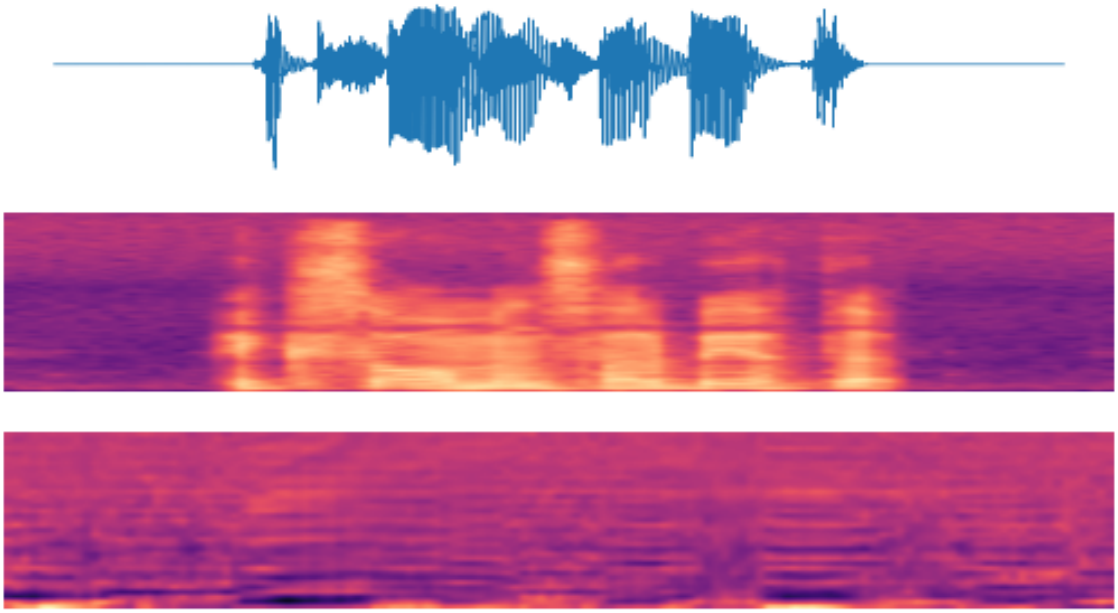


Machine Learning Methods for Limited Data Speaker Identification

The Design and Implementation of a Novel Real-Time System

Master's thesis in Complex Adaptive Systems

MATTIAS PATRICKS

Department of Electrical Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2019

Master's thesis 2019

Machine Learning Methods for Limited Data Speaker Identification

The Design and Implementation of a Novel Real-Time System

Mattias Patricks

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Electrical Engineering
Division of Signal processing and Biomedical engineering
Chalmers University of Technology
Gothenburg, Sweden 2019

Machine Learning Methods for Limited Data Speaker Identification
The Design and Implementation of a Novel Real-Time System
Mattias Patricks

Supervisor: Adam Andersson, Syntronic
Examiner: Irene Yu-Hua Gu, Department of Electrical Engineering

Master's Thesis 2019
Department of Electrical Engineering
Division of Signal processing and Biomedical engineering
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: The pulse-code modulated audio signal of the word 'examensarbete' and its corresponding spectrogram and cepstrogram.

Typeset in LaTeX
Gothenburg, Sweden 2019

Machine Learning Methods for Limited Data Speaker Identification
The Design and Implementation of a Novel Real-Time System
Mattias Patricks
Department of Electrical Engineering
Chalmers University of Technology

Abstract

With the recent development of speech-to-text algorithms it has becoming increasingly plausible to generate subtitles to a live stream or to transcribe the content of a meeting in real time. Most of these algorithms specialise in this speech-to-text task, while omitting the sometimes important information of who is saying what. The objective of this thesis project is to design and implement a real time speaker identification system to complement existing speech-to-text systems. The thesis compares using Gaussian mixture models (GMM) versus multi-layer perceptrons (MLP) as classifiers for mel-frequency cepstral coe□cients (MFCCs). The primary performance experiment was conducted on a continuous conversation-like test based on the LibriSpeech ASR corpus. For 12 seconds of training data per speaker the system achieved a 99 % identification accuracy for 2 speakers, but the performance declined linearly down to 88 % for 12 speakers. The result was therefore deemed unsatisfactory for the intended application and more modern features and classifiers are recommended.

Keywords: speaker identification, speaker recognition, limited data, real time, mel-frequency cepstral coe□cients, Gaussian mixture model, multi layer perceptron, neural networks, deep learning.

Contents

List of Figures

List of Tables

1

Introduction

Speaker recognition is the automatic recognition of a speaker's identity via speech. The recognition make use of speech differences between speakers that arise from both physiological characteristics of speech production, e.g. size and shape of the vocal folds and vocal tract, and speaking style, e.g. intonation and rhythm [1]. These differences further tend to vary between the genders and different age groups, and depend on the speaker's emotional and physical state [2]. Within speaker recognition, a distinction is made between speaker *identification* (identifying a known speaker from a set of candidates) and speaker *verification* (verifying the claimed identity of a given speaker). Another distinction is also made based on the constraints on the speech used in the application. In text-dependent speaker recognition the speech is constrained to sentences or words known in advance; in text-independent speaker recognition the speech is unconstrained [3].

As two examples of applications, speaker verification has been used as voice authentication for transactions over telephone, while speaker identification has been used for automatic speaker labelling of audio segments [3]. Speaker identification is also used to recognise the voices of the different users in consumer products such as Amazon Alexa and in military and police intelligence to monitor internet or telephone conversations [4]. A recent example of the latter is the EU-funded *Speaker Identification Integrated Project* that completed its final field test in November 2017 [5].

In this thesis, speaker identification will be used to identify the current speaker in an ongoing conversion. With the recent development of speech recognition, it is plausible to perform a real-time transcription of what is being said with increasing reliability. However, many of these algorithms focus exclusively on the speech recognition, while omitting the occasionally important information of who is saying what. This information could instead be provided by a dedicated speaker identification system that complements existing speech recognition systems. Possible applications include real time closed captioning of live streaming or transcription of meetings. While the concept of real time could be considered vague as all physical implementations will inherently introduce some delay, real time will in this thesis be specified to allow a delay of up to 0.1 seconds. For humans any delay less than 0.1 or 0.2, depending on the person, is perceived as simultaneous by humans [6].

1.1 Project Objective

The objective of this thesis project is to design and implement a text-independent speaker identification system. The system will let M different speakers speak for T seconds each in a setup phase, and should then be able to identify the current speaker, $m \in M$, as a conversation separate from the setup recordings is progressing. The problem can be considered a case of supervised learning, where the setup and identification phases corresponds to training and classification. The system is not to be expected to have any prior knowledge about the speakers and should be able to display the current speaker without a noticeable delay running on a normal laptop (specified to be 100 ms between the end of a speech frame and its identification). The number of speakers is expected to be small ($M \leq 12$) and the training time is constrained ($T \leq 12$ s), so that the system will be easy to configure. The system should be able work in a multi-language environment.

1.2 Previous Work

Besides commercially available systems like the previously mentioned Amazon Alexa, several scientific studies have been made on the design of speaker recognition systems. In a highly cited paper published in 1995, Reynolds & Rose introduced a method based on mel-frequency cepstral coefficients (MFCCs) and Gaussian mixture models (GMMs) as a computationally inexpensive and robust speaker identification system [7]. Since then, there has been several follow-up studies that explored different variants and applications of this system. While focusing on speaker verification, Cambell & Reynolds include various incremental improvements, e.g. using the first order derivative delta-cepstra values, that can be generalised to apply to speaker identification as well [3]. They also conclude that alternative cepstral features such as using linear prediction coding spectral analysis instead of Fourier transform based spectral analysis yields similar performance.

In recent years joint factor analysis has been introduced as a way of reducing channel variability between sessions [8]. Follow-up studies by the same authors have found that these features also contain speaker information. This observation was used to develop new features based on joint factor analysis called identity vectors (i-vectors) for speaker verification in an article published 2010 [9]. I-vectors have also been used in speaker identification with impressive results for limited training data compared to the MFCC-GMM method [10]. Instead of classifying the features directly, the i-vectors are based on the deviation from a pre-trained background model.

In addition to MFCCs and i-vectors several deep learning methods have also been studied. In a highly cited article published in 2009 [11], Lee et al demonstrate that features from pre-trained convolutional deep belief network (CDBN) together with support vector machines (SVMs) perform well in various speech and speaker recognition tasks. The CDBN was trained unsupervised on a large data set and the extracted features corresponded to phones/phonemes. The CDBN-SVM method outperformed an implementation of MFCC-GMM for speaker identification, espe-

cially with few training utterances. In another deep learning article a convolutional neural network (CNN) was applied on the spectrogram directly to identify and cluster speakers [12].

Neural networks have also been used as classifiers with MFCCs as features. In [13] an extreme learning machine (a type of artificial neural network where the hidden nodes are not updated) in combination with MFCCs is demonstrated to outperform MFCC-SVM for speaker verification with limited training data. In [14] MFCCs with or without delta-cepstral values are used with probabilistic neural network. While no direct comparison was performed, the initial experiment showed promising performance.

1.3 Scope

The particular application of speaker identification for the thesis provide some challenges that will affect the selection of base algorithm. Perhaps most notably the limited training data excludes direct deep learning approaches such as the CNN approach described in the previous work section. To use a pre-trained deep learning feature extractor, such as the CDBN, would however still be a valid option in this regard.

The data demanding process of training the feature extractor can be performed in advance, when data is available, and the resulting network saved for later application. It is however not clear how the pre-trained extractor will perform if the speech used in the session differs considerably from the speech used for training the extractor. If for example the extractor is trained on English, but the session language is Swedish, or if the session environment is more noisy, the accuracy is showed to decline.

The background model related to i-vectors suffers from the same problem. The performance is significantly degraded when the features in the session are less similar to the data in the pre-training. While methods of compensating or solving this have been researched [15], the limited time and resources of the project have to be considered. Instead, the more basic MFCCs will be used as features in the project with different classifiers. Specifically, a multi-layer perceptron will be compared with the standard Gaussian mixture model. Although modern (post 2010) speaker identification systems primarily use either i-vectors or some sort of deep learning, the performance should be reasonable for this introductory project. To limit the scope of the thesis, the session is assumed to be relatively noise free.

1.4 Research Aim

The research aim of this thesis project is to compare Gaussian mixture models and multi-layer perceptrons as classifiers for limited data text-independent speaker identification with mel-frequency cepstral coefficients as features, and determine the most suitable classifier for this specific application. This real-time application will

also require adaptation of existing algorithms which are not designed for real-time usage.

