Semester Thesis
at the Department of Information Technology
and Electrical Engineering

Audio-Visual Speech Activity Detection

Salome Mannale

Advisors: Tofigh Naghibi, Dr. Beat Pfister
Professor: Prof. Dr. Lothar Thiele

January 16, 2014
Abstract

Speech activity detection is an application in speech processing where the presence or absence of human voice is detected. In this project not only audio data but also video data of the lip movements is used in order to improve the performance of the detector. The algorithm can be separated into three blocks, namely the feature extraction, the audio and video classification and the hidden Markov model. It is shown, that the audio data is more reliable for speech activity detection than the video data, but adding video data leads to an improvement of the detector. Especially, in presence of speech type background noise the video data is of great value.
Acknowledgements

I would like to thank Prof. Dr. Lothar Thiele and Beat Pfister for giving me the opportunity to write my semester thesis in the speech processing group.

Furthermore, I would like to thank my supervisor Tofigh Naghibi. His support was very helpful in order to accomplish this project. I appreciated your patience and openness to discuss ideas and problems with me.

Salome Mannale
Zurich, Schweiz
December 2013
Contents

Abstract i

Acknowledgements iii

1 Introduction 1
  1.1 Motivation ................................................. 1
  1.2 Problem Statement ......................................... 1
  1.3 Outline ..................................................... 2

2 Audio-Visual Speech Activity Detection 3
  2.1 Feature Extraction .......................................... 4
  2.2 Classification ............................................... 7
  2.3 Hidden Markov Model ........................................ 11

3 Performance Evaluation 13
  3.1 Random Forest vs. Support Vector Machine ............... 13
  3.2 Number of Trees for Random Forest ..................... 15
  3.3 Number of PCA Features .................................. 16
  3.4 Weighting Audio and Video Classification ............... 17
  3.5 Improvement Hidden Markov Models ...................... 19
  3.6 Speaker Independent Evaluation ........................... 20

4 Conclusion and Future Work 23

List of Tables 25

List of Figures 27

References ................................................... 29
Introduction

1.1 Motivation

Speech activity detection is an application which detects the presence or absence of human voice. It is an important topic in the current speech recognition technology, as it is used in a variety of different applications such as in speech encoding or hands-free telephony. Often utterance detection is a necessary step prior to actually start processing speech.

The most common approach for speech activity detection is to evaluate the energy level of the input signal. This approach is simple but has the drawback that it cannot distinguish between speech and loud background noise. So for this project a statistical approach is proposed to detect speech. It works with the help of audio signals and video data.

1.2 Problem Statement

The goal of this thesis is to use audio as well as video data to improve the speech activity detector. The audio data contain the information about the presence or absence of voice, whereas the video data is used to detect speech with the help of lip movements.

In a first step the features which are describing some characteristics of speech are extracted from the video and audio signals. For the audio data spectral features are used and on the video data a principle component
analysis (PCA) is performed to get the PCA features.

In a second step the audio and video features are used to train two different classifiers, one for the audio data and one for the video data. A small part of the data is not used for training but to test the classification algorithm. Two different classification algorithms are proposed and compared in this thesis, namely the support vector machine algorithm and the random forest algorithm. The output of the classifiers for the test data is a probability indicating the confidence of the classification.

In a third step the two observation probabilities of the audio and video classifiers are weighted and combined. To include the time dynamics a hidden Markov model is used with two possible states eventually, speech and non-speech. Different transition probabilities are tested. The maximum likelihood state sequence is found with the Viterbi algorithm.

1.3 Outline

In Chapter 2 the whole procedure of the speech activity detection is described. The focus lies mainly on section 2.2 about classification algorithms and section 2.3 about the hidden Markov models. In Chapter 3 the performance of the procedure is evaluated. In chapter 4 the main results of the procedure are concluded and an outlook on future projects on this topic is given.
Audio-Visual Speech Activity Detection

In this chapter the procedure of the algorithm is explained step by step following the path in figure 2.1. The algorithm can be separated into three blocks. First the features of the audio and video data are extracted. Then the audio and video feature vectors $X_{aud}$ and $X_{vid}$ are classified, where the output of the classifier is a probability giving the confidence of the classification. The audio-visual speech activity detector can be improved.
by modeling it as a hidden Markov model in order to include the time
dynamics of the signals.

