CHALMERS
UNIVERSITY OF TECHNOLOGY

Audio and Speech Classification Applied to Child Sexual Abuse Investigation

Master's thesis in Computer Science: Algorithms, Languages and Logics

Oskar Montin, Gustav Mörtberg

Department of Computer Science
CHALMERS UNIVERSITY OF TECHNOLOGY AND UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2016

MASTER'S THESIS 2016

Audio and Speech Classification Applied to Child Sexual Abuse Investigation

OSKAR MONTIN
GUSTAV MÖRTBERG

Department of Computer Science And Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY AND UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2016

Audio and Speech Classification Applied to
Child Sexual Abuse Investigation

Supervisor: Krasimir Angelov, Department of Computer Science
Examiner: Bengt Nordström, Department of Computer Science

Master's Thesis 2016
Department of Computer Science
Division of Algorithms, Languages and Logic
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Abstract

The complexity and scale of seized media in criminal investigations has increased dramatically in recent times, not least in child sexual abuse investigations. Manual examination of material impose great stress on the investigator and innovative aids can play a crucial role mitigating this. The thesis evaluates the use of machine learning algorithms for automatic speech classification. More specifically, we present the components of a system that uses acoustic features to identify speech in noisy environments and classify the speakers gender and spoken language. For each of the tasks, separate approaches based on earlier research were developed and experiments were devised to validate them. The results of all classification tasks were satisfactory, but the language classifier were found not to scale well with the number of supported languages. In conclusion, the thesis shows that machine learning models are well suited for speech classification. The thesis was performed at Safer Society Group.

Keywords: machine learning, speech classification, support vector machines, deep belief networks, gaussian mixture models, voice activity detection, langauge identification, gender classification.

Contents

1 Introduction **1**
1.1 Goals . . . 1
1.2 Delimitations . . . 2
1.3 Thesis outline . . . 2

2 Audio theory **5**
2.1 Audio Basics . . . 5
2.2 Digital audio . . . 5
2.3 Audio Features . . . 6

3 Machine Learning theory **13**
3.1 Support Vector Machines . . . 14
3.2 Gaussian Mixture Models . . . 17
3.3 Deep Belief Networks . . . 18

4 Methods **23**
4.1 Datasets and data selection . . . 23
4.2 Technical information . . . 25
4.3 Merging of data sources . . . 25
4.4 Data preprocessing . . . 25
4.5 Classification smoothing . . . 27

5 Voice activity detection **29**
5.1 Background and approach . . . 29
5.2 Experimental setting . . . 30
5.3 Results . . . 32
5.4 Analysis . . . 34

6 Language classification **35**
6.1 Background and approach . . . 35
6.2 Experimental setting . . . 38
6.3 Results . . . 39
6.4 Analysis . . . 41

7 Gender classification **43**
7.1 Background and approach . . . 43
7.2 Experimental setting . . . 44

7.3 Results 45
7.4 Analysis 45

8 Discussion and future work 47
8.1 Comparison with previous research 47
8.2 Relating to CSA media 48
8.3 Data challenges 49
8.4 The effects of removing frames 50
8.5 Future work 50

9 Conclusion 53

Bibliography 55

1

Introduction

The complexity and scale of seized media in criminal investigations has increased dramatically in recent times, not least in child sexual abuse (CSA) investigations. This development can be attributed to several recent advancements in fields such as data storage, networking capability and prevalence of video recording devices made ubiquitous with the rise of modern smartphones. This changes the needs of forensic investigators dealing with seized media.

For obvious reasons manual examination of CSA material impose great stress on the investigator. Therefore, innovative methods that aid in examination of CSA media play a crucial role in helping investigators. One technique that the law enforcement forensic community has expressed great interest in is classification of audio. The ability to quickly find material that includes speech which satisfies certain conditions can play an important role in identifying both the victim and offender.

This thesis explores the use of machine learning techniques for automatic classification of speech in audio and classifying found speech segments on features such as the language spoken and gender of the speaker.

1.1 Goals

The goal is to develop a system which identifies active speech in noisy audio environments and classify speech segments on the speakers gender and spoken language.

In order to do this, the following intermediary milestones have been identified:

- Identify and extract features needed for classification.
- Develop a model which identifies segments of active speech in noisy audio, voice activity detection.
- Develop a model for classification of a segments spoken language.
- Develop a model for classification of the gender of a segments speaker.

1.2 Delimitations

The classification tasks are exclusively solved using acoustic properties of speech. This means that no syntactic approach is evaluated for any of the classification tasks. Even though it could be relevant to look at the grammatical structure of the speech segments for language classification, it is outside the scope of the thesis.

Only reasonable noise levels, where noise does not overpower the speech, is supported. This means that the harshest noise conditions in terms of noise level will be noise at the same level as the speech. Furthermore the voice activity detection models will not be evaluated on all types of noisy environments, but are limited to a set of predefined noisy environments ranging from domestic to outdoor environments.

For the language identification task, a limitation has been set on the number of languages used during evaluation. At most five languages will be simultaneously evaluated as solving the language identification task for a large set of languages is infeasible in the scope of a master thesis. Only closed set tests will be performed, where the number of languages are known beforehand, as classifying speech from unseen languages is outside the thesis scope.

1.3 Thesis outline

CHAPTER 2 Describes audio theory. It handles properties of digital audio and which features that are extracted from audio streams.

CHAPTER 3 Describes machine learning theory. It handles machine learning concepts and describes the theoretical background which the proposed models are based on. It also includes a description of the model's evaluation metrics.

CHAPTER 4 Describes the proposed system and methodology used to perform the experiments. It handles data selection and processing.

CHAPTER 5-7 Follows the same convention and describes how each of the intermediary milestones, defined in the goals section, was handled. Each consists of the following parts: a background and description of the approaches for the particular problem, description of the experimental setting, the results for each experiment and finally an analysis of the achieved results.

CHAPTER 8 Compares the achieved results with previous research, describes some of the choices made and challenges encountered. It also includes possible future extensions of the work done.

CHAPTER 9 Consists of the thesis conclusion.

2

Audio theory

This chapter introduces and explains basic concepts of audio, its digital representation and properties that can be derived from the audio.

2.1 Audio Basics

In its most basic form, sounds are pressure waves propagated through air which can be perceived by the human auditory system or an artificial auditory system such as a microphone. Speech is produced by pushing air through the vocal tract, producing these pressure waves. The resonant frequencies of produced sound are determined by the shape of the vocal tract and the current position of tongue and lips [1].

The resonance in the vocal tract gives rise to *formants*. Formants are concentrations of acoustic energy around particular frequencies, where each formant corresponds to a resonance in the vocal tract. There are several formants F_x ordered in rising frequency and it is primarily the locations of the first three formants frequencies $F_1 - F_3$ that distinguishes vowels [2]. One way to visualize formants is by plotting the audio spectrum in a spectrogram as can be seen in Figure 2.1.

Another important feature of speech is the *fundamental frequency*, which is the lowest frequency of the speech signal's waveform. The fundamental frequency can be used to measure the perceived pitch of the produced sound. Generally, the vocal tracts of males are roughly fifteen percent longer than those of females. This is one of the reasons why male speech generally have lower formant and fundamental frequencies than female speech [4].

2.2 Digital audio

In digital audio recordings, the waves are represented by a series of points, or *samples*, where each sample is the value of the waveform at a particular point in time. A digital signal is sampled at a *sample rate* f_s which is the number of measured data points per second. A visualization of sampling from a continuous signal can be seen in Figure 2.2.

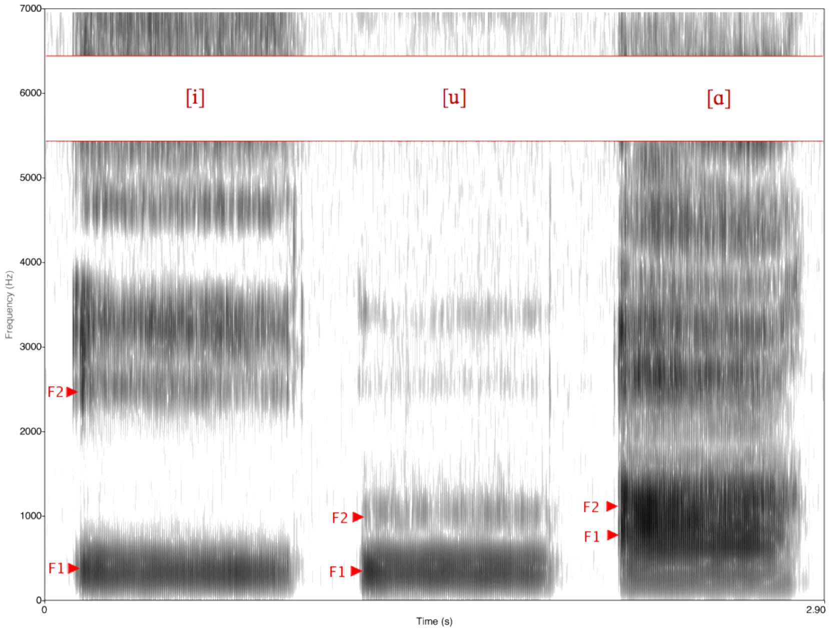

Figure 2.1: Spectrogram of American English vowels i, u and a. Labeled with the approximate frequencies of the first and second formants [3].

The sample rate bounds the maximum bandwidth of a signal. If a signal has no frequency higher than W Hz, it is completely determined by its ordinates at points spaced $\frac{1}{2}W$ seconds apart [5]. Thus, to perfectly reconstruct a signal with bandwidth W, the signal must be sampled at $2W$ Hz which often referred to as the Nyquist rate. Traditional telephone signals are sampled at 8 kHz for example, which means frequencies up to 4kHz can be recaptured.

All audio features in this section are calculated from series of samples in windows of fixed size, henceforth referred to as a *frame*.

2.3 Audio Features

This section describes the different audio properties extracted, also called features, which form the input to the classification models. The selection of features are based on earlier research, including [6] and [7].

2.3.1 Energy

The average energy in the frame, formally specified as equation 2.1 where N is the number of samples in the frame and x_i the value of sample number i.

$$\text{ENERGY} = \frac{\sum_{i=1}^{N-1} x_i^2}{N} \tag{2.1}$$

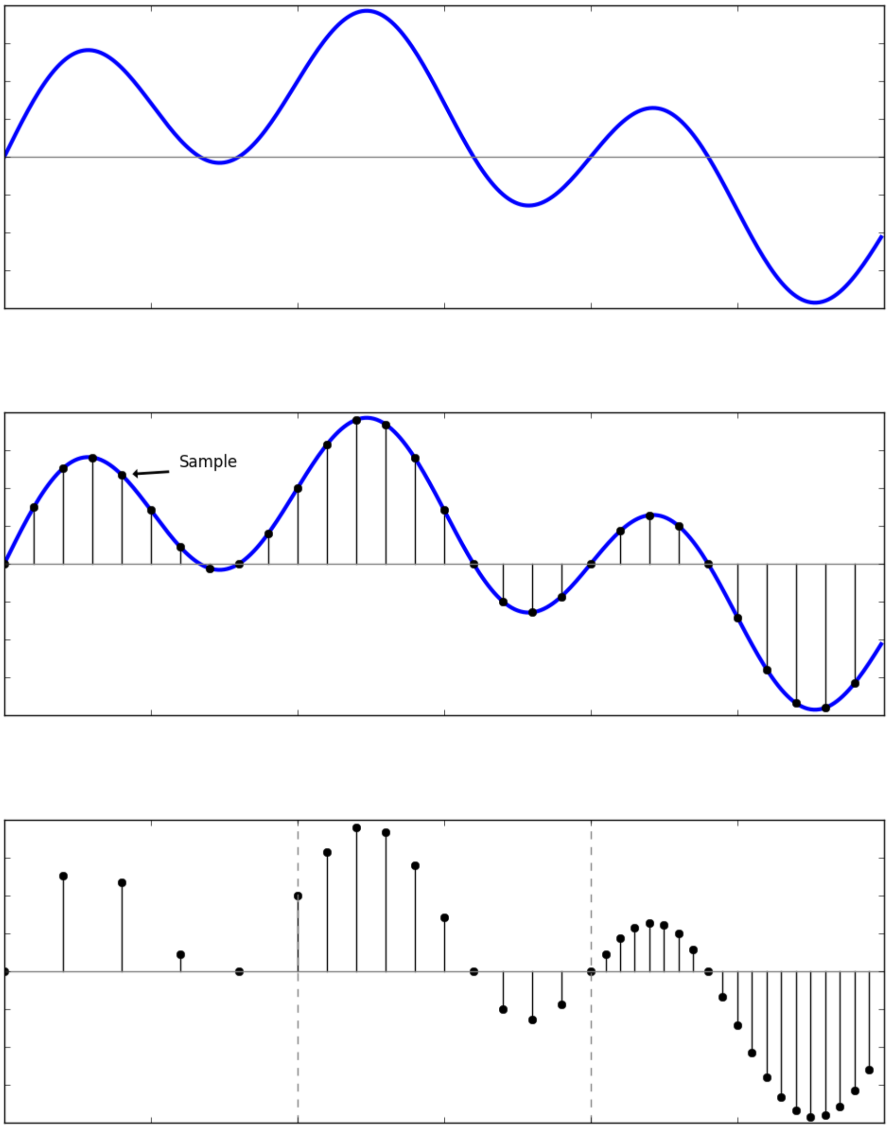

Figure 2.2: Top: A continuous signal. Mid: The signal sampled at a steady frame rate, showing the samples which will constitute the digital signal. Bottom: The signal sampled at three different rates. As can be seen, if the sample rate is low (as in the leftmost part) there is a risk of loosing information and if the rate is high there is a risk of storing redundant information.

