanation is that the features selected by FDR lead the classifiers to do the same misclassification results.

(2) Classifiers have their minimum classification accuracy with feature sets selected by Relief-F and FDR. Fuzzy 3-NN has its maximum classification accuracy with CMIM feature set. The performances of the other classifiers lie between 0.37 and 0.39. Combinations don't have higher recognition rate that individual classifiers.

(3) In case of MIFS, median combiner and LDC are the best classifiers. Fuzzy *k*-NN has by far the worst recognition accuracy in this case; it has 10% less accuracy than the other classifiers.

(4) In case of MRMR naive Bayes, mean and median combiners achieve a recognition rate greater or equal to 60%, with naive Bayes having the highest (63%) and fuzzy *k*-NN the lowest (44%). Also with the feature selected by MRMR we have the highest recognition accuracy all the experiments; the performance of naive Bayes.

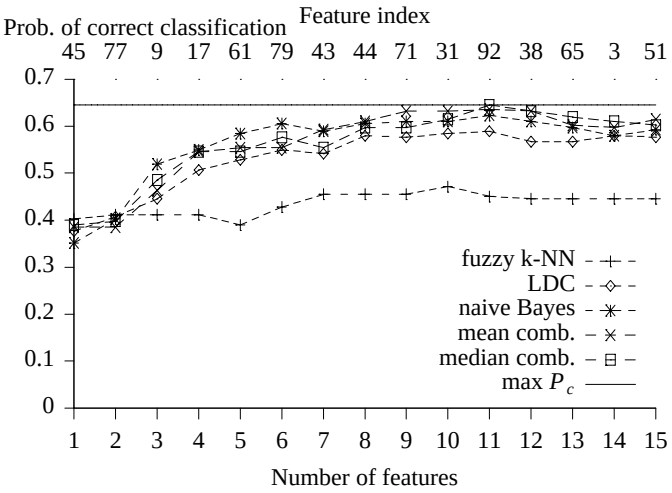

Figure 4.11: Selecting 15 features with MRMR method. The performance of all classifiers increases from the seventh step above 40%. The recognition accuracy of naive Bayes's classifier tends to retain around 50% from the seventh step till the end.

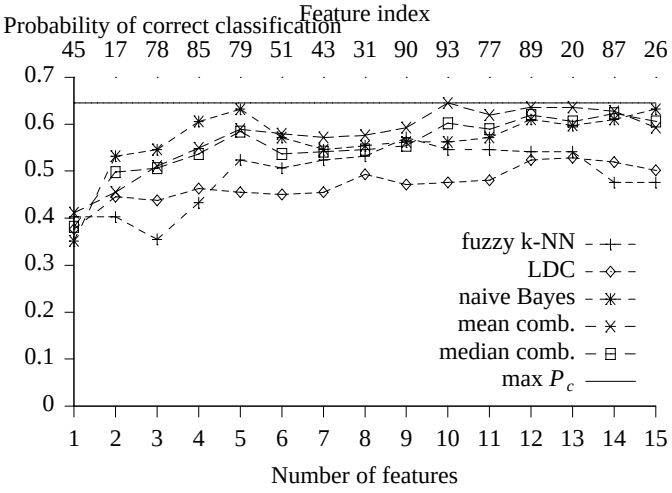

Figure 4.11: Selecting 15 features with CMIM method. With this method median combiner achieves the highest recognition accuracy in the experiment.

(5) With CMIM the performance of all classifiers increased. Mean, LDC, and fuzzy *k*-NN achieve their highest performance with CMIM compared to feature sets created by the other methods.

(6) The feature-selection algorithms and classifier pairs that achieved the best classification rate in the experiment are shown below:

- fuzzy *k*-NN – CMIM
- naive Bayes – MRMR
- LDC – CMIM

4.6. CONCLUSIONS

Feature selection has been a research topic with practical significance in many areas such as statistics, pattern recognition, machine learning, and data mining. In this Chapter we presented five feature-selection algorithms of the filter model: FDR, Relief-F, MRMR, MIFS, and CMIM. We select 15 out of 95 statistics of prosodic features from the speech signal using the aforementioned algorithms and evaluate their performance using as criterion the classification accuracy achieved by three individual classifiers: fuzzy *k*-NN,

naive Bayes, and LDC. Also, we combine classification decisions of individual classifiers about speech utterances to derive a more robust speech emotion recognition system.

From the results of experiments, we conclude that without knowledge of the pattern recognition problem, there is no best feature set. Many feature selection algorithms should be used and with the feature selected, more than one classifiers should be evaluated.

CHAPTER 5

EMOTION CLASSIFICATION USING ENSEMBLE METHODS AND CART TREES

Most pattern recognition methods based on feature vectors, use a natural measure of distance, between such vectors, called metric, in order to classify samples. Examples of such algorithms are *k*-NN, neural networks, etc.

However, there are cases where we can't use metric methods for classification, for example if the classification problem involves nominal data. In this case we can use a special type of classifier called decision tree. They are attractive types of models for three main reasons:

1) they have an intuitive representation
2) because decision trees of the fact that are non-parametric techniques, there is no need for the user to intervene on the data
3) there exist scalable algorithms for decision-tree construction models.

In the next section we introduce the concept of decision tree, and the concept of ensemble classification using decision trees.

5.1. DECISION TREES

Decision trees are a large class of nonlinear classifiers in which samples are classified into classes through a series of questions. A decision tree can be represented with an acyclic graph in the form of a tree. The root of the tree does not have any incoming edges. Every other node N has exactly one incoming edge and B outgoing edges which connect this node with subsequent nodes. The number B of outgoing edges from a node is called the node's *branching* factor. A node with branching factor $B = 0$ is called *leaf*, otherwise is called an *internal* node. The subsequent nodes that an internal node N is connected with are called its children.

The classification of a sample begins at the root node, which asks for the value of a particular property of the sample. Based on that answer, we follow the appropriate link to a descent node. The next step is to make the decision at the appropriate subsequent node, which can be considered the root of a sub-tree. We continue this way until we reach a leaf node, which has no further question. Each leaf node bears a category label and the test sample is assigned the category of the leaf node reached.

5.1.1. Binary Classification Trees

The most popular decision trees are binary classification trees (BCTs) where the internal and root nodes have a branching factor $B = 2$. One reason for their popularity is that any decision tree with $B > 2$ can be transformed into a binary decision tree. In these trees the sequence of questions to be answered is of the form ''*is feature* $x_i \leqq \alpha$?'' where α is a threshold value. This leads to a partition of the input space into hyper-rectangles with sides parallel to axes. The basic idea behind an BCT is demonstrated in Figure 5.1 where two dimensional samples are classified into four classes . Figure 5.2 shows a decision tree for the partition of Figure 5.1.