2

Theory

2.1 Feature Extraction

Amongst the most popular methods of feature engineering for speaker recognition (before 2010s) is to transforming the speech signal into feature vectors consisting of mel-frequency cepstrum coe□cients (MFCCs). The feature vector is thought of as a more compact, less redundant and more statistical modelling friendly way of representing speech. The coe□cients capture the speakers vocal tract structure, which is a substantial part in how the voices di□er. The following chapter explains how to extract these coe□cients [16].

2.1.1 Short-time Fourier transform

The first step in extracting mel-frequency cepstral coe□cients is to divide the pulse code modulated speech signal into smaller frames and perform a Fourier transform on each frame to determine the containing frequencies. This process is known as short time Fourier transform and allows the frames to be analysed separately. In speech processing the length of the frame is usually between 10 to 30 ms when the vocal tract parameters are assumed to be constant.

In order to reduce misclassification, a voice activity detector is often used to determine if the frame contains voice activity or if it's noise or silence that can be ignored. This activity detection can be performed on the frame directly, at a later stage of the extraction, or at the feature itself depending on the type of detector.

A pre-emphasis filter (a type of finite impulse response filter) that enhance high frequencies is optionally applied on the frame. The filter has the following form where $x(n)$ is a single sample point from the signal

$$x'(n) = x(n) - ax(n-1), \quad a \in [0.95, 0.97]. \tag{2.1}$$

Before the discrete Fourier transform is applied, the signal is tapered by using a Hamming window to reduce the spectral leakage. The Hamming has the following weight function

$$w(n) = \alpha - (1-\alpha)\cos\left(\frac{2\pi n}{N}\right). \tag{2.2}$$

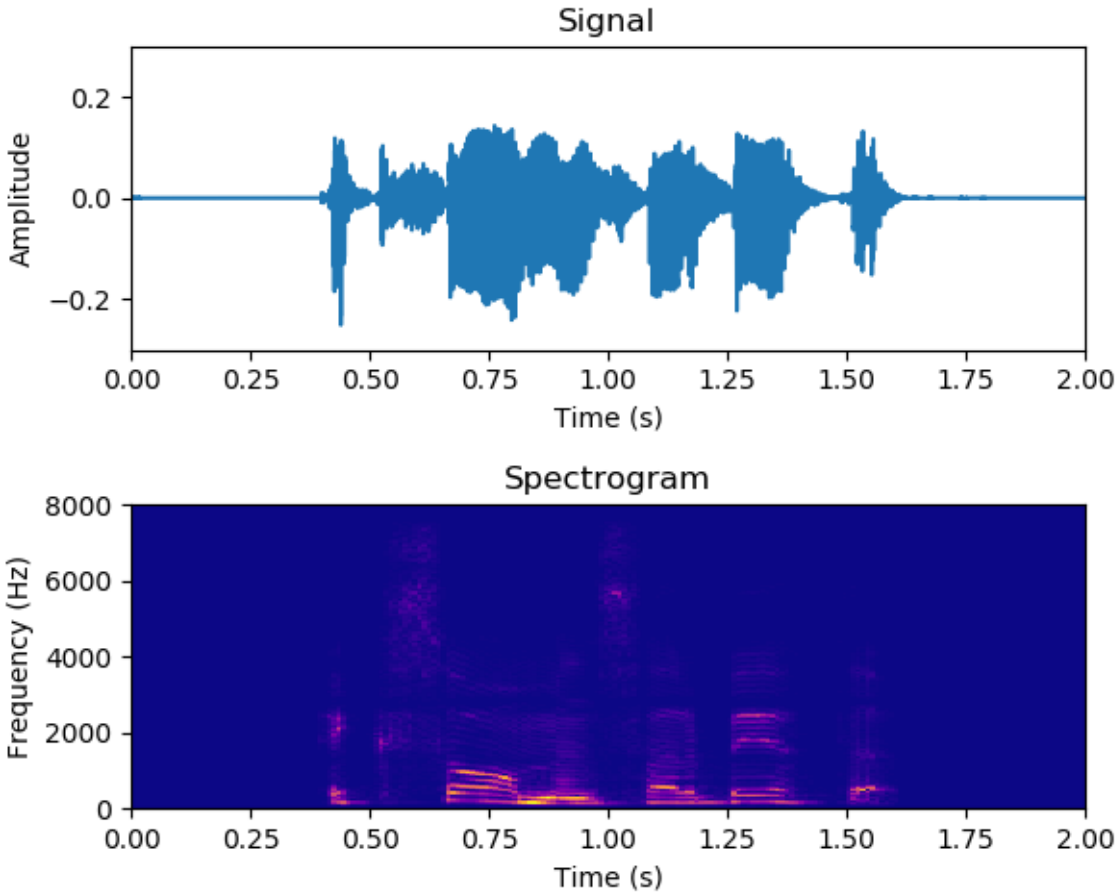


Figure 2.1: An pulse code modulated signal of the word “examensarbete” together with the corresponding magnitude spectrogram.

Here N is the number of samples in the frame, n is the sample index and $\alpha = 0.54$. The tapering implies a loss of data whereby a 50 % overlap between frames is used to compensate for this e□ect.

A fast Fourier transform is now applied with the sample points in the frame zero padded to nearest power of two resulting in a spectrum of (K) frequency bins X_k, k K. Since the output of the Fourier transform is symmetric, only the first half ($K/2$) is used to form the magnitude spectrum of the frequency bins $|X_k|$. The magnitude spectrum from several consecutive frames can be illustrated in a spectrogram as seen in Figure 2.1. In some articles, the power spectrum $|X_k|^2$ is used instead of the magnitude spectrum, however this will only result in a scaling of the end feature as will be apparent in the next section.

2.1.2 Mel-frequency spectral coefficients

In the second step of the feature extraction, mel frequency spectral coe□cients (MFSC) is generated from the magnitude of the frequency bins from the previous step.

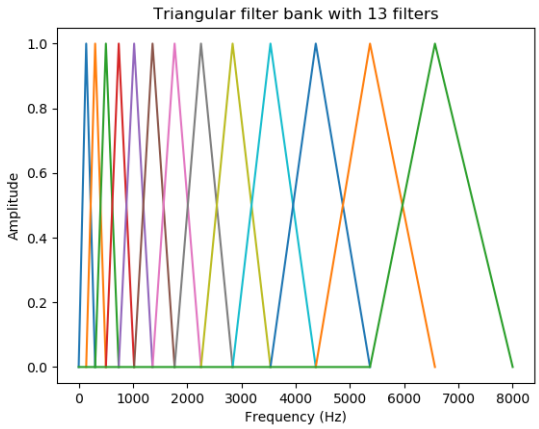


Figure 2.2: The mel-scale filter bank using 13 filters. The number of coefficients extracted (not all necessarily used) equals the number of filters.

Instead of using spectrum directly, a spectral envelope in the form of a mel-frequency filter bank is used to set the focus on the most significant information and reduce the size of the feature vector. The mel-frequency filter bank a series of triangular overlapping band pass filters where the placement of the filters is based on the mel-frequency scale. This frequency scale is designed to mimic the human hearing perception of equal distance between different pitches [17]. The relationship between a frequency f in Hertz and the corresponding mel-frequency f_m is given by

$$f_m = 2595 \log_{10}\left(1 + \frac{f}{700}\right). \tag{2.3}$$

If the number of filters in the filter bank is denoted by L, the $L + 2$ points needed to form the triangles are taken from the linearly spaced interval from f_m^{min} to f_m^{high} projected back to standard frequency scale $f_0, \ldots, f_{L+2}$. f_m^{min} and f_m^{high} are set to the limits of the voice frequency band, or to 0 and the Nyquist frequency ($f_{sample}/2$). The filter bank is illustrated in Figure 2.2 for $L = 13$. In other words, each individual filter $\{h_l[n]\}_{l=1}^{L}$ is given by

$$h_l[f] = \begin{cases} \frac{f - f_{l-1}}{f_l - f_{l-1}} & f_{l-1} \leq f \leq f_l \\ \frac{f_{l+1} - f}{f_{l+1} - f_l} & f_l < f \leq f_{l+1} \\ 0 & \text{otherwise} \end{cases} \tag{2.4}$$

The matrix of all discrete mel frequencies thus takes the form $H_L \quad R^{L \times (K/2)}$. The mel frequency spectral coefficients are now given by

$$S = H_L |X|, \tag{2.5}$$

where $|X| = \{|X_0|, \ldots, |X_{K/2}|\}$ is the magnitude spectrum as before.

In the source–filter model of speech production, the speech spectrum equals the product of the vocal source and the vocal tract. This can be expressed in magnitude

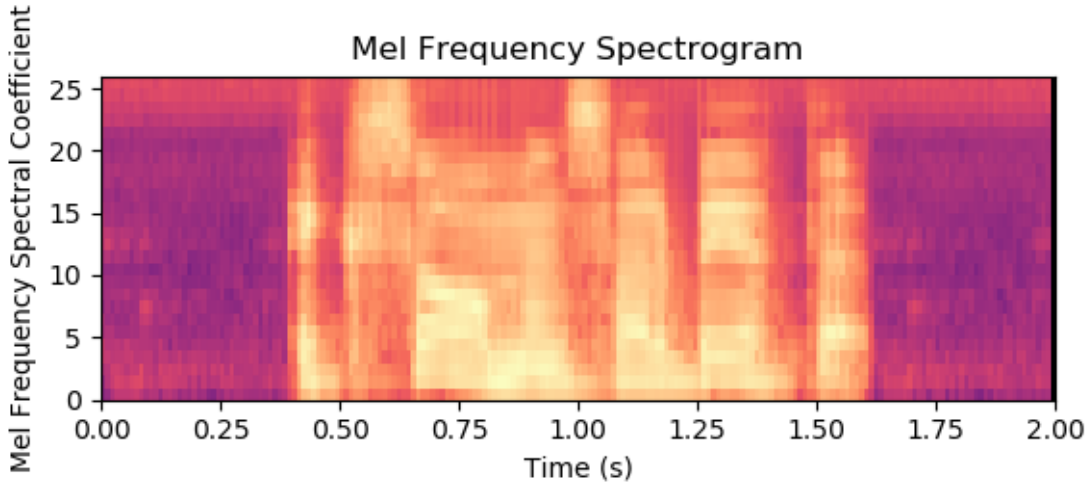


Figure 2.3: The mel-spectrogram in decibel of the same signal used in Figure 2.1.

form as $|F(\omega)| = |S(\omega)| \cdot |V(\omega)|$, where $|S(\omega)|$ is the vocal source spectrum and $|V(\omega)|$ is the magnitude of the Fourier transform of the impulse response of the vocal tract. To separate the two, they are decorrelated by converting all frequency spectral coe□cients into decibel scale $\tilde{S} = 20\log(S)$. An illustration of this mel-frequency spectrum in decibel is shown in Figure 2.3.

