Speech activity detection in videos

Viktor Andersson

Nelly Ostréus

LUND UNIVERSITY

Department of Automatic Control

MSc Thesis
TFRT-6171
ISSN 0280-5316

Department of Automatic Control
Lund University
Box 118
SE-221 00 LUND
Sweden

© 2022 by Viktor Andersson & Nelly Ostréus. All rights reserved.
Printed in Sweden by Tryckeriet i E-huset
Lund 2022

Abstract

Speech is an important way of communication all over the world. The speech information is encoded both aural and visual. More than 1.5 billion people have hearing loss and for those the visual information is even more important than for people with normal hearing. Lip reading is therefore an important research topic.

In this master thesis, machine learning algorithms were used to identify speech activity in realistic video with monologues and dialogues. Each video contained three persons speaking: one performing a monologue and two performing a dialogue. Support vector machines for linear, radial basis function, sigmoid and polynomial kernels were used to classify the audio as either speech or non-speech based on faces from realistic videos. A speech envelope was calculated and resampled to four Hertz. Based on a threshold of the envelope, the ground truth was created and each audio data point was selected to be either speech or non-speech. Convolutional neural networks using max-margin object detection were used to extract facial landmarks from the videos. Six different video features were calculated and used: the mouth opening distances, the variance of the mouth opening distances and the difference of mouth opening distances between several frames, the mouth area, the variance of the area and the difference of area between several frames.

The mean accuracy for the speech activity in the monologues were low. This was probably due to the unbalanced data in the monologues, since most data in the ground truth were classified as speech. For the dialogues, the accuracy were slightly higher than classifying everything as the most frequent class. The variance of the mouth area was the best performing feature. The performance varies between the videos and combining the best mouth opening distances feature with the best mouth area feature for the two best kernels, increased the accuracy for the best performing videos.

Acknowledgements

We would like to give a huge thanks to our supervisor at Department of Automatic Control, Bo Bernhardsson, and our supervisors from Oticon A/S Emina Alickovic, Martin Skoglund and Johannes Zaar. You have always been there when we needed you and given valuable advice. The discussions during the Monday meetings have been interesting and contributed to a better project. The opportunity to visit Eriksholm Research Centre was highly appreciated and the study trip led to a nice environmental change and a really fun day!

We would also like to thank all our colleagues in the master thesis room at the Department of Automatic Control. You have all truly lightened up our days!

The computations and the data handling were enabled by resources provided by the Swedish National Infrastructure for Computing (SNIC) at Chalmers Centre for Computational Science and Engineering (C3SE) partially funded by the Swedish Research Council through grant agreement no. 2018-05973.

Contents

List of Acronyms

CNN Convolutional Neural Network

CV Cross-validation

HOG Histogram of Oriented Gradients

MMOD Max-Margin Object Detection

Rbf Radial basis function

RMS Root Mean Square

STFT Short-Term Fourier Transform

SMA Simple Moving Average

SVM Support Vector Machine

1

Introduction

1.1 Motivation

Speech is the most common way humans communicate with each other. Most of the information gathered from speech is collected via the ears but some of the information is also gathered via the eyes and the visual cues a person emits. This makes it possible for hearing-impaired persons to use their vision to understand speech. The ability of using visual perception to understand speech is called lip reading. Lip reading can be a useful tool since more than 1.5 billion people worldwide are affected by hearing loss and 430 million of these people have moderate or high levels of hearing loss in the better hearing ear according to World Health Organisation. Without any help people with higher levels of hearing loss are more likely to be negatively affected. If hearing loss is not addressed it can have several severe negative effects on many aspect of a person's life. These aspects can for example influence communication, the development of language and speech in children, cognition, education, employment, mental health and interpersonal relationships. One solution suggested to combat hearing loss is the usage of sensory substitutions like sign language and lip reading. Lip reading is therefore a relevant and important topic [*World report on hearing: executive summary* 2021].

1.2 Aim of project

It is well-established that listeners with hearing loss have more difficulties following conversations when there are multiple speakers talking at the same time, than listeners with normal hearing [Shinn-Cunningham and Best, 2008; Lee et al., 2018]. Oticon, one of the largest manufactures of hearing aids in the world, and their research center Eriksholm Research Centre (ERH) are interested in investigating if these difficulties can be overcome by developing new algorithms that can support hearing-aid users during selective attention tasks.

The aim of this project is to investigate if there are some facial features that correlate to speech activity and how well these features perform. This is achieved

by using machine learning algorithms to classify speech/non-speech, given access to only video files with the audio recordings as ground truth of speech and non-speech.

1.3 Previous research

The last few years there has been an increase in models and systems for lip reading. There has been a push for systems with automatic lip reading and with it methods based on deep learning have become popular and advanced the achievable performance. Traditionally, automatic lip reading has mainly been based on the extraction of visual features and the classification and modeling of the spoken sequence. These systems addressed simple recognition tasks like alphabet or digit recognition. With the shift to more advanced deep learning architecture, more complex and realistic recognition tasks became possible, for example situations with lip reading. These advances were also possible with the increase of large-scale databases like the Lip Reading Words database and Lip Reading Sentences database that are based on recordings of BBC programs from 2010 to 2016 [Chung et al., 2017]. Most databases include only one face in every video and the face looks directly towards the recorder. However, other databases have been produced that include recordings of the face from different angles. The majority of databases are in English. Other languages like French and Spanish are much less frequent and for smaller languages it is hard to find even one database where both audio and visual data is included [Fernandez-Lopez and Sukno, 2018].

Most lip reading systems start by detecting face in the video and extracting the area of the mouth and areas surrounding it. Feature extraction is then applied to the speaker's lips. There are several techniques of feature extraction for visual speech recognition and there is no exact consensus of what is the best technique. Therefore, researchers have proposed several different visual features based on image transform, motion (optical flow), geometry (width and height of the mouth) and statistical model [Fernandez-Lopez and Sukno, 2018].

The shift to systems dealing with words and sentence recognition has both been seen in traditional automatic lip reading and in deep learning. Most of the traditional systems use image features with hidden Markov-model or support vector machine (SVM) as classifier while the deep learning systems have moved toward end-to-end deep neural network architectures. These are dominated by convolutional neural networks (CNN) features in combination with long short-term memories (LSTM) [Fernandez-Lopez and Sukno, 2018]. One of the leading end-to-end models is LipNet. It performs prediction on sequences of sentences for visual speech recognition. It uses a spatio temporal convolutional neural network, recurrent neural networks, and the connectionist temporal classification loss. When LipNet was performed on the database GRID which uses sentences as data, it gave an accuracy of 95.2 % of a sentence [Assael et al., 2016].

2

Background

2.1 Cocktail-party problem

The cocktail party phenomenon refers to our ability to focus on a single talker of interest in the presence of other competing speakers in noisy and complex environments [Cherry, 1953]. In most cases it is relatively easy for our brains to segregate and follow a sound source of interest while filtering out other voices [Bronkhorst, 2000]. The sounds of the environment are summed together linearly into a single sound stream per ear and the auditory system processes the single sound stream.

Only by separating features from different sources and by grouping together features origination from the same spatial source, can a listener single out the specific sound stream to listen to. This mechanism is called sound segregation [Marinato and Baldauf, 2019]. Intertwined with sound segregation is the ability to direct attention to the sound source of interest while ignoring others and switching attention between different sources, for example when following two conversations. The cognitive process can often only operate on one thing at a time and therefore selects a particularly sound source to focus on [McDermott, 2009]. For direct attention the binaural hearing, hearing from both ears, gives better ability to detect the location of incoming sound. This ability is reduced when one ear or both have the hearing capability reduced [Tris Atmaja, 2019].

However, these abilities to tune out other voices and focus on one specific voice or "selective hearing" is something not everybody can do. Since selective hearing uses the auditory system to understand speech in multiple-talker situations it can be difficult for hearing-impaired people to understand speech in these kinds of situations. These difficulties are often associated with the term cocktail-party problem or cocktail-party effect. The cocktail-party problem asks, in essence, the simple question "How do we recognise what one person is saying when others are speaking at the same time?" [Bronkhorst, 2000]. The problem was originally mentioned by Cherry (1953). Behavioural studies have attempted to answer this problem since the 1950s. There have been several attempts to solve the cocktail-party problem since then and progress has been made but there are still many things that are unknown and issues that are not explained in understanding how the activity of the

brain creates selective hearing. Reiss and Molis (2021) used stimulation of different stimuli in each ear with a variation in fundamental frequency to explore the presence of speech fusion (that is the blending of stimuli between the two ears). This was performed with two groups of listeners with normal hearing or hearing loss. Most participants from both groups reported hearing only one vowel (fused the vowels) when the stimulation did not differ in fundamental frequency in both of the ears. When vowel fundamental frequency increased between ears, listeners with normal-hearing sensitivity indicated the presence of two vowels, while listeners with hearing loss continued to report only one vowel. The authors concluded that this spectral blending between ears may degrade speech recognition in noisy cocktail-parting settings with competing talkers. This study showed the issue of the cocktail party problem and highlights the importance of finding ways for patients with hearing loss to manage these challenging listening situations.

2.2 Linguistics

A key part in visual speech processing and lip reading is the understanding of the movements of the mouth and tongue and how it creates speech. The classification of the sounds of the speech is called phonology. The classification is different for each language but all are structured around phonemes. Phonemes are the smallest detectable unit of sound in a language. It serves to distinguish words from another. Not all phonemes can be seen via the movements of the mouth and face. These visually distinctive units are called visemes and the number of visemes is much smaller than the number of phonemes.

This is because some phonemes are produced inside the mouth and throat, and cannot be seen. This means that several phonemes have the same viseme, for example /p/ and /b/ when spoken in English. Words that have the same visual distinction but have different phonemes are called homophones, for example the words "pet", "bell" and "men" all have the same movements but sound differently. Homophones are a crucial part of the problems lip reading has as a means of understanding what a person is saying without the usage of hearing. Even though the visual movements of the mouth are the same, other factors can contribute to the understanding of what is said, for example the timing and duration during the actual speech in terms of visual "signature" of a given gesture that cannot be captured with a single photograph. Conversely, some sounds which are hard to distinguish acoustically are clearly distinguished by the face in a frame [Chen, 2001]. For example, acoustically speaking English /l/ and /r/ can be quite similar especially in clusters, such as "grass" compared to "glass", yet visual information can show a clear contrast. This is demonstrated by the more frequent mishearing of words on the telephone than in person. Some linguists have argued that speech is best understood as bimodal (aural and visual), and comprehension can be compromised if one of these two domains is absent [McGurk and MacDonald, 1976]. Studies have shown that when English-

speaking subjects were tested on their ability to interpret lip reading, the accuracy of correct understood words was only 30 %. This was mostly due to homophones [National Deaf Children's Society, n.d.]

The number of phonemes in different languages is still highly debatable but in spoken English there are approximately 44 phonemes. The number of phonemes is not decisive and different numbers are given by different sources [Grønnum, 2005; Basbøll, 2005]. However, according to [Grønnum, 2005] there are eleven vowel phonemes (/i e a y ø œ u o /) and fifteen consonant phonemes (/m n p t k b d g f v s h l r j/) in spoken Danish. The vowel phonemes often have different, so called, allophones depending on length, especially in conjunction with /r/, while consonants phonemes have many different allophones. Allophones is when a letter's sound changes based on the word, for example, /ø/ is lowered when it occurs either before or after /r/, and /a/ is pronounced /æ/ when it is long. This adds another difficulty in differentiating vowels and consonants from each other in speech.