2.3.2 Zero crossing rate

Zero crossing rate (ZCR) Describes the rate at which the signal switches from positive to negative or vice versa in the frame. It can be seen as an indicator of the frequency at which the energy is concentrated in the frequency spectrum [8]. Calculated as in equation 2.2, where N is the number of samples in the frame.

$$ZCR = \frac{\sum_{i=1}^{N-1} \mid sgn[x_i] - sgn[x_{i-}] \mid}{N} \tag{2.2}$$

2.3.3 Pitch Estimation

As described in section 2.1, speech at a point in time has a fundamental frequency f_0 which can be be used to measure the perceived pitch of the produced sound.

To correctly estimate f_0 is an unexpectedly complex task, especially for different types of signals. For example, a method which performs well on music does not automatically perform well on speech signals and even a method which estimates well for clean speech can degrade heavily when applied to noisy speech [9]. This has led to the creation of several different widely used estimation methods, with each having its own preferred domain and quirks.

One of the simpler methods that works well with speech from a single speaker with low background noise [9], is the use of the ZCR as a basis for pitch estimation. It is also computationally inexpensive. The estimated pitch is calculated as following.

$$\text{PITCH} = \frac{\text{SAMPLE RATE}}{\mu_{ZCR}} \tag{2.3}$$

2.3.4 Discrete Fourier Transforms

The *Discrete Fourier transform* (DFT) decomposes a function into a set of sinusoidal waves whose sum represents the original function. In signal processing, the DFT decomposes a signal, that is a function of time, into the frequency domain representation of the original signal. In speech processing, the DFT gives the energy in different frequency bands (also called the *power spectrum*) which is closely related to the concept of formants described in section 2.1.

The sinusoidal waves used to model the DFT are referred to as the basis functions. These are sine and cosine waves with unity amplitude but different period lengths defined as equation 2.4 and 2.5, where k determines the number of periods of the wave which fits within a single window of N samples.

$$s_k(t) = sin(t * k * \frac{2\pi}{N}) \tag{2.4}$$

$$c_k(t) = cos(t * k * \frac{2\pi}{N}) \tag{2.5}$$

The DFT can be calculated as following.

$$DFT_k(x) = \sum_{n=0}^{N-1} x_n e^{-i2\pi k \frac{n}{N}} \tag{2.6}$$

A straightforward method of using this formula for each of the DFT points results in $O(N^2)$ calculations, but the complexity can be reduced to $O(NlogN)$ by using the the Cooley–Tukey fast Fourier transform (FFT) algorithm [10].

2.3.5 Mel Frequency Cepstral Coefficients

Mel frequency cepstral coefficients (MFCC) represent the perceived power spectrum of a signal in a set of certain frequency bands [11]. The calculation of MFCCs for a signal $s(n)$ is performed as following. First, a periodogram-based power spectral estimate is calculated in order to identify the power in the frequency band for a frame $s_i(n)$, as in equation 2.7.

$$P_i(k) = \frac{1}{N}|DFT_i(k)|^2 \tag{2.7}$$

Following this, a filter bank of triangular filters spaced based on the Mel scale are applied to the power spectrum. An example consisting of eleven filters can be seen in figure 2.3. This approximates the nonlinear frequency resolution of the human ear and results in an approximation of the energy of the frame in each filter's frequency range.

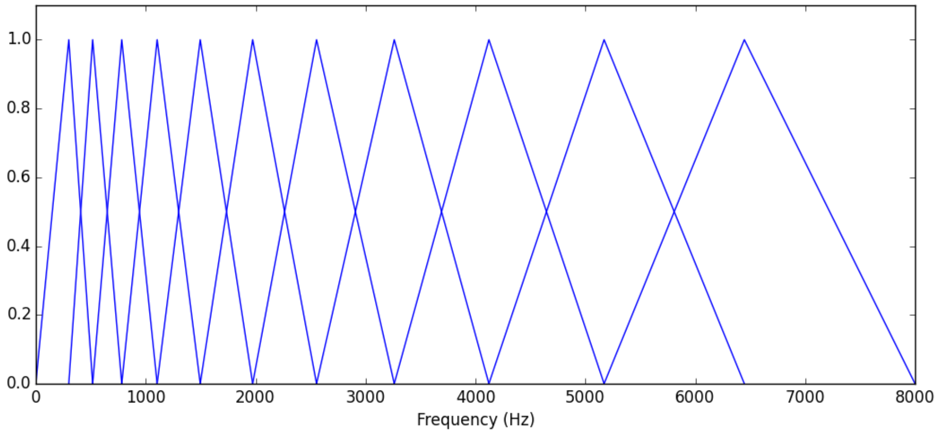

Figure 2.3: A filter bank of eleven Mel-spaced triangular filters.

To simulate the perceived loudness of the energies, the logarithms are computed. This resembles how humans perceive sound better, which is on a logarithmic scale over frequencies and not a linear scale. Since the filters in the filter bank are overlapping, the energies are correlated to each other. To decorrelate them, the discrete cosine transform for each filter's energy is calculated. From this, a number of coefficients which encapsulate the relevant frequency band are selected. These constitute the vector of MFCCs for the frame.

The MFCCs provides a good estimation of the local spectra, but they do not encapsulate temporal information. This can also be included as a separate feature, by estimating one or both of the MFCCs local derivatives (DMFCC) and the DMFCCs derivatives (DDMFCC). The DMFCCs are estimated as following (and the DDMFCC is computed the same way but using the DMFCCs as C), where $C_{i,t}$ is the ith cepstral coefficient of frame t and D is the distance in number of frames used in the delta computation.

$$\Delta C_{i,t} = \frac{\sum_{k=-D}^{D} k C_{i,t+k}}{\sum_{k=-D}^{D} k^2} \tag{2.8}$$

2.3.6 Shifted Delta Coefficients

Shifted delta coefficients (SDC) are included to model temporal information. Given an ordered list C of coefficients over time, temporal information for one coefficient $c_t \in C$ is gained by concatenating a fixed amount k of upcoming delta coefficients with equal spacing p as shown in Figure 2.4.

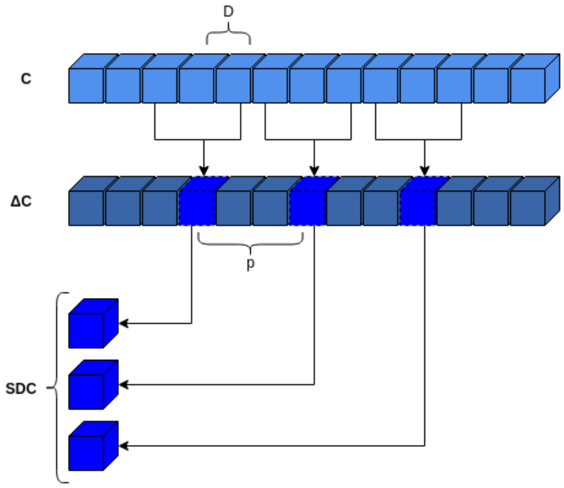

Figure 2.4: Figure showing the concatenation of SDCs for the first coefficient in a sequence of coefficients where p=3, k=3 and D used in delta computations set to 1.

In speech technology, temporal information that can model high order properties such as phonemes is achieved by taking SDCs consisting of DMFCCs derived from MFCCs with M cepstra. This results in a total number of SDC values per frame of $M \times k$.

2.3.7 Linear Predictive Coding Coefficients

Linear predictive coding (LPC) is a signal processing technique that derives a compact representation of spectral magnitude for a signal [12]. LPC analysis of a speech signal is performed by estimating the formants (see Section 2.1) and removing their effect on the signal. The formants are modeled as filters that mimic the effects of the vocal tracts shape. The analysis results in a set of p coefficients defining the formant filter as well as the residual signal after removing their effect.

A common way to estimate the coefficients is using a least square method, for example autocorrelation, where coefficients that minimize the mean energy of the residue signal are selected. The residue signal is defined in equation 2.9 where p is the number of poles in the filter, a the vector of filter coefficients and $x(t)$ the hamming windowed input signal.

$$r(t) = x(t) - \sum_{k=1}^{p} a_k x(t-k) \tag{2.9}$$

The autocorrelation method is used to estimate the coefficients that minimizes the expected value of the energy error:

$$E_{error} = \sum_{t=-\infty}^{\infty} r^2(t) = \sum_{t=-\infty}^{\infty} [x(t) - \sum_{k=1}^{p} a_k x(t-k)]^2 \tag{2.10}$$

The coefficients that minimize E_{error} are found by solving the p linear equations $\frac{\delta E_{error}}{\delta a_k} = 0$ for $k = 1...p$, this yields equation 2.11. $R(i)$ is the autocorrelation defined in 2.12 where N is the frame size.

$$R(i) = \sum_{k=1}^{p} a_k R(i-k) \tag{2.11}$$

$$R(i) = E[x(t)x(t-i)] = \sum_{t=i}^{N-1} x(t)x(t-i) \tag{2.12}$$

If the linear equations are formulated in matrix form $Ra = r$ where R is a Toeplitz matrix with elements $R_{ij} = R(|i-j|)$ and $r = [R(1), \ldots, R(p)]$ is the autocorrelation vector, they can be solved using the Levinson-durbin algorithm [13]. The resulting coefficients constitute the LPC values.

3

Machine Learning theory

Machine learning is a varied field with many application areas, generally speaking one can say that it encompasses techniques for drawing conclusions from data. This can be further separated in three major areas - classification, clustering and regression.

Classification is the task of assigning an observation to one of several sub-populations, or labels. If there are only two labels it is called *binary classification* and if there are more than two it is called *multi-class classification*. A prime example of a classification problem is spam filtering, where a new observation (an email) is classified as either spam or non-spam depending on its contents.

Classification models can be further separated into supervised and unsupervised models. Supervised models are shown observations and corresponding labels with the goal of finding general rules to assign labels to new observations. In unsupervised learning, the model is shown observations with the goal of finding patterns and inferring the structure of the observations.

All of the classifications performed in the thesis is of the supervised kind. The dataset shown to the model consists of a feature matrix X and a vector of true labels (or target labels) y. X is of the form $n \times d$, where n is the number of observations and d the dimensionality of each data point. The label vector y consists of n labels taken from a *label space*, consisting of the possible classifications for a data point.

This section introduces the theory of the models used in the thesis, how the performance of the models are measured and how the performance is verified to be generalizable.

Overfitting and cross validation

A concern is overfitting the model to the data, which is when the model captures every slight variation in the data. This leads to a model which performs well on seen data but does not generalize to new observations.

An example of overfitting can be seen in Figure 3.3, where the goal is to fit a function in such a way that it correctly describes a series of points. As can be seen, the left function has a smooth curve and approximately models the points while the function on the right describes all points but follows the points exactly. This function would probably not generalize well to new input data. To asses how well a model generalizes, the data is split

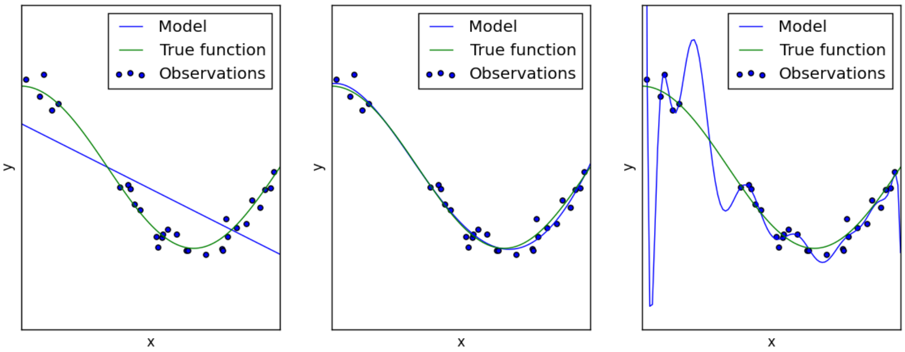

Figure 3.1: Fitting a function to a series of observations. Left: An underfitted function. Middle: A correctly fitted function. Right: An overfitted function.

into disjoint train and test sets, where the elements in each set is randomly selected. By doing this, the performance of the classifier is evaluated on unseen data. However, there is still a risk of splitting the data incorrectly, which can lead to unexpected results.

To further improve this validation scheme, k-fold validation can be applied. In k-fold validation the data is randomly split into k equally sized subsets, which enables validating the model k times by systematically using each of the subsets as test set and the rest for training. The metric used to validate the classifier is then the average over all folds.

3.1 Support Vector Machines

Support vector machines (SVM) is a supervised learning method for performing binary classification of linearly seperable input data. It was introduced by Vapnik et al. in the nineties and was shown to generalize well to varied sorts of data [14][15].