2.1.3 Mel-frequency cepstral coefficients

As a final step, the mel-frequency spectral coe□cients are transformed into mel-frequency cepstral coe□cients by applying discrete cosine transform as follows

$$c_i = \sum_{l=0}^{L-1} \tilde{S}_l \cos\left[i(l - \frac{1}{2})\frac{\pi}{L} \right], \quad i = 0, 1, ..., C \tag{2.6}$$

where $C \leq L$ and c_i are the mel-frequency cepstral coe□cients (MFCCs). The coe□-cients that correspond to frequencies irrelevant for speech are sometimes discarded. The cepstral coe□cients are thought to better capture the pitch of the speech in the frame and be more robust to noise compared with the spectral coe□cients [18]. Consecutive cepstral coe□cients form a cepstrum. A cepstrum can be seen in Figure 2.4

2.1.4 Delta coefficients

In order to capture dynamic information about how the extracted coe□cients change in time, linear regression is applied resulting in the following expression for each component

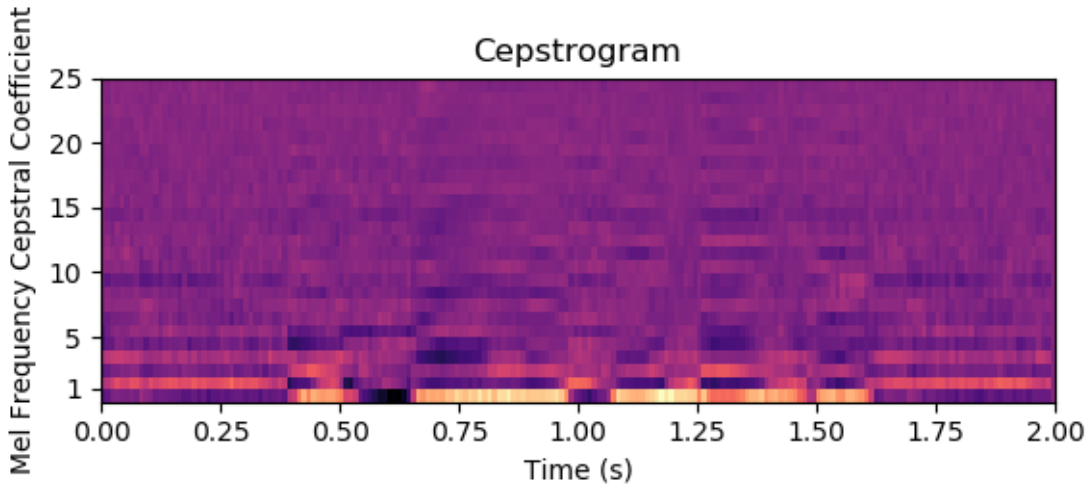


Figure 2.4: The cepstrum of the same signal used in 2.1. The first coe□cient has been removed for visualisation purposes.

$$\Delta c_i = \frac{\sum_{k=-t}^{t} k c_{i+k}}{\sum_{k=-t}^{t} |k|}. \tag{2.7}$$

Here Δc_i is the delta coe□cient for component i and t is the number of time adjacent coe□cients you want to include. The new vector of the same length as previous (M) is then appended to the feature vector. Since the new feature vector is not defined for the first or last t frames, these feature vectors are skipped [19].

2.2 Classifiers

The goal of classification is to learn a mapping of inputs $\mathbf{x}$ to outputs $y \in \{1, \ldots, M\}$, where M is the number of di□erent classes. Alternatively, it can also be thought of as a function approximation. If $y = f(\mathbf{x})$ for some unknown function f, the goal is to estimate a function $\hat{f}$ that predicts the label of an input $\hat{y} = \hat{f}(\mathbf{x})$. In supervised learning the classifier is given a set of training examples with their corresponding labels in order to learn its mapping (estimate its function). Ideally the retrieved rule generalise to also apply to new inputs separate from the training set [20].

2.2.1 Gaussian mixture models

A Gaussian mixture model (GMM) is a statistical model that mix K weighted Gaussian probability distributions to form a new mixture model. The mixture model probability density function p thus has the following form

$$p(\mathbf{x}_i|\theta) = \sum_{k=1}^{K} w_k N(\mathbf{x}_i|\mu_k, \Sigma_k), \tag{2.8}$$

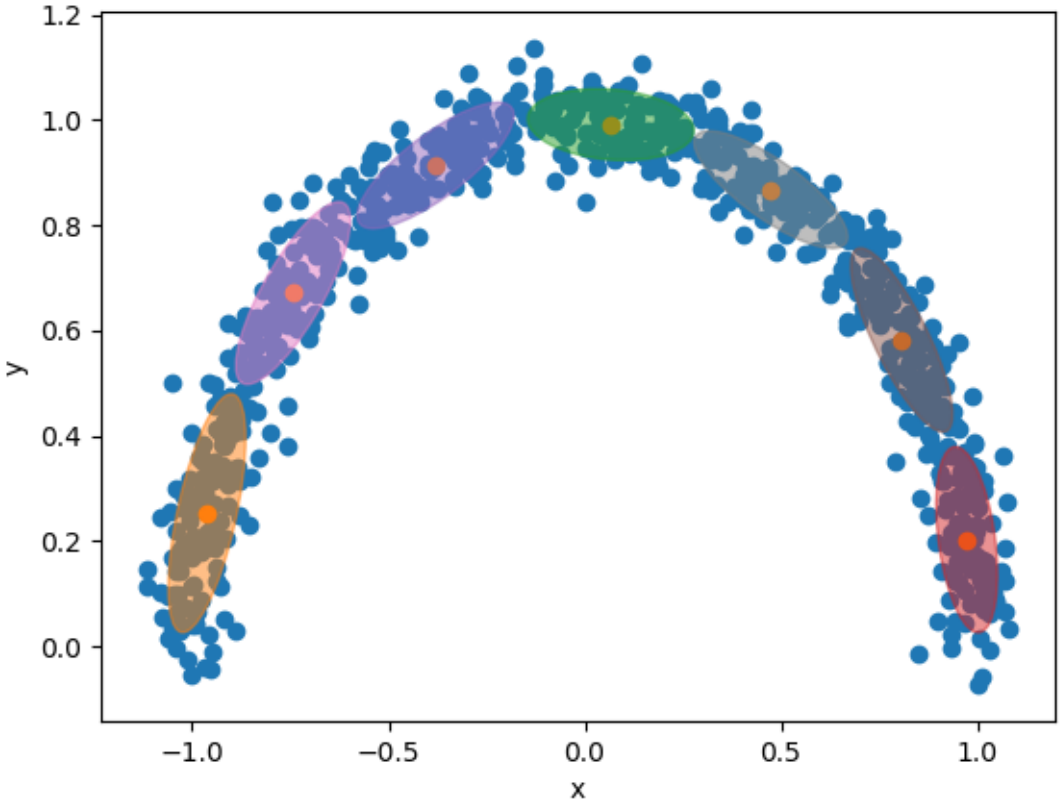


Figure 2.5: Seven mixture components trained to fit larger distribution represented by a 2D dataset in the shape if an arc.

with the sum of the weights $\sum_k w_k = 1$. Given enough number of mixture components, a GMM can approximate any density defined on R^D [20]. An illustration of how a mixture of Gaussian distribution is used to fit a dataset is illustrated in Figure 2.5

2.2.1.1 Training

For speaker identification, a separate model is trained for each speaker m with its corresponding parameters $\lambda_m = \{\mathbf{w}_m, \mu_m, \Sigma_m\}$. The individual component density of the GMM is thought to represent some underlying set of acoustic classes that characterise a speaker's voice that the model finds during the training. Put differently, the goal of the training is to iteratively update the parameters of the mixture model to maximise the likelihood function of the given training data observation. This is usually done in two steps. First the model is initialised on random or by using algorithms such as the k-means algorithm. Next, the expectation maximisation algorithm is applied. These will be explained the following subsections.

While the co-variance matrix Σ determines the shape of the Gaussian, it is important to note that any shape of a full co-variance matrix can be achieved using multiple

diagonal co-variance matrix components. It is both computationally faster and empirically more favourable to avoid full co-variance matrices [7].

2.2.1.1.1 K-means algorithm The goal of using the k-means algorithm is to initialise the K different mixture components before the expectation-maximisation algorithm (EM-algorithm) is applied. The initialisation is important as an attempt to avoid converging to local optima. In the k-means algorithm the mixture components are assumed to have fixed variance ($\Sigma_m = \sigma^2 \mathbf{I}_D$) and fixed weights ($w_k = \frac{1}{K}$), and the components are instead referred to as clusters. Only μ_m is left to be assigned. The algorithm is therefore sometimes also referred to as hard EM.

After the clusters have been initiated, either randomly assignment or using other algorithms, the goal is to assign each point to its closest euclidean centre

$$\arg\min_k ||\mathbf{x}_i - \mu_k||^2 \tag{2.9}$$

The cluster center is then updated by computing the mean of all its assigned points.

$$\mu_k = \frac{1}{N_k} \sum \mathbf{x}_i \tag{2.10}$$

The points are then reassigned and the processes repeated until convergence.

2.2.1.1.2 Expectation–maximisation algorithm Given K pre-initialised mixture components as described in the previous section we define the parameters in iteration t as $\{w_k, \mu_k, \Sigma_k\}^{(t)} = \theta^{(t)}$. Let $r_{ik} = p(z_i = k|\mathbf{x}_i, \theta^{(t-1)})$ denote the responsibility that cluster k takes for datapoint i for the latent variable z. The algorithm iterates over the E step and the M step.

Using this notation the E step (the expected value of the log likelihood function of $\mathbf{x}_i$ given θ_k^{t-1}) is simply

$$r_{ik} = \frac{w_k p(\mathbf{x}_i|\theta_k^{t-1})}{\sum_k w_k\, p(\mathbf{x}_i|\theta_k^{(t-1)})} \tag{2.11}$$

In the M step we instead try to find the parameters that maximise the log likelihood function with regard to $\mathbf{w}$ and θ_k. It follows that each parameter have the following update rule

$$w_k = \frac{1}{N} \sum_{i=1}^{N} r_{ik} = \frac{r_k}{N} \tag{2.12}$$

$$\mu_k = \frac{\sum_i r_{ik}\mathbf{x}_i}{r_k} \tag{2.13}$$

$$\Sigma_k = \frac{\sum_i r_{ik}(\mathbf{x}_i - \mu_k)(\mathbf{x}_i - \mu_k)^T}{r_k} = \frac{\sum_i r_{ik}\mathbf{x}_i\mathbf{x}_i^T}{r_k} - \mu_k\mu_k^T \tag{2.14}$$

These are now the new estimates $\theta^{(t)} = \{w_k, \mu \Sigma_k\}$ for $k = 1, ..., K$. The processes is repeated until convergence [20].