2.3 Sound processing

Sound is almost in every case a mono-dimensional signal or as it is also called function of time and it is supposed to represent the air pressure in the ear canal. When doing the digital audio sampling, an analogue-to-digital conversion, storage and transmission is needed. To perform processing on the audio signal it needs to be digitised and that means it must be converted to discrete samples of a discrete-time domain. This operation of turning a signal to discrete time from continuous time is called sampling. It is performed by picking values from the continuous time signal with an interval of T. From this a sampling rate F_s can also be calculated via equation (2.1). The unit of sampling rate is Hertz (Hz) or samples per second. This means that when interval of T decreases, the sampling rate increases and one get a more similar version of the discrete-time signal to the continuous-time signal [Rocchesso, 2003].

$$F_s = \frac{1}{T} \tag{2.1}$$

The fundamental rule of sampling is the equation (2.2). It describes the sampling of a continuous-time signal with a sampling rate and produces a discrete-time signal. The discrete-time signal frequency spectrum is a periodic replication of the spectrum of the original signal, and the replication period of F_s. The angular frequency ω for functions of discrete variable is converted to the frequency variable f as described by equation (2.2) [Rocchesso, 2003].

$$\omega = 2\pi f T = \frac{2\pi f}{F_s} \tag{2.2}$$

In the conversion from continuous-time signals to discrete-time signal, the Nyquist-Shannon sampling theorem limits how high the sample rate can be to allow

a discrete sequence of samples to represent all the information of continuous-time signal. To correctly represent the information of the continuous signal $x(t)$ that has spectral content limited to frequencies smaller than F_b the sampling rate needs to be $F_s > 2F_b$ [Rocchesso, 2003].

Another factor that affects the analogue-to-digital conversion is the bit depth. It is the number of bits of information in each sample and it directly corresponds to the resolution of the sample.

When dealing with extracting features of an audio signal $x(m)$, where m is the discrete index, one can do it both in the time-domain but also in the frequency domain. An example of a feature in the time domain is short-term energy, E_n. It is a simple and effective parameter for classifying speech and non-speech segments. Signal energy is described in equation (2.3), where E represent the energy of the audio signal $x(m)$ [Jalil et al., 2013].

$$E = \sum_{m=-\infty}^{\infty} x(m)^2 \tag{2.3}$$

The signal energy of a short-term speech signal can be described according to equation (2.4), where $w(n-m)$ is the window, n is the sample the analysis window is centred on and N is the window length [Jalil et al., 2013]. The window can be several different window types but in this project it is a hamming window and is defined according to equation (2.5) [Oppenheim, 1999]. The result E_n is the energy of N samples of a signal at index m, where the last sample has index n. In equation (2.4) $n-N+1$ represents the length of the signal.

$$E_n = \sum_{m=n-N+1}^{n} [x(m)w(n-m)]^2 \tag{2.4}$$

$$w(n-m) = 0.54 - 0.46\cos(2\pi\frac{n-m}{N}) \tag{2.5}$$

To get access to features in the frequency domain the audio signal needs to be converted from a function of time to a function of frequency X. This can be done with Short-term Fourier transform (STFT). STFT basically works as Fourier transform but the time domain signal is divided into smaller windows and the Fourier Transform is calculated for each window section to get the frequencies, see equation (2.6) where n is the time index [Sairamya et al., 2019].

$$X_n(e^{j\omega}) = \sum_{m=-\infty}^{\infty} x(m)w(n-m)e^{-j\omega m} \tag{2.6}$$

An example of features from the frequency domain is the spectral spread S_t of the signal. It is derived from the spectral centroid C_t, see equation (2.8), where $X_t(k)$ is the magnitude of the discrete Fourier Transform coefficients of the t:th audio frame and W_L is the window length. The magnitude of the discrete Fourier

coefficients, $X_t(k)$, is defined in equation (2.7) can be treated as a measure of the intensity with which the respective frequency participates in the signal $x(m)$ [Giannakopoulos and Pikrakis, 2014].

$$X_t(k) = |\sum_{m=0}^{N-1} x(m) e^{-\frac{j2\pi}{N}km}|, \quad k = 1, ..., W_L \tag{2.7}$$

$$C_t = \frac{\sum_{k=1}^{W_L} kX_t(k)}{\sum_{k=1}^{W_L} X_t(k)} \tag{2.8}$$

The spectral spread is used as a measure of distribution around the spectral centroid and it defines the variance from the spectral centroid. Spectral spread can be calculated using the equation (2.9) [Krishnamurthi et al., 2022; Giannakopoulos and Pikrakis, 2014].

$$S_t = \sqrt{\frac{\sum_{k=1}^{W_L} (k - C_t)^2 X_t(k)}{\sum_{k=1}^{W_L} X_t(k)}} \tag{2.9}$$

2.4 Image processing

A pixel is short for "picture element" and is the smallest, controllable part of an image visible on a screen. Each pixel in an image has a colour and the colour can be manipulated with different algorithms. An algorithm that transforms the pixel value independent on the pixel's neighbours is called a point operator. If the algorithm is using some of the values from the pixel's neighbours to change the pixel's value, the algorithm is a neighbourhood operator. A general image operator takes an image, or multiple ones, as input and returns an output image. Mathematically, this can be expressed as in equation (2.10) for the discrete domain, where $\mathbf{x} = (i)$ is pixel location, h is the image processing operator, f is the input image and g is the output image [Szeliski, 2021; Russ, 2008].

$$g(i) = h(f(i)) \tag{2.10}$$

The point operators are algorithms that affect all pixels the same, no matter where in the image the pixel is located or how the surrounding of the pixel looks like. Examples of popular point processors are algorithms to adjust the contrast or the brightness in the image. The contrast and the brightness of the image could be adjusted according to equation (2.11), where a changes the contrast of the image and b adjusts the brightness of the image [Szeliski, 2021].

$$g(\mathbf{x}) = af(\mathbf{x}) + b, \quad a > 0 \tag{2.11}$$

A commonly used neighbourhood operator is the linear filter. When using a linear filter, each pixel value becomes a weighted sum of the pixel values from the close surrounding to the pixel. This can be done as described in equation (2.12), where $\mathbf{x} = (i, j)$ is pixel location. This can also be denoted as in equation (2.13) [Szeliski, 2021].

$$g(i,j) = \sum_{k,l} f(i+k, j+l)h(k,l) \tag{2.12}$$

$$g = f \otimes h \tag{2.13}$$

2.5 Machine Learning

Traditional programming is a sequence of instructions which a computer is following step by step. Programming using machine learning is a computer science area, where the computer is programmed to learn from data based on different mathematical models. The computer is handling new data based on what it has learned from the data it has previously seen. Tom Mitchell's definition of machine learning is commonly used and is as follows: "A computer program is said to learn from experience E with respect to some task T and some performance P, if its performance on T, as measured by P, improves with experience E" [Mitchell, 1997]. The computer is programmed to learn to do some task, often classifying between different classes. This is done by exploiting mathematical models. These can be improved by learning from some experience, in machine learning referred to as the training data. More training data improves the model and hence the performance of the task. The model is evaluated by using validation data. The validation data is new data which the model never has seen before. Based on some performance measure, for example accuracy or precision, the model is adjusted and fine-tuned to improve the performance of the task. When the developer is satisfied with the model, the model's performance is tested on a test set with new, previously unseen data, to see how good the model actually is.

Overfitted and underfitted models

If the performance of the validation data is good but the performance on the test data is not as good, the model is probably overfitting the training data. The overfitted model is often complex and has parameters that have found a pattern where there is no pattern. The model is therefore well adjusted to the specific cases in the training data but not as good at classifying the unseen test data. To avoid overfitted models, it is good with a large dataset and a flexible model with fewer parameters. A large dataset means that the machine learning model has learned from more data and is less adjusted to a specific training data point. Fewer parameters make the model less adjusted to the specific data points, minimises the number of irrelevant parameters

and makes the model more general. In practice, overfitting cannot be completely avoided and is something the developer should be aware about and try to avoid as much as possible [Géron, 2019; Zhou, 2021].

An underfitted model is a model that cannot perform the task as well as it could do. The model is so generalised that it cannot tell much about the data. It has not learned the general properties of the training data good enough to be able to handle the test data as good as possible. Making the model more complex by adding parameters to the model, makes the model less underfitted. Using better features to train the model can also decrease the underfitting of the data [Géron, 2019; Zhou, 2021].

Supervised and unsupervised learning

In supervised machine learning, the data is labelled. In unsupervised learning the data is not labelled. For example, a collection with images is going to be classified as either an image of an animal or an image of a plant and then the algorithm will be evaluated. For supervised learning, all the images have a label that states if the image is representing an animal or a plant. After the classification is done, the images will have been classified in two classes, one representing "animal" and one representing "plant". Since each image also has a label, the evaluation of the model can be done by for example calculating the accuracy, the precision or the recall.

If the animal and plant images were to be classified by unsupervised learning, this means that the images are not labelled "animal" or "plant". The algorithm then returns a number of clusters. The user often specifies how many clusters should get returned, in this case two, or how large distance it should be between the different clusters. If a distance measurement is used, a number of classes could be returned. In the animal/plant case, three clusters based on for example plants, sponges and other animals could be returned. Since the data were unlabelled, it is not possible to tell if an image were correctly or incorrectly clustered. The performance of unsupervised learning is not as easy to measure as the performance of the supervised learning. Different statistical measurements are usually used to verify that there are patterns in the data which could be clustered based on some non-random features. It is also checked whether the number of clusters which were created from the model are the same as the number of true clusters or not. Statistics from the data from each cluster are also compared and see if they match what would be expected from the clusters. Finally, the results from different models could also be visually compared. In the animal/plant case, the clusters with images could be compared between the models to see which model seems to perform best [Palacio-Niño and Berzal, 2019].

In this project, when using the video landmarks to predict the audio, the audio is labelled as either speech or non-speech. Therefore, methods for supervised learning are used in the classification. In this project, support vector machine (SVM) is used. Both the accuracy, the precision and the recall were studied for the different models. The accuracies for the models are presented in this thesis since it was from that the

main conclusions were made.

Balanced and unbalanced data

Ideally, the dataset should contain equally many samples from each class. However, that is often not the case in practice. The dataset is often unbalanced, with one class occurring more frequently than another. This can lead to an algorithm having a high accuracy not because it is performing great at the classification task, but only since it classifies everything or almost everything as the most common class. There is no easy solution to this, but three methods are commonly used to reduce the problem of unbalanced data. The first solution, undersampling, is simply to ignore some of the data of the most common class to obtain a more balanced relationship between the classes. The problem of this is that the algorithm is not trained on as much data, which could become a large problem if the dataset is small. Another method is to oversample the data with the least common class, to increase its occurance. The problem with this solution is that is could easily lead to an overfitted model. A third solution is to change the threshold for a data point being classified as either class. This could lead to the data being wrongly classified, even though it with the original threshold was correctly classified [Zhou, 2021].

Cross-validation

Cross-validation (CV) is a popular way to avoid overfitting. One of most popular cross-validation methods is k-fold cross-validation. When using k-fold cross validation, the data is first divided into k subsets or folds as they also are called. The subsets will be, if it is possible to split the data that way, of equal size and disjoint, meaning that the data is not occurring in several folds. Then the model is trained with $k-1$ folds at a time, with the final fold serving as test data. This is then done k times, so that all folds have been used as a test data at some point. This is illustrated in Figure 2.1, where blue represents the unseen data and the red represents the seen data for the model. The variable V in the figure stands for validation data. This way, the model is tested on unseen data several times, which decreases the risk of the model to overfit. The test scores are then averaged for the different test folds and used as a final test score. The large disadvantage of CV is the computational time [Zhou, 2021; Suthaharan, 2016].

Deep learning and convolutional neural network

Deep learning is a subset of machine learning and tries to mimic the human brain by creating an artificial neural network with artificial neurons that perform a weighted summation of its incoming signals and outputs a new signal. The artificial neuron is the basic element that the neural networks are built with. The nodes are connected with each other and this is called network architecture. There are two major groups of network architectures: feed-forward architecture and feed-back architecture. In feed-forward networks, there are no closed loops for signal transfer. They are often

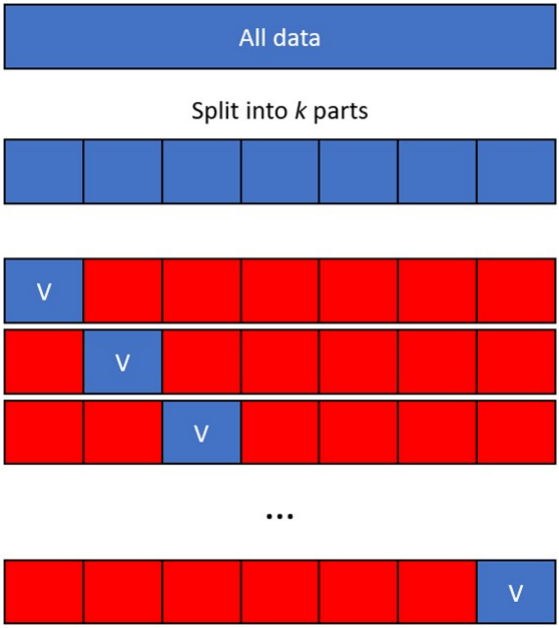

Figure 2.1 Drawing of a *k*-fold cross-validation. Blue represents the unseen data and the red represents the seen data. *V* stands for validation data.

structured to just let input values propagate forward through the layers of nodes to an output layer. For feed-forward networks one can view the input as a "question" and the output as the artificial neural networks "answer" [Goodfellow et al., 2016].