Given a set of n data points $x_i \in R^d$ and their corresponding labels $y_i \in \{-1, 1\}$, the idea is to find a hyperplane $w^T x - b = 0$ separating the two classes where b is the bias of the hyperplane. If b is zero, the hyperplane cuts the origin and is said to be unbiased.

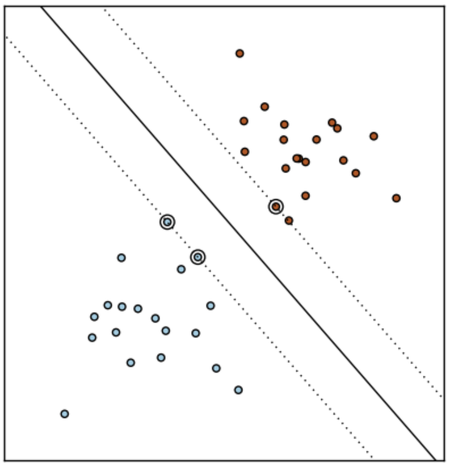

Figure 3.2: Example of separating hyperplane with a hard margin. The circled data points constitute the support vectors.

There is an infinite number of possible separating hyperplanes and the goal is to find the hyperplane that maximizes the margin between the two classes. This is equivalent to solving the following quadratic optimization problem:

$$\min_{w,b} \frac{1}{2} w^T w \quad s.t. \, \forall y_i : y_i(w^T x_i + b) \geq 1 \tag{3.1}$$

In the best case scenario the data is linearly separable which makes it possible to select two parallel hyperplanes with a distance between them as large as possible as seen in Figure 3.2. This is the *hard margin* case where the maximum-margin hyperplane becomes the hyperplane in the middle of the two class bordering hyperplanes. The data points closest to the optimal hyperplane are called *support vectors*.

If the data is truly linearly separable, a hard margin implies a perfect classification. However, this is often not the case in real world applications and forcing a hard margin will result in an overfitted model. This is remedied by relaxing the constraint into the *soft margin* version of the problem.

The soft margin version modifies the optimization problem in 3.1 to the primal formulation as in equation 3.2 where ζ is the *hinge loss* function and C is a regularisation constant.

$$\min_{w,b,\zeta} \frac{1}{2} w^T w + C \sum_{i=1}^{n} \zeta_i \quad s.t. \, \forall y_i : y_i(w \cdot x_i + b) \geq 1 - \zeta_i \tag{3.2}$$

$$\zeta_i = max(0, 1 - y_i(w \cdot x_i + b)$$

Solving the primal formulation optimally is computationally intractable for higher dimensions, which motivates the use of the dual formulation in 3.3 instead. This simplified problem is obtained by solving the Lagrangian dual of the primal and is a quadratic function of α subject to linear constraints, making it efficiently solvable using quadratic programming algorithms.

$$\max_{\alpha} \sum_{i}^{n} \alpha_i - \frac{1}{2} \sum_{i}^{n} \sum_{j}^{n} y_i y_j \alpha_i \alpha_j (x_i \cdot x_j) \quad s.t. \; \sum_{i}^{n} y_i \alpha_i = 0, \; \forall i : 0 \leq \alpha_i \leq C \tag{3.3}$$

The weight vector is obtained from the α-values as in 3.4. Note that α_i is only non zero for datapoints x_i chosen as support vectors. The bias is calculated as $b = w \cdot x_i - y_i$ for one of the support vectors x_i.

$$w = \sum_{i}^{n} \alpha_i y_i x_i \tag{3.4}$$

When the SVM is trained, the class of a new datapoint x_{new} can be predicted according to equation 3.5.

$$predict(x_{new}) = sgn(w \cdot \varphi(x_{new}) + b) = sgn([\sum_{i=1}^{n} \alpha_i y_i k(x_i, x_{new})] + b) \tag{3.5}$$

3.1.1 Kernels

The solution above works well with linear separable data, but the use of SVMs that only handles linear separation is limited. Even though the data is not linearly separable in the current feature space R^N, it might be in a higher feature space R^M. Thus by transforming the input to a higher feature space, the data might be separable while still using the above formulation. Increasing the feature space can however be very expensive in computation and space, especially in cases where M grows exponentially with N.

This is solved using *kernel functions*, which is a function $k : R^N \times R^N \to R$ that given two vectors implicitly computes their inner products in a higher dimension R^M without explicitly transforming the vectors to R^M. Kernels provides a measure of similarity between two vectors if it satisfies equation 3.6 where $\varphi(x)$ is the vector transformed to R^M.

$$k(x_i, x_j) = \varphi(x_i) \cdot \varphi(x_j) = k(x_j, x_i) \geq 0 \tag{3.6}$$

Using kernels, a dataset can be implicitly transformed to a higher-dimensional feature space without a memory overhead and only a small increase in complexity. The kernels are incorporated into the model by replacing the inner products in equation 3.3 with $k(x_i, x_j)$.

Kernel functions

To use kernels to model the solution to linearly separable data, a kernel defined by the function $k(v, w) = v^T w$ is used. It is useful if the feature space is high dimensional and each individual feature is informative[16]. If this is the case, it is likely that the decision boundary can be represented as a linear combination of the original features which renders the implicit transformation redundant. However, for data where all features are not individually informative, non-linear kernels such as polynomial or *radial basis function* kernels perform better.

The polynomial kernel is defined as $k(v, w) = (v^T w)^d$ where d is the polynomial degree and $w, v \in R^N$. It provides a way to measure similarity of not only individual features but also their combinations. The implicit feature space is in $R^{d \times N}$.

The *radial basis function* (RBF) kernel is a universal kernel, defined as a Gaussian function of two vectors as equation 3.7 where γ is a parameter usually set to $1/2\sigma^2$ where σ is.. .

$$k(v, w) = exp[-\gamma \|w - v\|^2] \tag{3.7}$$

RBF kernels have an infinite implicit feature space [17]. Because it is a universal kernel, SVMs using RBF kernels guarantees globally optimal predictors for a large set of classification problems as long as an appropriate regularization parameter C is chosen [17][18].

3.2 Gaussian Mixture Models

A *Gaussian mixture model* (GMM) is a probabilistic model which assumes that the distribution of observations can be modelled as a combination of Gaussian distributions. Mixture models are used in statistics to model sub-populations within the full population and can be thought of as a generalization of k-mean clustering to incorporate information of statistical properties of the data points.

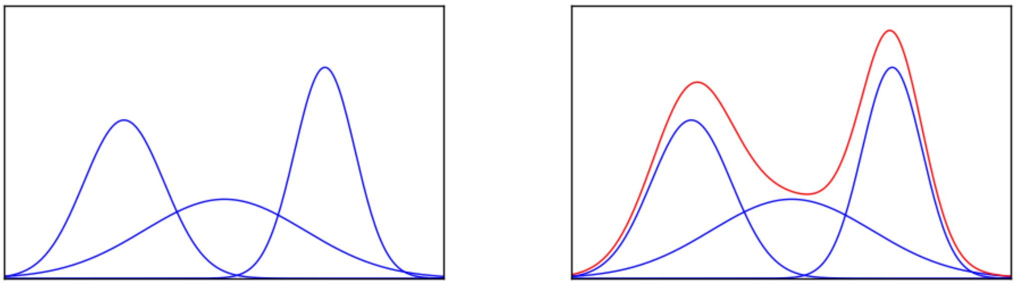

Figure 3.3: GMMs enables modelling a complex distribution (the top distribution in the rightmost figure) as a combination of Gaussian distributions (each of the three distributions in the leftmost figure).

Formally a distribution f is a Gaussian mixture of K components modeled as a linear combination of K Gaussian distributions with density function

$$f(x) = \sum_{k=1}^{K} \omega_k N(x; \mu_k, \sigma_k^2) \tag{3.8}$$

where ω_k is the mixture weights satisfying $\sum_{k=1}^{K} \omega_k = 1$ and $\omega_k > 0$ for $k = 1, ..., K$. Thus the model is defined by the parameters $\lambda = \{\omega_1, ..., \omega_k, \mu_1, ..., \mu_k, \sigma_1^2, ..., \sigma_k^2\}$.

Training a GMM entails estimating the μ, σ^2 and w parameters for all K components and is typically done using the EM-algorithm [19] for finding *maximum a postoriori* (MAP) estimates of the parameters. The EM-algorithm is an iterative algorithm where each iteration involves an expectation (E) step where log-likelihood expectation functions are formed using the current parameters and a maximization (M) step to find new parameters that maximizes the expected log-likelihoods from the E-step.

In multi-class classification problems separate GMMs can be trained for the different classes and classification of an input x means finding the most likely model. Two models can be compared by performing a likelihood ratio test as follows. Given an input x, the null hypothesis model λ_{null} and the alternative model λ_{alt}, the likelihood ratio for λ_{null} is calculated as in equation 3.9 and expresses how many times more likely the data are under λ_{null} than λ_{alt}. If the likelihood ratio for λ_{null} is higher than a constant C it is accepted, otherwise λ_{null} is rejected in favour of the alternative model.

$$\Lambda(x) = \frac{P(x|\lambda_{null})}{P(x|\lambda_{alt})} \tag{3.9}$$

3.3 Deep Belief Networks

Deep belief networks (DBN), proposed by Hinton et al [20, 21, 22], is a methodology for designing and training deep neural networks. Its many layers of non-linearities have been shown to be able to express both highly variant and highly non-linear functions better than shallow models such as SVMs [23]. One of the strengths of the approach is its ability to discover underlying regularities of multiple features while still being highly generalizable to new observations [24]. In recent years, DBNs has been applied successfully in varied areas of research such as image recognition [25], natural language processing [26] and speech analysis[24].

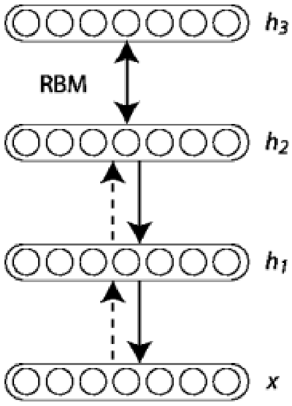

Figure 3.4: Overview of DBN architecture.

The high level architecture of a DBN can be seen in Figure 3.4. It is composed of several layers of stacked *restricted boltzmann machines* (RBM), which will soon be described in more detail. The top layer represents the input layer and is bidirectionally connected to the second layer, forming an associative memory [21].

The network is defined as the following joint distribution, where x is the input and g^i the hidden variables at layer i:

$$P(x, g^1, g^2, ..., g^l) = P(x \mid g^1)P(g^1 \mid g^2) \cdots P(g^{l-2} \mid g^{l-1})P(g^{l-1} \mid g^l) \tag{3.10}$$

All the conditional layers $P(g^i \mid g^{i+1})$ are factorized conditional distributions for which computation of probability and sampling are easy. In Hintons original paper [21], each hidden layer g^i is considered a binary random vector with n^i elements g^i_j :

$$P(g^i \mid g^{i+1}) = \prod_{j=1}^{n^i} P(g^i_j \mid g^{i+1}) \tag{3.11}$$

$$P(g^i_j = 1 \mid g^{i+1}) = sigm(b^i_j + \sum_{k=1}^{n^{i+1}} W^i_{kj} g^{i+1}_k)$$

where $sigm(t) = \frac{1}{1+e^{-t}}$, b^i_j is the bias of unit j in layer i and W^i is the *weight matrix* for layer i.

As noted, the building blocks of the network are RBMs, an example of which can be seen in figure 3.5. An RBM consists of two layers, the top one denoted as the visible layer and the second as the hidden layer. Each node in the visible layer is directionally connected to every node in the hidden layer and there are no connections between nodes within layers.

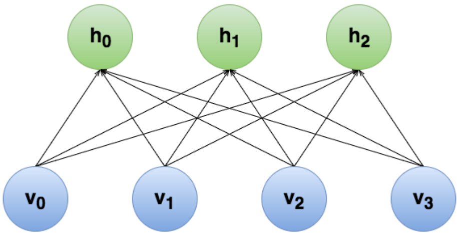

Figure 3.5: A single restricted boltzmann machine.

The joint distribution of the two layers are defined as in equation 3.12, where Z is the normalization constant for the distribution, b is the bias vector for the visible units, c the bias vector the hidden units and W the weight matrix for the layer.

$$P(\mathbf{v}, \mathbf{h}) = \frac{1}{Z} e^{\mathbf{h}' W \mathbf{v} + b' \mathbf{v} + c' \mathbf{h}} \tag{3.12}$$

The energy function is defined as the negation of the argument in the exponent:

$$energy(\mathbf{v}, \mathbf{h}) = -\mathbf{h}' W \mathbf{v} - b' \mathbf{v} - c' \mathbf{h} \tag{3.13}$$

Hinton et al [21] introduced a greedy layer-wise unsupervised learning algorithm which greatly improved the efficiency of DBN training and made using them feasible in practice. It begins by setting g^0 to the input vector **x**, and then sampling the empirical distribution of the RBM using *contrastive divergence* (CD). By continually passing each RBM's result up through the network, **x** is transformed to a new representation in layer g^l. CD is a method for quickly estimating the gradient of the log-likelihood of an RBM using Gibbs Markov chain sampling over the visible and hidden units in the RBM, with a low sampling periodicity ($k = 1$ in most cases).