2.1 Feature Extraction

The goal of the feature extraction is to extract the information of the
audio and video data which is characteristic for speech or non-speech.
The classifier is then trained based on the extracted features. So
the performance of the classification depends highly on the feature
extraction. In this chapter it is explained what kind of features are used
to feed the classifier.

The audio samples and video frames used in this thesis are from 6
different speakers, who were recorded in a computer room. In this room
the speech is always accompanied by noise coming from the computers.
Sometimes other people are speaking in the room and the squeaking of
an opening and closing door can be heard. Each of the six individual
data sets is balanced, which means that the number of speech samples is
approximately equal to the number of non-speech samples.

Fig. 2.2: The six recorded speakers in the computer room.

2.1.1 Audio Data

The audio features are derived from the audio samples. The first 13
features are spectral features. The audio data is divided into frames
which are overlapping in time. Each frame is transformed with a discrete
Fourier transformation into frequency domain. The audio data in the
2.1. Feature Extraction

The frequency domain is then split into 13 frequency bands and the spectrum is averaged in each of this 13 bands. The next 26 features are delta-features and delta-delta features. Delta features are calculated as follows:

\[ \Delta c[m] = \frac{\sum_{i=1}^{k} i(c[m + i] - c[m - i])}{2 \sum_{i=1}^{k} i^2}, \]  \tag{2.1}

where \( c[m] \) is the vector of the 13 spectral features at time \( m \) and \( \Delta c[m] \) is the respective delta-feature-vector. The delta-delta features are calculated similarly with equation [2.1] but \( c[m] \) are not the spectral features but the delta-features. After the delta-delta features we have 10 features of linear predictive coding coefficients. Linear predictive coding uses the information that adjacent audio samples are not statistically uncorrelated. The \( n-th \) audio sample \( s[n] \) is estimated by a weighted sum of the previous samples \( s[n - 1], s[n - 2], ..., s[n - K] \), where the weights \( a_k \) called LPC-coefficients are derived by minimizing

\[ e^2[n] = (s[n] - \tilde{s}[n])^2 = \left(s[n] + \sum_{k=1}^{K} a_k s[n - k]\right)^2. \]  \tag{2.2}

The last feature contains the zero crossing rate of the frame.
2.1.2 Video Data

To get the video features the mouth region of the image has to be extracted first, as it is shown in figure 2.5. This image is then transformed into frequency domain by a two dimensional discrete cosine transform (2DCT). The discrete cosine transform is similar to the discrete Fourier transform and is calculated as follows:

\[ X[k] = \frac{1}{2}(x[0] + (-1)^k x[N - 1]) + \sum_{n=1}^{N-2} x[n] \cos \left( \frac{\pi}{N-1} nk \right), \quad k = 0, \ldots, N - 1. \]  

(2.3)

The low frequencies contain more information. Hence the 192 features with the lowest frequencies in \( x \)- and \( y \)-directions are extracted in zigzag order. After these 192 features, there are 8 features containing the \( x \)- and \( y \)-coordinates of the most left, the most right, the uppermost and the most lowest point of the lip. Then there is one feature containing the width and one feature containing the height of the lip.

![Fig. 2.5: Example of closed and open mouth image.](image)

Fig. 2.5: Example of closed and open mouth image.

| 192 | 8  | 1   |

Fig. 2.6: The structure of the video features.