A form of feed-forward network is convolutional neural network (CNN) which is designed to handle the spatial relation between inputs. In convolutional networks the first argument to the convolution is called the input and all the other arguments are called kernels. CNNs consist of an input layer, hidden layers and an output layer. The hidden layers include not just the input and output masked by the activation function and final convolution but also layers that perform convolutions. The convolutional layers consist of three blocks: the convolution stage, the activation stage and the pooling stage. The first two blocks are of high importance while the pooling stage is not always necessary. One key attribute of convolutional networks is its use of sparse connectivity. It means that the kernel is smaller than the input, which decreases the number of parameters in the network. This leads to smaller memory requirements and the output requires fewer operations. These improvements make the CNN more efficient so it can better handle larger input datasets than fully connected feed-forward neural networks. This also allows units in the deeper layers

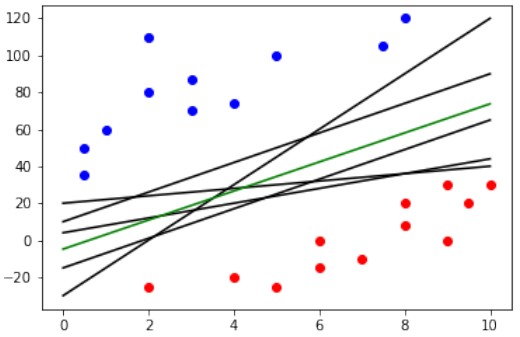

Figure 2.2 Multiple hyperplanes can separate the data points from the two classes. The optimal hyperplane to separate the blue and red data points is coloured green.

to indirectly interact with a larger part of the input and makes the network able to describe more complicated relations between many variables [Géron, 2019; Sutharan, 2016; Goodfellow et al., 2016].

SVM

Support vector machine (SVM) is a supervised method often used in classification problems. The method is commonly used in brain-computer interface related machine learning problems. The idea of SVM is that the data points form clusters based on their classes. By drawing a hyperplane, these clusters can be separated from each other with each cluster representing one class. In two dimensions, the hyperplane is a straight line and in three dimensions the hyperplane is a plane. In other words, the hyperplane is a straight line in multiple dimensions [Noble, 2006]. There is never just one hyperplane that separates the points, there are be multiple as illustrated in Figure 2.2.

The optimal hyperplane is obtained by maximising the distance to the data points. In Figure 2.2, the hyperplane that maximises the distance to the data points in red and blue is coloured green. The data points closest to the hyperplane have the largest impact on the position of the hyperplane. Since the hyperplane is "supported" by the data points with the minimal distance to the hyperplane, these data points are called "support vectors". The aim is to maximise the distance between the support vectors while minimising the prediction error.

A soft margin in the algorithm makes the classifier accept some outliers even though the prediction error increases. The regularisation parameter C regulates how sensitive the SVM should be to miss-classification of the training data points. A larger value on C will chose a smaller margin between the support vectors and the hyperplane if that increases the amount of correctly classified training data points.

However, to obtain the optimal hyperplane, a single outlier data point should not have too large impact on the position of the hyperplane. A large value on C therefore improves the accuracy, but at the same time increases the risk of overfitting the model. The parameter C is often set to a lower number at first and then gradually increased, and by trial-and-error the best value on C is selected [Marius, 2020; Noble, 2006; Géron, 2019; Suthaharan, 2016; Pedregosa et al., 2011].

The data points cannot always be separated by a straight line. By adding dimensions to the data, it could be possible to separate the data points from the different classes by a straight line. This is mathematically done by a kernel function. The linear, the polynomial, the sigmoid and the radial basis function (rbf) kernels are some of the most common kernel functions [Noble, 2006].

The linear kernel function The linear kernel function is the simplest one. When using many different features, the linear kernel is often applied to avoid the use of a complex, overfitted model. The large benefit of the linear kernel, compared to the other kernel functions, is that it is not computational heavy.

A linear hyperplane, described by equation (2.14), divides the data points into two subspaces D_1 and D_2, defined according to equation (2.15). The feature data points is denoted as $\mathbf{x}$. Each feature has a weight and the weights are denoted by $\mathbf{w}$. The intercept b is the bias. A linear combination of all features and their corresponding weights predicts the classification label . The hyperplane described in (2.14) is also known as the max-margin hyperplane [Suthaharan, 2016].

$$\mathbf{w}\mathbf{x}' + b = 0 \tag{2.14}$$

$$\begin{aligned} D_1 &= \{\mathbf{x} : \mathbf{w}\mathbf{x}' + b \leq 0\} \\ D_2 &= \{\mathbf{x} : \mathbf{w}\mathbf{x}' + b > 0\} \end{aligned} \tag{2.15}$$

The max-margin hyperplane should have as large distance as possible to D_1 and D_2. This is done by having two linear hyperplanes which serve as boundaries for the subspaces. These two hyperplanes are described by equation (2.16) [Suthaharan, 2016].

$$\begin{aligned} \mathbf{w}\mathbf{x}' + b &= 1, \mathbf{x} \in D_1 \\ \mathbf{w}\mathbf{x}' + b &= -1, \mathbf{x} \in D_2 \end{aligned} \tag{2.16}$$

The distance d between the parallel hyperplanes described in equation (2.16) can be calculated according to equation (2.17). The largest distance d is searched to create the max-margin hyperplane. Maximising d is the same as minimising the term $\|\mathbf{w}\|^2$. The weight vector is easy to derive, since its derivative is $\mathbf{w}$, which makes it easier to minimise the the weight vector instead of maximising the distance [Suthaharan, 2016].

$$\frac{d^2}{2} = \frac{1}{\frac{\|\mathbf{w}\|^2}{2}} \tag{2.17}$$

The polynomial kernel function The polynomial kernel functions can be used if there is a linear relationship between the classes of data, when more dimensions are added to the data. The polynomial kernel function can be described as equation (2.18), where $\mathbf{x}$ is the feature vector, $\mathbf{w}$ the weight of the vectors, b some constant and d the degree of the polynomial. When comparing equation (2.18) with the linear hyperplane in equation (2.14), the main difference is the degree d [Suthaharan, 2016].

$$\phi(\mathbf{x}, \mathbf{w}) = (\mathbf{x}'\mathbf{w} + b)^d \tag{2.18}$$

The mathematical method used to add more dimensions to be able to separate the data is often called the "kernel trick". The problem with adding dimensions, is that the more dimensions that are added, the more overfitted the model could become. However, if the polynomial degree is too low, the model will not be able to handle a more complex dataset. It is therefore common to start with a low degree on the polynomial function and increase it gradually [Géron, 2019].

The radial basis function (rbf) kernel The rbf kernel is often used for non-linear data. The downside of this method is that it for large datasets can require a lot of computational power. The rbf kernel function can be described by equation (2.19), where $\|\mathbf{x} - \mathbf{w}\|$ is the euclidean distance between each data point x and each data point l.

$$\phi_\gamma(\mathbf{x}, \mathbf{w}) = exp(-\gamma \|\mathbf{x} - \mathbf{w}\|^2) \tag{2.19}$$

The rbf kernel function contains the parameter γ, which is a measure on how large distance each support vector has influence on. A low value on γ means that the distance is large and a high value means that the influence distance is small. This means that if γ is low, points from far away are considered when the position of the hyperplane is calculated. A low value on γ therefore decreases the risk of overfitting and a large value on γ increases the risk of overfitting [Géron, 2019; Pedregosa et al., 2011].

The sigmoid kernel function The sigmoid kernel function, or hyperbolic tangent kernel function which it also is called, can be described by equation (2.20), where $\mathbf{x}$ is the feature vector, $\mathbf{w}$ is the weight of the vectors, b is some constant and γ is related to the distance from which a training data point has influence. An SVM classifier with the sigmoid kernel works as a neural network with two layers [Géron, 2019].

$$\phi(\mathbf{x}, \mathbf{w}) = tanh(\gamma\, \mathbf{x}'\mathbf{w} + b) \tag{2.20}$$

2.6 Facial landmarks detection

There are several different methods for facial landmark detection. The landmarks are either unique key locations in the face or interpolated points between these key points. The aim of the landmark detection method is to predict the locations of D landmarks $x = x_1, y_1, x_2, y_2, ..., x_D, y_D$, where x is the coordinate in horizontal direction and y is the coordinate in vertical direction of each facial landmark in the image.

Deep learning based methods, often based on CNN, for facial landmarks detection have recently become more popular than traditional algorithms such as holistic methods, constrained local methods (CLM) and regression-based methods. The deep learning facial landmark detection algorithms can be dived into pure-learning methods and hybrid deep learning methods. The pure-learning methods use only deep learning and the hybrid deep learning methods use both some 3D computer vision method and the deep learning methods. The advantage of the hybrid deep learning methods is that the computer vision detection algorithm often is better at handling faces from different angles than deep learning methods [Wu and Ji, 2018].

All methods is searching for a face and placing the facial landmarks coordinates within a bounding box given by some face detector. The algorithm is searching for different key points in the face which are easier to find. The unique key locations is often somewhere in the face where there is a sharp edge, since edges because of discontinuities are easier to detect. An example of key points, which thanks to the discontinuities of the colours in the image are easier to detect, is the corners of the eyes. The other landmarks around the eyes are then interpolated between the key points that marks the corners of the eyes [Wu and Ji, 2018].

3

Materials and methods

3.1 The Dataset

The dataset was created at Eriksholm Research Centre (ERH) by Sascha Bilert for his master thesis [Bilert, 2020]. The dataset is owned by Oticon and is not publicly available. The dataset consists of the stimuli (including audio and video), eye-tracking data and EEG data. In this thesis, only the stimuli is used.

The stimuli from the dataset consists of 24 different videos, with the corresponding audio, of monologues and dialogues in Danish. The dialogues and monologues are performed by four professional Danish actors (two males and two females). All video clips contain three of the Danish actors, of which one performed an improvised monologue at the same time as two the others performed a dialogue. The two actors who performed the dialogue were given an image each and were asked to find the twelve differences between the two images without showing them for each other. In Figure 3.1, the setup for video 1 can be seen, where the first person is performing a monologue and the second and third person are performing the dialogue. The position every actor has in each video can be seen in Table 3.1.

Figure 3.1 One frame from video 1 in the stimuli with the monologue to the left and the dialogue to the right.

Table 3.1 Description of the videos from the stimuli. Every actor has been given a number from 1 to 4. The letters M and D represent what type of speech the actor is performing in the video, where M represent a monologue and D represent a dialogue.