After this initial pre-training step, the entire network is fine-tuned using a supervised gradient descent algorithm on the negative log-likelihood cost function given a prediction y for an input vector **x**. This prediction is given using a logistic regression classifier to classify an input vector **x** into a class y given the result of feeding **x** up through the network to layer g^l. This can be seen as either an independent system component or as the final layer of the network.

3.3.1 Performance metrics

Some of the most widely used performance metrics for binary classifiers are accuracy, precision and recall. These can all be derived from the following classification matrix, which describes the different outcomes of a classification.

	Value Positive	Value Negative
Classified Value Positive	True Positive (TP)	False Positive (FP)
Classified Value Negative	False Negative (FN)	True Negative (TN)

Table 3.1: Classification outcomes of a binary classifier.

Using these different cases, the metrics are calculated as following.

$$\text{ACCURACY} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FN} + \text{FP}} \qquad \text{PRECISION} = \frac{\text{TP}}{\text{TP} + \text{FP}} \qquad \text{RECALL} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

Where

- Accuracy describes the proportion of classified values that were correct.
- Precision describes the proportion of values classified as positive that were positive.
- Recall describes the proportion of positive values that were correctly classified as positive.

The relationship between precision and recall are often described by the use of either a *precision/recall* (PR) curve or *receiver operating characteristics* (ROC) curve [27].

Precision and recall can also be combined to form the *F1-score*, which is defined as

$$F1 = 2 * \frac{\text{PRECISION} * \text{RECALL}}{\text{PRECISION} + \text{RECALL}}$$

Another metric which is of importance when deciding between different models is how the training time scales with the input size as well as the time it takes for the model to classify new observations.

By combining all of these different metrics, a well rounded measure on the quality of the classification model can be described. Depending on the context of the classifier, some of the metrics might be of more importance that others. For example, if the classification needs to be performed in a live setting (say, a live speech translator) the time the classifier needs to classify new observations is very important.

4

Methods

This section introduces the proposed system and describes how each of the components fits together. It continues with an in-depth description of each individual component and what considerations has gone into developing them.

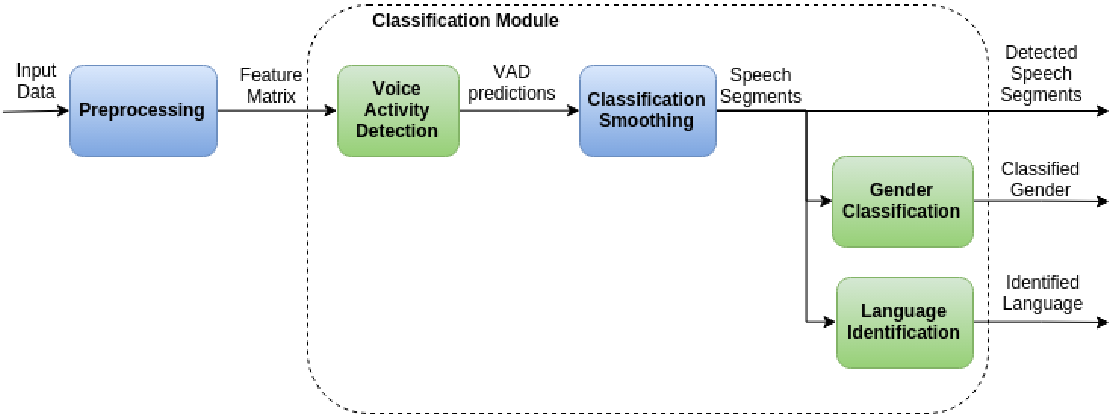

Figure 4.1: The proposed system from input data to classifications.

4.1 Datasets and data selection

The data used as input to a classification model is paramount to the models quality. There is always a risk that the model performs well on a particular dataset during experimentation but performs substantially worse during real world use. This hints that the model solves a different problem than the one asked for.

Due to the nature of the media that the trained models will classify, it is not viable to perform experiments using real data. This forced us to look for alternative data to use as input during experimentation. In addition to being representative, there are a few more requirements that the data needs to satisfy; it needs to be tagged with the proper information needed for the classification tasks, it has to follow a license that allows for use in both academic as well as commercial applications and it has to be free of charge due to constraints of the thesis. The data used came from two sources, which will be described in more detail in the upcoming sections.

4.1.1 VoxForge speech recordings

The VoxForge speech recordings dataset is a collection of crowdsourced transcribed speech recordings, initially created for use in developing open source speech recognition engines. The dataset consists of user submitted recordings created using an online tool which provides feedback on the quality of the recording. Each recording is collection of ten audio files containing a single utterance. A summary of the metrics for each language can be seen in table 4.1.

All recordings are licensed under the GNU general public license version 3 [1].

	English	**Spanish**	**French**	**Italian**	**Russian**	**Total**
Utterances	80007	22885	18591	9612	8883	139 978
Speakers	1182 (3257)	460 (696)	283 (625)	196 (336)	199 (469)	2320 (5383)
Male	1081	285	261	184	186	1997
Female	95	74	24	14	14	221
Undefined gender	41	23	43	20	22	160
Max length	82,25s	37,13s	27,13s	16,81s	147,5s	147,5s
Min length	0,4s	1,32s	0,75s	1,5s	0,78s	0,4s
Average length	4,98s	8,03s	5,92s	6,82s	9,73s	7,09s
Total length	110h 37m	51h 04m	30h 34m	18h 12m	24h 01m	234h 28m

Table 4.1: Summary of the VoxForge dataset. The parenthesized value in the speakers row is because recordings can be labeled as "Anonymous". The first value is the number of uniquely labeled speakers, and the value within parentheses is this number plus the number of speakers labeled "Anonymous (where the same anonymous speaker can provide several recordings). This means that the real number of speakers lies somewhere in the span between the two values.

4.1.2 DEMAND

The Diverse Environments Multichannel Acoustic Noise Database (DEMAND) is a collection of multichannel recordings from a diverse set of noise environments [28]. The dataset contains five minutes of recordings from 18 different environments, recorded using a microphone array of 16 microphones. This totals to 90 minutes of audio in 16 slightly different variations, giving 24 hours of total audio.

The different noise environments encompass five main themes; domestic environments, natural environments, office environments, public environments and transportation environments. A description for each of the included environments can be found in [28]. The dataset is licensed under the Create Commons Attributions-ShareAlike 3.0 license [2].

[1] http://www.gnu.org/licenses/gpl-3.0.html

[2] http://creativecommons.org/licenses/by-sa/3.0/legalcode

4.2 Technical information

The project was written in the Python programming language, which was chosen for its wide availability of high quality machine learning and scientific libraries as well as the prevailing notion that it is well suited for experimentation. The machine learning components of the thesis was performed using two different libraries; *scikit-learn* [29] for general purpose machine learning and Theano [30] [31] to develop deep learning networks. Both of these rely heavily on the *scipy* [32] and *numpy* [33] libraries, which include functionality for high performance scientific computing.

For data handling, a system using an sqlite database and Googles protocol buffer objects were developed.

All experiments throughout the thesis are conducted on the same machine with the following specifications:

CPU: 2 × Intel Xeon CPU E5-2620 2.00 GHz 12 cores
RAM: 56 GB, 1334 Hz
OS: Windows Server 2012 R2 (64-bit)

4.3 Merging of data sources

For some experiments, data from different sources were merged to create the final dataset. This provides several benefits: full control of the data and labeling is maintained, it can be verified that the *signal to noise ratio* (SNR) is the same over the entire dataset and more data can be synthesized by generating permutations of the data.

In order to minimize the number of frames incorrectly labeled as speech after merging, the silent parts of the speech files are initially removed. The speech segments are then normalized in order to achieve a certain SNR to the background noise by applying gain on the segment. This is performed by satisfying equation 4.1 where P_s and P_N is the respective power of the speech and noise and s_{dB} and N_{dB} their loudness in decibel.

$$SNR_{dB} = 10\, log_{10}(\frac{P_s}{P_N}) \Leftrightarrow SNR_{dB} = s_{dB} - N_{dB} \tag{4.1}$$

4.4 Data preprocessing

Before using data as input to the models they need to be put through a series of preprocessing steps, this section describes them. The steps used varies between experiments.

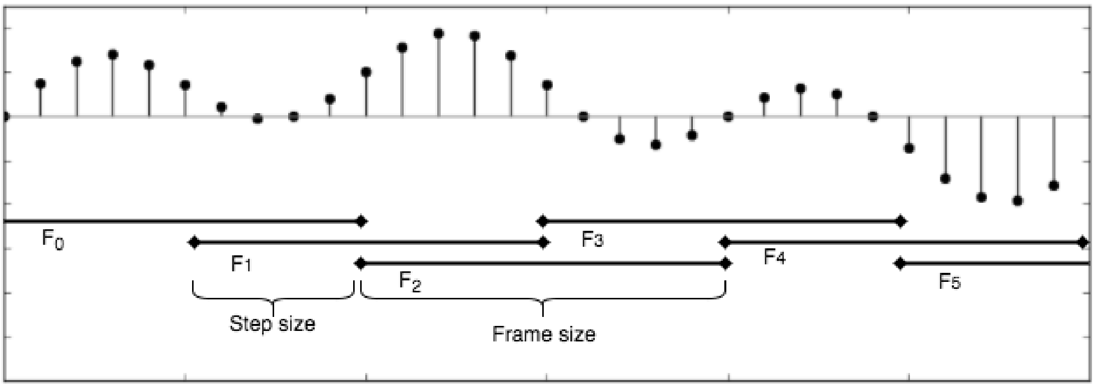

Figure 4.2: Example of segmentation of samples into frames F_0-F_5, with *frame size* of 10 samples and *step size* of 5.

4.4.1 Feature extraction

The feature extraction module converts an audio signal into series of data points which are fed as input to the classification models. The signal and its corresponding class labels are segmented into frames using a specified size and step length, as described in Section 2.2.

Each data point consists of a set of extracted features concatenated into a *feature vector* which represents the acoustic properties of the frame. The features extracted are a subset of the features described in section 2.3 and vary between experiments.

Special consideration is taken when extracting features for the last frame of an audio file. The system defines the last frame as the first frame which extends beyond the length of the audio file. For this to make sense, the samples in the frame are padded with the value of the last sample.

The module facilitates merging of feature vectors calculated using different window sizes, as long as they have the same step length. This means that each frames feature vector contains information on different time spans originating at the same point.

4.4.2 Feature Scaling

Because the different acoustic features yield values in very varied ranges, the experiments include different forms of scaling so that they can be used in conjunction with each other. The different forms of scaling are: scaling each value between zero and one for each feature, scaling each value between zero and one for each frame and scaling each feature so that it has centers on the mean and has a unit variance.

4.4.3 Silence removal

For some experiments, removal of silent parts from utterances were performed before feeding them as training data to the model. This is performed to maximize the quality of the training data by minimizing the risk of feeding the model with mislabeled speech frames. It is important that silence removal is performed after calculating the full feature vectors as some of the features are based on the upcoming frames and if the silent frames were removed before extraction, the features would no longer be representative of real speech.

The silence removal is based on energy and works as follows. Given a list of frames, the energy for each frame is first calculated as described in section 2.3.1. All frames with energy under a threshold τ are considered silent and are removed from the feature matrix. The threshold is defined in equation 4.2 where c is a constant satisfying $0 < c < 1$.

$$\tau = \max_{\forall x \in X}(energy(x)) \cdot c \tag{4.2}$$

4.5 Classification smoothing

When classifying audio per frame on acoustic features, the resulting classifications is often fragmented. This is expected, especially if the features used for classification does not incorporate acoustic features over a longer time span. This can be remedied by applying a *classification smoothing* scheme on the given classifications. Such a scheme can be designed in a number of ways, from simple ratio comparisons such as the one implemented in the thesis to more advanced schemes based on hidden Markov models [34] or state machines [35].

The smoothing scheme used in the thesis scans the classification vector linearly and makes a new decision based on the surrounding classifications. The algorithm takes a window size that defines how many upcoming and past frames to take into consideration when making a decision. For each frame, the fraction of the past and upcoming classifications that are the same as the current classification are computed. If both fractions are under a set threshold, the frame is reclassified.

5

Voice activity detection

Voice activity detection (VAD) is the task of separating areas of speech and non-speech in an audio signal, which is a problem that arises in a wide range of areas such as speech coding in digital telephony, signal-to-noise estimation and speech recognition.

5.1 Background and approach

If the audio signal is relatively clean, i.e contains low or no background noise and no non-speech acoustic events such as ringing doorbells or barking dogs, VAD is a simple problem that can be solved using a speech detector based entirely on energy. However, if such noise is present in the background, the complexity of finding segments containing speech is greatly increased and requires the use of more sophisticated analysis approaches. The thesis has developed two approaches in parallel, which are presented in this chapter.