5.1.2. Classification and Regression Trees

The order in which features are tested in the BCTs plays an important role for the classification performance of the tree. For example an obvious question for the tree of Figure 5.2 is why to consider x_1 as the first tested feature and not x_2. This question is answered by Classification and Regression Trees (CART), a generic decision tree learning algorithm.

The scheme used in decision-tree learning for selecting attributes is designed to minimize the depth of the final tree. The idea is to pick the most informative feature, in the sense that it goes as far as possible toward providing an exact classification. In the terminology of decision trees such features lead to ''purer''

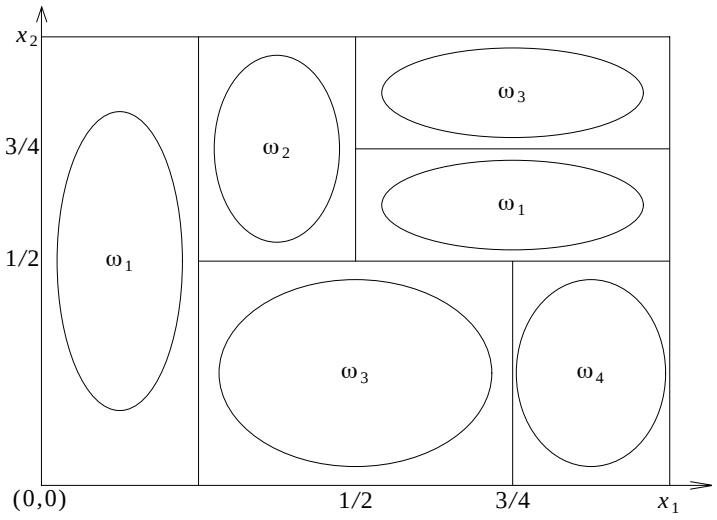

Figure 5.1: Decision tree partition of the space.

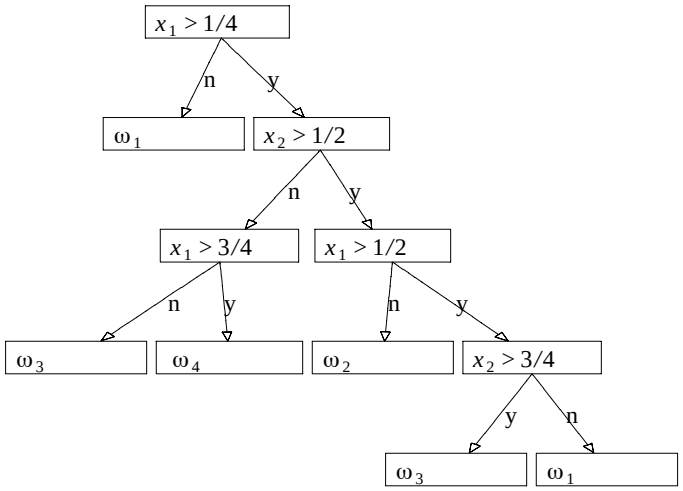

Figure 5.2: An example of decision tree for the case of Figure 5.1.

splits compared to the ancestor's node. One suitable measure is the amount of information provided by that feature.

5.1.3. Choosing Feature Tests

Let N be a node in a BCT and N_l, N_r its children respectively (Figure 5.3). Also let X_N be the set associated with node N that is split into subsets X_{N_l}, X_{N_r} associated with N_l and N_r respectively (Figure 5.4). For every split the following are true:

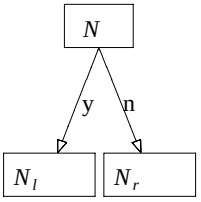

Figure 5.3: A binary split of node N into two descendant nodes N_l and N_r.

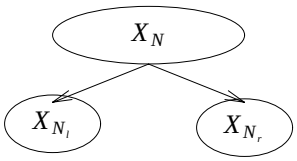

Figure 5.4: The dataset X_N associated with parent node N is split into two disjoint subsets X_{N_l}, X_{N_r} associated with left and right child nodes respectively.

$$X_{N_l} \cap X_{N_r} = \varnothing$$

$$X_{N_l} \cup X_{N_r} = X_N$$

While we are growing a BCT we seek for a property test T at each node N that makes data reaching N_l and N_r as ''pure'' as possible. Thus we need a definition of impurity for a node. Motivated by information theory (see Section 4.4.3), we define the information impurity $i(N)$ of node N as:

$$i(N) = -\sum_j \hat{P}(\omega_j \mid N) \log_2 \hat{P}(\omega_j \mid N) \tag{5.1}$$

where $\hat{P}(\omega_j \mid N)$ is the fraction of patterns at node N that are in category ω_j. The above definition has the property that $i(N) = 0$ when all the patterns that reach the node N bears the same label, and is large if the categories are equally presented. This is nothing else but the entropy associated with node N and subset X_N.

The key question now is given a tree down to node N, what value s should we chose for the property test T at N? An obvious heuristic is to chose the test that decreases the impurity of node's children as much as possible. The decrease in node impurity is defined as

$$\Delta i(s,\ N) = i(N) - \hat{P}_l\, i(N_l) - (1 - \hat{P}_l)\, i(N_r) \tag{5.2}$$

where s is the one of the possible split values for test T, N_l, N_r are the left and right descendant nodes, $i(N_l)$, $i(N_r)$ their impurities, and $\hat{P}_l$ the fraction of patterns at node N that will go to N_l when the property test T is used. Then the best value s^* for T is the one that maximizes $\Delta i(s,\ N)$.

5.1.4. Stop Splitting Rule

If a tree is grown fully until each leaf node corresponds to the lowest impurity, then it has been over fitted. A possible stop-split criterion is to adopt a threshold β and stop splitting if the maximum value of $\Delta i(s,\ N)$, over all possible splits s, is less than β. Other alternatives is to stop splitting when the node represents fewer than some fixed percentage of points of the total training set.

5.1.5. Class Assignment Rule

For the choice of label in leaf nodes we can use the simple majority rule, i.e., the leaf is labeled as ω_i where

$$i = \arg\ \max_j \hat{P}(\omega_j \mid N)$$

In other words, the leaf node N is assigned to the class where the majority of samples in X_N belongs to.