2.2.1.2 Classification

Once a model with the corresponding parameters for every speaker has been trained $\lambda_m, m \quad \{1, ..., M\}$, the classified identity of a speaker $\hat{m}$ for a given feature vector $\mathbf{x}_t$ from frame t is given by the maximum posterior probability

$$\hat{m} = \arg \max_{1 \le m \le M} p(\lambda_m | \mathbf{x}_t) = \arg \max_{1 \le m \le M} \frac{p(\mathbf{x}_t | \lambda_m) p(\lambda_m)}{p(\mathbf{x}_t)} \tag{2.15}$$

from Bayes' rule. Assuming that all speakers are equally likely ($p(\lambda_m) = 1/M$), the equation simplifies to

$$\hat{m} = \arg \max_{1 \le m \le M} p(\mathbf{x}_t | \lambda_m). \tag{2.16}$$

The performance can generally be improved by averaging over a segment of consecutive T frames $X = \{\mathbf{x}_1, ..., \mathbf{x}_T\}$. The same assumptions still holds and Equation 2.16 instead becomes

$$\hat{m} = \arg \max_{1 \le m \le M} p(X | \lambda_m). \tag{2.17}$$

Using logarithm and the independence between observations, this simplifies to

$$\hat{m} = \arg \max_{1 \le m \le M} \sum_{t=1}^{T} \log p(\mathbf{x}_t | \lambda_m). \tag{2.18}$$

There is a trade-off in the choice of T. A large value of T is good for averaging out bad frames, but produce errors when the conversation transitions to another speaker [7].

2.2.2 Multi-layer perceptron

A multi-layer perceptron is a type of feed forward artificial neural network with at least three layers. In this simple type of neural network, there are no cycles or loops in the network, instead the information is fed forward from the input layer down to the output layer. In the MLP the neurons are fully connected, meaning that every neuron is connected to all of the neurons in the following layer. For multi-classification the size of the input layer is simply the size of the feature, and the output layer is simply the number of different classes. An illustration of multi-layer perceptron is seen in Figure 2.6.

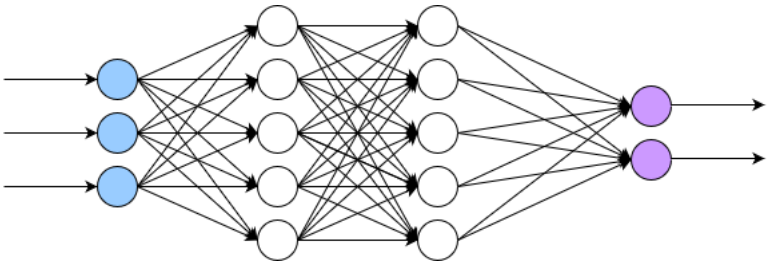


Figure 2.6: A multi-layer perceptron with three neurons in the input layer (blue), two hidden layers of five neurons and two neurons in the output layer (purple).

2.2.2.1 Activation Function

Except for the weights connecting the neurons and possibly a bias, the neurons also need an activation function to achieve non-linearity. While different activation functions have been experimented in the past, the standard choice today is the rectified linear unit.

$$g(\mathbf{x}) = max(0, \mathbf{x}) \tag{2.19}$$

However, in classification the activation function in the output layer is the softmax function, which allows the output to be interpreted as probabilities.

$$p(m = i|\mathbf{x}) = g(\mathbf{x}) = \frac{e^{\mathbf{x}_i}}{\sum_{m=1}^{M} e^{\mathbf{x}_m}} \tag{2.20}$$

2.2.2.2 Training

Once a network has been initiated with starting weights and biases, they need to be updated to minimise the cost function C. Depending on application there are several cost functions to choose between, but for classification, cross entropy is the most popular. The cross entropy for N datapoints and M classes is defined as

$$C(\theta; y_{im}, p_{im}) = -\frac{1}{N} \sum_{i=1}^{N} \sum_{m=1}^{M} y_{im} \log p_{im} \tag{2.21}$$

Here y_{im} is a latent variable indicating if label m is the correct classification for data point i, and p_{im} is the probability data point i being classified as label m.

Using this loss function, the feed forward network has historically been updated simply using gradient descent backpropagation. Modern learning algorithms instead rely on variants of stochastic gradient descent, which only use a small sample (batch) of training data for each step. This has proved especially advantageous for stochastic objective functions. One such algorithm is *Adam* [21]. Given a step size α and exponential decay rates $\beta_1 \& \beta_2$, the gradient is calculated as $g_t = \quad C(\theta_{t-1}; \mathbf{y}, \mathbf{p})$ for each iteration t. The gradient is then used to retrieve the first and second moments m_t and v_t as

$$m_t = \beta_1 m_{t-1} - (1 - \beta_1) g_t \tag{2.22}$$

$$v_t = \beta_2 v_{t-1} - (1 - \beta_2) g_t^2 \tag{2.23}$$

A bias adjustment is then performed as

$$\hat{m}_t = m_t / (1 - \beta_1^t) \tag{2.24}$$

$$\hat{v}_t = v_t / (1 - \beta_2^t) \tag{2.25}$$

The parameter is finally updated according to

$$\theta_t = \theta_{t-1} - \alpha \hat{m}_t / (\hat{v}_t + \ ) \tag{2.26}$$

2.2.2.2.1 Regularisation As artifical neural networks (ANNs) have a tendency to overfit on the training data, L^2 regularisation is sometimes applied to discourage parameters to become too strong. This is achieved by adding a penalty function to the loss function. One common such penalty function is the L^2-regression.

$$\alpha ||vec(\theta_r)||^2 \tag{2.27}$$

2.2.3 Classification

With softmax as the activation function for the output layer, the classification is functionally identical for the multi-layer perceptron and the Gaussian mixture models. A small difference might be that MLP consists of a single model with the number of outputs equal to the number of classes, while the GMM has one model per class.

Although primarily discussing speech recognition, ANNs have been argued to be preferable to GMMs in speech processing. ANNs are specifically thought to be better at modelling data that lie on or near a non-linear manifold in the feature space [22].

3

Method

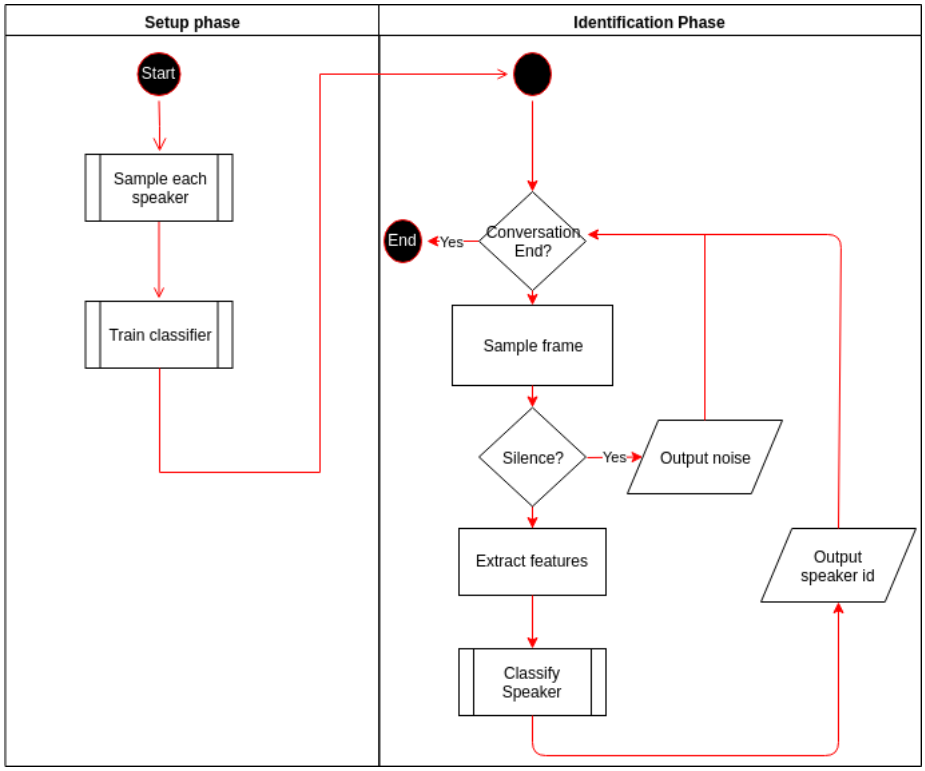


Figure 3.1: A schematic flowchart over the speaker identification system and its two phases.

3.1 System Prototype Implementation

In order to fulfil the project objective and research aim, a prototype of the system needs to be implemented. This system di□ers from most other speaker identification systems in three notable ways: the limited data constraint, the real-time constrain and the way it accounts for an ongoing conversation.

The first two constraints are self-explanatory. As mentioned in Chapter 1, in order to make the system practically useful there is a strong constrain on the recording time per speaker ($T \leq 12$ s) in the training phase. This limitation on training data excludes a naive implementation of certain machine learning methods. The real-time

constraint furthermore, implies that the implementation needs to be computational e□cient in order to display the result in real-time. This scaled down prototype also has to leave a margin for additional delay from the full system.

The third notable di□erence is the process of identifying the current speaker in an *ongoing conversation* between multiple speakers. This di□ers from most other systems described in the scientific literature where a speaker is identified in a segment containing a single speaker. This distinction is motivated by the additional challenges that the ongoing conversation brings.

In a segment-based case, the system can analyse the entire segment under the assumption that there is only one speaker to identify. If some frames are uncertain, the combination of frames in the segment will hopefully provide a more clear result. Either if the system is given presegmented snippets containing only a single speaker or if the system can continue to sample until the system has reached a certain level of confidence. Variants of this type of test are common in the literature as it is easy to implement and simple enough to avoid test complexity. Real applications that takes this form could be if the speakers all used di□erent distinguishable source microphones.

In contrast, in an ongoing frame-based case, no such assumption about neighbouring frames belonging to the same speaker can be safely assumed. Instead it is up to the system to detect when the current speaker is switched. Since the conversation is also ongoing and not pre-recorded, the system can only access frames up to the current frame, but not beyond. This means the system has to present the result, no matter the confidence level, in order to meet a real-time criterion. This is closer to the intended application of this thesis.

To simplify the development process of the system, a prototype was implemented in Python with the actual live audio recording replaced by a speech corpus module designed to have a similar behaviour. The prototype was designed so that this speech corpus module could be easily replaced with a recording module once it was implemented. A schematic overview of the prototype can be seen in Figure 3.1.