	Position 1		Position 2		Position 3	
Video	Actor	M/D	Actor	M/D	Actor	M/D
1	1	M	2	D	3	D
2	3	D	2	D	1	M
3	3	D	2	D	1	M
4	2	M	3	D	4	D
5	2	M	3	D	4	D
6	2	M	3	D	4	D
7	4	D	3	D	2	M
8	4	D	3	D	2	M
9	3	M	4	D	1	D
10	3	M	4	D	1	D
11	4	M	1	D	2	D
12	2	D	1	D	4	M
13	2	D	1	D	4	M
14	2	D	1	D	4	M
15	1	M	2	D	3	D
16	3	D	2	D	1	M
17	1	M	4	D	3	D
18	1	D	4	D	3	M
19	1	D	4	D	3	M
20	1	D	3	D	4	M
21	4	M	3	D	1	D
22	4	M	3	D	1	D
23	1	D	3	D	4	M
24	3	M	4	D	2	D

The videos were recorded until all differences in the images were discovered by the actors performing the dialogue, meaning that length of the different video clips are not the same. The videos have a frame rate of 25 frames per second. All actors were asked to try to keep the attention of the listeners by talking as engaged as possible. All actors wore microphones that recorded the audio with a sampling rate of 48 kHz. The actors who performed the dialogues sat directed toward each other and therefore there is a larger amount of noise in the dialogue audio files. All audio files have been preprocessed to only contain the audio from the actor who wore the microphone even if other people were talking next to the actor. However, the audio files of the dialogues still contain small levels of noise from the opposite speaker [Bilert, 2020].

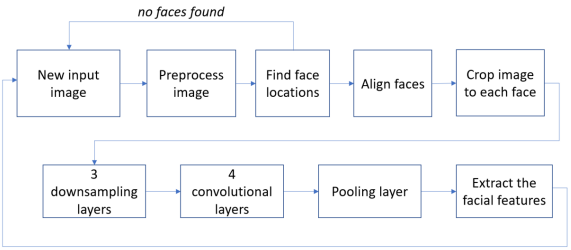

Figure 3.2 An overview of the CNN/MMOD algorithm for facial landmark detection.

3.2 Facial landmarks extraction from videos

Different facial landmarks were extracted from the faces in the videos using an algorithm for face detection from the software library dlib [King, 2009]. There are two different face detection algorithms implemented in dlib, one using histogram of oriented gradients (HOG) and linear support vector machine (SVM) and one based on a convolutional neural network (CNN) using max-margin object detection (MMOD). The CNN-algorithm using max-margin object detection is better at detecting faces that are turned away from the camera than the HOG/SVM algorithm. The downside of the CNN/MMOD method is that it is not as computational efficient as the HOG/SVM method. The computational power has not been a problem and we have therefore chosen to use the CNN/MMOD algorithm and are able to more thoroughly detect faces in the frames of the video. The algorithm has successfully been used on several different face detection and face recognition databases [Rosebrock, 2021; Sharma et al., 2016].

There are other facial landmarks extraction methods that are more complex. However, since the objective is to only classify speech activity and not for example words or sentences, a more simple method was chosen. The dialogues from the stimuli were turned towards each other and not towards the camera. This led to a requirement for a robust algorithm that could handle faces that were not centred in the image and turned towards the camera. The CNN/MMOD algorithm fulfilled these criteria and was therefore used in this project.

The CNN/MMOD algorithm is described with a drawing in Figure 3.2. The method first does some preprocessing to the input image. The image is normalised based on colour, brightness and contrast and is then upsampled. The algorithm then first finds the locations of the faces, then aligns the faces, crops the image to become an image for each face which just covers the face and finally extracts the facial landmarks from the cropped images. The first three layers in the CNN are downsampling the input images. After that, there are four convolutional layers which finds features in the image. Each convolutional layer takes the input and, based on the input, gives

Figure 3.3 Facial landmarks marked with white on a face.

a feature map as output. These layers uses local receptive fields, which gives each neuron a small area of the input to focus on. The convolutional layers are also sharing weights for the features for different parts of the image. There are also pooling layers which downsamples the feature maps. The last layer extracts the features [Sharma et al., 2016; King, 2009].

The CNN model is using the loss function Max-Margin Object Detection loss. This optimises the chances that a face is correctly found in an image. The function maximises the margin and chooses the alternative with the largest predicted margin to have a correctly classified image [King, 2015].

In total 68 landmarks were extracted from each face, of which 20 were around the lips. Twelve of the landmarks track the exterior side of the lips while the other eight landmarks track the interior side of the lips. Only the exterior landmarks have been used in this project. The landmarks consists of an x and an y coordinate. A drawing of the landmarks of the exterior lips together with the number of the coordinate of each landmark can be seen in Figure 3.4. These landmarks were later used to create the video features.

3.3 Processing of audio

To be able to classify speech and non-speech in the video it was necessary to have a ground truth. This ground truth came from the recorded audio from each actor in the video used to detect when there is speech and when there is not speech in the audio files. There are multiple ways of detecting speech in an audio file.

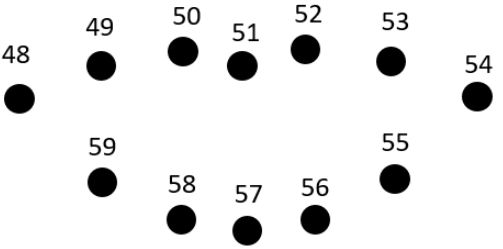

Figure 3.4 A drawing of the structure of the exterior facial landmarks around the lips.

Speech envelope using moving root mean square

One of them is by extracting the envelope of the speech and examine it. In order to extract the envelope of the speech several methods were studied, both on a ten second period and the entire audio file of one of the actors speaking in a monologue. The methods tested for extraction of the envelope of the audio signal were Hilbert transformation, moving root mean square (RMS) and peak envelope. These methods were first tested in MATLAB because all of them were available in the MATLAB function *envelope*. This made it easy to switch between them by only changing the input parameters to the function and a quick comparison could be made of the plots of different methods on the audio file.

Hilbert transformation [Oppenheim, 1999] gave an envelope that was too detailed for the case of only needing to know if the actor was speaking or not, see Figure 3.5. A too detailed envelope can cause some part of words with lower amplitude to be classified as non-speech as the ground truth, despite it still being speech. The issues with the Hilbert transformation were the opposite for the peak envelope method. It had the problem of sometimes not reaching lower amplitudes between the sounds as can be seen in Figure 3.6. Moving RMS gave a result that followed the envelope of the audio while still detecting periods when there was no sound, see Figure 3.7. This meant that the best method was moving RMS.

Moving Root Mean Square The method moving RMS is used to provide an estimated temporal evolution of the signal energy [Caetano and Rodet, 2011]. The envelope amplitude can be estimated with a sliding window to the RMS equation (3.1).

$$x_{RMS}(t) = \sqrt{\frac{1}{N}\sum_{i=1}^{N} w_i(t)x_i^2(t)} \tag{3.1}$$

In equation (3.1) the $x_i(t)$ is the i:th sample of the data centred around t as seen through the window $w_i(t)$. The variable t is the number of samples the analysis win-

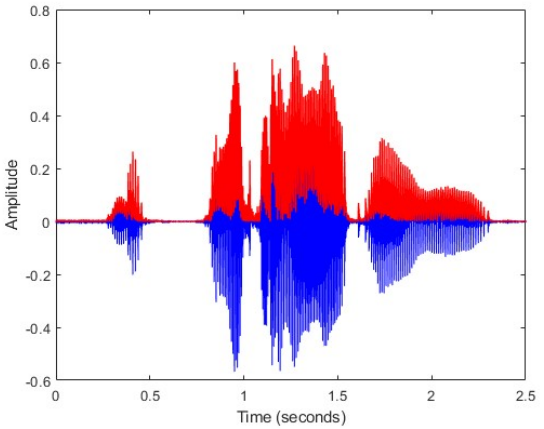

Figure 3.5 Speech envelope using Hilbert transform over 2.5 seconds of the monologue from video 1. Blue line: original audio. Red line: unfiltered speech envelope on top of the original audio.

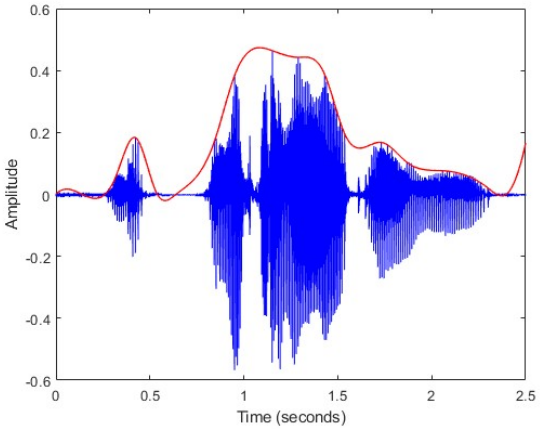

Figure 3.6 Audio envelope using peak envelope method over 2.5 seconds of the monologue from video 1. Blue line: original audio. Red line: speech envelope.

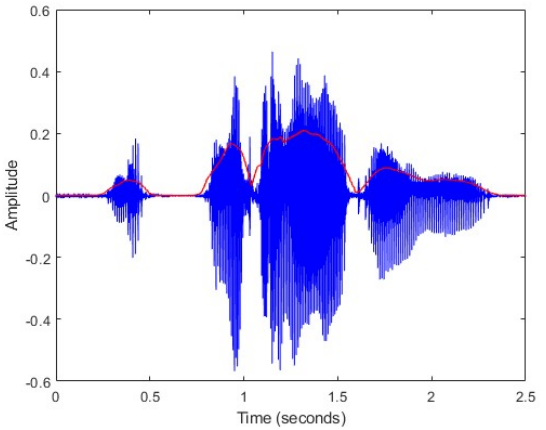

Figure 3.7 Audio envelope using moving RMS over 2.5 seconds of the monologue from video 1. Blue line: original audio. Red line: speech envelope.

dow moves and N is the length of the window. Different kinds of windows can be used with this method but in this project the most common is used, which is the rectangular window [Caetano and Rodet, 2011]. The moving RMS method functions as a moving average and a low-pass filter that smooths the signal. The length of the window gives a trade-off between the temporal sample rate of the envelope and how much information it represents. Small window sizes create an envelope that is sensitive to smaller changes in the audio file, for instance sudden changes in amplitude while presenting ripples in more steady regions. Larger window values smooth out the ripples but tend to lag behind abrupt energy changes. The moving RMS is calculated for each position in the data vector resulting in a moving RMS and a vector of several RMS. To compute the moving RMS at the edges of the data, where the algorithm does not have enough data to fill the window, the algorithm fills the empty slots in the window with zeros. This continues until the window has moved enough to be able to fill an entire window with data.

Calculating speech and non-speech The moving RMS was calculated for all the recorded audio files of the actors speaking in the videos. Since there was no right length of the window several lengths were calculated and tested. The best performing window length was 5000 data points, which corresponded to 104 milliseconds. To be able to compare it with the video it needed to have the same sample rate as the video's frame rate of 25 frames per second. The audio envelope was resampled by collecting every 1920:th value, because the sampling rate of the audio is 48 kHz. A vector was then created with 25 values of the audio envelope for each second. This was performed for every audio file with the window length of 5000 data points from each person in the video and resulted in 72 data files.

These 72 data files were later changed and updated after the realisation that with 25 frames per second it became to detailed and if the goal was to only separate when there is speech and non-speech a lower sample rate was needed. This sample rate was four frames per second. The resampling process was also changed from picking every 12000:th value of the audio envelope to calculating the mean value over every 12000 values without any overlap between the windows that the mean was calculated over. This resulted in a vector with four values for every second and each value representing the mean of 12000 data points of the audio envelope. A threshold was used to differentiate between speech and non-speech. Several values of the threshold was tested, for example the median value of the audio file. However, ultimately a constant threshold was chosen since with the median value it changed too much between monologue, dialogue and it became difficult to find a good solution.

Speech detection using short-term energy and spectral spread

When these 72 data files were later used as ground truth in the machine learning models they performed poorly and it was suggested that the decision making between speech and non-speech needed to detect the smaller details in the audio, for example letters in spoken words. Therefore, another algorithm was tested to see if it performed better. The difference between this algorithm and RMS windowing in Speech envelope is the addition of using spectral spread (2.9) for determining the threshold instead of only using moving RMS as was the case in Speech envelope.

The algorithm is called *detectSpeech* and has been modified from the original algorithm to use short-term energy and spectral spread as features instead of short-term energy and spectral centroid [The MathWorks, 2020; Giannakopoulos, 2009].