5.1.6. CART Implementation Issues

CART algorithm uses loops in a multi-dimensional space, in order to select the best split for each node. At each node, every feature x_j is processed in order to find the best split value for the question $x_j < s_j$. The best choice of split value for a test T associated with feature x_j is found using the following procedure:

- consider each feature x_j at a time
- order the values in vector x_j in descending order

ALGORITHM: CART

IN: $Q \times R$ matrix X where rows are samples and columns are variables

OUT: binary classification tree G

- begin with the root node, i.e., $X_{root(G)} = X$
- for each new node N
  - for every feature x_j
    - for every split value s
      - generate X_{N_l} and X_{N_r} w.r.t the answer in '' $x_j \leqq s$?''
      - compute the impurity decrease
    - choose s leading to the maximum decrease w.r.t x_j
  - choose x_j and associated s leading to the overall maximum decrease of impurity
  - if stop splitting criterion is true, declare node N as leaf and assign to it a label using the majority rule
  - else if stop splitting criterion is false, generate two children nodes N_l and N_r with associated subsets X_{N_l} and X_{N_r}, depending on the answer to question $x_j \leqq s$
- return tree G

Figure 5.5: Pseudocode describing the tree growing algorithm CART.

- for each vector x_j calculate the possible splitting values s_j as the middle of adjacent values $(x_j^i + x_j^{i+1})/2$
- among all questions $x_j \leqq s_j$ choose the value s_j with highest change of impurity
- repeat the procedure for the next feature vector x_{j+1}

Figure 5.5 shows the CART algorithm in pseudocode form. The main point in CART is that every node in the tree tries to solve a maximization problem of the form $\underset{x_j \leqq s,\ j=1,\ldots,R}{arg\max} [i(N) - \hat{P}_l i(N_l) - \hat{P}_r i(N_r)]$.

5.2. ENSEMBLE LEARNING

In the previous sections we described methods where only one hypothesis is used to predict the output of a classification. The key idea behind ensemble learning is the selection of an ensemble of hypotheses and their combination. For example we can generate thousand CART classifiers and choose the best hypothesis for the classification of a new sample.

The motivation for ensemble learning is the following. Consider an ensemble of $M = 5$ hypotheses and suppose that we combine their predictions using simple majority voting. For the ensemble to misclassify a new example, at least three of five hypotheses have to misclassify it. Two well known methods of ensemble learning are boosting and bagging.

5.2.1. Boosting

Boosting introduces the concept of *weighted training set.* A weighted training set is a set of training samples, where each sample x_i has a weight $w_i \geqq 0$ associated with it. As w_i increases, the sample becomes more important during the learning process.

The boosting algorithm takes as input a training set of N samples $S = \{(x_1, y_1), \ldots, (x_N, y_N)\}$ where x_i is an instance drawn from some space X and y_i is the class label associated with x_i, where y_i takes discrete values from the set $\{1, \ldots, K\}$, and a weak classifier L. Weak classifiers always returns a hypothesis on the training set with accuracy that is slightly better than random guessing (i.e., $50\% \pm \varepsilon$ for binary classification).

ALGORITHM: ADABOOST.M1

IN: K, number of classes

$Y = \{1, \ldots, K\}$, set with cardinality K with the number of possible labels

S, set of N training samples $\{(x_1, y_1), \ldots, (x_N, y_N)\}$ where $y_i \in Y$

L, weak classifier

OUT: final hypothesis $h_{fin}(x)$

- initialize weight vector $\vec{w} = [w_i, \ldots, w_N]$, i.e., $\vec{w}(i) = 1/N$
- for $m = 1$ to M do
  - $\vec{h}_m \leftarrow L(S, \vec{w})$
  - $\varepsilon_m \leftarrow 0$
  - for $j = 1$ to N do
    - if $\vec{h}_m(x_j) \neq y_j$ then $\varepsilon_m \leftarrow \varepsilon_m + \vec{w}(j)$
    - else $\vec{w}(j) \leftarrow \vec{w}(j) \times \varepsilon_m / (1 - \varepsilon_m)$
  - normalize $\vec{w}$
  - $z_m \leftarrow \log(1 - \varepsilon_m)/\varepsilon_m$
- return $h_{fin}(x) = \underset{y \in Y}{arg\max} \sum_{m: \vec{h}_m(x) = y} z_m$

Figure 5.6: The ADABOOST.M1 variant of the boosting method for ensemble learning. The algorithm generates hypotheses by successively re-weighting the training examples.

Boosting starts with $w_i = 1$ for all samples x_i, $i = 1, \ldots, N$, in the training set. Hypothesis h_1 comes out using the initial weighted training set. According to h_1, some samples will be correctly classified and other will be misclassified. We would like the second hypothesis, h_2, to perform better on the misclassified samples, so we increase their weights while decreasing weights of correctly classified samples. From the new weighted training set, comes out hypothesis h_2. The same process continues until the creation of M hypothesis, where M is a user defined input to the boosting procedure. Final ensemble hypothesis is a weighted-majority combination of all M hypotheses, each one weighted according to its performance on the training set.

5.2.1.1. ADABOOST

Arguably, the best known boosting method is ADABOOST. ADABOOST is an iterative procedure that combines many weak classifiers in order to increase the overall classification accuracy. ADABOOST has many variations, such as ADABOOST.M1 for classification problems where each classifier can attain a weighted error of no more than 1/2, ADABOOST.M2 for those weak classifiers that cannot achieve this error (for regression problems), among many others. In the next subsubsections we describe ADABOOST.M1 and ADABOOST.M2.

5.2.1.2. ADABOOST.M1

Figure 5.6 describes the ADABOOST.M1 algorithm. One important theoretical property about ADABOOST.M1 is that if weak hypotheses consistently have error only slightly better than 1/2, the error of the final hypothesis h_{final} drops to zero exponentially.

The main disadvantage of ADABOOST.M1 is that it is unable to handle weak hypotheses with error grater than 1/2. The expected error of a hypothesis which randomly guesses the label is $1 - 1/K$, where K is the number of possible labels. Thus ADABOOST.M1 requirement for $K = 2$ is that the prediction is just slightly better than random guessing. However, when $K > 2$, the requirement is much difficult to be met.

5.2.1.3. ADABOOST.M2

ADABOOST.M2 attempts to overcome the difficulty of ADABOOST.M1 when $K > 2$. Also ADABOOST.M2 introduces a more complex requirement for the performance of the weak classifier. Rather than using the prediction error as ADABOOST does, it introduces the concept of pseudo-loss. More formally, a mislabel is a pair (i,y) where i is the index of a training sample and y is an incorrect label associated with example i. Let B be the set of all mislabels:

$$B = \{(i,y) : i \in \{1, \ldots, K\},\ y \neq y_i\} \tag{5.4}$$

A mislabel distribution is a distribution defined over the set B of all mislabels. At each iteration m of boosting, ADABOOST.M2 (Figure 5.7) supplies the weak classifier L with a mislabel distribution D_m. In response L output a ''soft'' hypotheses $\vec{h}_m : X \rightarrow [0,1]^K$. The j^{th} component of this vector represents a ''degree of belief'' that the correct label is j. The components with values close to 1 or 0 correspond to those labels considered to be plausible or implausible, respectively.