2.1.2.1 Principal Component Analysis

Each video sample contains 202 features whereas the audio samples only contain 51 features. On one hand the large number of video samples slows the simulation down and on the other hand not all of the video features contain the same amount of information. So the goal is to find a transformation matrix \( T \) to reduce the number of 2DCT features from 192 to \( L \) transformed PCA-features, where \( L < 192 \). The transformed \( L \) features should contain as much information as possible. Since the last 10 features are very informative the transformation is executed solely on the first \( n = 192 \) features.
This transformation matrix $T$ can be found by the principal component analysis as follows:

Suppose we have a matrix $A \in \mathbb{R}^{m \times n}$ with $m$ feature vectors containing $n$ features. The covariance matrix $A^T A$ of $A$ can be decomposed into

$$A^T A = V \Lambda V^{-1},$$

(2.4)

where the diagonal entries $\lambda_i$ of $\Lambda$ are called eigenvalues of $A^T A$ and $V$ is a unitary matrix containing the eigenvectors of $A^T A$.

The transformation matrix $T$ consists of the eigenvectors in $V$ corresponding to the $L$ largest eigenvalues.

$$T = V_L,$$

(2.5)

where $T \in \mathbb{R}^{n \times L}$. So each feature vector $v \in \mathbb{R}^{1 \times n}$ can be transformed to a feature vector with $L$ features by

$$\hat{v} = vT,$$

(2.6)

where $\hat{v} \in \mathbb{R}^{1 \times L}$ is the new feature vector with reduced dimension. The evaluation of a good choice of $L$ is done in section 3.3.

Fig. 2.7: The new structure with the $L$ PCA-features.

### 2.2 Classification

Classification is a problem of statistics and machine learning. It is assumed that each feature vector belongs to a certain group, called class labels. In this project each feature vector belongs to one of two different groups, namely speech denoted with the class label $Y = 1$ or non-speech denoted with the class label $Y = -1$. Two different
classification algorithms are proposed to solve the classification problem, namely the random forest and the support vector machine algorithm. Both algorithms are explained with the help of a simple example.

2.2.1 A Binary Classification

Assume we have two groups of sample points generated from different Gaussian distributions as you can see in figure 2.8. Each feature vector contains two features: the \( x_1 \)- and \( x_2 \)-coordinates. For both groups the \( x_1 \)- and \( x_2 \)-coordinates are distributed with a standard deviation \( \sigma = 2 \). For the blue group the sample points are centered around the mean \( \mu = (4, 4)^T \) and for the green group they are centered around the mean \( \mu = (8, 8)^T \) as demonstrated in figure 2.8. The class labels of the green and blue group are known and therefore they are the training data set. The class labels of the red feature vectors are unknown, some of them belong to the green group and some of them belong to the blue group. The goal is to find the class labels of the red feature vectors with the help of the training data. To get a reliable classification the set of the training data has to be sufficiently large.

Fig. 2.8: Simple 2 dimensional classification example
2.2.2 Random Forest

The class labels of the test data can be found by using the random forest algorithm. The random forest algorithm uses decision trees to determine the class labels. One of these trees is depicted in figure 2.9. By following the branches of this tree every red feature vector can be classified.

![Classification tree generated by the training data.](image)

The random forest algorithm generates an ensemble of decision trees. Each of these trees is generated by randomly selecting training data and randomly selecting some of the features. The majority vote of the trees is the final decision to which of the two groups a feature vector belongs to. The output of the random forest can also be represented as probability. If 5 such trees are generated and the decision of 3 trees was $Y = 1$ and the decision of two trees was $Y = -1$, the probability that the feature vector belongs to group one is $3/5$, whereas it belongs to group $Y = -1$ with probability $2/5$, respectively. The more features each feature vector contains, the more decision trees have to be generated. Further information about the random forest algorithm can be found in chapter 15 of [1].
2.2.3 Support Vector Machine

The support vector machine (SVM) algorithm describes a different approach for the classification problem. It is assumed that the training data can be split into two separable groups. Often this is not possible by a linear function. In order to achieve this, the data is transformed into a higher dimensional space with a transformation called kernel function. In this higher dimensional space a separation is hopefully possible resulting in a decision boundary as shown in figure 2.10. To determine the class labels of the red feature vectors, it is evaluated on which side of the decision boundary these points are laying. So all the red points situated on the green side of the decision boundary are assumed to be part of the green group. The output of the support vector machine can also be represented as a probability of being part of a certain class. This probability is derived from the distance to the decision boundary. The closer a feature vector is to the decision boundary, the less certain is its classification. In the above example the feature vector only consisted of the two features $x_1$ and $x_2$ in order to image the algorithms. Of course the problem can be extended into higher dimensions as it is the case for the audio and video feature vectors. In this project a radial basis function is used as the kernel function to determine the decision boundary. Further information about the support vector machine algorithm can be found in chapter 12 of [1].