3.1.1 Feature extraction

As stated in the introduction, mel-frequency cepstral coe□cients were used as feature representation of the audio for the system to classify. The algorithm for extracting these coe□cients from a speech frame was implemented in Python from scratch based on the theory described in the Section 2.1.

Pre-emphasis was disabled as it appeared to deteriorate the performance. The magnitude spectrum was used instead of the power spectrum in the Short-Time Fourier Transform step as it appears to increase the performance. The use of delta coe□cients are questionable for this real-time system as they would require information about the upcoming frame, or alternatively shift identification backwards with at least 10 ms for each value of t. Because of this only $t = 1$ was considered.

The parameters in the extraction of mel-frequency ceptral coe□cients can be tuned extensively with varying results. To achieve the best result, they should be included

Table 3.1: The best practice values used for feature extraction parameters used in the hyperparameter optimisation.

Parameter	Value
Frame size	20 ms
Frame overlap	50 %
Number of filters	26
Number of coe□cients	20
Delta k	1
Pre-emphasis	Disabled

in the hyperparameter search. In order to save simulation time however, the best practice parameters seen in Table 3.1 are used instead. The hyperparameter search is generally be considered a di□cult task in machine learning and some sort of limitation is inevitable.

After first attempting to implement other voice activity detectors based on analysis of the frame as well as the extracted features, the Python package `py-webrtcvad` for the voice activity detector from the WebRTC project[1] was used instead. This voice activity detector outperformed all previous attempts, both in terms of false positive and false negatives. Voice activity detection could in general constitute a separate thesis. In hindsight it can be stated that it would have been more time-e□cient to use the existing implementation of the feature extraction from the open source package `python-speech-features`[2], with some modifications for this real-time application.

3.1.2 Classifiers

As stated in the introduction, Gaussian mixture models and multi-layer perceptrons were compared as classifier for the system. The classifiers were implemented so that the system could easily switch between them. Both classifiers used the implementations from Scikit-learn[3] as base classifiers. These implementations follow the theory as described in Section 2.2. The hyperparameter optimisation is described in Chapter 4.

For the Gaussian mixture model, one instance of the `GaussianMixture` class was user per speaker with a diagonal covariance matrix, a varying number of components and the EM-algorithm with tolerance set to default 0.001. The log likelihood of each frame was then compared to identify the speaker. For the multi-layer perceptron, a single instance of the `MLPClassifier` class was used. Each speaker was treated as a separate label and softmax was used in the output layer to emulate probability. The hidden layer was varied, but the rest was set to default. The training continued until the loss value (cross entropy) did not improve more than a threshold of 0.0001 over two iterations.

[1] https://webrtc.org/
[2] https://python-speech-features.readthedocs.io/
[3] http://scikit-learn.org

While the speech is not segmented in advance, a moving segment can be used to increase the performance since it is safe to assume that the current speaker is not switched every frame. There is obviously a trade-off. An increased size of this moving segment will improve accuracy for longer passages, but conversely deteriorate performance when the speaker actually changes and this assumption is false. To compensate this, a linear weighting was applied to give more recent frames higher importance.

$$\hat{y} = \arg\max_{i} \sum_{n=1}^{N} w_n p_i, \tag{3.1}$$

where p_i denote the likelihood or the emulated probability of label i from the GMM and MLP (see Section 2.2), N as segment length and w_n as linearly scaled scalar weight.

3.2 System Evaluation

In order to compare the classifiers for the research aim and tweak parameters of the system, an evaluation process was designed and implemented. The evaluation processed system was designed to be able to handle both an ongoing conversation and the more standard segment-based test for benchmarking with other articles. It was also designed to handle multiple separate speech corpora. Two different corpora were used in the test.

3.2.1 Speech corpora

Instead of recording training and testing material, pre-existing speech corpora was be used for this prototype. Unfortunately there is no single standard speech corpus favoured in the literature, which would have made comparisons between algorithms easier. Different corpora feature different languages, speaker compositions and levels of background noise etc. More distinct speakers, less background noise and richer utterances can be assumed to make the classification easier and improve the accuracy. The closest corpus of being considered standard within speaker identification might be the *TIMIT Acoustic-Phonetic Continuous Speech Corpus* by the United States Defense Advanced Research Projects Agency [23], but as this is a non-free corpus, this thesis will instead use *LibriSpeech ASR corpus* and *English Language Speech Database for Speaker Recognition*. Both corpora have divided the speech from each speaker into separate labelled files that can be combined for longer sessions.

3.2.1.1 LibriSpeech ASR corpus

The primary speech corpus used in this thesis project is the *LibriSpeech ASR corpus*, which is a free-to-use corpus derived from audiobook readings from the LibriVox project [24]. The full corpus contain approximately 1000 hours of speech, but for

this project only the `train-clean-100` subset was used. This subset contains on average 25 minutes of audio book readings (different source texts) from 251 different speakers (126 males and 125 females) speaking English with an American accent recorded with high quality. For technical reasons the files where converted from original Free Lossless Audio Codec (flac) format to Waveform Audio File Format (wav) in advance. The speakers were recorded with a sample rate of 16 kHz.

The benefit of using this corpus is the ample amount of data it provides, but for the intended application, the different recording settings (microphone etc) could be a potential source of error. It could be the case that the system will include characteristic of the recording setting in the identification of a speaker instead of just the speech itself. Another potential issue is that reading aloud from a book is not necessary the same sort of speech as used in conversations.

3.2.1.2 English Language Speech Database for Speaker Recognition

As a complement to *LibriSpeech*, the smaller free-to-use *English Language Speech Database for Speaker Recognition* (ELSDSR) will also be used. This corpus consist of 22 (12 males and 10 females) different non-native English speakers (mostly Danish) recorded with the same recording setting.

The corpus is split into a training set and a test set. In the training set, the speakers all read the same source text (on average 83 s). In the test set, the source text is different (on average 17.6 s) [25].

The limitation of ELSDSR is the comparatively limited number of speakers and data. The benefit for this application however is the use of the same microphone for all speakers and the optional ability to compare the performance of having a training set where the speakers use the same source text.

3.2.2 Training

What differs the training of this system compared to many other machine learning applications is the maximum allowed training data. For each speaker, the system was given T seconds of consecutive speech to train on. The speech was taken from the beginning of file, until T seconds had been reached. The file was concatenated with another file if it wasn't long enough. The selection to the file was either selected so that the used the same source text if possible, or complete at random if this was to be avoided

The training files were then put aside and separated from the test set. In the end application, the system has to be retrained for each session and while the training should obviously not generalise for other speakers, it should ideally generalise for the same acoustic classes used in a different context.

3.2.3 Testing

As described in the Section 3.1, the most common type of speaker identification system and the corresponding test is a system that expects pre-segmented series of speech containing only one speaker per segment. The main test for this system was instead an ongoing conversation that contains multiple speakers, returning one frame at the time. The pre-segmented test was however implemented to be able to compare the performance with other articles.

3.2.3.1 Ongoing frame-based test

The ongoing frame-based test was designed to resemble an ongoing conversation in order to be more representative of the intended application. However, a "conversation" is a vague concept that can take many shapes and forms depending on situation, language and culture. Parameters to consider include the length of the replies and the length of the gap between them, the level of overlapping speech and the level of background noise etc. It is obvious that a quick exchange of information between two close friends on a music festival is a very di□erent conversation from a formal meeting with 12 participants in a quiet meeting room. As for language/cultural di□erences, the average length of the gap between replies was determined to be 470 ms in Danish, while only 7.3 ms in Japanese [26].

It is not certain that there is a definite optimal solution for all types of conversations (the bias-variance trade-o□) and it could therefore be argued that separate performance evaluations should be implemented for the various types of conversations. To simplify the process however, a general test was be performed in this case.

For the ongoing frame-based test, it is important to note that not all frames will contain voice activity. The correct classification of these frames will be undefined, so these frames will instead be referred to as silence or noise. The voice activity detection from the WebRTC project will be used here as well. Depending on how strong the threshold for voice activity is, the accuracy when using this measurement can be adjusted to only include "good" frames which are easier to classify. Through out this test the voice activity detection in WebRTC will be set to level 1.

The actual test was implemented as follows: Given a list of snippet lengths, the test system takes uninterrupted snippets of at least these lengths from the corpus. The actual snippet could however be longer, since the end cut of the snippet is not made until the first inactive frame after the specified length is discovered. This adds some additional randomness to the test as the resulting snippet could be substantially longer than the specified minimum. The snippets are then shu□ed and concatenated into a complete test session with a uniformly distributed silent gap separating them (on average 470 ms). The speaker identification system was fed one frame at the time. The accuracy score ($A_{ongoing}$) is then calculated as the number of correctly classified frames divided by the number of active frames (Equation 3.2). An illustration of a test session and its classification can be seen in Figure 3.2.

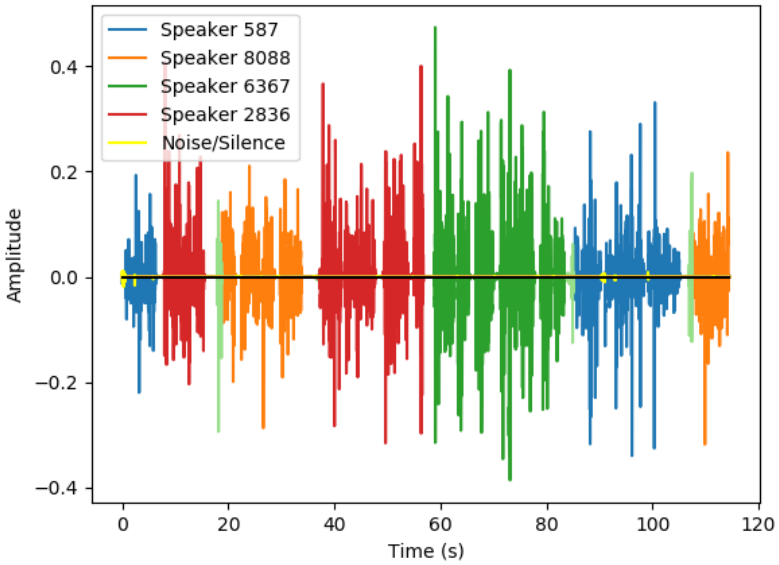


Figure 3.2: An illustration of a completed ongoing frame based test consisting of four different speakers from the LibriSpeech corpus. The colours indicate the classification of each frame, with the lighter shades indicating a misclassification. The total accuracy was 95 %. The session consisted of two utterances for each speaker, one shorter (>3 s) and one longer (>8 s). Notice that the utterances from speaker 6367 are placed next to each other.

$$A_{ongoing} = \frac{\text{Number of correctly classified frames}}{\text{Total number of frames} - \text{Number of noise frames}} \tag{3.2}$$

This test is more similar to the end application than the segment-based test, yet some obvious simplifications can be noted. The speakers will politely "speak in turn" with no overlapping speech in the frames, as it would be undefined who the correct speaker is in these mixed frames. The rhythm of the conversation will be unnatural as the segments will be taken out of context and no additional noise will be added to the test session (although noise from the source material could be included).