Intuitively, (i,y) represents a binary question of the form: ''Do you predict that the label associated with sample x_i is y_i (the correct label) or y (one of the incorrect labels)?'' With this interpretation, the weight $\vec{w}_m(i,y)$ assigned to this mislabel represents the importance of distinguishing incorrect labels y on sample x_i.

Hypothesis $\vec{h}_m$ is interpreted as a set of ''plausible'' labels for a given sample x. Intuitively, it is easier for the weak learner to identify a set of labels which may plausibly be correct, rather than selecting a single label. If $\vec{h}_m(x_i,y_i) = 1$ and $\vec{h}_m(x_i,y) = 0$, then $\vec{h}_m$ has correctly predicted that x_i's label is y_i, and not y (since $\vec{h}_m$ deems y_i to ''plausible'' and y ''implausible''). Similarly, if $\vec{h}_m(x_i,y_i) = 0$ and $\vec{h}_m(x_i,y) = 1$, then $\vec{h}_m$ has incorrectly made the opposite prediction. If $\vec{h}_m(x_i,y_i) = \vec{h}_m(x_i,y)$, then $\vec{h}_m$ is taken to be a random guess.

Now the pseudo-loss of hypothesis $\vec{h}_m$ with respect to weights $\vec{w}_m$ is defined as:

$$\varepsilon_m = \frac{1}{2} \sum_{(i,y) \in B} \vec{w}_m(i,y)\left(1 - \vec{h}_m(x_i,y_i) + \vec{h}_m(x_i,y)\right) \tag{5.5}$$

The goal of the weak learner is to minimize the pseudo-loss. The pseudo-loss function is minimized when correct labels y_i are given values near 1 and incorrect labels $y \neq y_i$ values near 0. The final combined hypothesis $h_{fin}(x)$, for a given example x, chooses the single label which occurs most frequently in the plausible label sets chosen by the weak hypotheses (possibly giving more or less weight to some of the weak hypotheses). In Figure 5.7 the algorithm for ADABOOST.M2 is presented.

5.2.2. Bagging

Bagging, which stands for *bootstrap aggregating*, is another popular ensemble learning method and one of the earliest in the field. Diversity of classifiers in bagging is obtained by using bootstrapped replicas of the training data. That is, different data sets are randomly chosen with replacement from the entire dataset. Each training dataset is used to train different classifiers of the same type. Individual classifiers are then combined by taking the majority vote of their decisions. For any given sample, the class chosen by most classifiers is the final ensemble decision. Since training datasets may overlap, additional measures can be used to increase diversity. Pseudocode of Bagging algorithm is provided in Figure 5.8.

5.2.3. Random Forests

Random forests, proposed by [42], and add an additional layer of randomness to bagging. Random forests use CART trees as weak classifiers. In addition to constructing each tree using a different bootstrap training set of data, random forest change the process of constructing CART tress. In CART trees, each node is split using the best split among all feature variables. In random forests, each node is split using the best among a subset of feature variables randomly chosen at that node. This somewhat counterintuitive strategy turns out to perform very well compared to many other classifiers, including discriminant analysis, support vector machines and neural networks.

ALGORITHM: ADABOOST.M2

IN: S, set of N samples $\{(x_1,y_1),\ldots,(x_N,y_N)\}$

K, number of classes

$Y=\{1,\ldots,K\}$, set with cardinality K with the number of possible labels

L, weak classifier

OUT: final hypothesis $h_{fin}(x)$

- Let $B=\{(i,y):i\in\{1,\ldots,N\},y\neq y_i\}$
- initialize $\vec{w}_1(x,y)=1/|B|$ for $(i,y)\in B$
- for $m=1$ to M do
  - $\vec{h}_m\leftarrow L(\vec{w}_m,S)$
  - calculate pseudo-loss ε_m as

$$\varepsilon_m\leftarrow 1/2\sum_{(i,y)\in B}\vec{w}_m(i,y)(1-\vec{h}_m(x_i,y_i)+\vec{h}_m(x_i,y))$$

  - $\beta_m\leftarrow\varepsilon_m/(1-\varepsilon_m)$
  - $\beta_m\leftarrow\beta_m^{\frac{1}{2}(1+\vec{h}_m(x_i,y_i)-\vec{h}_m(x_i,y))}$
  - $\vec{w}_{m+1}(i,y)\leftarrow\vec{w}_m(i,y)\times\beta_m,$ for every $(i,y)\in B$
  - normalize $\vec{w}_{m+1}$
- return $h_{fin}(x)=\underset{y\in Y}{arg\max}\sum_{m:\vec{h}_{m(x)}=y}\log\frac{1}{\beta_m}\vec{h}_m(x,y)$

Figure 5.7: The ADABOOST.M2 variant of the boosting method for ensemble learning. ADABOOST.M2 overcomes the difficulties of ADABOOST.M1 for the multiclass classification case.

ALGORITHM: BAGGING

IN: K, number of classes

$Y=\{1,\ldots,K\}$, set with cardinality K containing the number of possible labels

S, set of N samples $\{(x_1,y_1),\ldots,(x_N,y_N)\}$, $y_i\in Y$

L, weak classifier

OUT: final hypothesis h_{fin}

- for $m=1$ to M
  - generate bootstrap training set S'_m with size $n'\leqq N$, by sampling examples from S uniformly and with replacement
  - $h_m\leftarrow L(S'_m)$
- for $i=1$ to N
  - let $v_{m,j}=\begin{cases}1, & \text{if } \vec{h}_m(x_i) \text{ decides } j,\ j\in Y\\ 0, & \text{otherwise}\end{cases}$
  - obtain total vote received by each class, $V_j=\sum_{m=1}^{M}v_{m,j},$ for every $j\in Y$
  - choose the class that receives the highest total vote as the final classification

Figure 5.8: The Bagging algorithm.

ALGORITHM: random forests

IN: S, set of N samples $\{(x_1, y_1), \ldots, (x_N, y_N)\}$

K, number of classes

$Y = \{1, \ldots, K\}$, set with cardinality K with the number of possible labels

L, weak classifier

OUT: final hypothesis $h_{fin}(x)$

- for $m = 1$ to M
  - generate bootstrap training set S'_m with size $n' \leqq N$, by sampling examples from S uniformly and with replacement
  - $\vec{h}_m \leftarrow L(S'_m, n')$, i.e., grow a complete CART tree with the following modification: at each node, rather than choosing the best split among all feature variables, randomly sample $n_s = \lfloor \log_2 N + 1 \rfloor$ of the feature variables and choose the best split from among those.
- for $i = 1$ to N
  - let $v_{m,j} = \begin{cases} 1, & \text{if } \vec{h}_m(x_i) \text{ decides } j,\ j \in Y \\ 0, & \text{otherwise} \end{cases}$
  - obtain total vote received by each class, $V_j = \sum_{m=1}^{M} v_{m,j,}$ for every $j \in Y$
  - choose the class that receives the highest total vote as the final classification

Figure 5.9: Random forests algorithm uses a different split criterion for the construction of classification tree.