![Fig. 2.10: Decision boundary found by the training data.](image-url)
2.3 Hidden Markov Model

To improve the performance of the classifier a hidden Markov model is used. The classifier determines the class labels of the feature vectors independent of time. But in reality the class labels are not independent of time at all. For example it is not possible for someone to speak just one microsecond. So if a frame belongs to the class label speech, it is much more likely that the adjacent frames also belong to the class label speech than that they belong to the class label non-speech. As depicted in figure 2.11 in this project we have two possible states: state 1 (speech) and state -1 (non-speech). $p_{11}$ and $p_{22}$ are the probabilities that the state remains the same and $p_{12}$ and $p_{21}$ are the probabilities that there is a state change. For this project it assumed that the speech sequences have more or less the same length as the non-speech sequences so $p_{11} = p_{22}$ and $p_{12} = p_{21}$. Since the state has to be constant for a certain amount of time $p_{11} > p_{12}$ and $p_{22} > p_{21}$.

The current state $y_n$ can be estimated i.e. computed by

$$p(y_n|x_1, \ldots, x_n) = \frac{p(y_n, x_1, \ldots, x_n)}{p(x_1, \ldots, x_n)} \propto p(y_n, x_1, \ldots, x_n) \quad (2.7)$$

To find the maximum likelihood sequence of states $(y_1, y_2, \ldots, y_n)$ the Viterbi algorithm is used. A description of the Viterbi algorithm can be found in [2]. The drawback of the hidden Markov model is the occurring delay. In order to prevent a long delay, the hidden Markov model is reset after a certain time $T$, such that the resulting delay equals $T$. More information about hidden Markov models can be found in chapter 5 of [3].
Performance Evaluation

To evaluate the performance of the speech activity detection procedure some parameters have to be set. These values are defined in table 3.1. A 12-fold cross validation method is used, which means the data set is split into 12 groups. To get an averaged result, the simulation runs 12 times, where each of the 12 groups serves once as a test data and the remaining 11 are used as training data for classification.

As a quality measure the classification rate is used:

\[
\text{classification rate} = \frac{\text{number of samples classified correctly}}{\text{total number of samples}} \quad (3.1)
\]

<table>
<thead>
<tr>
<th>Variable</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classification Algorithm</td>
<td>Random Forest</td>
</tr>
<tr>
<td>Number of Trees</td>
<td>250</td>
</tr>
<tr>
<td>Reduction of Video Features</td>
<td>(L = 20) features</td>
</tr>
<tr>
<td>Reset time (T) of HMM</td>
<td>60 samples = 1 sec</td>
</tr>
</tbody>
</table>

Tab. 3.1: Default values for the simulation.

3.1 Random Forest vs. Support Vector Machine

As described in section 2.2, two different classification algorithms are used to classify the audio and video frames. In this section the two algorithms are compared. For each of the 12 groups from the cross-validation method
we get a classification rate as defined in equation (3.1). These values vary from group to group depending on the difficulty of the respective data set.

3.1.1 Audio Classification

![Audio Classification Graph](image)

Fig. 3.1: Comparing audio classification for SVM and random forest.

<table>
<thead>
<tr>
<th>Audio Classification</th>
<th>Classification Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random Forest</td>
<td>85.09% ± 8.18%</td>
</tr>
<tr>
<td>SVM</td>
<td>85.95% ± 7.18%</td>
</tr>
</tbody>
</table>

Tab. 3.2: Average values over the 12 groups for audio classification.