The algorithm can be seen in Figure 3.8. The algorithm is implemented by first converting input, the audio signal, to a time-frequency representation using STFT, see equation (2.6). The window length WL and the percentage of allowed overlap OL between windows in the STFT was 0.025 seconds and 20 %. From the converted audio signal short-term energy and spectral spread is calculated and saved into feature segments for every frame in the audio file. These segments are marked as Energy and Spread in Figure 3.8. Histogram are then created from each segment and a smoothing filter is applied on both histograms. Based on the respective smoothed histogram a threshold is determined according to equation (3.2). The variables M_1 and M_2 in the equation are the positions of the first and second local maxima of the specific histogram and the variable W is set to five. The two feature segments are then compared to their respective threshold. If the threshold is lower than the feature value the corresponding audio frame is added to a segment with only speech in it. As a post-processing step the detected speech segments are lengthened by five short term windows, which corresponds to 250 milliseconds, on both sides. Regions that are declared as speech are merged together if the distance between them are less than MD of accepted distance between two areas of detected speech. For this algorithm the parameter MD was set to 0.1 seconds. An audio sequence is plotted in

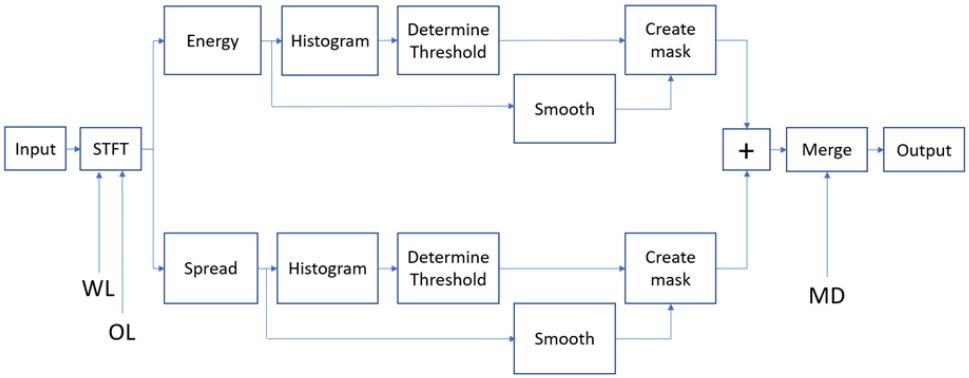

Figure 3.8 A drawing of a high-level overview of the *detectSpeech* algorithm. The variables *WL*, *OL* and *MD* stand for the length of the window, what kind of percentage of the windows are allowed to overlap, and the longest distance two speech areas are allowed to be apart in order to merge them into one area of speech.

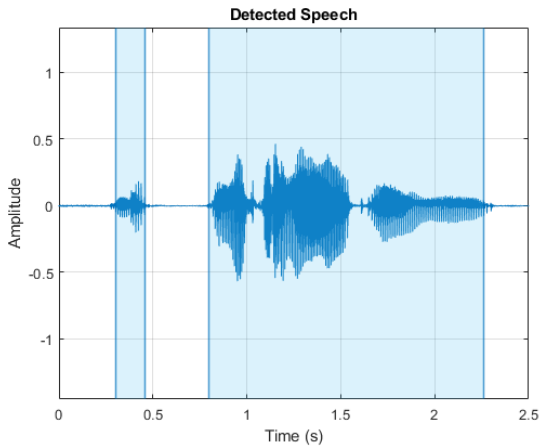

Figure 3.9 Plot of 2.5 seconds of audio where the blue area is classified as speech and the non-blue area is classified as non-speech.

Figure 3.9, where the blue areas are declared as speech according to the algorithm.

$$Threshold = \frac{W \times M_1 + M_2}{W + 1} \tag{3.2}$$

Since the separation of speech and non-speech was already made in the algorithm, the data files of all audios could quickly be made with ones representing

speech and zeros representing non-speech.

Final speech envelope

The method finally used was the moving RMS. This was then resampled to four frames per second. A constant threshold was used to differentiate if the data points from the audio envelope were classified as speech or non-speech. This method was chosen as the final version because it gave the best result when dividing the audios into speech and non-speech by visually comparing the different methods results to each other.

3.4 Video feature processing

Different features based on the landmarks from the lips were tested during the project on a couple of videos. The feature that worked best were the mouth opening distances and the area of the mouth opening. These features have been used just as they are, that is the area or the mouth opening distances for each landmark. The variance of these features has also been looked into as well as the difference of these features over time. To be able to later match the video features with the audio envelope features, the video features were resampled to four samples per second.

Feature 1 (F1). Mouth opening distances

The first feature, F1, used was the distances of the mouth opening in number of pixels. Three distances were calculated in horizontal direction and three in vertical direction. Originally, only the vertical mouth opening distances were used as part of the feature. However, after testing on a couple of videos it became clear that also the horizontal mouth opening distances are important in order to classify the speech activity. The mouth opening distances in x direction were calculated from the coordinate pairs 49-53, 48-54, 59-55 which were the coordinates in the vertical middle according to Figure 3.4. These coordinate pairs consist of one coordinate on the left side of the lips and one coordinate on the right side of the lips. The distances in x direction between each coordinate pair was called d_x and was calculated as specified in equation (3.3), where x_{right} was the x coordinate from the right side of the lips and x_{left} was the x coordinate from the left side of the lips.

$$d_x = x_{right} - x_{left} \tag{3.3}$$

The three coordinate pairs of the lip in the horizontal middle, 50-58, 51-57 and 52-56, which can be seen in Figure 3.4, were used to calculate the mouth opening distances in y direction. These coordinate pairs all consists of an upper (50, 51 and 52) coordinate and a lower (58, 57 and 56) coordinate. The distances in y direction, d_y, between these coordinate pairs were then calculated according to equation (3.4). There the distances in y direction is calculated by subtracting y_{lower}, which is the

y coordinate on the lower lip from the coordinate pair, from y_{upper}, which is the y coordinate on the upper lip from the coordinate pair.

$$d_y = y_{upper} - y_{lower} \tag{3.4}$$

These distances were then filtered by a simple moving average (SMA) filter, with the filter length $k = 6$ frames. This decision was made in order to filter over the same sample frequency as the feature would later transform to. The simple moving average for the distances $\bar{d_k}$ over the last k data points was calculated according to equation (3.5). For the edges of the data, where there are fewer than k data points to calculate the SMA over, data points were reflected so that the correct number of data points were obtained. The variable n is the position of the vector with the data points of the distance d_i.

$$\bar{d_n} = \frac{1}{k} \sum_{i=n-k+1}^{n} d_i \tag{3.5}$$

After the SMA filter were applied, samples were evenly picked out to match the sample frequency of four Hz.

Feature 2 (F2). Variance of mouth opening distances

The second feature was the sample variance of the mouth opening distances. The hypothesis was that if someone is talking, their mouth opening varies a lot but when they are not talking, the mouth is still and does not move. Therefore, the variance was used as a feature.

To calculate the variance, a simple moving average over the k previous data points were firstly calculated according to equation (3.5). The different value on the variable k was tested on a couple of videos and since the highest accuracy for these videos were obtained with $k = 6$, k was set to six. For the edges of the data, where there are fewer than k data points to calculate the SMA over, the number of available samples are used and k is therefore as high as it can be, but at most six. The SMA was calculated for both the distances d_x and the distances d_y from equations (3.3) and (3.4).

$$\bar{d_k} = \frac{1}{k} \sum_{i=n-k+1}^{n} d_i \tag{3.6}$$

After the SMA was calculated, the variances σ_k^2 were calculated over a sliding window of $k = 6$ frames according to equation (3.7).

$$\sigma_k^2 = \frac{1}{k} \sum_{i=n-k+1}^{n} (d_i - \bar{d_k})^2 \tag{3.7}$$

The SMA filter described in equation (3.5) was then applied at the data and finally the features were resampled to four Hz.

Feature 3 (F3). Difference of mouth opening distances

The third mouth opening feature was the difference of the mouth opening distances Δd between several frames. This feature was done by taking the difference between consecutive data points from the feature F1, as described in equation (3.8). Different values of k were tried during the project, but based on performance, we decided to use $k = 1$. For the first data point, the difference were set to zero.

$$\Delta d_i = d_i - d_{i-k} \tag{3.8}$$

Feature 4 (F4). Area of the mouth

The fourth feature was the area of the mouth. The centroid x_{cen} of the x coordinates and the centroid y_{cen} of the y coordinates were calculated according to equation (3.9). The variable m is the number of the coordinates, as shown in Figure 3.4. The centroid was calculated for each frame.

$$\begin{aligned} x_{cen} &= \frac{1}{m} \sum_{m=48}^{59} x_m \\ y_{cen} &= \frac{1}{m} \sum_{m=48}^{59} y_m \end{aligned} \tag{3.9}$$

After that, the area of the irregular polygon formed by the lip landmarks were calculated. Triangles A_m were created by two landmarks next to each other and the centroid. The area of each triangle was calculated as in equation (3.10).

$$A_m = \begin{cases} |\frac{1}{2}(x_m(y_{m+1} - y_{cen}) + x_{m+1}(y_{cen} - y_m) + x_{cen}(y_m - y_{m+1}))| & \text{if } m \neq 59 \\ |\frac{1}{2}(x_m(y_{48} - y_{cen}) + x_{48}(y_{cen} - y_m) + x_{cen}(y_m - y_{48}))| & \text{if } m = 59 \end{cases} \tag{3.10}$$

To calculate the area of the whole irregular polygon A_{tot} created by the coordinates around the lips, the small triangles were then summed up, as described by equation (3.11).

$$A_{tot} = \sum_{m=48}^{59} A_m \tag{3.11}$$

The area is then, just like the mouth opening distances features, filtered by a SMA filter described in equation (3.5) with the filter length six and resampled to four Hertz.

Feature 5 (F5). The variance of the mouth area

The fifth feature was the variance of the area. Just like the calculations of the second feature, the SMA was firstly calculated for the areas. This was done according to equation (3.12), where $k = 6$ was chosen.

$$\bar{A_k} = \frac{1}{k} \sum_{i=n-k+1}^{n} A_t o t_i \tag{3.12}$$

After the SMA was calculated, the variances σ_k^2 were calculated over a sliding window of $k = 6$ frames according to equation (3.13).

$$\sigma_k^2 = \frac{1}{k} \sum_{i=n-k+1}^{n} (A_i - \bar{A_k})^2 \tag{3.13}$$

After the variance is calculated, the feature is filtered with the same uniform filter as before, see equation (3.5), and then resampled to four Hertz.

Feature 6 (F6). The difference of the mouth area

The sixth feature is the difference between the area between difference frames. Exactly as the third feature, the difference is calculated between consecutive samples, here described by equation (3.14). For the first data point, the difference were set to zero.

$$\Delta A_i = A_i - A_{i-k} \tag{3.14}$$

3.5 Proposed SVM

SVM was used to classify the audio as speech or non-speech based on the video features. The different features described above were used separately in different models to compare the features with each other. The algorithms were trained and tested on each face in each video separately, since the monologues and dialogues are dissimilar as well as the actors behaved differently with various amounts of movements.

A new model is trained and tested for each face in each video. The stimuli contained 24 videos, which means that the algorithm was trained on $24 \times 3 = 72$ faces with their corresponding audio as ground truth for each feature. Since the stimuli is quite small, cross-validation was used to reduce the risk of overfitting. Cross-validation was made with only $k = 5$ folds, since we wanted each test set to contain both speech and non-speech for all video clips for all faces. Linear, rbf, sigmoid and polynomial kernel functions were used. The parameter C was tested on a scale from 1-6 for each face and the value with which gave the highest accuracy was chosen as parameter value. Grid search was also used to calculate the highest accuracy for the degrees 2-5 for the polynomial kernel. This was also done for each face in each video.

After all the features were tested separately, the best kernel functions with the best feature F1-F3 and the best feature F4-F6 were selected to train some new models. The new models were only trained on the videos that had received highest accu-

racies on the models for feature F1-F6. This was done to see if the features worked better together and returned higher accuracies as well as to study the performance of videos with high accuracy in more detail.