Breiman's recommended size for the random subset is $N_s = \lfloor \log_2(N) + 1 \rfloor$. Thus, with 100 feature variables, every time that a tree node needs to be split, a random sample of 11 features is drawn. The random forests algorithm is presented in Figure 5.9.

5.3. EXPERIMENTAL RESULTS

We compared the performance of previous ensemble classification algorithms, i.e., BAGGING, ADABOOST.M2, and random forests using the EmoDB database. Initially, each speech sample x_i was represented by a column vector $\vec{x}_i \in R^{95}$ composed by fundamental frequency features, energy features, spectral features, and several statistics derived from their contours. The framework for this experiment is the same as CHAPTER 4 (see Section 4.1 for details about the extracted features and Section 4.3 for the classification approach).

In order to eliminate redundant features, we ran the MIFS feature selection algorithm (see Section 4.4.3 for details) and selected 15 out of 95 initial features. Thus, after feature selection, each speech sample x_i is represented by a column vector $\vec{x}_i \in R^{15}$. Features selected by MIFS are presented in Table 5.1.

All algorithms used CART trees as weak classifiers. We ran each algorithm for $i = 330$ iterations. At each iteration i, the ensemble size for each algorithm is i, i.e., tested algorithms train i weak classifiers and use their predictions to take the final decision for this iteration. Figure 5.10 shows the probability of correct classification of the algorithms as a function of the ensemble size i. The x-axis shows the number of rounds and the y-axis the test error of each algorithm.

Accuracy of ADABOOST.M2 is increasing until iteration $i = 100$, where it reaches its maximum value; thereafter it's accuracy tends to stabilize around 0.4422. ADABOOST.M2 is unable to keep up with

TABLE 5.1
Selected features by MIFS algorithm

45	17	77	63	85	36	71	67	75	33	25	74	29	92	94

Indices in table are references to features presented in Section 4.1

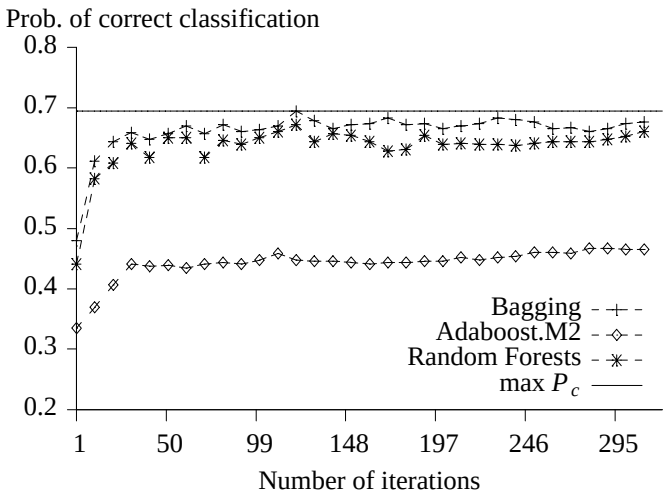

Figure 5.10: Comparison of learning methods using CART as the weak classifier.

the other two ensemble methods and never reaches the recognition accuracies of either BAGGING or random forests.

BAGGING and Random Forests reach their maximum performance in the range $100 < i < 170$. There does not seem to be any significant advantage of using random forest over BAGGING from recognition accuracy point of view.

5.4. CONCLUSIONS

The presented ensemble algorithms has been proved to produce very promising results. The experiments performed on the EmoDB database showed that bagging and random forest achieve a maximum classification accuracy of 69% and 67% respectively, using features selected by the MIFS algorithm. We believe that ensemble classification techniques are promising classifiers in the task of emotion speech recognition and should be further investigated in the future.

CHAPTER 6

COMBINING SPECTRAL, PROSODIC, AND ENERGY FEATURES FOR SPEECH EMOTION RECOGNITION

In this Chapter we show how a state of the art speech-emotion recognition system can be improved by combing spectral, prosodic, and energy features. We exploit the combination of our proposed TEMFCC features (refer to Section 3.4 for details) with statistics extracted from the pitch, formant frequencies, and short-time energy contours (refer to Section 4.1 for details). Specifically, we train three different classifiers, one for spectral features and two for prosodic features, and fuse their decision. The decision fusion is based on knowledge from our previous experiments. Finally we compare the results between individual classifiers and fusion system to show an improvement in classification accuracy n the order $\sim 7\%$.

6.1. INTRODUCTION

Cepstral features, like MFCC, Log Frequency Power Cepstral coefficients (LFPC) [25], and TEMFCC belong to a category of features called short-term features. The process of extracting short-term features involves the partition of speech signal in frames. Features obtained on portions of speech equal to one frame are called short-term features. Features obtained on portions of speech longer than one frame are called long-term features. Prosodic features and their statistics capture variations in intonation, timing, and loudness that are specific to the speaker.[43].

The speech emotion recognition system described in CHAPTER 3 was based solely on short-term features (MFCC, TEMFCC, T-MFCC) and random forest, which described in CHAPTER 6, on long-term features. In this experiment we try to combine the aforementioned systems in order to improve classification rate. This is done by fusing their decisions about a speech sample $s(n)$ in a way that is described in the following section.

6.2. FEATURES AND CLASSIFIERS

We combine decisions taken by three classifiers, denoted as e_1, e_2 and e_3 respectively, based on different feature vectors. e_1 is based on TEMFCC and follows a frame based classification approach (refer to Section 4.6.2 for details). Every frame is represented by a feature vector $\vec{x}_i$ containing TEMFCC plus their first and second order differences, known also as dynamic spectral features. e_1 is trained using $\vec{x}_i$ and outputs a decision for a speech utterance u_j by taking a simple majority vote of the labels of its frames.

e_2 and e_3 are based on statistics of several prosodic contours, like mean values of pitch contour, mean values of rising slopes of first formant etc, and follow an utterance based classification approach. The set of prosodic features extracted is the set of features described in Section 4.1 augmented with four statistics based on zero-crossing rate:

[96-99] mean, median, min and max of zero-crossings.

We number them starting from 96 because we assume that they are continuation of the feature list presented in Section 4.1

6.3. CLASSIFICATION SYSTEM OVERVIEW

Every classifier e_k is given as input an utterance u_j and outputs a decision $e_k(u_j) = c$, where $c = 1, \ldots, C$ is one of the C available class labels and $k = 1,2,3$ the classifiers. e_1 is a GMM based classifier trained with TEMFCC features. e_2 and e_3 are a naive Bayes and a random-forest classifier respectively, trained with statistics extracted from the prosodic contours of u_j. Figure 6.1 is a graphical representation of the combination system.