For audio classification the performance of the SVM and random forest classifiers are very similar for all twelve groups as it can be seen in figure 3.1. The averaged classification rates for SVM and random forest algorithm resumed in table 3.2 are very close as well, although the SVM algorithm reaches a slightly higher classification rate with a smaller standard deviation.

3.1.2 Video Classification

Both classifiers have more trouble classifying video frames than classifying audio frames, which implies that audio classification is more
3.2. Number of Trees for Random Forest

The number of trees necessary for classification depends on the number of features and the number of training data. The more features and the more training data available the higher the number of trees necessary in order to get a stable result. Of course the more decision trees the better but there is a tradeoff between accuracy and computation time. An significant than video classification. The performance seems to depend a lot on the data set, since both classifiers show high variances. For video classification the random forest algorithm performs considerably better than the SVM algorithm. The overall classification rate for random forest is 70.29%, whereas it is only 55.81% for the SVM classifier. Since high classification rates have to be achieved for both audio and video classification the random forest is the more suitable algorithm for this task. Any further evaluations are done using the random forest algorithm.

![Diagram](image)

**Fig. 3.2:** Comparing video classification for SVM and random forest.

<table>
<thead>
<tr>
<th>Video Classification</th>
<th>Classification Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random Forest</td>
<td>70.65% ± 16.21%</td>
</tr>
<tr>
<td>SVM</td>
<td>55.81% ± 13.40%</td>
</tr>
</tbody>
</table>

Tab. 3.3: Average values over the 12 groups for video classification.

3.2 Number of Trees for Random Forest

The number of trees necessary for classification depends on the number of features and the number of training data. The more features and the more training data available the higher the number of trees necessary in order to get a stable result. Of course the more decision trees the better but there is a tradeoff between accuracy and computation time. An
evaluation for audio and video classification with different numbers of classification trees reveals that 250 classification trees is a good tradeoff between computation time and stability. Starting from 250 trees the classification rate seems to be stable for audio and video classification. In figure 3.3 it can be seen, that the audio classification already seems to be stable with 100 trees and the video classification seems to be stable with 250 trees. The default value of 250 trees is therefore a reasonable choice.

3.3 Number of PCA Features

In section 2.1.2 the dimension reduction of the video features is explained. $L$ is the number of the PCA-features and $L + 10$ is the total length of the feature vector. The smaller $L$ the faster the algorithm, but if $L$ is too small there is not enough information in the feature vectors anymore. If $L$ is too large, the feature vector has features with very little information, which are weighted equally as the features containing important information about speech or non-speech states. Figure 3.4 shows that the number of PCA-features should be around 20 to achieve a good classification rate with the video data.
3.4 Weighting Audio and Video Classification

The outputs of the audio and video classifier are probabilities indicating the certainty of the respective class label decisions. These two statistical outputs of the audio and video classifier have to be combined. Since audio classification reaches a higher classification rate than video classification the two probabilities are weighted differently. The combined probability depending on the feature vectors $X_{aud}$ and $X_{vid}$ is defined as follows:

$$p_{tot}(Y = y|X_{aud}, X_{vid}) = \alpha \cdot p_{aud}(Y = y|X_{aud}) + (1 - \alpha) \cdot p_{vid}(Y = y|X_{vid}),$$  \hspace{1cm} (3.2)

where $\alpha \in [0, 1]$ is the weight of the combination and $y \in \{-1, 1\}$ denotes the two possible outputs non-speech $Y = -1$ and speech $Y = 1$.

The decision $\hat{Y}$ is based on the combined probability $p_{tot}$ as follows:

$$\hat{Y} = \begin{cases} -1, & \text{if } p_{tot}(Y = -1|X_{aud}, X_{vid}) > p_{tot}(Y = 1|X_{aud}, X_{vid}) \\ 1, & \text{if } p_{tot}(Y = 1|X_{aud}, X_{vid}) > p_{tot}(Y = -1|X_{aud}, X_{vid}) \end{cases}$$  \hspace{1cm} (3.3)

The goal is to find a weight which maximizes the classification rate. Since the audio classification performs generally better than the video classification, the optimal $\alpha$ is expected to be close to 1. But it seems that the video classification is also significant in order to achieve a high
classification rate. The optimal weight for audio and video classification is \( \alpha = 0.62 \) as it is shown in figure 3.5.