4

Result

The result from the SVM models are presented in different box plots. All the box plots contain one box which represents the proportion of the the most frequent class, speech or non-speech. Since the video is not 50% speech and 50% non-speech, it is important to compare the accuracy with these proportion. For example, the first face in the video achieves an accuracy of 85-86% which could be interpreted as a good result and a high accuracy. However, when compared to the fact that the person speaks in 85% of the video, it is clear that the accuracy is not high. A model for that face could classify all the data as speech and get an accuracy of 85%. When studying the result in the figures, it is therefore important to compare the accuracies with the proportion speech and non-speech for each speaker.

In Figure 4.1, the accuracies for the monologue and the accuracies for the dialogues were compared with the proportion of the most frequent class for the mono-

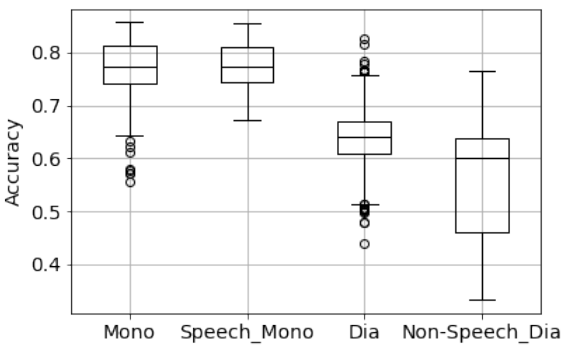

Figure 4.1 The accuracies for all the models trained on features F1-F6 for the monologues (Mono) and the dialogues (Dia), as well as the proportion of the most frequent class in the videos for the monologues (Speech_Mono) as well as the dialogues (Non-Speech_Dia).

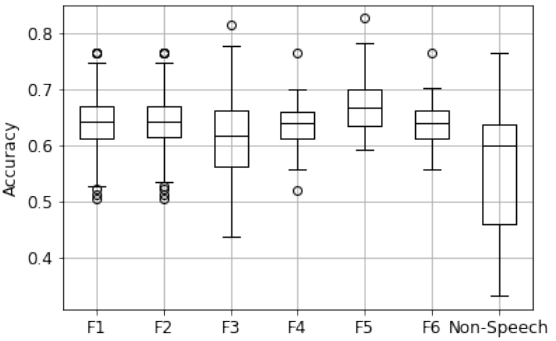

Figure 4.2 The accuracies for all the models trained on the dialogues for each features F1-F6, as well as the proportion of the most frequent class (Non-speech) for the dialogues.

logues, named "Speech_Mono" in the figure, respective the dialogues, named "Non-Speech_Dia" in the plot. In the monologues the relationship between the classes speech and non-speech are less balanced than for the dialogues. The most dominant class for the monologues are speech and the most dominant class for the dialogues are non-speech. The predicted classes for the monologues have about the same accuracy as classifying everything as speech. For some monologues the accuracy is even worse than classifying all data as speech. For the dialogues, the mean accuracy is a few percentage better than classifying everything as non-speech.

Since the dialogues had slightly better result than classifying all data as the dominant class, the result for the dialogues were studied in more detail. In Figure 4.2, the accuracies for the dialogues are divided between the different features F1-F6. The feature F5, which was the variance of the mouth area, has the highest mean accuracy. It is also F5 which has the highest accuracy for the first quartile. It is also notable that feature F3 has the longest whiskers among the features, meaning that F3 has the largest difference between the maximum and minimum accuracy. When studying in more detail, it was seen that F3 is better than F1-F3 for the rbf kernel but is worse for the polynomial kernel.

The four actors in the videos behave differently with uniquely moving patterns. Therefore, the accuracies for the four actors for the dialogues were compared to see if there were any differences. This can be seen in Figure 4.3. The mean for the third actor is slightly higher than for the other three actors. The other three actors have around the same mean accuracy.

The accuracies for the three different positions in the videos can be seen in Figure 4.4. This was interesting to investigate since the actors in the three positions are turned towards each other in different amounts. Since the second position always

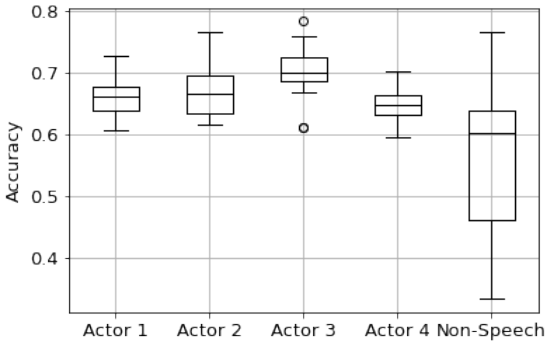

Figure 4.3 The accuracies for all the models trained on the dialogues for each actor, as well as the proportion of the most frequent class (Non-speech) for the dialogues.

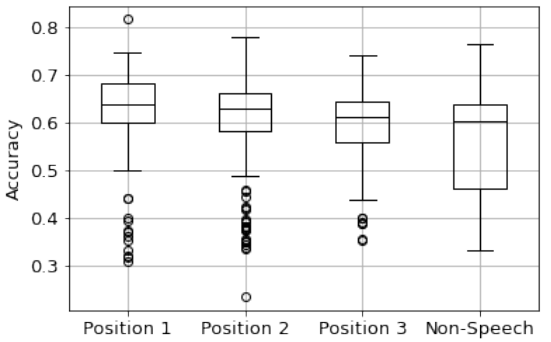

Figure 4.4 The accuracies for all the models trained on the dialogues for each video position, as well as the proportion of the most frequent class (Non-speech) for the dialogues.

has a person performing a dialogue, see Table 3.1, the second position contains more dialogue data than the other two positions. However, when studying the Figure 4.4, there is no significant difference in accuracy between the positions.

After studying the result, one can see that some features and some kernels perform better than others. The linear and the rbf kernel perform better than the sigmoid and polynomial kernel functions. For the linear kernel, the variance of the mouth opening distances (F2) is the best performing mouth opening feature, when

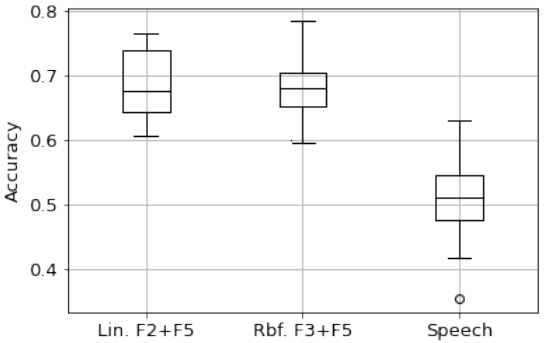

Figure 4.5 The accuracy for the fifteen faces with highest accuracy for the linear kernel trained on F2 and F5 and the rbf kernel trained on F3 and F5, as well as the proportion of the most frequent class (Speech) for the faces with the highest accuracy.

the data is not classifying all data as the dominant class. Of the area features, the variance (F5) was also the best feature for the linear kernel function. For the rbf kernel, the best features were the differences of the mouth opening distances (F3) and the variance of the area (F5). Therefore, new models were produced that use the linear kernel trained on both the variance of the mouth opening distances and variance of the area and new models using the rbf kernel trained on the differences of mouth opening area and the variances of the area. The model was only ran on faces that had a higher accuracy than the proportion of the most frequent class in the video. In total, fifteen faces were chosen. The two new models were then trained on this small dataset. The result can be seen in Figure 4.5.

For the dialogues used to train the models with the combinations of features, the speech was slightly more frequent than the non-speech class. Therefore, the proportion speech is plotted in Figure 4.5. The combination of features for the best faces has a higher accuracy than the proportion speech in these videos.

For the interested reader, the results for all the models for feature F1-F3 can be seen in Table A.1 and the results for feature F4-F6 in Table A.2 in Appendix.

5

Discussion

During the project's duration, there were many decisions made that affected the performance of the models. These choices need to be discussed and evaluated in order to get an understanding of the results we got. The discussion of the result and the decisions made is divided into speech detection, face detection and video features.

Speech detection

From the results there was an indication that there could be some issues with the stimuli and how the models classified the test data because in the monologues the model classifies almost all the data points as speech. This can be seen in the accuracy of the monologues and Speech_Mono in Figure 4.1. The reason why it does this can be because of the unbalanced data in the ground truth. Unbalanced data can lead to misleading results and it might be true in our case. In the monologues, the actors spoke almost the entire audio. This meant that the ground truth was not equally divided between speech and non-speech. This leads to the models classifying everything as speech in the monologues.

In order to get better results from the monologues some changes could have been tested. One solution could be to undersample and ignore some speech data to obtain a proportional relationship between speech and non-speech of 50 %. However, with such a low amount of non-speech in the monologues the total amount of data used for training after this procedure would be small and would therefore probably still lead to poor results. Since each actor performs a monologue between five and eight times, a possible solution to the problem with the small amount of data could be to merge the monologue data from each actors to four larger sequences. This could make the data more balanced without decreasing the amount of data too much.

Another solution is to oversample the non-speech data to gain a better relation between speech and non-speech. This solution could easily lead to an overfitted model. However, it could be worth trying generating artificial non-speech data from the already existing data, to eventually be able to create a better model for the monologues.

During the process of the project there was a discussion of what the the ground truth would represent and how detailed the threshold for speech activity should be. This discussion came down to the size of the window length when calculating the speech envelope using moving RMS. With a smaller window length, smaller details could be detected in the speech envelope such as the tracking of letters in the words spoken by the actors instead of just the tracking of speech of whole words or sentences. A window length of 5000 samples, which corresponds to 104 milliseconds, was ultimately chosen since a smaller window length was not needed to separate speech and non-speech. The conclusion was with other words, that the small details in when a letter is being pronounced, is too detailed for only deciding the speech activity.

Face Detection

When using the face detection algorithm there were some frames in the video that detected more than three faces. This was probably often caused by large head movements made by the actors. It was also caused sometimes by the actors turning their face away to a certain angle from the camera. The face detection algorithm thought then there were two faces instead of just one on one actor. One solution to this problem could be to remove frames which found more than three faces from the input. This also means that these frames also needed to be removed from the audio file. Another solution to this issue could be to set the missing landmark to the landmarks of the previous frame. This could be motivated by the small difference in coordinates in such small time windows. However, if more than three faces is detected over a longer time, the landmarks will be constant for a relative long time which could affected the stimuli and the possibility to gain a high accuracy. However, studying the videos this was not the case for most of the time. In most of the videos a small amount of data were lost due to this phenomena and the first solution were therefore used. For three of the videos, video 2, 11 and 24, the face detection algorithm does not work well and there is a great loss of data. A continuation of the project could therefore be to either remove the frames when no faces are detected from the dataset but also to change face detection algorithm and see if that improves the result.

Another problem the algorithm had with the videos in the stimuli was that when the actors turned their face from the the recorder the algorithm had a harder time finding a face. This only happened in some of the dialogues since the actors in the dialogues were not looking into the recorder but instead were facing each other. This issue is probably connected with the stimuli trying to recreate a realistic environment. Because in a realistic environment a person, who is trying to lip read, would not always be able to capture all the mouth movements. They will not always see the entire mouth area and if a person can not see the mouth movements then it is not fair to expect the algorithm to be able in the same situation to capture the face and its lip movements. However, this does not explain the poor results that were got from the monologues since they looked straight at the camera and the algorithm

Figure 5.1 A frame of an actor with black dots on the frame representing the coordinates of the landmarks created by the algorithm of where it thinks the mouth is supposed to be.

has almost no problems detecting landmarks on the actors faces. It is therefore reasonable to expect a lower accuracy in more realistic videos where the actors is not directed towards the camera.

Another issue discovered during the process of reviewing the landmarks was that the coordinates of the landmarks did not always align to where the actual mouth was on the frame. As an example see Figure 5.1 where the landmarks for the mouth was placed on the actors cheek. This meant that the algorithm could recognise that there was a face but the landmarks did not align with the face. However, further investigation of this problem, did show that even though the landmarks were placed on the cheek they still seem to correlate with the movements of the mouth. When the mouth was closed, the lip landmarks on the cheek also visualised a closed mouth and when the mouth was open, the landmarks showed an open mouth with approximately the same shape. When the actor is directed partly away from the camera, the face detection algorithm finds the face and aligns the landmarks in the centre of the bounding box containing the detected face. The problem with the landmarks on the cheek could therefore have occurred since the aligning was bad and not because of error in the lip landmark detection. This is however not certain and the problem with the landmarks on the cheeks is still a source of error. The features could still be used because the features use landmarks that are placed correctly in relation to each other. For example the third feature uses the difference between landmarks and these distances is still the same even though the landmarks are not placed on

the mouth.