The final decision c of combination system, is based on $e_k(u_j)$, $k = 1,2,3$ and is taken by the algorithm shown in Figure 6.2. According to this algorithm, if e_1, e_2, and e_3 disagree on their decision about the label of u_j, then the final decision c is taken to be that of e_1. In all other cases, the final decision is a

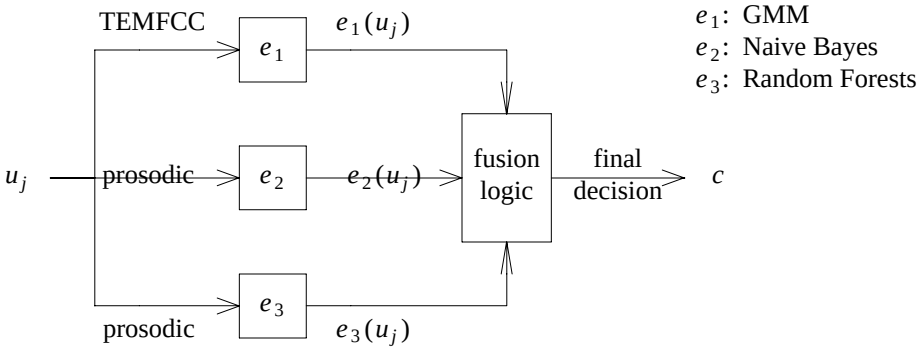

Figure 6.1: Graphical representation of combination system. The input given to classifiers is an utterance u_j. The final decision c is taken by a fusion logic based on the decisions made by individual classifiers.

ALGORITHM: Final decision of combination system

IN: u_j, speech utterance

$e_k(u_j)$, $k=\{1,2,3\}$, decisions of 3 individual classifiers, where e_1 is based on TEMFCC and e_2, e_3 on statistics of long-term features

OUT: final decision c for speech utterance u_j

- if $e_k(u_j) \neq e_l(u_j)$ for every $k \neq l$, $k,l \in \{1,2,3\}$ then
  - $c \leftarrow e_1(u_j)$
- else
  - $c \leftarrow$ simple majority vote of $e_k(u_j)$, $k=\{1,2,3\}$

Figure 6.2: The final decision for u_j is equal to the decision of $e_1(u_j)$ when all classifiers ''disagree''. In all other cases is equal to a simple majority vote taken over $e_k(u_j)$, $k=1,2,3$.

simple majority vote of $e_k(u_j)$'s. One may wonder: *''Why choose e_1's decision when all they disagree, instead of choosing at random?''*. The answer depends on the performance of e_1. In CHAPTER 3, the performance of e_1 with TEMFCC features was found to be ~0.76 at noise-free samples (refer to Table 3.1). In CHAPTER 4, we tested five feature selection algorithms in order to select 15 out of 95 prosodic features and 3 classifiers (among them was naive Bayes) and found that with these features sets naive Bayes achieve a maximum classification rate of ~0.6 (Table 4.2). In CHAPTER 5, the performance of random forest was found ~0.66 (Figure 5.10). Considering the results mentioned above, we assumed that e_1 is most likely to make the right choice when all classifiers disagree.

6.4. EXPERIMENTAL RESULTS

We compared the performances of individual classifiers e_k, $k=1,2,3$ to their combinations e_1 and e_2. Classifiers e_1 and e_2 follow the fixed combining rules r_1 and r_2 respectively. The first rule, r_1, is described in Figure 6.2. In the second rule, r_2, instead of selecting $e_1(u_j)$ decision when all classifiers disagree, we chose one of three decisions at random.

We used 5-fold cross validation to estimate the classification rates. Table 6.1 shows the results of cross-validation tests and Figure 6.1 is a graphical representation of Table 6.1. Naive Bayes and random forests classifiers achieve a recognition rate 62% and 66% respectively, using 15 out of 99 prosodic features with CMIM algorithm. Table 6.2 shows the indices of these features. GMM classifier achieves a correct rate of 70% using only spectral features. e_1 scores a recognition rate of 74.5%, while e_2 scores 73.5%. The improvement we get with rule r_1 is ~6% while with r_2 is ~4.5%.

We get more insight into the performance of of classifiers by looking their confusion matrices. Each element $a[i,j]$, in a confusion matrix, represents the count of instances whose known group labels are

TABLE 6.2
Selected features by CMIM algorithm

45	17	99	78	85	79	51	43	31	90	93	77	98	89	20

Indices in table are references to features presented in Section 4.1

TABLE 6.1

Cross validation results of experiments. There is a significant improvement in classification accuracy from the combination of classifiers.

Cross-validation iteration	GMM *TEMFCC*	Naive Bayes *prosodic*	Random Forest *prosodic*	combiner 1 *prosodic + TEMFCC*	combiner 2 *prosodic + TEMFCC*
1	0.73	0.65	0.65	0.76	0.75
2	0.70	0.55	0.68	0.75	0.73
3	0.80	0.61	0.66	0.80	0.75
4	0.60	0.60	0.57	0.64	0.66
5	0.68	0.70	0.75	0.80	0.80
Correct classification rate	0.70	0.62	0.66	0.75	0.74

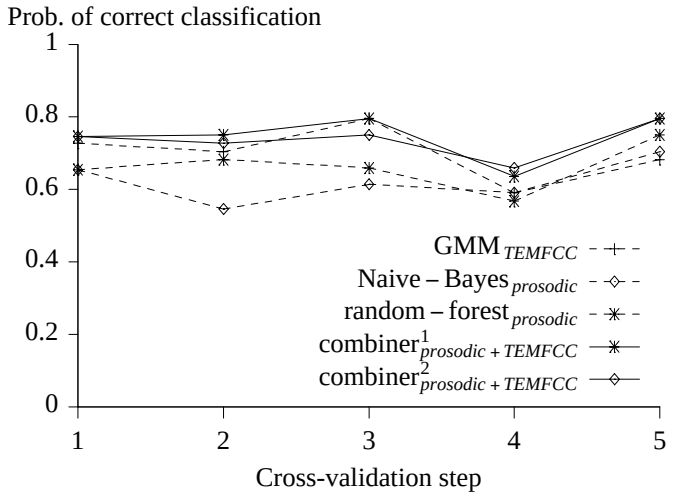

Figure 6.3: Graphical representation of Table 6.1.