5.1.1 Deep belief network based VAD

In recent years, research on the application of DBNs on speech classification has shown good results outperforming traditional acoustic modeling techniques such as GMMs [36]. The approach in this thesis is influenced by the work done by Zhang et al [24], but while their primary focus lies in showing that DBNs can work as a powerful tool to fuse a wide range of acoustic features we have a focus on developing a system which is better tailored for VAD. They used a merged feature vector consisting of 273 acoustic features calculated from three different window lengths.

The proposed solution differs mainly on the reduced size of the feature vector and the parameters used for the network during training. The reduced vector also necessitates a revised network architecture to successfully fuse the features.

5.1.2 Support vector machine based VAD

VAD solutions based on support vector machines are prevalent in research. Enqing et. al. implemented a rudimentary implementation [37]. Their results were later improved upon and refined by Ramirez et. al [38]. SVMs with RBF kernels were applied to the VAD task by Kinnunen et. al.[39], with MFCC and their derivatives as input features.

The proposed solution is based on [39] but the RBF kernel was approximated to decrease the computational complexity during training.

Kernel approximation

A disadvantage of the SVM approach is that standard kernelized SVMs do not scale well with bigger datasets[40]. This is a problem in the VAD task since the dataset needs to include examples from several different background environments and for each there needs to be enough speech data to capture the characteristics of speech. This problem is remedied by approximating the explicit feature map of the RBF kernel and performing linear classification, instead of applying the kernel trick. The data is explicitly mapped using a randomized mapping $z : R^d \rightarrow R^D$, where D is much lower than the actual explicit feature space of the kernel, so that the inner products of two transformed vectors approximates the kernel according to Equation 5.1. The advantage of doing this instead of using feature maps implicitly is that explicit feature mappings can be better suited for online training and decreases the training time with larger datasets [41].

$$k(x_1, x_2) = \varphi(x_1) \cdot \varphi(x_2) \approx z(x_1)^T z(x_2) \tag{5.1}$$

In the proposed solution, the explicit feature mappings are approximated by Monte Carlo approximation of the RBF kernels Fourier transform using the RBF sampler supplied by *scikit learn*[1], which implements a variant of the *random kitchen sink procedure* proposed in [42]. The RBF sampler takes a γ (see RBF γ in 3.1.1) as well as an integer parameter, number of components, that determines the new feature mapping dimensionality. The resulting feature mappings are used in a linear SVM trained with stochastic gradient descent.

5.2 Experimental setting

A dataset consisting of Voxforge utterances and DEMAND background environments were merged using the methodology described in Section 4.3. Sixteen five minute long audio files were created for each of the 17 environments, totaling 1360 minutes. Each merged audio file consists of 50% speech segments on top of noise and 50% only noise. Separate datasets were create for each of three different signal to noise ratios; 0dB, 5dB and 10dB.

In the experiments, unintelligible speech such as background chattering is considered non-speech. Noise environments involving unintelligible speech are included in the experiments to capture some harder use cases.

[1] http://scikit-learn.org/stable/modules/generated/sklearn.kernel_approximation.RBFSampler.html

All experiments were performed using five fold cross validation. For each fold, 80% of the frames from each environment were merged to create the entire training set while 20% of the frames for each environment were kept separated in environment wise test sets.

The test sets contain consecutive frames, which enable the use of classification smoothing scheme. The scheme applied is of the type presented in 4.5 with a window size of 40 ms.

5.2.1 SVM settings

Features for a single frame were calculated using a frame window size of 25ms and a step size of 10ms. The features extracted from the audio signal are presented in Table 5.1. The kernel was approximated using RBF sampler with γ set to 1 and a number of components set to ten times the original vector length. This results in new feature vectors of length 390, used by the linear classifier to approximate the result of using an RBF kernelized SVM in the original feature space.

Feature	**Dimensions**
MFCC	13
DMFCC	13
DMFCC	13
Total	**39**

Table 5.1: Features used in SVM based VAD experiments.

5.2.2 DBN settings

Features for a single frame were calculated using a combination of different window sizes, as described in Section 4.4.1. All use a step size of 10 ms but the three different sizes included are window sizes of 25, 200 and 400 ms.

The feature vector used can be seen in Table 5.2, they were selected loosely based on the features used in [24].

The network used consists of two hidden layers, the first consisting of 50 nodes and the second of 20 nodes. The learning rate for the pre training algorithm was 0.001 and the fine-tuning algorithm 0.01. The max number of epochs for both the pre training and fine-tuning were set to 100 epochs.

Feature	Dimensions
$Pitch_{25}$	1
LPC_{25}	12
DFT_{25}	16
DFT_{200}	16
DFT_{400}	16
$MFCC_{25}$	20
$MFCC_{200}$	20
$MFCC_{400}$	20
Total	**121**

Table 5.2: Features used in DBN based VAD experiments. Subscripted value is the window size in ms.

5.3 Results

The total accuracies of the models are the average classification accuracy of all supported noise environments.

Environment	Accuracy			Precision			Recall			F1-score		
SNR	0	5	10	0	5	10	0	5	10	0	5	10
Kitchen	86.2	89.5	90.1	81.3	86.0	85.9	94.5	94.7	96.2	87.4	90.1	90.8
Living	87.1	91.0	93.0	87.2	92.8	92.0	86.9	88.9	94.1	87.0	90.8	93.0
Washing	93.1	94.1	94.7	95.2	95.3	94.6	90.9	92.9	94.9	93.0	94.1	94.7
Field	90.7	92.2	92.5	88.9	91.0	90.4	93.6	94.1	95.6	91.2	92.5	92.9
Park	79.2	86.9	87.7	73.1	82.9	82.9	95.3	94.5	96.3	82.7	88.3	89.0
River	71.3	81.4	91.2	90.4	93.9	93.5	47.4	67.1	88.5	62.1	78.2	90.9
Hallway	90.6	92.5	92.9	88.5	91.3	90.7	93.7	94.3	95.9	91.0	92.8	93.2
Office	91.0	92.9	93.0	87.8	91.0	90.6	95.4	95.3	96.1	91.4	93.1	93.2
Cafeteria	71.9	85.3	89.3	66.2	85.3	86.8	89.6	86.0	92.6	76.1	85.6	89.6
Restaurant	73.5	79.5	90.4	89.0	96.8	95.6	53.9	61.2	84.7	67.1	75.0	89.8
Station	69.9	81.0	92.0	99.0	98.9	98.0	40.7	63.1	85.9	57.7	77.0	91.5
Cafe	82.1	89.0	91.7	79.9	91.2	90.6	86.6	86.9	93.5	83.1	89.0	92.0
Square	87.6	91.0	91.5	84.3	89.6	89.0	92.6	92.8	94.9	88.2	91.2	91.8
Traffic	87.5	91.0	93.5	96.2	96.1	94.6	78.5	85.8	92.5	86.4	90.7	93.5
Bus	90.1	92.4	92.9	87.3	90.9	90.7	94.3	94.6	95.9	90.7	92.7	93.2
Car	89.4	92.0	92.5	85.2	89.4	89.4	95.8	95.6	96.6	90.2	92.4	92.9
Metro	86.8	87.7	92.1	86.6	85.3	90.3	87.1	89.9	94.3	86.8	87.5	92.3
Total	**84.0**	**88.8**	**91.8**	**86.2**	**91.0**	**90.9**	**83.3**	**86.9**	**93.4**	**83.1**	**88.3**	**92.0**

Table 5.3: Results for SVM experiments after applying the classification smoothing scheme.

Environment	**Accuracy**			**Precision**			**Recall**			**F1-score**		
SNR	0	5	10	0	5	10	0	5	10	0	5	10
Kitchen	78.0	84.9	86.1	74.2	82.0	82.0	86.8	90.0	93.2	80.0	85.8	87.2
Living	81.0	93.2	95.6	78.2	96.2	98.2	86.0	90.0	92.9	81.9	93.0	95.5
Washing	86.3	95.6	96.8	82.5	97.5	98.9	92.6	93.7	94.7	87.2	95.5	96.7
Field	86.8	96.0	97.0	83.1	98.0	99.1	93.3	94.2	95.0	87.9	96.1	97.0
Park	75.0	87.3	92.4	69.4	83.6	91.0	93.3	94.2	94.8	79.6	88.6	92.9
River	73.5	89.3	93.2	76.0	96.1	97.5	68.7	82.0	88.7	72.2	88.5	92.9
Hallway	84.8	94.2	95.9	80.7	95.1	97.5	92.2	93.3	94.4	86.1	94.2	95.9
Office	86.0	95.2	96.6	81.8	96.7	98.6	93.1	93.7	94.6	87.1	95.2	96.5
Cafeteria	72.2	87.0	93.3	68.7	87.8	96.1	81.9	86.1	90.2	74.7	86.9	93.0
Restaurant	67.2	84.4	90.7	69.5	94.2	97.1	61.9	73.5	84.0	65.5	82.6	90.1
Station	76.1	91.2	94.8	77.3	96.3	98.2	74.7	85.9	91.4	76.0	90.8	94.7
Cafe	78.3	91.1	94.8	76.4	93.6	97.6	82.9	88.6	92.0	79.5	91.1	94.7
Square	83.0	93.4	95.5	77.7	93.5	96.9	92.9	93.4	94.1	84.6	93.5	95.5
Traffic	82.4	94.2	95.9	80.4	97.7	98.6	86.3	90.7	93.2	83.3	94.1	95.8
Bus	84.5	93.5	95.9	80.1	93.3	96.6	92.6	94.0	95.4	85.9	93.7	96.0
Car	86.1	94.1	97.1	81.5	93.6	98.1	93.8	94.8	96.1	87.3	94.2	97.1
Metro	80.5	88.5	94.5	76.3	84.6	94.7	88.6	92.2	94.4	82.0	88.2	94.5
Total	**80.1**	**91.4**	**94.5**	**77.2**	**92.9**	**96.3**	**86.0**	**90.0**	**92.9**	**81.2**	**91.2**	**94.4**

Table 5.4: Results for DBN experiments after applying the classification smoothing scheme.

	Fine-tuning epochs		
Fold	**SNR dB 0**	**SNR 5 dB**	**SNR 10 dB**
One	51	65	93
Two	93	100	75
Three	100	61	100
Four	5	100	100
Five	47	100	41

Table 5.5: Fine-tuning epochs run per cross validation fold with max epoch limit set to 100.

5.4 Analysis

As can be seen in Tables 5.3 and 5.4, the models achieve good performance for most environments and the DBN outperforms the SVM approach on both SNR 5 and 10. In these SNRs, the DBN generally has high precision values for all environments, meaning most of the frames classified as speech are indeed speech. However, in the most difficult SNR, the DBNs precision drops sharply for all environments. This means that a lot of non-speech is classified as speech. This is surprising, since the intuition for the use of DBNs was it would be able to find non-linear patterns in the input data and thus being able to differentiate speech even in harder settings.

In the intended application of the classifier, missing speech segments is worse than classifying non-speech incorrectly. Therefore, the most important metric of the models are the recall value. As can be seen in the results, both classifiers have a pretty good average recall value for SNR 5 and 10. However, the SVM classifiers recall values wildly fluctuates between environments while the DBNs values are more stable.

The most difficult environment for both approaches are three environments with unintelligible background chatter; Restaurant, Café and Cafeteria. Since they include background chatter, the noise exhibits similar characteristics as speech, especially in lower signal-to-noise ratios. In the higher SNR levels, the unintelligible speech will be lower than the intelligible speech and therefore the values of the DFT and MFCC are also lower. This is not the case for SNR 0, since the unintelligible speech is at the same level as the intelligible speech. However, both approaches performs satisfactory for the hard environments in higher signal-to-noise ratios.

The running times between the two approaches are very different, while training the DBN takes roughly 15 hours the SVM only takes a matter of minutes. While this might seem like a deal breaker, this is not necessarily the case since training is only performed once. The time to classify a new observation for both approaches are insignificant.

Fine-tuning the DBN network amounts to running the fine-tuning algorithm until the squared error converges, the ratio of change drops below a specified threshold or a specified limit on the number of fine-tuning epochs are reached. As can be seen in table 5.5, for the three experiments the number of fine-tuning epochs ran until the specified limit was reached (100 epochs) in roughly half of the cases. This means that re-running the experiments with a higher max epoch count most likely will result in higher classification scores at the expense of longer running times. Since this step is only performed during training of the network it is recommended to increase the maximum epochs in a production environment.

6

Language classification

Language classification can be seen as either the tasks of language identification or verification [7]. Identification deals with identifying which language is spoken, only given the audio signal while verification tries to confirm or deny a hypothesized language. This chapter describes the proposed *language identification* (LID) system, the experiments performed using this system and an analysis of the results achieved.

6.1 Background and approach

The architecture of the proposed LID system can be seen in Figure 6.1 and are based on GMMs, which are trained on acoustic and temporal acoustic data from a set of supported languages. The solution is based on research proposed by Reynolds et al [43], where they train a *universal background model* (UBM) which forms the basis for each class-specific GMM. Reynolds et al [43] successfully applied the methodology to the related problem of speaker verification and it has also been applied on the problem of language identification [44]. It has since been one of the dominant techniques for acoustic based language identification [7].

The main reason for using GMMs to model speech is its ability to model arbitrary densities, which is needed to model the complicated distributions of speech [45]. In the proposed solution, each mixture models a distribution of acoustic properties. Each of these distributions represent an acoustic class, which is an abstract concept not to be confused with a language class.