3.2.3.2 Segmented-based test

The segmented-based test is quicker to implement and less flexible, making it more suitable for benchmarking in academic contexts. In some version of this test, the segments are set to the different files in the corpus. In this testing system, however the segments are generated by taking a specified number of snippets of a fixed length generated from one or several files from the speech corpus. The classification is then performed by taking the average log probability of each frame in the entire segment. Inactive frames are excluded from the averaging in hope of increasing the accuracy. The accuracy was then calculated the number of correctly classified frames divided by the total number of frames (Equation 3.3).

$$A_{segment} = \frac{\text{Number of correctly classified segments}}{\text{Total number of segments}} \tag{3.3}$$

4

Results

A total of six different experiments were conducted and are presented in this chapter along with their results. To make the comparison between the classifiers fair, a simplified hyperparameter optimisation was performed for each data point in the experiment, unless otherwise specified. The reported accuracy was then given by the system with the best performing hyperparameters averaged over 50 repetitions. A schematic flowchart of this process can be seen in Figure 4.1.

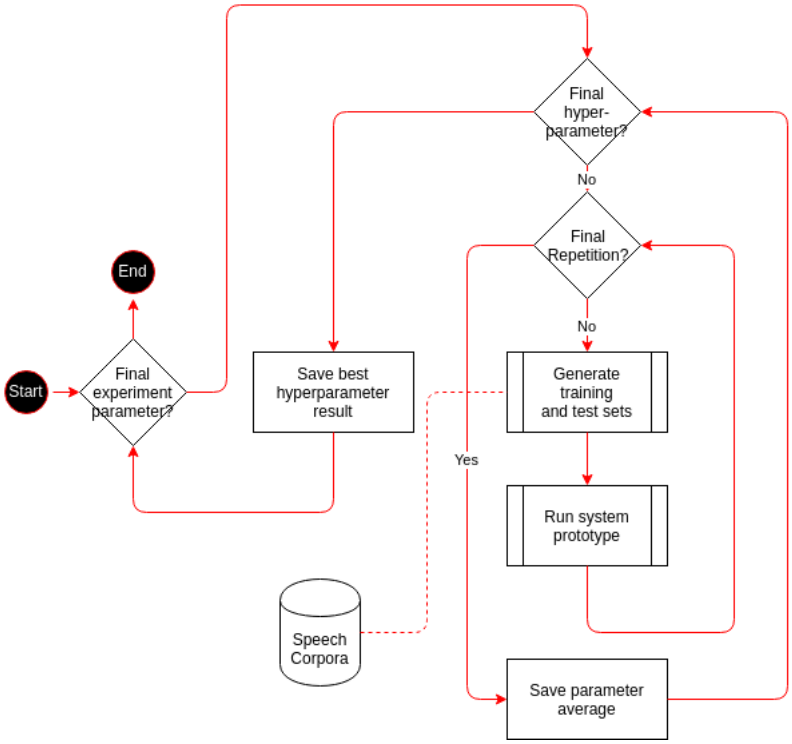


Figure 4.1: A schematic flowchart over the evaluation process of the speaker identification system.

Since MFCCs are a well researched topic by now, the best practice parameters were used as fixed hyperparameters to limit the search space to the hyperparameters of the classifiers. Also, only the number of mixture components were considered for the GMM and only the size and shape of the hidden layers were considered for the MLP. The rest of the parameters were left as their default values in Scikit-learn. This is obviously not ideal, but necessary to meet the project

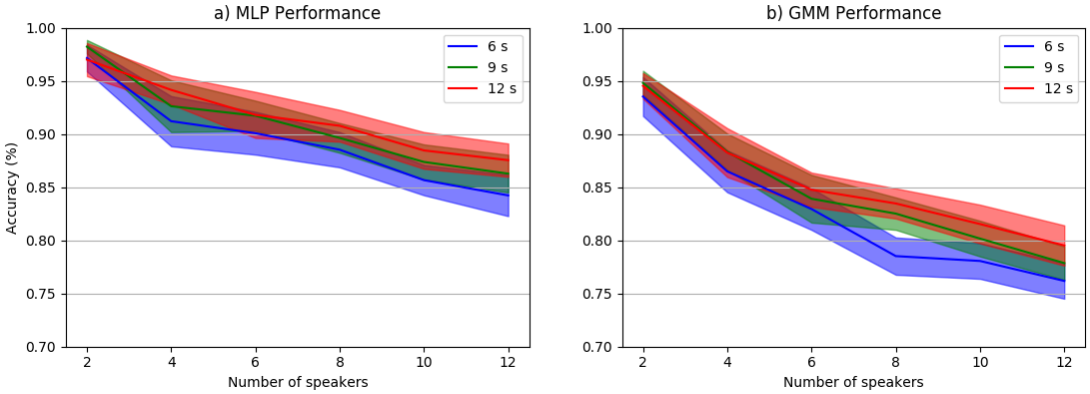


Figure 4.2: The average identification accuracy over 50 repetitions for a given number of speakers and total length of sampled speech for training. The 95 % confidence interval is illustrated by the shades

time scope. In this simplified grid search, the number of mixture components for the GMM were [8, 16, 32, 64, 128] and the hidden layer shapes for the MLP were [(30,), (50,), (100,), (30, 30), (30, 50), (30, 100), (30, 30, 30), (50, 50, 50), (100, 100, 100)].

Unless otherwise specified the lengths of the speech snippets in the frame-based test were [3, 8] seconds. An equal number of males and females were randomly selected for each repetition.

4.1 Performance and the Amount of Training Data

In the main performance experiment of the thesis, the accuracy of the system prototype was tested for a given M number of speakers, and T seconds of sampled speech as training data for each speaker. Using the same setup as described in the chapter introduction, each combination of $M = [2, 4, 6, 8, 10, 12]$ and $T = [6, 9, 12]$ used a separate hyperparameter optimisation. The result is presented in Figure 4.2. Each data point in the result plot represents the average of 50 repetitions for the discovered best setup of hyperparameters. As seen in the figure, the performance declines with an increasing value of M and a decreasing value of T. The MLP consistently outperforms the GMM. For up to around 6 speakers the performance seems to be similar for the di□erent lengths of training speech, but longer speech seem to have a more defined accuracy advantage with 8 speakers or more.

4.2 Speech Corpora Comparison

To get an indication of how the speech corpus itself a□ect the result, the performance of using ELSDSR and LibriSpeech was compared. The comparison used a fixed number of speakers $M = 6$ and $T = 12$ seconds of sampled speech for training. In order to fit the limited test files in ELSDSR, the minimum test utterance

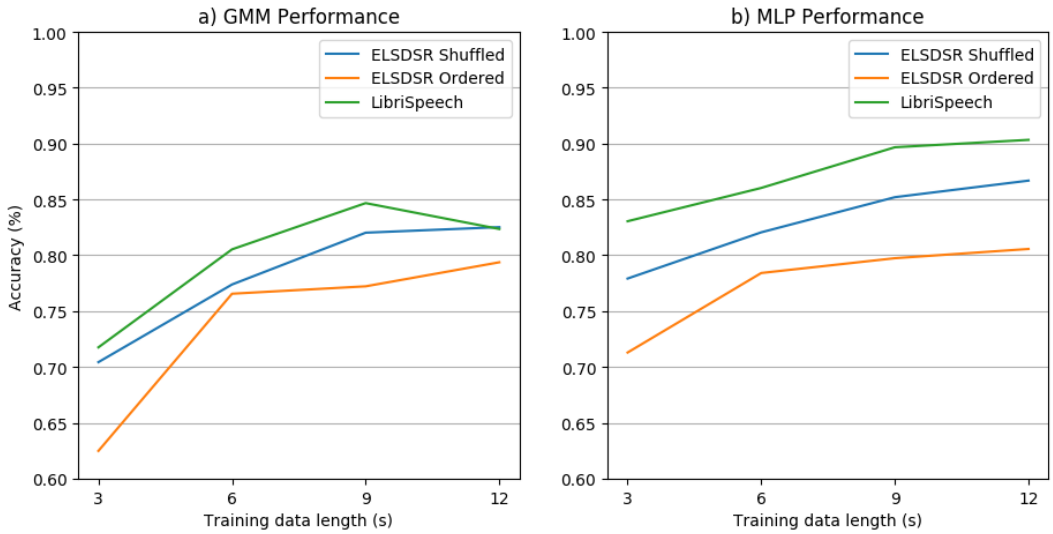


Figure 4.3: The performance of the identification system when using different speech corpora.

lengths was set to [2, 4] instead of [3, 8]. With the exception of the number of speakers, the utterance lengths and the sampled training data, the same hyperparameter optimisation as described in the chapter introduction was used. Two separate configurations of ELSDSR were used. In the ordered configuration the speech used as training data was selected so that all speakers were reading the same source text. In the shuffled configuration the training data were randomly selected.

The performance when using different speech corpora can be seen in Figure 4.3. Using LibriSpeech generates a higher accuracy than ELSDSR. Using the same source text within ELSDSR decreases the performance. MLP outperforms GMM, but the general relationship between the performance of using the different corpora is the same.

4.3 Performance and the Speaker Set Composition

To see how different speaker compositions affect the performance, the gender ratio in a set of $M = 8$ speakers were varied by changing the number of males. Each speaker was sampled for $T = 12$ seconds for training data. With exception of this gender ratio, the number of speakers and the sampled training data were constant. The same hyperparameter optimisation was performed as described in the chapter introduction. The result can be seen in Figure 4.4. As seen in the figure, the accuracy peaks for a balanced mixture of the two genders. However the result is also tilted so that a majority male composition results in a better performance than majority female.

As an additional result a confusion matrix of a single test is displayed in Figure 4.5.