TABLE 6.3
Confusion matrix for naive Bayes

	Anger	Happiness	Neutral	Sadness	Disgust
Anger	0.68	0.03	0	0.26	0.11
Happiness	0.1	0.5	0.19	0	0.18
Neutral	0	0.4	0.74	0.03	0.07
Sadness	0.08	0	0	0.54	0
Disgust	0.15	0.07	0.08	0.17	0.64

group *j* and whose predicted group labels are group *i*. Table 6.3 is the confusion matrix for the case of the naive bayes classifier. Naive Bayes has 74% recognition rate for neutral speech, and 50% for happiness. Many samples from neutral are misclassified as happiness, but not as many as happiness that are misclassified as neutral. This is common for all classifiers, and denotes a difficulty to distinguish these two emo-

TABLE 6.4
Confusion matrix for random forests

	Anger	Happiness	Neutral	Sadness	Disgust
Anger	0.90	0.03	0.04	0.34	0.28
Happiness	0.01	0.57	0.12	0	0.25
Neutral	0.02	0.35	0.81	0.03	0.18
Sadness	0.05	0.02	0.02	0.63	0.21
Disgust	0.03	0.01	0.01	0	0.07

TABLE 6.5
Confusion matrix for GMM

	Anger	Happiness	Neutral	Sadness	Disgust
Anger	0.69	0.06	0.019	0.09	0.25
Happiness	0	0.48	0.12	0	0.04
Neutral	0.02	0.39	0.85	0	0.11
Sadness	0.08	0	0	0.91	0
Disgust	0.21	0.07	0.02	0	0.61

TABLE 6.6
Confusion matrix for combiner 1

	Anger	Happiness	Neutral	Sadness	Disgust
Anger	0.90	0.02	0.02	0.23	0.29
Happiness	0	0.54	0.08	0	0.10
Neutral	0	0.39	0.88	0.02	0.07
Sadness	0.04	0	0	0.75	0
Disgust	0.06	0.05	0.02	0	0.54

tions. Table 6.4 is the confusion matrix for the random-forests classifier. Random forests have an outstanding recognition accuracy for anger equal to 90%. Also they have 81% recognition accuracy for neutral, 57% for happiness, 63% for sadness, and 7% for disgust. Their performance on detecting disgust is by far the worst among all classifiers. GMM have accuracy on recognizing neutral speech 81% and sadness 91%, the greatest among individual classifiers; classification rate for happiness and disgust is 48% and 61% respectively.

Table 6.6 is the confusion matrix for classifiers e_1. Diagonal elements of this table, represent the recognition rate for each emotion and are approximate equal to the mean of the diagonal elements of individual classifiers. Element $[5,5]$ is the recognition rate for disgust. The reason we don't get any improvement for disgust is the low recognition accuracy of random forests for this emotion. Similar performance to e_1 has e_2 whose confusion matrix is Table 6.7.

TABLE 6.7
Confusion matrix for combiner 2

	Anger	Happiness	Neutral	Sadness	Disgust
Anger	0.92	0.04	0.02	0.26	0.21
Happiness	0	0.52	0.08	0	0.11
Neutral	0	0.39	0.87	0.03	0.07
Sadness	0.03	0.02	0	0.71	0.07
Disgust	0.05	0.04	0.04	0	0.54

6.5. CONCLUSIONS

In this Chapter we built a speech emotion recognition system capable of using prosodic, spectral and energy features. This system, is a combination of classifiers designed in Chapters 3, 4, and 5. In detail, the combination was done by fusing the decision of different classifiers trained on different feature vectors. Three classifiers were examined; GMM trained with spectral features, naive Bayes and random forests trained with prosodic, energy and formants statistics features. The combination of their decisions achieve a significant classification error reduction of order ~7%. If the requirement of ''disagreement independence'' among individual classifiers holds, i.e., apply different classification principles for each individual classifier, then using different representations (feature sets) leads to a reduction in the correlation between the outputs of individual classifiers, since there is almost always less correlation between different input vectors.

CHAPTER 7

CONCLUSIONS AND FUTURE WORK

7.1. CONCLUSIONS

The research in this thesis aimed at applying state of the art techniques in emotion recognition from speech signals, examine them with a critical eye, and try to keep the best aspects and combine them in a proper way. Searching for a proper way, several experiments have been designed, each aiming to address one or more questions. Below we draw some conclusions learned during the progress of this thesis.

7.1.1. Models of Emotion

The initial step was choosing a model of emotion. The chosen model is based on the idea that emotions are regarded in a discrete way. From the practical perspective the model is appropriate for using machine learning techniques, and results of several tests show that it is a successful choice.

7.1.2. Feature Representation of Speech Signals

The features extracted from speech signals, are key parameters for the design of a speech emotion recognition system. They influence not only the performance of the system but also its overall structure. Cepstral features have been used extensively in the literature, as well as prosodic features, but their combinations should be further examined. Finding the most suitable set of features, that will yield the best performance and will include no redundancies, is still very challenging. The obtained sets of features in the literature are database-dependent, and high differences are observed between acted and natural speech datasets. Finding a set of features that is optimal and data-independent is still an unsolved problem. In our study, we found out that using different types of features, different classifiers and in the end fusing everything in an optimal manner can lead to strong improvements of the results.

7.1.3. Future Work

Even though it is something that researchers said many times before, we need to mention one more time that there is a strong need for new databases and of course especially databases of real emotional speech. It is obvious that all the research in this area is depending on the databases, and databases of real speech bring new challenges that need to be overcome. This way the research will get closer to the real-life application purpose. Besides this, we felt the need of standards in labelling of emotional states. We believe that there is a strong opportunity to build more robust and portable systems by using extended corpora.

ACKNOWLEDGMENTS

I would like to thank my supervisor Vassilis Digalakis for his trust in me and his understanding. I thank my parents for always being there for me. Of course, non of this would have been possible without my friends.