![Optimal Weight](image)

**Fig. 3.5:** Classification rates evaluated for different \( \alpha \).

In general the classification rate gets slightly better by including video classification. For audio classification only (\( \alpha = 1 \)) a classification rate of 85.10% and for video classification only (\( \alpha = 0 \)) 70.22% can be achieved, whereas with an optimal combination of the two classifications a rate of 86.68% can be achieved as it is resumed in table 3.4.

<table>
<thead>
<tr>
<th>Classification Type</th>
<th>Classification Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Audio</td>
<td>85.09% ± 8.18%</td>
</tr>
<tr>
<td>Video</td>
<td>70.65% ± 16.21%</td>
</tr>
<tr>
<td>Audio and Video</td>
<td>86.82% ± 8.66%</td>
</tr>
</tbody>
</table>

**Tab. 3.4:** Maximal classification rates achieved.

More interesting though is the ROC curve. It gives information about what kind of misclassifications are done by the classifiers. There are two possible misclassifications: false accept and false reject. False accept means, that a feature vector is wrongly classified as speech and false reject means, that a feature vector is wrongly classified as non-speech. In figure 3.6 it can be seen that video classification performs better with respect to false accept than audio classification. So including video classification can lower the false accept rate of the audio classification.
at a cost of a higher false reject rate. This decrease of the false accept rate results of misclassifications done by the audio classifier when other people are speaking in the computer room, which does not affect the video classification. So if an application is sensitive to speech type background noise, including video data might be very valuable. On the opposite if the video classification is given too much weight, the result is a high false reject rate and also the false accept rate increases.

![ROC curve for varying weight α.](image)

### 3.5 Improvement Hidden Markov Models

As explained in section 2.3 a hidden Markov model is used to improve the performance of the classification algorithm by including the time dynamics of speech activity. In order to do this, appropriate transition probabilities as in figure 2.11 have to be chosen. These probabilities are resumed in the transition probability matrix

$$
Tr = \begin{pmatrix}
    p_{11} & p_{12} \\
    p_{21} & p_{22}
\end{pmatrix} = \begin{pmatrix}
    p_s & 1 - p_s \\
    1 - p_s & p_s
\end{pmatrix},
$$

(3.4)

where $p_s$ is the stay-in-state-probability. For $p_s = 0.5$ the HMM has no influence on the classification rates, since a change of state is equally likely as a stay in state. For $p_s = 1$ the HMM is very strong, meaning that the state sequence stays in state for the hole sequence. So if $p_s$ is large it is
important to reset the HMM in regular and short intervals, which is done anyway to prevent a long delay. The reset interval is chosen to be 60 frames long which equals one second.

In [3.7] it can be seen, that for $p_s = 0.5$ (without using the HMM) the lowest classification rate is obtained. These classification rates are depicted as dashed lines in figure [3.7]. So for $p_s > 0.5$ the HMM improves the classification rate. As expected the highest classification rates are achieved with a combination of the audio and video classification. The improvement and resulting $p_s$ are resumed in table [3.5].

![Classification HMM with varying stay-in-state probability $p_s$](image)

<table>
<thead>
<tr>
<th>Classification Rates</th>
<th>Without HMM</th>
<th>With HMM</th>
<th>$p_s$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Audio</td>
<td>84.95% ± 8.46%</td>
<td>87.69% ± 10.01%</td>
<td>1</td>
</tr>
<tr>
<td>Video</td>
<td>71.31% ± 15.41%</td>
<td>73.70% ± 18.43%</td>
<td>0.9</td>
</tr>
<tr>
<td>Audio and Video</td>
<td>86.76% ± 8.35%</td>
<td>90.59% ± 7.74%</td>
<td>0.6</td>
</tr>
</tbody>
</table>

Tab. 3.5: Classification rates without and with using HMM.