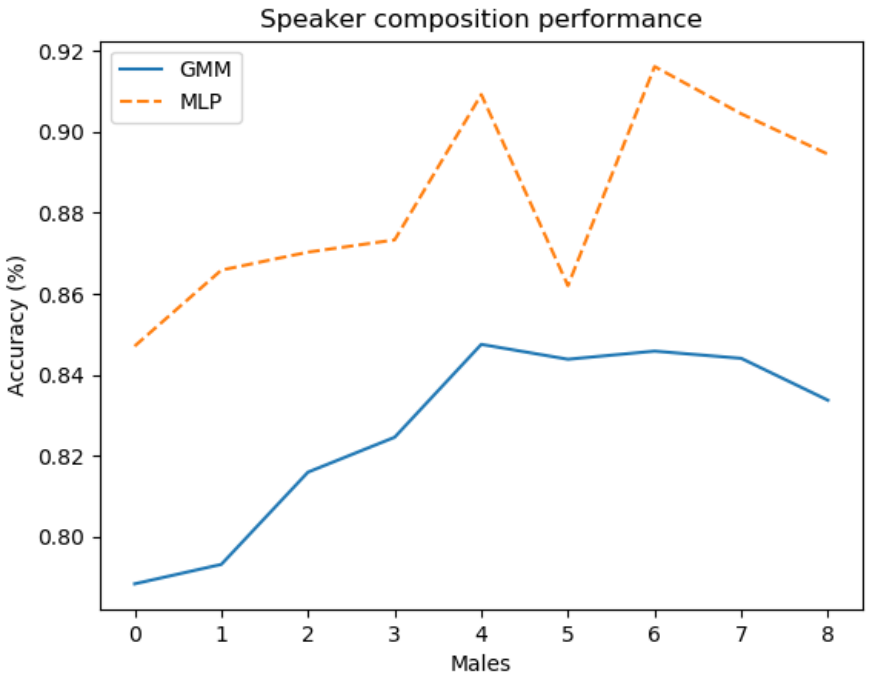


Figure 4.4: The average identification accuracy of different speaker set compositions.

As seen in this figure misclassifications were mostly made within the gender groups.

4.4 System Setup Comparison

To get an indication of how different aspects of the identification system contribute for the total accuracy, different setups of the system were compared. The comparison also serves as a motive for assuming certain parameters to be fixed in the hyperparameter optimisation used in the other experiments. The experiment start with a base configuration and then one additional aspect is activated per different level. These configuration setups were:

- **Setup 0** - The base case, the classification is based on the single current frame. The delta coefficients are turned off.
- **Setup 1** - As setup 0, but with the classification now extended to be made on a 0.7 seconds long moving segment of the frames leading up to the current frame. 0.7 was selected as the optimal segmentation time based on a simplified grid search for this configuration.
- **Setup 2** - In this setup, the weighting of classification segment is added. This lets the more recent frames have a higher importance. The classification length is now set to 0.9 after a random search.
- **Setup 3** - Finally the delta coefficients are turned on which corresponds to

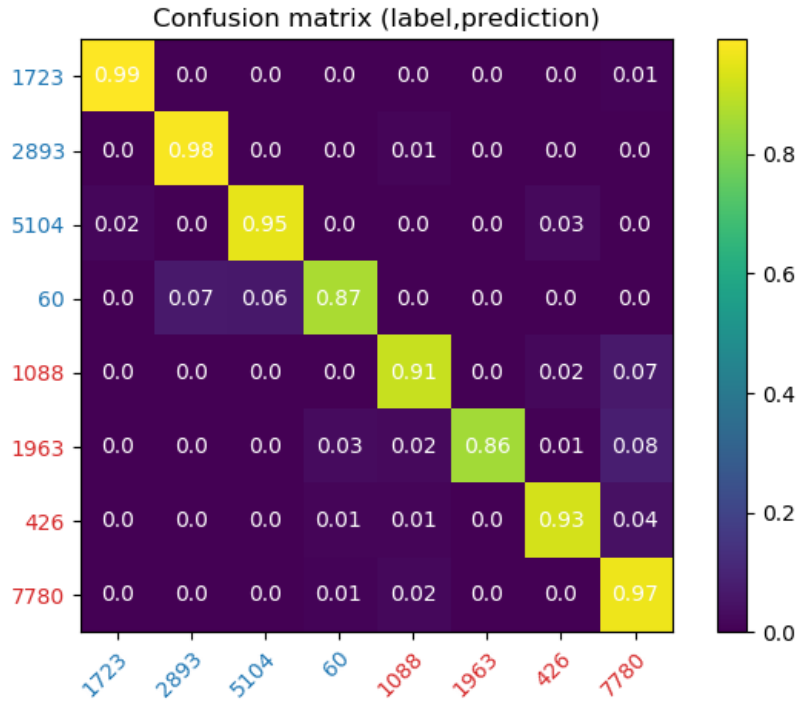


Figure 4.5: A confusion representing the share of frames belonging to a given speaker (per row) by how it was classified (per column). The labels are the IDs from LibriSpeech. Males are in blue and females are in red.

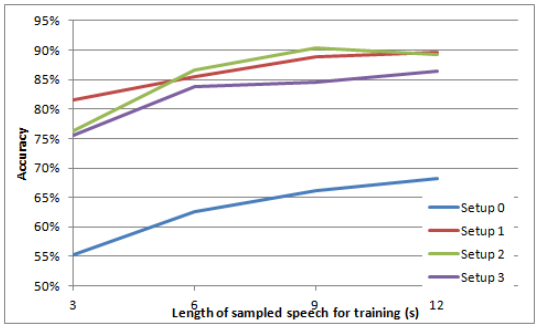
the full system.

The performance of di□erent configuration setups is shown in Figure 4.6. As seen in the figure, the introduction of classification segment improves the classification drastically. For the segmentation weighting and delta coe□cients the result is less clear. The segmentation weighting seems negligible at best, while the delta coe□cients seem to decrease the performance.

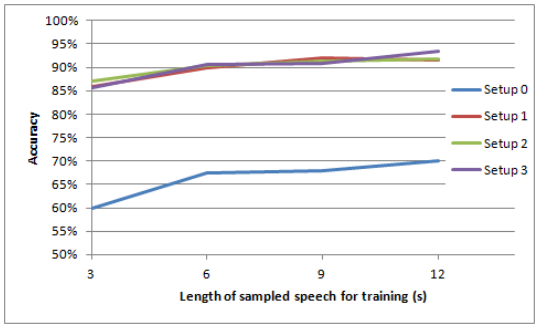
4.5 Performance Benchmark

To compare the system prototype implementation and the selection of speech corpus with others, the speaker identification experiment from [11] is repeated. This article use a 168 speaker (112 males, 56 females) subset of TIMIT with 10 utterances per speaker. Their own system and an implementation of Reynolds-Rose are trained on a given number of these utterances and then tested on the remaining utterances. The experiment can't be fully repeated without the expensive TIMIT corpus, but some approximation of it can be performed using free corpora. The full corpus consist of 6300 utterances with a total length of 5.4 hours. This results in a total of 3.1 seconds per utterance on average.

Using this approximation, the system prototype was tested with a segment-based test similar to the test in their report, but with TIMIT replaced by LibriSpeech.



(a) GMM



(b) MLP

Figure 4.6: The performance of the identification system when using different system setups.

Table 4.1: The table shows a re-creation of the speaker identification experiment from [11]. The prototype of the system implemented in this thesis is compared with the Reynolds-Rose algorithm implemented in [11] and the convolutional deep belief network and support vector machine combination introduced in [11]. For the system prototype, LibriSpeech replaced TIMIT as corpus.

Number of utterances	System prototype	Reynolds-Rose in [11]	CDBN + SVM in [11]
1	63.5 %	40.2 %	90.0 %
2	78.5 %	87.9 %	97.9 %
3	84.5 %	95.9 %	98.7 %
5	89.5 %	99.2 %	99.2 %
8	94.4 %	99.7 %	99.7 %

The result, together with the result presented in their report, is shown in Table 4.1. Again note that the comparison is only approximate and the performance will depend on the corpus (see section 4.2), but based on the comparison, the system prototype seems to be on par with their Reynolds-Rose implementation, but lag behind their new system.

4.6 Real-Time Plausibility

For the system prototype, the actual recording module and graphical interface is still missing. This makes the real time requirement difficult to verify correctly. However, the plausibility of the requirement can be examined by clocking the extraction time of a feature from a frame and the MLP classification time of a feature.

Once the training data is sampled, there will inevitably be a small delay for the training of the classifiers as well. If the delay is too long, the user experience will deteriorate. Therefore the training time for the system was clocked as well.

In this thesis real-time is defined as less than 100 ms delay between speech and identification. Using Python's `time` library the best hyperparameters for $M = 12$ speakers with $T = 12$ seconds of training data were used. All other applications

were closed while running the test. The result were averaged over 25 repetitions and can be seen in Table 4.2.

Table 4.2: Table showing average execution time for different components of the system.

Measurement	Average Execution Time
Feature Extraction	160 µs
Feature Classification	120 µs
Training Time	9.2 s

As seen in the table both feature extraction time and classification time are several orders of magnitude lower than the allowed delay, and the training time is lower than the sampling time for a single speaker.

5

Discussion

As a general comment, the MLP seems to outperform the GMM in every test. This is a surprising result since MFCC-GMM was considered to be the preferred combination for speaker identification until the introduction of i-vectors and more advanced deep learning methods. However, it is important to keep in mind, that the application in this thesis is different from most articles in the field and that this result might not apply to speaker identification in general. Because of the simplified hyperparameter search, where all feature extraction parameters were fixed and the hyperparameters of the classifiers were limited to the number of mixture components (GMM) and the shape of the hidden layer (MLP), it cannot be guaranteed that the optimal performance for any classifier was achieved. It could be the case that the solution found was just a local optimum, but missed the global optimum if any. It seems reasonable however to believe that the conclusion would still be the same even with different parameters.

The key experiment of the report was how the amount of training data affected the performance. From the project objective stated in the introduction, the system could at maximum be allowed $T = 12$ seconds of training data. The result of the experiment showed that the accuracy of the system for $T = 12$ started at 99 % of all active frames for two speakers, and declined almost linearly down to 88 % for 12 speakers. Considering that: this was the maximum allowed amount of training data, this was only the average performance, the actual performance could be even worse as indicated by the speaker corpora comparison, the session was relatively noise free and contained no overlapping frames; the result has to be deemed as unsatisfactory for the intended application. While 100 % might be unrealistic, 88 % is too unreliable for practical purposes. If the system was run as a complement to a speech recognition system, and it could use the output from the speech recognition to more easily derive the correct speaker from the context of the speech. As this is not the case for this setup, an accuracy below 95 % would imply too many mistakes for practical purposes.

Having said that, there are two simple adjustments that would automatically improve the accuracy. The first would be to ease the constraint of the sampling time being limited to 12 seconds. This would make the system less convenient to use, but certainly increase the accuracy up to some point. The second adjustment would be to let the voice activity detector be more aggressive in rejecting frames. A prediction based on a somewhat noisy frame is more error prone than a prediction based on a clean frame. A more aggressive voice activity detector would remove more of the

noisy frames and thereby improve the accuracy.