References

1. Takayuki Kanda, Kayoko Iwase, Masahiro Shiomi, and Hiroshi Ishiguro, ‘‘A tension-moderating mechanism for promoting speech-based human-robot interaction,’’ *IRIOS*, p. 527−532 (2005).
2. Burkhart F., Ajmera J., Englert R., and Huber R., ‘‘Detecting anger in automated voice portal dialogs,’’ *INTERSPEECH* (2006).
3. Ai H., Litman D.J, Forbes-Riley K., Rotaru M., Tetreaut J., and Purandare A., ‘‘Using system and user performance features to improve emotion detection in spoken tutoring dialogs,’’ *INTER-SPEECH* (2006).
4. Yang B. and Lugger M., ‘‘Emotion recognition from speech signals using new harmony features,’’ *Elsevier, Signal Processing* (2009).
5. Dimitrios Ververidis and Constantine Kotropoulos, ‘‘Emotional speech recognition: Resources, features, and methods,’’ *Elsevier, Speech Communication* **48**, pp. 1162-1181 (2006).
6. Engberg I. S. and Hansen A. V., ‘‘Documentation of the Danish Emotional Database,’’ *Internal AAU reposrt*, Center for Person Kommunikation.
7. J. H. L. Hansen and S. E. Bou-Ghazale, ‘‘Getting started with SUSAS: A speech under simulated and actual stress database,’’ *Proceeding on the European Conference on Speech Communication and Technology (EUROSPEECH)*, Rhodes, Greece, pp. 1743-1746.
8. Anssi Klapuri and Manuel Davy, *Signal processing methods for music transcriptions*, Springer (2006).
9. Dubnowski J. J., Schafer R. W., and Rabiner L. R., ‘‘Real-time digital hardware pitch detector,’’ *IEEE Transactions on Acoustic, Speech, and Signal Processing* **24**, pp. 2-8 (1976).
10. Schafer R. W. and Rabiner L. R., ‘‘System for automatic formant analysis of speech,’’ *Journal of Acoustical Society of America* **47**, pp. 634-648 (1970).
11. D. Talkin, ‘‘A Robust Algorithm for Pitch Tracking (RAPT),’’ *Speech Coding & Synthesis*, pp. 495-518, Elsevier (1995).
12. Schaffer H. L., Cohen A., Freudberg R., Manley H. J., and Ross M. J., ‘‘Average magnitude difference function pitch extractor,’’ *IEEE Transactions on Acoustic, Speech, and Signal Processing* **22**, pp. 353-362 (1974).
13. Markel J. D., ‘‘The SIFT algorithm for fundamental frequency estimation,’’ *IEEE Transactions on Audio and Electroacoustics* **20**, pp. 367-377 (1972).
14. Lawrence R. Rabiner, Michael J.Cheng, Aaron E. Rosenberg, and Carol A.McGonegal, ‘‘A Comparative Performance Study of Several Pitch Detection Algorithms,’’ *Transaction on Acoustics, Speech, and Signal Processing* **24**, pp. 399-418, IEEE (1976).
15. American National Standards Institute and American Standards Association, *American national psycho-acoustical terminology*, 197.
16. Frank Dellaert, Thomas Polzin, and Alex Waibel, ‘‘Recognizing Emotion In Speech,’’ *Proceedings of the CMC*, pp. 1970-1973 (1996).
17. James F. Kaiser, ‘‘On a Simple Algorithm to Calculate the ‘energy’ of a Signal,’’ *IEEE Proc. ICASSP* **90** (1990).
18. Atal B. and Schroeder M., ‘‘Predictive coding of speech signals,’’ *IEEE Transactions on Acoustics, Speech and Signal Processing*, pp. 360-361 (1967).
19. Vlasenko B., Schuller B., Wendemuth A, and Rigoll G., ‘‘Combining Frame and Turn-Level Information for Robust Recognition of Emotions within Speech,’’ *INTERSPEECH* (2007).
20. Hu H., Xu M. X., and Wu W., ‘‘Fusion of Global Statistical and Segmental Spectral Features for Speech Emotion Recognition,’’ *INTERSPEECH*, pp. 2269-2272 (2007).
21. Wolpert D. H. and Macready W. G., *No Free Lunch Theorems for Search*, 1996.

22. H. M. Teager, ‘‘Some Observation On Oral Air Flow During Phonation,’’ *IEEE Trans. Acoustics, Speech, Signal Processing* **28**, pp. 599-601 (1980).

23. Thomas F. Quatieri, *Discrete-Time Speech Signal Processing*, Prentice Hall Signal Processing Series (2002).

24. Maragos Petros and Bovik Alan C., ‘‘Image demodulation using multidimensional energy separation,’’ *Journal of the Optical Society of America* **12**, pp. 1867-1876 (1995).

25. T. L. Nwe, S. W. Foo, and L. C. De Silva, ‘‘Detection of Stress and Emotion in Speech Using Traditional and FFT Based Log Energy Features,’’ *Proceedings of the 4th International Conference on Information, Communications and Signal Processing*, Singapore (2009).

26. Guojun Zhou, John H. L. Hansen, and Jamed F. Kaiser, ‘‘Nonlinear Feature Based Classification of Speech Under Stress,’’ *IEEE Transaction of Speech and Audio Processing*, pp. 201-216 (2001).

27. Dimitrios Dimitriadis, Petros Maragos, and Alexandros Potamianos, *Auditory Teager Energy Cepstrum Coefficients for Robust Speech Recognition*.

28. Hemant A. Patil and T. K. Basu, ‘‘Identifying Perceptually Similar Languages Using Teager Energy Based Cepstrum,’’ *Engineering Letters* **16** (2008).

29. C. A. Bouman, *An Unspurvised algorithm of for modeling Gaussian Mixtures*, 1997.

30. J. Kittler, M. Hatef, R. P. W. Duin, and J. Matas, ‘‘On Combining Classifiers,’’ *IEEE Transactions on Pattern Analysis and Machine Intelligence* **20** (1998).

31. Philip Loizou, *Colea: A matlab software tool for speech analysis*, 2003.

32. Deller J. R., Hansen J. H. L., and Proakis J.G, *Discrete-Time Processing of Speech Signals*, Wiley, N.Y.

33. J. M. Keller, M. R. Gray, and J. A. Givens, Jr, ‘‘A Fuzzy K-Nearest Neighbor Algorithm,’’ *IEEE Transactions on Systems, Man, and Cybernetics* **15**(4), pp. 580-585.

34. Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*, Wiley-Interscience (2000).

35. Liu H. and Motoda H., *Computational Methods of Feature Selection*, Chapman & Hall (2008).

36. Kolomenko L., ‘‘Estimating Attributes: Analysis and extension of RELIEF,’’ pp. 171-182 in *Proceedings of the European Conference on Machine Learning*, ed. Bergadano F. and De Raedt L., Springer-Verlang, Italy, Catania (1994).

37. Fano R., *Transmission of Information: Statistical Theory of Communications*, Wiley, New York.

38. Latham, PE and Roudi, Y, ‘‘Mutual information,’’ *Scholarpedia* **4**, p. 1658 (2009).

39. Battiti R., ‘‘Using mutual information for selecting features in supervised neural net learning,’’ *IEEE Transactions on Neural Networks* **5**, pp. 537-550 (1994).

40. Peng H., Long F., and Ding C., ‘‘Feature Selection Based on Mutual Information: Criteria of Max-Dependency, Max-Relevance, and Minimum Redundancy,’’ *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **27**, pp. 1226-1238 (2005).

41. Fleuret F., ‘‘Fast Binary Feature Selection with Conditional Mutual Information,’’ *The Journal of Machine Learning Research* **5**, pp. 1531-1555 (2004).

42. Leo Breiman, ‘‘Random Forests,’’ *Mach. Learn.* **45**(1), pp. 5-32, Kluwer Academic Publishers (2001).

43. Shriberg E., ‘‘Higher-Level Features in Speaker Recognition,’’ in *Speaker Classification I*, ed. Christian Muller, Sprigner, Heidelberg (2007).

TABLE OF CONTENTS