There were also the issue when collecting landmarks from the face detection algorithm. The actors sometimes covered their face with their hands or looked closely at the printed image they were given so that they hid their face behind the paper. This led to data being lost from a number of frames but the number of lost frames were negligible compared to the total number of frames that was collected so its effect on the result was very small.

According to Figure 4.3 the accuracy depends on the actor and again in Figure 4.3 actor three has the best accuracy overall and reasons why this is has been discussed above. Some differences noticed when studying the videos when this actor is talking compared to the other actors are that she does not move her head nor cover her face and mouth as much as the other three actors. This probably made it possible to collect more data and made the data more accurate compared to her speech.

Video features

For the monologue, there was no correlation s between the video landmarks and the speech activity. This could have been because of the unbalanced data set, which is more discussed in the discussion subsection *Speech detection*. It could also be since the features did not work. A good continuation could therefore be to continue investigating features. An example could be to use landmarks from the whole face and to eventually find a correlation between facial features and speech activity.

Several parameter values were chosen to best fit the data. Examples of these parameters are the number of frames which the difference of mouth opening or area were calculated between in feature F3 and F6 as well as the length of the sliding window for the variance in feature F2 and F4. Different values of the parameters of the video features were firstly evaluated based on performance on the monologue in the first video. This was done since the actor performing the monologue is directed straight to the camera and the feature calculations therefore were more accurate. The features were also evaluated on the other faces in the first video as well as partly on the third face in the ninth video. The parameters were checked on all faces on the first video to study the values and their result of the dialogues. The ninth video at the third position was tested since this video gave better result. For most of the result, where the models not were better than classifying all the data as the most frequent class, it would probably not have made any difference. It would however be interesting to change the parameters to see if it would have made any difference for the videos with an accuracy better than random.

Actor number two was moving a lot in several of the videos but it is hard to see any effect of this in Figure 4.3. The difference between his moving and the other moving of the other actors is that he is sometimes moving more towards and backwards from the camera. When he is getting closer to the camera, the area of the mouth as well as the mouth opening distances, will get larger. To decrease this source of error, the features could have been normalised against for example the

size of the face.

From the results there were some interesting observations. Feature F3 in Figure 4.2 had the lowest accuracy and a large whiskers compared to the other features over all the models. Since feature F3 was based on the distances between landmarks, the lower accuracy could be a result of the actor speaking slower or faster. With faster speech, the lips move quicker and the frames sampled for four Hertz can miss situations between these four frames per second. There could be situations where the speaker has already opened and closed the mouth and it is therefore not detected by the feature. This motivation can also be applied on Feature 6 but we do not see the same accuracy or whiskers as Feature 3 in Figure 4.2. It seems more robust to have features that use the variance like feature F2 and F5 since they do not risk missing single variations of the mouth opening. The combination of theses two features can be seen in Figure 4.5 with a linear kernel and it showed a significant increase compared to proportion of speech.

The combination of a mouth opening feature and an area feature, which was used to train a SVM with a linear and rbf kernel, returned better result than the proportion of the most frequent class. They also gave higher result than the linear kernel trained on the second feature as well as the rbf trained on the feature F3 respective F5. Therefore, the features are complementing each other so that a higher accuracy is obtained.

6

Conclusion

Using the methods from this project, there was no correlation between the facial features and the speech activity for the monologues. The entire videos of the monologues were in most cases predicted as speech, probably due to the unbalanced data.

For the dialogues, the mean accuracy were a little higher than classifying everything as the most frequent class but the accuracy for the dialogues varied a lot between the videos. Overall, the variance of the mouth area (feature F5) was the feature with the highest mean accuracy for the dialogues. There was no difference between the position of the actor in the dialogues, however there was a small difference in mean accuracy for the actors. The third actor gave a better result in the dialogues. With other words, for some of the videos there were a correlation between the facial features and the speech activity.

For the videos with a significant correlation between facial features and speech activity, the mean accuracy for the linear kernel trained with feature F2 and F5 and the rbf kernel trained with feature F3 and F5 were between 65-70%. This can be compared to the mean amount of speech in these videos which was almost 51%.

7

Future work

There is a lot more which could be done in the continuation of the work with the dataset with the long term goal of improving hearing aids. A more complex facial landmarks detection algorithm could be used in order to find the face also in videos where the algorithm used in this project had problems. There are a lot of different deep learning networks specialised on lip reading purpose. The downside of these algorithms is often that they are computationally heavy. However, since the method tried in this master thesis did not return good result, another facial landmarks detection method should be tried.

Other features could also be tried in order to achieve better result. It could be beneficial to use landmarks from the whole face and not just around the lips. This could be important to do since facial expression is a large part of our way of communicating.

In this project, SVM was used as a classifier to classify the speech activity. Apart from SVM, a different classifier could also be tried in the project. Two of the most common classifiers used in previous research are SVM and a neural network using CNN. Therefore, it is reasonable to also try a deep learning classifier using CNN to classify the speech activity.

The different actors have been treated separately in this project. However, since the dialogues consists of two actors interacting, it could be a good idea to look at the interaction to better predict the speech activity. For example, it is likely that when one actor is speaking the other is silent and so on. Likewise, if one of the actors starts to speak, the other actor probably finished talking. Therefore, it could be interesting to look into the interactions of the actors to better predict the speech activity of the dialogues.

A

Appendix

A.1 Results of accuracies for features F1-F6

Table A.1 Accuracy for the models trained on the mouth opening features (F1-F3) with linear, rbf, sigmoid and polynomial kernel for each face. The proportion speech (sp.) and non-speech (non-sp.) for each person is also given in the table. "Vid." is short for video and "pos." is the position 1-3 of the speaker.

				Linear			Rbf			Sigmoid			Poly		
Vid.	Pos.	Sp.	Non-Sp.	F1	F2	F3	F1	F2	F3	F1	F2	F3	F1	F2	F3
1	1	0.85	0.15	0.85	0.85	0.85	0.85	0.85	0.86	0.85	0.85	0.84	0.85	0.85	0.86
1	2	0.40	0.60	0.67	0.67	0.60	0.63	0.63	0.64	0.60	0.60	0.53	0.68	0.68	0.64
1	3	0.39	0.61	0.61	0.61	0.61	0.61	0.61	0.61	0.61	0.61	0.56	0.62	0.62	0.61
2	1	0.40	0.60	0.67	0.67	0.60	0.64	0.64	0.67	0.57	0.57	0.53	0.68	0.68	0.63
2	2	0.33	0.67	0.67	0.67	0.67	0.67	0.67	0.71	0.67	0.67	0.58	0.70	0.70	0.58
2	3	0.84	0.16	0.84	0.84	0.84	0.84	0.84	0.84	0.83	0.83	0.80	0.84	0.84	0.84
3	1	0.36	0.64	0.67	0.67	0.64	0.64	0.64	0.68	0.63	0.63	0.64	0.68	0.68	0.65
3	2	0.35	0.65	0.65	0.65	0.65	0.67	0.67	0.72	0.65	0.65	0.58	0.66	0.66	0.69
3	3	0.84	0.16	0.84	0.84	0.84	0.84	0.84	0.84	0.84	0.84	0.81	0.85	0.85	0.84
4	1	0.78	0.22	0.78	0.78	0.78	0.78	0.78	0.78	0.78	0.68	0.68	0.79	0.79	0.79
4	2	0.49	0.51	0.60	0.60	0.58	0.58	0.58	0.64	0.51	0.51	0.53	0.57	0.57	0.53
4	3	0.39	0.62	0.65	0.65	0.61	0.63	0.63	0.68	0.61	0.61	0.51	0.65	0.65	0.65
5	1	0.69	0.31	0.69	0.69	0.69	0.69	0.69	0.69	0.69	0.69	0.57	0.71	0.71	0.69
5	2	0.44	0.56	0.64	0.64	0.58	0.63	0.63	0.71	0.56	0.56	0.54	0.64	0.64	0.58
5	3	0.39	0.61	0.61	0.61	0.61	0.61	0.61	0.68	0.61	0.61	0.51	0.61	0.61	0.65
6	1	0.70	0.30	0.70	0.70	0.70	0.73	0.73	0.70	0.69	0.69	0.58	0.73	0.73	0.71
6	2	0.40	0.60	0.72	0.72	0.60	0.70	0.70	0.72	0.60	0.60	0.59	.072	0.72	0.65
6	3	0.36	0.64	0.64	0.64	0.64	0.64	0.64	0.68	0.64	0.64	0.55	0.64	0.64	0.68
7	1	0.35	0.65	0.65	0.65	0.65	0.65	0.65	0.66	0.57	0.57	0.54	0.65	0.65	0.65
7	2	0.38	0.62	0.75	0.75	0.62	0.74	0.74	0.72	0.62	0.62	0.57	0.74	0.75	0.66
7	3	0.67	0.33	0.67	0.67	0.67	0.68	0.68	0.67	0.67	0.67	0.56	0.68	0.68	0.67
8	1	0.32	0.68	0.68	0.68	0.68	0.68	0.68	0.68	0.68	0.68	0.60	0.68	0.68	0.68
8	2	0.46	0.54	0.62	0.62	0.57	0.60	0.60	0.73	0.54	0.54	0.54	0.60	0.60	0.62
8	3	0.69	0.31	0.69	0.69	0.69	0.69	0.69	0.69	0.69	0.69	0.58	0.69	0.69	0.70
9	1	0.83	0.17	0.83	0.83	0.83	0.83	0.83	0.83	0.83	0.83	0.83	0.83	0.83	0.83
9	2	0.36	0.64	0.64	0.64	0.64	0.64	0.64	0.70	0.64	0.64	0.57	0.67	0.67	0.66
9	3	0.59	0.52	0.60	0.60	0.56	0.62	0.62	0.64	0.51	0.51	0.44	0.67	0.62	0.56
10	1	0.81	0.19	0.81	0.81	0.81	0.81	0.81	0.81	0.81	0.81	0.70	0.83	0.83	0.81
10	2	0.42	0.58	0.58	0.58	0.58	0.58	0.58	0.61	0.58	0.58	0.53	0.58	0.58	0.59
10	3	0.51	0.49	0.58	0.58	0.58	0.63	0.63	0.63	0.54	0.54	0.48	0.63	0.63	0.55
11	1	0.72	0.28	0.72	0.72	0.72	0.74	0.74	0.73	0.69	0.69	0.57	0.75	0.75	0.73
11	2	0.66	0.34	0.66	0.66	0.66	0.66	0.66	0.66	0.66	0.66	0.52	0.66	0.66	0.66
11	3	0.35	0.65	0.72	0.72	0.65	0.72	0.72	0.71	0.64	0.64	0.55	0.74	0.74	0.69
12	1	0.32	0.68	0.68	0.68	0.68	0.68	0.68	0.69	0.68	0.68	0.68	0.68	0.68	0.69
12	2	0.65	0.35	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.50	0.66	0.66	0.65
12	3	0.77	0.23	0.77	0.77	0.77	0.77	0.77	0.78	0.77	0.77	0.71	0.77	0.77	0.77