### 3.6 Speaker Independent Evaluation

As stated in section 2.1 the audio and video data are from 6 different speakers. With a 12-fold cross validation method each data set is divided into two. One of the 12 groups is always used as test data and the
remaining 11 are used for training. Hence there is always data from the same speaker of the test data in the training data. Since there are many applications where no speech of the user is given for training, the performance of the algorithm for a speaker independent setting is evaluated. Still a 12-fold cross validation method is used with the exception that the training data from the same speaker as the test data is removed from training.

In figure 3.8 it can be seen that for audio classification the speaker independent setting performs approximately equally good as the speaker dependent setting. This is a very pleasing result, as speech from the user of a certain application of this algorithm is not necessarily needed for training the application, which simplifies its use. For the video classification the speaker independent setting has a significantly lower classification rate than the speaker dependent setting. It seems that the video data of the lip movements is more dependent on the speaker than the audio data.

<table>
<thead>
<tr>
<th>Classification Rates</th>
<th>Speaker Dependent</th>
<th>Speaker Independent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Audio</td>
<td>85.09% ± 8.18%</td>
<td>84.80% ± 8.11%</td>
</tr>
<tr>
<td>Video</td>
<td>70.65% ± 16.21%</td>
<td>55.62% ± 10.56%</td>
</tr>
<tr>
<td>Audio and Video</td>
<td>89.22% ± 7.49%</td>
<td>88.16% ± 8.43%</td>
</tr>
</tbody>
</table>

Tab. 3.6: Averaged results speaker dependent and independent setting.
Conclusion and Future Work

For this application the random forest algorithm is more suitable, since its classification rate for video classification is considerably higher than the classification rate of the support vector machine algorithm. Including video data leads to an improvement of the classification rate especially if a hidden Markov model is used. In presence of speech type background noise the video classification might be of great help for the audio classification. Nevertheless audio classification is more significant for speech activity detection and is therefore higher weighted in the combination of audio and video classification. The speaker independent setting shows that for audio classification a similar classification rate is achieved as for the speaker dependent setting. But for video classification the speaker independent setting leads to a significant decrease of the classification rate, because the features extracted of the lip images seem to be highly dependent on the speaker.

In future projects it might be interesting to try other classification algorithms and compare them to the basic approach of detecting the energy level of the audio signal. The video classification rate could be increased by a better lip movement extraction algorithm. Another possibility to reach a higher classification rate could be achieved by a more complex hidden Markov model, which is trained with the given training data.
List of Tables

3.1 Default values .................................................. 13
3.2 Audio classification for SVM and random forest ............ 14
3.3 Video classification for SVM and random forest ............. 15
3.4 Maximal classification rates achieved ......................... 18
3.5 Classification rates without and with using HMM ............. 20
3.6 Averaged results speaker dependent and independent setting ..................................................... 21
# List of Figures

2.1 Proposed procedure ........................................... 3  
2.2 The six recorded speakers in the computer room ............. 4  
2.3 Audio data of speaker one ...................................... 5  
2.4 The structure of each audio feature vector .................... 5  
2.5 Example of closed and open mouth image ..................... 6  
2.6 The structure of the video features ........................... 6  
2.7 The new structure with the \( L \) PCA-features ............... 7  
2.8 Simple 2 dimensional classification example ................. 8  
2.9 Classification tree generated by the training data ........... 9  
2.10 Decision boundary found by the training data ............... 10  
2.11 Hidden Markov model with two states ....................... 11  

3.1 Comparing audio classification for SVM and random forest. 14  
3.2 Comparing video classification for SVM and random forest 15  
3.3 Audio and video classification rates for varying number of trees ........................................... 16  
3.4 Classification rates and standard deviations for varying number of PCA features .............................. 17  
3.5 Classification rates evaluated for different \( \alpha \) .............. 18  
3.6 ROC curve for varying weight \( \alpha \) ............................ 19  
3.7 Classification HMM with varying stay-in-state probability \( p_s \) ......................................................... 20  
3.8 Speaker dependent and independent setting .................... 21
Bibliography