The underlying hypothesis when modeling language identification using GMMs is that each language has a unique distribution of acoustic classes. If this assumption is correct, the quest becomes trying to estimate the parameters of this underlying distribution for each language. This is is done by training a GMM for each language with examples of speech from that particular language. With these, we can perform language classification by comparing a new example to each of the trained models and selecting the one with the highest likelihood.

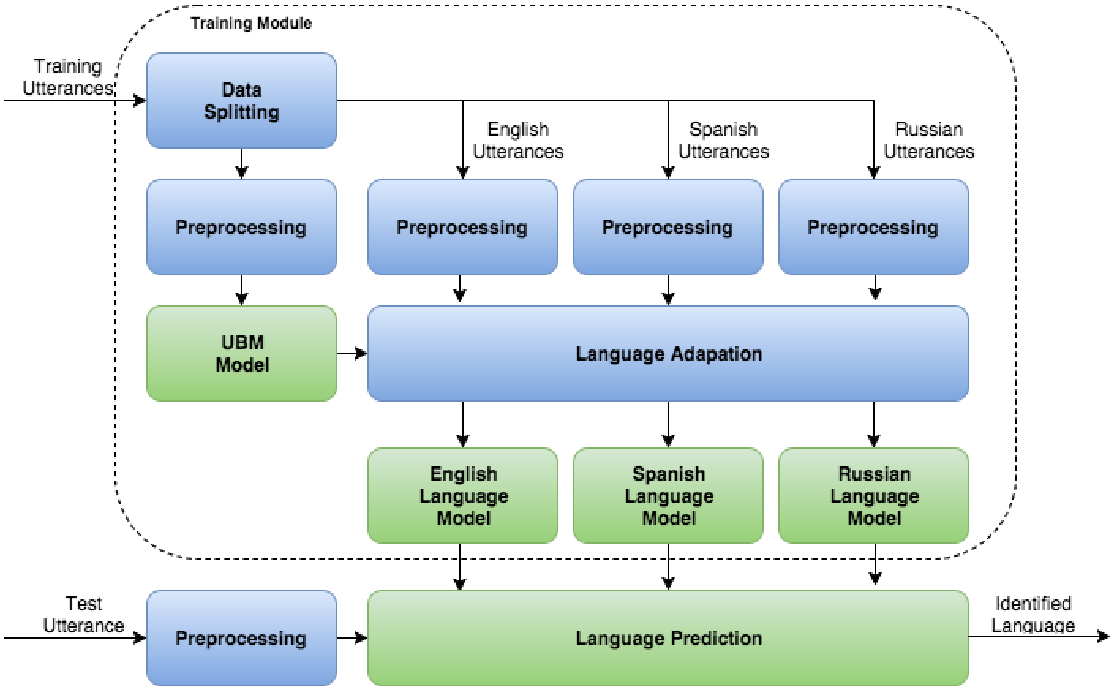

Figure 6.1: An overview of the proposed UBM-GMM LID system for an experiment including three languages, but can be generalized to more languages.

Universal Background Model

The universal background model(UBM) is a GMM trained on a subset of the data from each of the classes, whose main purpose is to capture the language independent distribution of all speech. This is especially useful when performing classification using the language models since the UBM can be used to represent the general characteristics of speech. This provides a way for the model to indicate that an utterance has the properties of speech but does not show any of the unique characteristics the languages trained on. Another positive effect of using the UBM as a base for the language specific GMMs is that it makes the training of the system more efficient, both in training time and the amount of data needed to train the language specific models [43].

In order for the UBM to be universal, the data used for its training need to be diverse enough to cover most of the language independent characteristics of speech. It is especially important that the training data has minimal bias to any of the languages, since a bias in the UBM can make the whole LID system biased. Because of this, roughly equal amount of speech data from all languages is used in the UBM train set.

Language Adaptation

As stated, each supported language is represented by a GMM λ_l providing a unique distribution of acoustic classes for each language l. The language models are adapted from the UBM using a form of *Bayesian adaptation* based on a modified version of the EM-algorithm as presented in [43]. The algorithm used during the adaptation procedure is described below, given a train set $X_l = \{x_1, \ldots, x_N \mid x_k \in R^d\}$ containing N speech frames as feature vectors of d dimensions from language to adapt.

STEP ONE Given a UBM with m mixtures, initiate a language GMM λ_l with the UBMs parameters.

STEP TWO Predict the language model's posterior probability for each data point x_k under each mixture given the current parameters.

$$P(i|x_k, \lambda_l) = \frac{w_i p_i(x_k|\lambda_l)}{\sum_{j=1}^{m} w_j p_j(x_k|\lambda_l)}, \quad for\ i = 1, ..., m \tag{6.1}$$

STEP THREE Compute the sufficient statistics for the parameters μ_i, w_i and σ_i^2.

$$c_i = \sum_{k=1}^{N} P(i|x_k) \tag{6.2}$$

$$E_i[X_l] = \frac{1}{c_i} \sum_{k=1}^{N} P(i|x_k) x_k \tag{6.3}$$

$$E_i[X_l^2] = \frac{1}{c_i} \sum_{k=1}^{N} P(i|x_k) x_k^2 \tag{6.4}$$

STEP FOUR Update the parameters of each mixture i. α is an adaptation coefficient controlling the balance between former and new estimates, with r as a relevance factor. After all mixture weights have been updated they are scaled by multiplying with a scaling factor γ to ensure that they sum to unity.

$$\hat{\mu}_i = \alpha_i E_i[X_l] + (1 - \alpha_i)\mu_i \tag{6.5}$$

$$\hat{\sigma}_i^2 = \alpha_i E_i[X_l^2] + (1 - \alpha_i)(\sigma_i^2 + \mu_i^2) - \hat{\mu}_i^2 \tag{6.6}$$

$$\hat{w}_i = \alpha_i \frac{c_i}{N} + (1 - \alpha_i) w_i \tag{6.7}$$

$$\alpha = \frac{c_i}{c_i + r} \qquad \gamma = \frac{1}{\sum_{j=1}^{m} \hat{w}_j}$$

STEP FIVE Iterate from STEP TWO if the likelihood change is under a set threshold, otherwise the adaptation has converged and is completed.

Predicting through log likelihood ratios

To predict the spoken language of a speech segment X, the likelihood of X given the pre-trained models, including the UBM are used to find the most likely model. For convenience the log likelihood is used and the log likelihood of X given a model λ_l is defined as the average log likelihood of all descriptive (as described in the end of this section) frames as in Equation 6.8.

$$ln\ P(X|\lambda) = \frac{1}{N}\sum_{k=1}^{N} ln\ P(x_k|\lambda_l) \tag{6.8}$$

For each language model a log likelihood ratio is computed as in Equation 6.9, where the UBM is used as the alternative hypothesis. Using the UBM as the alternative hypothesis gives the benefits of not having to compute the models mutual log likelihood ratios and since the UBM is trained to model the language independent characteristics of speech, it could be interpreted as representing *any* of the languages. A language can be verified given a threshold c, if $\Lambda(X|\lambda_l) > c$ and rejected if $\Lambda(X|\lambda_l) < c$. In all experiments, $c = 0$. This means that if a frame has a higher likelihood of belonging to a specific language than the UBM it is considered classifiable. The identified language is the language with maximum log likelihood ratio.

$$log\ \Lambda(X|\lambda_l) = log\ P(X|\lambda_l) - log\ P(X|\lambda_{ubm}) \tag{6.9}$$

As noted in the first paragraph, the frames that are included when computing the average log likelihood of X given a model λ_l are those which have been deemed as descriptive. A frame is labeled as descriptive if the frames log likelihood ratio is positive for at least one language model. The reasoning behind this is the notion that if the log likelihood ratios for all language models are negative, the frame does not include any corroborative evidence for a particular language model and can thus be excluded from the decision for X without loosing important information.

6.2 Experimental setting

Classification were performed using a dataset consisting of VoxForge utterances from each of the included languages. The main experiment included three languages - English, Russian and Spanish, and the secondary experiment added two more - Italian and French. The second experiment was designed to test the systems scalability to additional languages. The two additional languages belong to the same language family as Spanish, with the justification that similar languages should be more difficult to differentiate between acoustically.

Features for a single frame were calculated using a frame window size of 25ms and step size of 10ms. The features included in the experiments can be seen in Table 6.1. The SDC

Feature	Dimensions
MFCC	13
DMFCC	13
SDC	26
Total	**52**

Table 6.1: Features used in GMM based language experiments.

features were calculated before the removal of silent frames with number of coefficients k set to 3 and the distance p between them set to 7.

To minimize the risk of feeding the model silent data points, for example the beginning and end of recordings and pauses between sentences, each of the utterances were preprocessed using the silence removal algorithm from Section 4.4.3. This removal is performed after the extraction of features since SDC encodes temporal information that would be lost or corrupt if they were extracted after removal. Following removal step, a total of five hours of speech per language were selected as data for the experiments.

Each UBM mixture is initiated using a diagonal covariance matrix with zero mean and identity covariance. During training, the maximum EM iterations were set to 50.

Only the μ parameter of each mixture was updated during model adaptation. The motivation for this was found in [43], where experiments showed that only updating μ resulted in significantly decreased running times without loss of classification performance. Because their models were applied to speaker verification and not language identification, brief experiments were performed to make sure that their findings were transferable to this domain. By training GMMs with 64 components and comparing results from updating all parameters and only updating μ we found that this was the case and it was assumed to hold for higher components counts as well.

Both experiments were performed using five fold cross validation, stratified over languages. For each fold, 80% of the entire dataset constituted the training set while 20% were kept separate for testing. From each language specific training set, roughly 10% were added to the training set for the UBM.

The threshold for the log likelihood ratio test was set to zero, which means that if a frame has a higher likelihood of belonging to a specific language than the UBM it is considered classifiable.

6.3 Results

The results for the first experiment can be seen in Table 6.2. In both Table 6.2 and 6.3, the column "Unclassified" is the percentage of the language files which has been deemed unclassifiable. This is the case when the mean log likelihood ratio for the entire utterance is negative for all language models.

Components	Language	Accuracy	Precision	Recall	F1-score	Unclassified
16	**English**	-	85.48	68.38	75.98	1.70
	Spanish	-	76.52	80.55	78.84	1.38
	Russian	-	65.59	78.19	71.34	1.93
	Total	**74.61**	**75.87**	**75.70**	**75.38**	**1.67**
32	**English**	-	87.62	74.07	80.28	1.24
	Spanish	-	81.26	80.40	80.82	1.44
	Russian	-	68.25	81.66	74.35	1.44
	Total	**78.02**	**79.04**	**78.71**	**78.48**	**1.36**
64	**English**	-	89.71	79.48	84.28	0.76
	Spanish	-	84.52	83.39	83.96	1.42
	Russian	-	73.64	84.45	78.68	1.31
	Total	**82.00**	**82.62**	**82.44**	**82.30**	**1.10**
128	**English**	-	91.13	82.18	86.42	0.34
	Spanish	-	87.21	85.01	86.10	0.60
	Russian	-	77.98	90.12	83.61	0.59
	Total	**84.44**	**85.44**	**85.77**	**85.37**	**0.48**
256	**English**	-	92.2	84.81	88.35	0.04
	Spanish	-	89.76	86.39	88.04	0.02
	Russian	-	77.98	90.12	83.61	0.13
	Total	**86.78**	**86.64**	**87.10**	**86.66**	**0.06**
512	**English**	-	92.89	86.59	89.63	0.04
	Spanish	-	90.98	87.47	89.19	0.04
	Russian	-	80.26	91.43	85.48	0.00
	Total	**88.23**	**88.00**	**88.49**	**88.10**	**0.03**

Table 6.2: Classifiation performance for first experiment. Accuracy is only presented for all languages, while the other metrics are presented per language.

Components	Language	Accuracy	Precision	Recall	F1-score	Unclassified
512	**English**	-	86.66	85.52	84.54	0.30
	Spanish	-	74.50	81.52	77.85	0.36
	Russian	-	70.60	87.26	78.05	0.44
	French	-	84.95	79.00	81.86	0.74
	Italian	-	75.68	59.44	66.59	3.91
	Total	**78.14**	**78.48**	**78.55**	**77.78**	**1.09**

Table 6.3: Classification performance for the second experiment, where the model is taught five languages. Accuracy is only presented for all languages, while the other metrics are presented per language.

Languages	Components	UBM training	Adaptation	Test	Total Runtime
3	16	00:10:53	00:02:19	00:01:10	00:14:22
3	32	00:36:55	00:04:47	00:01:37	00:43:19
3	64	01:17:43	00:13:29	00:01:56	01:33:08
3	128	02:35:46	00:16:20	00:02:27	02:54:33
3	256	05:01:26	01:13:22	00:03:19	06:18:17
3	512	09:54:44	03:32:07	00:05:08	13:31:59
5	512	15:41:35	05:32:20	00:14:45	21:28:40

Table 6.4: Running times for the two experiments averaged over all five folds. The running time for model adaptation consists of adapting all languages and the test running time is the time to make predictions for the entire test set.