The thesis introduced the distinction between the ongoing frame based test and the segment-based prerecorded test. To my knowledge, no other articles have used a similar ongoing test, which is more telling about the performance in the end application, but makes it more difficult to compare with others. As demonstrated by both the speech corpora comparison speaker set comparison, the corpus and configuration have a substantial impact on the accuracy. It is therefore import to find a standardised test using the free corpora.

The result from the speech corpora comparison seems to indicate that the system learned to recognise features of the recording environment to identify the speakers. In the intended application, these features would largely be unavailable as everyone is recorded by the same microphone. To some extent the distance and intensity of the speech could possibly be used. The finding that using the same source text deteriorated the performance was also unexpected. Intuitively, it seems like an easier task to contrast the different speakers based on the pronunciation of the same acoustic class. Given this result it seems more like that placed the classes to near each other and thus increased the errors when there was doubt.

As mentioned in the introduction, there are several traits that differ the groups in terms of voice. These include age, emotional and physical state etc. For ELSDSR and LibriSpeech it was easy to make a partition based on gender. In terms of voice the most distinguishing aspect between the groups is the fundamental frequency of the voice. It is thereby not surprising that a heterogeneous group of the two genders results in a higher accuracy than a homogeneous group, or that it seems most misclassifications occur within the same gender group. What is more surprising is the tilting of the curve that shows a majority male group generates a higher accuracy than a majority female group. This result demonstrates that the optimal solution could depend on the speaker set composition. The explanation behind the difference is beyond the scope of the thesis, but a possible explanation could be a lower variance among the voices (or the MFCC representation) within either this set or within the group. Another hypothesis is that males have a more stable monotonous voice over both training and classification, while females vary more. The variations could be either within the corpus set or within the group at large.

The result from the setup comparison was very surprising. The two additional features of weighting the moving segment and including the delta k coefficients sounded good in theory, but seems to have been superfluous in practice.

The performance benchmark test demonstrate the difference between the frame-based and the segment-based models. The performance for 12 speakers in the frame-based test was 88 %, for 168 speakers in this test the performance was 89.5 %. It was nonetheless disappointing that the system was not closer in performance to the one used as reference. Without more details about the speech corpus it is difficult to say if this due to the speech corpus difference or error in the system.

Given the performance from the real-time plausibility experiment, the system seems to be well on its way to reach the real-time constraint. The remaining steps of recording the live speech and displaying it on a graphical interface, now have almost

two thirds of the maximum delay allowed left to use.

5.1 Future Work

As with every project, a scope has to be placed to make the project feasible to complete within the project time. Beyond what was treated in this thesis, there are several other directions to explore.

Some simple modifications to improve the performance of the system were suggested in the previous section of this chapter. This included extending the sampling time of the training data and making the voice activity detector more aggressive. While these changes most certainly would improve the accuracy, the extent of the improve might still not reach satisfactory result. Instead the the use of modern features such as i-vectors and CDBNs mentioned in the section 1.2. It is still not clear how these features would perform in a more noisy environment or when the voices in the session differ substantially from the pre-training. The language issue could be solved by first detecting language used, and then switch between different pre-trained configurations to match the session language. Instead of using MLP as classfier, ELM has shown promising results in other articles although a direct comparison has not been found.

Some improvements could probably also be achieved by analysing the the log-probabilities directly further. Unreasonably short classification segments before or in a longer segment of a given speaker is a sign of misclassification that could be adjusted. Especially if the likelihood value make it is a close call.

When switching from the static segment based test to an ongoing frame based test, the base algorithms had to be adjusted. A different approach could have been to focus on converting the ongoing session into a series of segments by instead attempting to detect the change of speaker. This would have been a very different project and more difficult if the real time requirement was to be kept.

The frame-based test was designed to closer model the intended end application, but should only be seen as a primitive model. For more reliable results, more careful definition of the end application is needed and the test modified accordingly. But even with a more refined model, a real life test is also needed.

At the start of the project, additional functionality to detect new speakers not in the training session was planned. In the end, there was no time to implement this and it thus left as future work.

6

Conclusion

The results of this thesis suggest that neither classifier, Gaussian mixture models or multi-layer perceptron, in combination with mel-frequency cepstral coe□cients generate satisfactory performance for the intended application. It would probably still be possible to increase the performance of the existing system by either easing the constraints or making the voice activity detector more aggressive in rejecting frames. To reach the desired performance without these changes,however, more modern features and classifiers are recommended.

The intended application motivated adjustments of existing algorithms to suit this specific application rarely covered in the literature. This make the study and its results unique. A significant proportion of the project time was spent on designing and implementing a more proper evaluation method. While being far from a complete model, it provides some aspects left out in traditional benchmarks.

The result is di□cult to compare with other articles as the application is di□erent and the thesis relied on free, but uncommon speech corpora. To get indications of how the speaker set composition and the corpus a□ect the accuracy, two separate experiments were conducted. The result from these experiments indicate that these choices can a□ect the accuracy with around two percentage point each, which is worth taking note of when reading other articles. Especially the speaker set composition could be investigated further to find an explanation and possibly a solution.

Bibliography

[1] Leena Mary and SpringerLink (e-book collection). *Extraction and representation of prosody for speaker, speech and language recognition*. English. 2012th ed. New York: Springer Science+Business Media, 2011. isbn: 9781461411598.

[2] Florian Eyben, Klaus Scherer, Björn Schuller, et al. "The Geneva Minimalistic Acoustic Parameter Set (GeMAPS) for Voice Research and A□ective Computing". English. In: *IEEE transactions on a ective computing* 7.2 (2016), pp. 190–202.

[3] W. M. Campbell D. A. Reynolds. "Springer handbook of speech processing". In: ed. by Jacob Benesty, M. M. Sondhi, and Yiteng Huang. Berlin;London; Springer, 2008. Chap. Text-Independent Speaker Recognition, pp. 763–781.

[4] Ava Kofman. "Forget about Siri and Alexa - When it comes to voice identification, the "NSA Reigns Supreme"". In: *The Intercept* (Jan. 2018). url: https://theintercept.com/2018/01/19/voice-recognition-technology-nsa/.

[5] "Speaker Identification Integrated Project passes final field test". In: *Interpol News and Media* (Nov. 2017). Press Release. url: https://web.archive.org/web/20180125002333/https://www.interpol.int/News-and-media/News/2017/N2017-165.

[6] Robert B Miller. "Response time in man-computer conversational transactions". In: *Proceedings of the December 9-11, 1968, fall joint computer conference, part I*. ACM. 1968, pp. 267–277.

[7] D. A. Reynolds and R. C. Rose. "Robust text-independent speaker identification using Gaussian mixture speaker models". English. In: *IEEE Transactions on Speech and Audio Processing* 3.1 (1995), pp. 72–83.

[8] P. Kenny, G. Boulianne, P. Ouellet, et al. "Joint Factor Analysis Versus Eigenchannels in Speaker Recognition". English. In: *IEEE Transactions on Audio, Speech, and Language Processing* 15.4 (2007), pp. 1435–1447.

[9] Najim Dehak, Patrick J. Kenny, Réda Dehak, et al. "Front-End Factor Analysis for Speaker Verification". English. In: *IEEE Transactions on Audio, Speech, and Language Processing* 19.4 (2011), pp. 788–798.

[10] Misaki Tsujikawa, Tsuyoki Nishikawa, and Tomoko Matsui. "I-vector-based speaker identification with extremely short utterances for both training and testing". English. In: IEEE, 2017, pp. 1–4.

[11] Honglak Lee, Peter Pham, Yan Largman, et al. "Unsupervised feature learning for audio classification using convolutional deep belief networks". In: *Advances in neural information processing systems*. 2009, pp. 1096–1104.

[12] Yanick Lukic, Carlo Vogt, Oliver Durr, et al. “Speaker identification and clustering using convolutional neural networks”. English. In: IEEE, 2016, pp. 1–6.

[13] Yuan Lan, Zongjiang Hu, Yeng C. Soh, et al. “An extreme learning machine approach for speaker recognition”. English. In: *Neural Computing and Applications* 22.3 (2013), pp. 417–425.

[14] Khan S. Ahmad, Anil S. Thosar, Jagannath H. Nirmal, et al. “A unique approach in text independent speaker recognition using MFCC feature sets and probabilistic neural network”. English. In: IEEE, 2015, pp. 1–6.

[15] Pulkit Verma and Pradip K. Das. “i-Vectors in speech processing applications: a survey”. English. In: *International Journal of Speech Technology* 18.4 (2015), pp. 529–546. issn: 1381-2416.

[16] Frédéric Bimbot, Jean-François Bonastre, Corinne Fredouille, et al. “A Tutorial on Text-Independent Speaker Verification”. English. In: *EURASIP Journal on Advances in Signal Processing* 2004.4 (2004), pp. 1–22.

[17] S. S. Stevens, J. Volkmann, and E. B. Newman. “A Scale for the Measurement of the Psychological Magnitude Pitch”. English. In: *The Journal of the Acoustical Society of America* 8.3 (1937), pp. 185–190.

[18] A. M. Noll. “Short-Time Spectrum and “Cepstrum” Techniques for Vocal-Pitch Detection”. English. In: *The Journal of the Acoustical Society of America* 36.2 (1964), p. 296.

[19] S. Furui. “Speaker-independent isolated word recognition based on emphasized spectral dynamics”. English. In: vol. 11. 1986, pp. 1991–1994.

[20] Kevin P. Murphy. *Machine Learning - A Probabilistic Perspective*. MIT Press, 2012.

[21] Diederik P. Kingma and Jimmy Ba. “Adam: A Method for Stochastic Optimization”. In: *CoRR* abs/1412.6980 (2014). arXiv: 1412.6980. url: http://arxiv.org/abs/1412.6980.

[22] Geo□rey Hinton, Li Deng, Dong Yu, et al. “Deep Neural Networks for Acoustic Modeling in Speech Recognition”. In: *Signal Processing Magazine* (2012).

[23] John S Garofalo, Lori F Lamel, William M Fisher, et al. “The DARPA TIMIT acoustic-phonetic continuous speech corpus cdrom”. In: *Linguistic Data Consortium* (1993).

[24] Vassil Panayotov, Guoguo Chen, Daniel Povey, et al. “Librispeech: An ASR corpus based on public domain audio books”. English. In: IEEE, 2015, pp. 5206–5210.

[25] Ling Feng and Lars Kai Hansen. “A new database for speaker recognition”. In: (Jan. 2005).

[26] Tanya Stivers, N. J. Enfield, Penelope Brown, et al. “Universals and Cultural Variation in Turn-Taking in Conversation”. English. In: *Proceedings of the National Academy of Sciences of the United States of America* 106.26 (2009), pp. 10587–10592. issn: 0027-8424.