		Predicted		
		En	**Sp**	**Ru**
Actual	**En**	86.63	3.60	9.77
	Sp	4.60	87.51	7.89
	Ru	5.50	3.07	91.43

	Predicted				
	En	**Sp**	**Ru**	**Fr**	**It**
En	82.77	2.23	7.67	3.98	3.35
Sp	3.96	81.82	5.61	3.16	5.45
Ru	4.05	2.00	87.65	3.13	3.17
Fr	3.00	2.36	9.52	79.59	5.53
It	7.20	18.38	6.95	5.61	61.86

Table 6.5: Confusion matrix in percent over predictions for the two experiments with 512 mixture components. Each row sums to 100 and the value at each row gives the classification distribution of a language dataset.

6.4 Analysis

As can be seen in Table 6.2, the number of mixture components used greatly influences the performance of the classifier as well the number of unclassifiable speech segments. The higher the count, the better our classifier performs. This enforces the notion that GMMs are well suited to model the complex distributions of speech, and by increasing the component count the model can represent more complex distributions.

But as Table 6.4 shows, there is a trade-off between the component count and the running time of the model. This is expected since the computation complexity scales linearly with the number of mixtures for the UBM training, model adaptation and predictions. The main contributor to the total running time is the training of the UBM using the EM-algorithm.

The results also highlight the benefit of using an UBM approach compared to stand-alone GMMs. By basing the language specific models on the UBM, which has been trained on the general characteristics of speech, the training time per added language is substantially reduced resulting in a much shorter overall training time. Without the UBM each language model would need to be trained from scratch, increasing the running time with the equivalence of running the EM algorithm to convergence with an input of the entire language data. In the proposed solution, the added input only consists of the size of of the language datas UBM split which is considerably smaller.

A common error in both experiments is incorrectly classifying other languages as Russian, which has the highest recall and the lowest precision. This means that most of the Russian speech segments were correctly classified, but a lot of speech from the other languages was incorrectly identified as Russian. A possible explanation for this could be that sounds present in all languages has been "assigned" to the Russian language model, resulting in the seen behavior.

The second experiment shows that the system's performance decrease with additional languages, which hints that an acoustic approach is not sufficient for language identification between acoustically similar languages.

The most difficult language for the model to classify is Italian, where only 59.4% of the speech segments are correctly classified. It is also the only language with a considerable number of unclassifiable segments. An explanation for the high unclassifiable rate could be that the Italian language model has not successfully captured the intricacies of the language. The reason for this could be that not enough data has been provided or that the quality of the data needs to be improved. Even though this could explain the unclassified rate, the main contributor to the low performance of the model is that it is often incorrectly classified as other languages.

The most common error is incorrectly classifying Italian as Spanish, occurring for almost 20% of the speech segments. Some degree of classification errors within the Romance family can be expected, but the interesting part is that erroneously identifying Spanish as Italian is not nearly as common. This could be a result of the two languages sharing distinguishing sounds, but that the shared sounds are more inherent to Spanish. For example, if a specific word exists in both the Spanish and Italian vocabulary but is more frequently used in Spanish, frames within that word will have a higher likelihood for the Spanish model than the Italian model.

7

Gender classification

Identifying the gender of a speaker is of big interest in several different applications and there has been considerate research to solve the problem.

7.1 Background and approach

The proposed solutions in the thesis are based on two approaches which have been described in more detail in earlier sections, support vector machines (SVM) and Gaussian mixture models (GMM). They were chosen because they have been applied in earlier research on the gender classification problem and to explore how well the models developed for speech and language identification generalize.

7.1.1 SVM based gender classifier

Several earlier papers propose slightly different variations of SVM for gender classification. Sedaaghi presents a comparative study of several different models applied on gender classification, including both SVM and GMM [46]. Kotti applies SVMs on gender classification of emotional speech with good results [47].

In the proposed SVM gender classifier, SVMs with RBF-kernels are fitted to labeled frames of male and female speech. To decrease the computational complexity the RBF-kernels are approximated the same way as in the VAD experiments in 5.1.2. The classification of a speech segment is performed by classifying each frame separately and classifying the segment as the majority class.

7.1.2 GMM-UBM based gender classifier

GMMs has also been successfully applied on the problem. Zeng et al achieved very good results (in the top 90% range) by applying a GMM approach [48]. Metze et al used a similar approach, but where they produced a combined age and gender classifier, which renders their results incomparable to our [49].

The proposed solution utilizes an almost identical approach as in the GMM-UBM language identification system described in 6, with the exception of adapting gender models instead of language specific models. The classification methodology remains unchanged.

7.2 Experimental setting

The experiments were performed using utterances between three and five seconds long from the VoxForge dataset. It includes utterances from all of the five languages described in Section 4.1.1 to ensure that the classifier is language independent. To minimize the amount of non-speech frames fed to the models, silence was removed using the method descried in Section 4.4.3. Following silence removal, 86 minutes of continuous speech from each gender class remained to train and test the models.

Both of the approaches use a feature vector containing pitch and MFCCs. Zeng et al shows that pitch is a strong indicator of gender, but also that there is some overlap between high pitched males and low pitched females [48]. To mitigate this overlap, we include the MFCCs and the final vector can be seen in Table 7.1. The feature vectors were calculated using a frame size of 25 ms and step size of 10 ms and both experiments were performed using five fold cross validation.

SVM settings

Each utterance's feature matrix was scaled per vector. The RBF kernel was approximated using RBF sampler with number of components set to 1000 and γ set to 2. The transformed RBF vectors are used in a linear SVM with hinge loss function, to approximate a RBF kernelized SVM in the original feature space.

GMM-UBM settings

From the training data, 10% of each gender is moved to the UBM training data. Each utterance's feature matrix was scaled per feature. The classifiers used 512 component mixtures, and each UBM mixture is initiated using a diagonal covariance matrix with zero mean and identity covariance. During training, the maximum EM iterations were set to 100.

As in the LID experiments, only the μ is updated during model adaptation and the log likelihood ratio threshold was zero.

Feature	Dimensions
Pitch	1
MFCC	13
Total	**14**

Table 7.1: Features used in the gender experiments.

7.3 Results

Gender	**Accuracy**	**Precision**	**Recall**	**F1-score**
Male	-	88.0	97.2	92.4
Female	-	96.8	86.8	91.6
Average	91.9	92.5	92.0	91.9

Table 7.2: Performance metrics for gender classification using SVM.

Gender	**Accuracy**	**Precision**	**Recall**	**F1-score**
Male	-	95.9	93.9	94.9
Female	-	93.8	95.8	94.8
Average	94.8	94.8	94.8	94.8

Table 7.3: Performance metrics for gender classification using GMM.

7.4 Analysis

As can be seen in the results, both of the approaches perform the classification well but the GMM based approach has both the best accuracy and consistent results between genders.

The SVM on the other hand does not have as consistent results. It is biased towards male speech, with the effect that male precision and female recall are low. This indicates that there is an overlap between the acoustic features for some male and female speakers which could not be separated using the approximated feature mapping. However, when conducting experiments using a higher number of components in the RBF sampler the bias was consistent. This hints that the features chosen are not sufficient for gender differentiation using this SVM approach.

The running times for the two approaches on the supplied input are low. The training for the SVM is under one minute and for the GMM-UBM roughly one hour, while the test time for both approaches are insignificant.

8

Discussion and future work

8.1 Comparison with previous research

All proposed solutions have to some extent been based on earlier research. The aim of this section is to place the achieved results in context. It is however important to note since the data differs, comparing results might give an inaccurate view of the models quality.

8.1.1 Voice activity detection

Zhang et al. developed a solution based on deep belief networks [24] which are evaluated on the AURORA2 corpus [50], which includes similar noise environments in the same signal to noise ratios as the proposed system. They also include wide-ranging comparison of their results with other research and approaches available at the time of writing, some of which we have included in our comparison as well. Table 8.1 present their results achieved with a similar setting on similar background noises. Note that the results for the thesis are after applying the rudimentary classification smoothing scheme. As noted earlier, we can not draw any direct conclusions from the comparison, as they are evaluated on different data, but they are included to put the results in context.

Environment	SVM	DBN	Zhang DBN [24]	Zhang SVM [24]	Ramirez [38]
Restaurant	75.9	84.4	83.6	82.09	69.59
Car	92.0	94.1	87.0	86.34	77.68
Subway	87.7	88.5	85.8	83.58	73.16
Average	85.2	89.0	85.5	84.0	72.14

Table 8.1: Comparison of VAD results with a signal-to-noise ratio of 5 dB. Average is presented for only these three environments.

8.1.2 Language identification

The language identification task is a very open problem and the evaluation methodology differs a lot depending on the problem formulation. The National Institute of Standards and Technology (NIST) has conducted bi-annual evaluations of automatic language recognition systems since 1996 and results from these evaluations are often presented in previous research. However, in the recent NIST LRE tasks the task is to, given a segment of speech and a pair of languages, decide which one of these two languages is spoken in the speech segment. Since this is not the same as our problem formulation, using results from the competition in a comparison with the proposed system would give an inaccurate view and is therefore excluded.

8.1.3 Gender classification

Table 8.2 present results for the gender classifier and some of the highlighted earlier research which the solution is loosely based upon.

Both of the alternative SVM solutions are evaluated on the Danish emotional speech (DES) [51] dataset, which consists of a total of 30 minutes of speech from two male and two female speakers. This dataset differs significantly from the dataset used for the proposed solution, which consists of hundreds of speakers. The presented results from both Seedaghi et al [46] and Kotti et al [47] is the solution they denote as "SVM1", which both are based on and RBF kernel. As seen, the achieved performance is a bit below their results which can be attributed to our models bias towards male speakers, as described in the the gender analysis in section 7.4.

Zeng et al [48] applied a GMM based approach to the problem, but not one based on a universal background model as in the proposed solution. They are evaluated on the TIMIT speech corpus [52], which consists of 276 male and 184 female speakers. This setting is more comparable to the corpus used in the proposed solution than in the SVM based solutions. As noted, since they use different evaluation data, we can not draw any direct conclusions from the comparison but it is included to put the results in context.

	SVM	**Seedaghi [46]**	**Kotti [47]**	**GMM**	**Zeng [48]**
Male	97.2	95.6	N/A	93.9	97.9
Female	86.8	93.9	N/A	95.8	98.7
Total	**92.0**	**94.8**	**99.4**	**94.8**	**98.2**

Table 8.2: Comparison of gender classification results with research.

8.2 Relating to CSA media

The intended use case of the classification systems is automatically classifying audio within large seizes, in particular involving child sexual abuse (CSA) media. Even though this has been considered throughout the thesis work, it has not really had a large impact on the proposed system. This section highlights two specific considerations in regards to this.

The background environments included in the VAD experiments are very varied and it might seem like some of them could be irrelevant for the intended application. However, large seizes in CSA investigations can involve non-CSA media relevant for the investigation. This motivates solving the general problem by including all environments instead of only environments which intuitively are more prevalent in CSA media.

In order for the proposed solutions to be specifically geared towards CSA media, they would need to be evaluated on data which includes prepubescent speech. This has not been done, mainly due to the lack of available speech corporas including such speech. As it is, the proposed systems would probably not perform well if evaluated on prepubescent speech. The acoustics of speech change a lot during puberty, which affects all classification tasks in general and the gender task in particular. This necessitates a revised approach altogether, for example training separate models and using an age classifier to select the appropriate one.

To conclude, we expect that the underlying theory of the proposed solutions would still be valid, but implementation and execution choices would need to be revised to properly facilitate prepubescent speech.

8.3 Data challenges

Selecting what data to use has been the biggest challenge during the thesis. For obvious reasons, it was never an option to test the models on CSE media. The preferred alternative would have been to use some of the benchmarking datasets often seen referred to in research, such as one of the NIST LRE evaluation sets or the TIMIT acoustic dataset[52]. As noted in section 8.1, this would have enabled a direct comparison of our results with research which would have validated the results. But due to cost and time constraints, we could not gain access to them. This forced us to look at alternative options, which had its pros and cons.

The main strength and issue with the VoxForge dataset is that it consists of user submitted data. The effect of this is that the speech is varied both when it comes to unique speakers and recording environments, but its quality is not assured and it could include both mislabeled and low quality recordings.

A drawback of the DEMAND dataset is that it includes several background scenarios which might not be as interesting to the use case of the classifiers. However, they were included to make the models well-versed and to increase the total time available for data.

As mentioned in Section 5.2, the data used for the VAD experiments were generated through merging Voxforge and DEMAND data. However, since they were recorded in different recording environments the generated segments containing active speech will not be identical to segments where the speech was present in the background recording from the beginning. The intuition was that if the unnatural noise is inaudible, it would not have a significant effect on the VAD classifiers decisions but there is a risk that this is not the case.

Further on, it is important that the data is considerate of sub populations within the data when selecting data for a particular problem. For example, when performing language classification, it is also important to verify that both genders are represented roughly equally. This was an insight we had after the fact, with the implication that the trained language models might be biased towards the skewed sub populations of the data.

8.4 The effects of removing frames

As described in the UBM-GMM section, frames with a negative log likelihood ratio for all adapted models were excluded during classification. The intuition behind this is that frames that do not reinforce any of the adapted languages can safely be excluded without affecting the final outcome of the classifier, which is almost correct if all new observations will belong to one of the seen classes. The information lost in this case is how much the negative ratios for class differ from each other, which could be used to inform a classification decision.

If new observations comes from an unseen class, which could occur in language classification, a different outcome is achieved. By removing frames, three things happen; the confidence in the UBM will decrease, the confidence in the language models will increase and the number of unclassifiable utterances will be minimized. This makes the model biased towards classifying as one of the seen classes which can be both positive and negative, depending on the intended use of the classifier. Especially whether or not it can be expected that untrained classes will be present as new observations.A model can not be expected to be trained on all possible languages, but new genders will probably not materialise out of thin air.

8.5 Future work

Several possible future extensions which would be interesting to evaluate has been found during the thesis work, but due to either scope or time constraints these have been left out. In addition to solving related problems such as approximate age identification or speaker verification, we want to highlight some of the most interesting extensions.

8.5.1 Advanced smoothing scheme

As noted in Section 4.5, the smoothing scheme implemented in the thesis is rudimentary and there are more advanced schemes described in research. We have seen that applying smoothing schemes have a big impact on the final metrics, which hints that giving the implementation some more thought could definitely improve the achieved metrics. A more advanced smoothing scheme could also be designed to enable setting a limit on minimum segment lengths found by the VAD, which would be desirable in a production environment.

8.5.2 Evaluating different audio lengths

A different metric which would be interesting to explore, especially when comparing competing approaches as during gender classification, is how well the classifier performs on speech segments of differing lengths. It is expected that there is a drop in performance when classifying two second segments as compared to ten second segments. How much of difference there is and if there is difference between different models would be interesting to present.

8.5.3 Extending language classifier

As discussed in the analysis of the language classification results, the main issue of the proposed system is that it does not scale well when adding new languages - particularly acoustically similar languages. It gets harder to distinguish languages acoustically using our current approach when the acoustic space gets more crowded. We have identified a few promising changes and extensions to our approach.

Our findings hints that an acoustic classifier might not be enough for a large scale LID system. A future extension could be incorporating higher order abstractions of speech into the models, which could include phonotactic, prosodic, lexical and synctactic features. This is consistent with advanced LID systems described in literature, especially [7].

However, a purely acoustic approach has its merits. A promising extension would be to implement a hierarchical LID system, where a tree structured is used and at each level the classifier distinguishes between language subgroups. The best subgroups is not necessarily the traditional language families, but should be obtained by clustering languages on their pair-wise acoustic distances. Each tree level in such system is a classifier, and its outcome determines which subgroup the utterance most likely belongs to, and thus which classifier to use in the next level. This is done down the tree until a language leaf is reached.

The classifiers for the different language groups must not necessarily be of the same type. For instance, a GMM-UBM classifier could be used on the first level to classify in larger subgroups. The different subgroups could then use different types of features and/or model types depending on what is best for that specific subgroup.

The main benefit of a hierarchical LID system is its modularity and scaling capabilities. Adding additional languages will mostly affect the bottommost classifier levels. A hierarchical LID system could also include levels with high order feature classifiers. A further extension to the hierarchical LID system would be to add an additional level under the language leafs to classify on accents within the languages.

8.5.4 Cross pollination between classifiers

Using the results of one classification task as an input to the other classification tasks would be an interesting extension. Ponder that we have a functioning age classifier, it would be reasonable that using its output as input to the gender classifier would improve performance.

9

Conclusion

Solutions to detect active speech in noisy environments and classifying the speakers gender and spoken language has been presented. These have all been grounded in earlier research and evaluated on a series of evaluation metrics where they perform satisfactory, with the exception that the language classifier does not gracefully scale with additional languages. In conclusion, it is shown that machine learning models are well suited for speech classification.

Bibliography

[1] M. P. Gelfer and V. A. Mikos, “The relative contributions of speaking fundamental frequency and formant frequencies to gender identification based on isolated vowels,” *Journal of Voice*, vol. 19, no. 4, pp. 544–554, 2005.

[2] D. O’shaughnessy, *Speech communication: human and machine*. Wiley-IEEE Press, 2000.

[3] Ish Ishwar, “Spectrogram of the american english vowels,” 2007. [Online; accessed March 03, 2016 from https://commons.wikimedia.org/wiki/File:Spectrogram$_iua-.png]$.

[4] R. O. Coleman, “A comparison of the contributions of two voice quality characteristics to the perception of maleness and femaleness in the voice,” *Journal of Speech, Language, and Hearing Research*, vol. 19, no. 1, pp. 168–180, 1976.

[5] C. E. Shannon, “Communication in the presence of noise,” *Proceedings of the IRE*, vol. 37, no. 1, pp. 10–21, 1949.

[6] M. J. Carey, E. S. Parris, and H. Lloyd-Thomas, “A comparison of features for speech, music discrimination,” in *Acoustics, Speech, and Signal Processing, 1999. Proceedings., 1999 IEEE International Conference on*, vol. 1, pp. 149–152, IEEE, 1999.

[7] E. Ambikairajah, H. Li, L. Wang, B. Yin, and V. Sethu, “Language identification: a tutorial,” *Circuits and Systems Magazine, IEEE*, vol. 11, no. 2, pp. 82–108, 2011.

[8] R. Bachu, S. Kopparthi, B. Adapa, and B. Barkana, “Separation of voiced and unvoiced using zero crossing rate and energy of the speech signal,” in *American Society for Engineering Education (ASEE) Zone Conference Proceedings*, pp. 1–7, 2008.

[9] D. Gerhard, *Pitch extraction and fundamental frequency: History and current techniques*. Regina: Department of Computer Science, University of Regina, 2003.

[10] J. W. Cooley and J. W. Tukey, “An algorithm for the machine calculation of complex fourier series,” *Mathematics of computation*, vol. 19, no. 90, pp. 297–301, 1965.

[11] S. B. Davis and P. Mermelstein, “Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences,” *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 28, no. 4, pp. 357–366,

1980.

[12] D. O'Shaughnessy, "Linear predictive coding," *Potentials, IEEE*, vol. 7, no. 1, pp. 29–32, 1988.

[13] J. Durbin, "The fitting of time-series models," *Revue de l'Institut International de Statistique*, pp. 233–244, 1960.

[14] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *Proceedings of the fifth annual workshop on Computational learning theory*, pp. 144–152, ACM, 1992.

[15] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[16] K. P. Murphy, *Machine learning: a probabilistic perspective*. MIT press, 2012.

[17] I. Steinwart, "On the influence of the kernel on the consistency of support vector machines," *The Journal of Machine Learning Research*, vol. 2, pp. 67–93, 2002.

[18] C. A. Micchelli, Y. Xu, and H. Zhang, "Universal kernels," *The Journal of Machine Learning Research*, vol. 7, pp. 2651–2667, 2006.

[19] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the royal statistical society. Series B (methodological)*, pp. 1–38, 1977.

[20] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[21] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[22] G. Hinton, "A practical guide to training restricted boltzmann machines," *Momentum*, vol. 9, no. 1, p. 926, 2010.

[23] Y. Bengio, P. Lamblin, D. Popovici, H. Larochelle, *et al.*, "Greedy layer-wise training of deep networks," *Advances in neural information processing systems*, vol. 19, p. 153, 2007.

[24] X.-L. Zhang and J. Wu, "Deep belief networks based voice activity detection," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 21, no. 4, pp. 697–710, 2013.

[25] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng, "Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations," in *Proceedings of the 26th Annual International Conference on Machine Learning*, pp. 609–616, ACM, 2009.

[26] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *Proceedings of the 25th international conference on Machine learning*, pp. 160–167, ACM, 2008.

[27] J. Davis and M. Goadrich, “The relationship between precision-recall and roc curves,” in *Proceedings of the 23rd international conference on Machine learning*, pp. 233–240, ACM, 2006.

[28] J. Thiemann, N. Ito, and E. Vincent, “The diverse environments multi-channel acoustic noise database: A database of multichannel environmental noise recordings,” *The Journal of the Acoustical Society of America*, vol. 133, no. 5, pp. 3591–3591, 2013.

[29] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, *et al.*, “Scikit-learn: Machine learning in python,” *The Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[30] F. Bastien, P. Lamblin, R. Pascanu, J. Bergstra, I. J. Goodfellow, A. Bergeron, N. Bouchard, and Y. Bengio, “Theano: new features and speed improvements.” Deep Learning and Unsupervised Feature Learning NIPS 2012 Workshop, 2012.

[31] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio, “Theano: a CPU and GPU math expression compiler,” in *Proceedings of the Python for Scientific Computing Conference (SciPy)*, June 2010. Oral Presentation.

[32] E. Jones, T. Oliphant, P. Peterson, *et al.*, “SciPy: Open source scientific tools for Python,” 2001–. [Online; accessed 2016-02-15].

[33] S. Van Der Walt, S. C. Colbert, and G. Varoquaux, “The numpy array: a structure for efficient numerical computation,” *Computing in Science & Engineering*, vol. 13, no. 2, pp. 22–30, 2011.

[34] J. Sohn, N. S. Kim, and W. Sung, “A statistical model-based voice activity detection,” *Signal Processing Letters, IEEE*, vol. 6, no. 1, pp. 1–3, 1999.

[35] J. G. Wiese, M. G. Shlipak, and W. S. Browner, “The alcohol hangover,” *Annals of internal medicine*, vol. 132, no. 11, pp. 897–902, 2000.

[36] A.-r. Mohamed, G. E. Dahl, and G. Hinton, “Acoustic modeling using deep belief networks,” *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 20, no. 1, pp. 14–22, 2012.

[37] D. Enqing, L. Guizhong, Z. Yatong, and Z. Xiaodi, “Applying support vector machines to voice activity detection,” in *Signal Processing, 2002 6th International Conference on*, vol. 2, pp. 1124–1127, IEEE, 2002.

[38] J. Ramírez, P. Yélamos, J. M. Górriz, C. G. Puntonet, and J. C. Segura, “Svm-enabled voice activity detection,” in *Advances in Neural Networks-ISNN 2006*, pp. 676–681, Springer, 2006.

[39] T. Kinnunen, E. Chernenko, M. Tuononen, P. Fränti, and H. Li, “Voice activity detection using mfcc features and support vector machine,” in *Int. Conf. on Speech and Computer (SPECOM07), Moscow, Russia*, vol. 2, pp. 556–561, 2007.

[40] A. Rahimi and B. Recht, “Random features for large-scale kernel machines,” in *Advances in neural information processing systems*, pp. 1177–1184, 2007.

[41] “Kernel approximation.” `http://scikit-learn.org/stable/modules/kernel_approximation.html`. [Accessed 2016-03-14].

[42] A. Rahimi and B. Recht, “Weighted sums of random kitchen sinks: Replacing minimization with randomization in learning,” in *Advances in neural information processing systems*, pp. 1313–1320, 2009.

[43] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn, “Speaker verification using adapted gaussian mixture models,” *Digital signal processing*, vol. 10, no. 1, pp. 19–41, 2000.

[44] E. Wong, J. Pelecanos, S. Myers, and S. Sridharan, “Language identification using efficient gaussian mixture model analysis,” in *Australian International Conference on Speech Science and Technology*, vol. 4, pp. 7–6, 2000.

[45] D. A. Reynolds and R. C. Rose, “Robust text-independent speaker identification using gaussian mixture speaker models,” *Speech and Audio Processing, IEEE Transactions on*, vol. 3, no. 1, pp. 72–83, 1995.

[46] M. H. Sedaaghi, “A comparative study of gender and age classification in speech signals,” *Iranian Journal of Electrical Electronic Engineering*, vol. 5, 2009.

[47] M. Kotti and C. Kotropoulos, “Gender classification in two emotional speech databases,” in *Pattern Recognition, 2008. ICPR 2008. 19th International Conference on*, pp. 1–4, IEEE, 2008.

[48] Y.-M. Zeng, Z.-Y. Wu, T. Falk, and W.-Y. Chan, “Robust gmm based gender classification using pitch and rasta-plp parameters of speech,” in *Machine Learning and Cybernetics, 2006 International Conference on*, pp. 3376–3379, IEEE, 2006.

[49] F. Metze, J. Ajmera, R. Englert, U. Bub, F. Burkhardt, J. Stegmann, C. Muller, R. Huber, B. Andrassy, J. G. Bauer, *et al.*, “Comparison of four approaches to age and gender recognition for telephone applications,” in *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, vol. 4, pp. IV–1089, IEEE, 2007.

[50] H.-G. Hirsch and D. Pearce, “The aurora experimental framework for the performance evaluation of speech recognition systems under noisy conditions,” in *ASR2000-Automatic Speech Recognition: Challenges for the new Millenium ISCA Tutorial and Research Workshop (ITRW)*, 2000.

[51] I. S. Engberg and A. V. Hansen, “Documentation of the danish emotional speech database des,” *Internal AAU report, Center for Person Kommunikation, Denmark*, p. 22, 1996.

[52] W. M. Fisher, V. Zue, J. Bernstein, and D. S. Pallett, “An acoustic-phonetic data base,” *The Journal of the Acoustical Society of America*, vol. 81, no. S1, pp. S92–S93, 1987.