				Linear			Rbf			Sigmoid			Poly		
Vid.	Pos.	Sp.	Non-Sp.	F1	F2	F3	F1	F2	F3	F1	F2	F3	F1	F2	F3
13	1	0.39	0.61	0.61	0.61	0.61	0.61	0.61	0.61	0.59	0.59	0.58	0.61	0.61	0.61
13	2	0.61	0.39	0.61	0.61	0.61	0.63	0.63	0.61	0.61	0.63	0.54	0.64	0.64	0.61
13	3	0.75	0.25	0.75	0.75	0.75	0.76	0.76	0.77	0.75	0.75	0.63	0.78	0.78	0.77
14	1	0.31	0.69	0.70	0.70	0.69	0.70	0.70	0.70	0.69	0.69	0.67	0.70	0.70	0.70
14	2	0.62	0.28	0.62	0.62	0.62	0.62	0.62	0.62	0.62	0.62	0.51	0.62	0.62	0.62
14	3	0.75	0.25	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.64	0.75	0.75	0.75
15	1	0.81	0.19	0.81	0.81	0.81	0.81	0.81	0.82	0.81	0.81	0.75	0.81	0.81	0.82
15	2	0.38	0.62	0.63	0.63	0.62	0.61	0.75	0.75	0.62	0.62	0.51	0.67	0.67	0.65
15	3	0.56	0.44	0.62	0.62	0.56	0.63	0.63	0.66	0.56	0.56	0.50	0.64	0.64	0.56
16	1	0.53	0.47	0.64	0.64	0.56	0.65	0.65	0.67	0.53	0.53	0.50	0.65	0.65	0.57
16	2	0.24	0.76	0.76	0.76	0.76	0.76	0.76	0.77	0.76	0.76	0.68	0.76	0.76	0.78
16	3	0.81	0.19	0.81	0.81	0.81	0.81	0.81	0.81	0.81	0.71	0.71	0.81	0.81	0.81
17	1	0.81	0.19	0.81	0.81	0.81	0.81	0.81	0.81	0.81	0.73	0.83	0.83	0.83	0.81
17	2	0.35	0.65	0.65	0.65	0.65	0.65	0.65	0.69	0.56	0.56	0.57	0.65	0.65	0.66
17	3	0.54	0.46	0.60	0.60	0.54	0.62	0.62	0.61	0.54	0.54	0.48	0.61	0.61	0.54
18	1	0.67	0.33	0.67	0.67	0.67	0.67	0.67	0.82	0.67	0.67	0.57	0.67	0.67	0.68
18	2	0.38	0.62	0.62	0.62	0.62	0.62	0.62	0.64	0.62	0.62	0.56	0.62	0.62	0.65
18	3	0.75	0.25	0.75	0.75	0.75	0.76	0.76	0.75	0.75	0.75	0.62	0.77	0.77	0.75
19	1	0.63	0.37	0.63	0.63	0.63	0.63	0.63	0.63	0.63	0.63	0.52	0.63	0.63	0.63
19	2	0.37	0.63	0.63	0.63	0.63	0.64	0.64	0.64	0.63	0.63	0.54	0.63	0.63	0.53
19	3	0.73	0.27	0.73	0.73	0.73	0.74	0.74	0.75	0.73	0.73	0.61	0.76	0.76	0.74
20	1	0.56	0.44	0.64	0.64	0.56	0.62	0.62	0.56	0.56	0.56	0.51	0.65	0.65	0.56
20	2	0.42	0.58	0.66	0.66	0.58	0.65	0.65	0.72	0.59	0.59	0.54	0.69	0.69	0.62
20	3	0.77	0.23	0.77	0.77	0.77	0.77	0.77	0.77	0.77	0.77	0.79	0.79	0.79	0.78
21	1	0.78	0.22	0.78	0.78	0.78	0.78	0.78	0.78	0.78	0.78	0.76	0.78	0.78	0.78
21	2	0.42	0.58	0.66	0.66	0.58	0.70	0.70	0.70	0.58	0.58	0.55	0.70	0.70	0.63
21	3	0.60	0.40	0.67	0.67	0.60	0.69	0.69	0.68	0.60	0.60	0.52	0.68	0.68	0.60
22	1	0.80	0.20	0.80	0.80	0.80	0.82	0.82	0.82	0.80	0.80	0.75	0.82	0.82	0.80
22	2	0.38	0.62	0.67	0.67	0.62	0.68	0.68	0.70	0.62	0.62	0.57	0.70	0.70	0.64
22	3	0.60	0.40	0.71	0.71	0.60	0.71	0.71	0.70	0.60	0.60	0.52	0.71	0.71	0.60
23	1	0.63	0.37	0.73	0.73	0.63	0.75	0.75	0.64	0.63	0.63	0.53	0.74	0.74	0.64
23	2	0.35	0.65	0.73	0.73	0.65	0.73	0.73	0.75	0.65	0.65	0.60	0.75	0.75	0.68
23	3	0.80	0.20	0.80	0.80	0.80	0.80	0.80	0.82	0.80	0.80	0.72	0.80	0.80	0.82
24	1	0.77	0.23	0.77	0.77	0.77	0.77	0.77	0.77	0.77	0.77	0.70	0.77	0.77	0.77
24	2	0.54	0.46	0.66	0.66	0.54	0.60	0.60	0.66	0.54	0.54	0.54	0.60	0.60	0.59
24	3	0.48	0.52	0.64	0.64	0.57	0.64	0.64	0.57	0.52	0.52	0.50	0.67	0.67	0.56

Table A.2 Accuracy for the models trained on video features 4-6 (F4-F6) with rbf kernel for each face. The proportion speech (sp.) and non-speech (non-sp.) for each person is also given in the table. "Vid." is short for video and "pos." is the position 1-3 of the speaker.

				Rbf		
Vid.	Pos.	Sp.	Non-Sp.	F4	F5	F6
1	1	0.85	0.15	0.85	0.85	0.85
1	2	0.40	0.60	0.62	0.63	0.62
2	3	0.39	0.61	0.61	0.61	0.61
2	1	0.40	0.60	0.64	0.61	0.64
2	2	0.33	0.67	0.69	0.62	0.64
2	3	0.84	0.16	0.84	0.85	0.84
3	1	0.36	0.64	0.65	0.70	0.70
3	2	0.35	0.65	0.68	0.70	0.70
3	3	0.84	0.16	0.84	0.84	0.84
4	1	0.78	0.22	0.82	0.78	0.78
4	2	0.49	0.51	0.59	0.68	0.61
4	3	0.39	0.62	0.65	0.70	0.67
5	1	0.69	0.31	0.69	0.75	0.69
5	2	0.44	0.56	0.64	0.72	0.66

				Rbf		
Vid.	Pos.	Sp.	Non-Sp.	F4	F5	F6
5	3	0.39	0.61	0.61	0.64	0.64
6	1	0.70	0.30	0.71	0.70	0.70
6	2	0.40	0.60	0.68	0.75	0.70
6	3	0.36	0.64	0.64	0.67	0.66
7	1	0.35	0.65	0.65	0.65	0.65
7	2	0.38	0.62	0.70	0.69	0.65
7	3	0.67	0.33	0.68	0.67	0.67
8	1	0.32	0.68	0.68	0.68	0.68
8	2	0.46	0.54	0.56	0.71	0.65
8	3	0.69	0.31	0.69	0.70	0.69
9	1	0.83	0.17	0.83	0.83	0.83
9	2	0.36	0.64	0.66	0.69	0.65
9	3	0.59	0.52	0.60	0.63	0.57
10	1	0.81	0.19	0.81	0.82	0.82
10	2	0.42	0.58	0.58	0.60	0.59
10	3	0.51	0.49	0.52	0.63	0.57
11	1	0.72	0.28	0.72	0.72	0.72
11	2	0.66	0.34	0.66	0.66	0.66
11	3	0.35	0.65	0.65	0.65	0.67
12	1	0.32	0.68	0.68	0.68	0.68
12	2	0.65	0.35	0.65	0.65	0.65
12	3	0.77	0.23	0.77	0.77	0.77
13	1	0.39	0.61	0.63	0.62	0.61
13	2	0.61	0.39	0.62	0.61	0.61
13	3	0.75	0.25	0.75	0.75	0.75
14	1	0.31	0.69	0.69	0.69	0.70
14	2	0.62	0.28	0.62	0.66	0.62
14	3	0.75	0.25	0.75	0.75	0.75
15	1	0.81	0.19	0.81	0.81	0.81
15	2	0.38	0.62	0.66	0.70	0.65
15	3	0.56	0.44	0.56	0.70	0.61
16	1	0.53	0.47	0.60	0.70	0.61
16	2	0.24	0.76	0.76	0.76	0.76
16	3	0.81	0.19	0.81	0.81	0.81
17	1	0.81	0.19	0.81	0.81	0.81
17	2	0.35	0.65	0.65	0.65	0.65
17	3	0.54	0.46	0.56	0.67	0.61
18	1	0.67	0.33	0.67	0.67	0.61
18	2	0.38	0.62	0.63	0.63	0.64
18	3	0.75	0.25	0.75	0.75	0.75
19	1	0.63	0.37	0.63	0.67	0.63
19	2	0.37	0.63	0.67	0.63	0.64
19	3	0.73	0.27	0.74	0.75	0.75
20	1	0.56	0.44	0.61	0.64	0.56
20	2	0.42	0.58	0.66	0.76	0.70
20	3	0.77	0.23	0.77	0.77	0.77
21	1	0.78	0.22	0.78	0.78	0.78
21	2	0.42	0.58	0.61	0.70	0.64
21	3	0.60	0.40	0.61	0.71	0.64
22	1	0.80	0.20	0.81	0.80	0.80
22	2	0.38	0.62	0.66	0.73	0.63
22	3	0.60	0.40	0.61	0.73	0.60
23	1	0.63	0.37	0.66	0.70	0.63
23	2	0.35	0.65	0.69	0.78	0.70
23	3	0.80	0.20	0.80	0.80	0.80
24	1	0.77	0.23	0.77	0.77	0.77
24	2	0.54	0.46	0.58	0.59	0.58
24	3	0.48	0.52	0.63	0.65	0.61

Bibliography

Assael, Y. M., B. Shillingford, S. Whiteson, and N. de Freitas (2016). "Lipnet: sentence-level lipreading". *CoRR*. arXiv: 1611.01599. URL: http://arxiv.org/abs/1611.01599.

Basbøll, H. (2005). *The Phonology of Danish*. The Phonology of the World's Languages. OUP Oxford. ISBN: 9780191519680. URL: https://books.google.se/books?id=zwdREAAAQBAJ.

Bilert, S. P. (2020). *Decoding Attention in Real-world Listening*. MA thesis. Technical University of Denmark, Kongens Lyngby.

Bronkhorst, A. (2000). "The cocktail party phenomenon: a review of research on speech intelligibility in multiple-talker conditions". *Acta Acustica united with Acustica* **86**, pp. 117–128.

Caetano, M. and X. Rodet (2011). "Improved estimation of the amplitude envelope of time-domain signals using true envelope cepstral smoothing". In: *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4244–4247. DOI: 10.1109/ICASSP.2011.5947290.

Chen, T. (2001). "Audiovisual speech processing". *IEEE Signal Processing Magazine* **18**:1, pp. 9–21. DOI: 10.1109/79.911195.

Cherry, E. C. (1953). "Some experiments on the recognition of speech, with one and with two ears". *The Journal of the acoustical society of America* **25**:5, pp. 975–979.

Chung, J. S., A. Senior, O. Vinyals, and A. Zisserman (2017). "Lip reading sentences in the wild". In: *2017 IEEE conference on computer vision and pattern recognition (CVPR)*. IEEE, pp. 3444–3453.

Fernandez-Lopez, A. and F. M. Sukno (2018). "Survey on automatic lip-reading in the era of deep learning". *Image and Vision Computing* **78**, pp. 53–72. DOI: https://doi.org/10.1016/j.imavis.2018.07.002. URL: https://www.sciencedirect.com/science/article/pii/S0262885618301276.

Géron, A. (2019). In: *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*. Vol. 2. O'Reilly Media, Sebastopol, CA. ISBN: 978-1-492-03264-9.

Giannakopoulos, T. (2009). *Silence removal in speech signals*. URL: https://www.mathworks.com/matlabcentral/fileexchange/28826-silence-removal-in-speech-signals. accessed: 05.05.2022).

Giannakopoulos, T. and A. Pikrakis (2014). "Chapter 4 - audio features". In: Giannakopoulos, T. et al. (Eds.). *Introduction to Audio Analysis*. Academic Press, Oxford, pp. 59–103. ISBN: 978-0-08-099388-1. DOI: https://doi.org/10.1016/B978-0-08-099388-1.00004-2. URL: https://www.sciencedirect.com/science/article/pii/B9780080993881000042.

Goodfellow, I., Y. Bengio, and A. Courville (2016). *Deep Learning*. http://www.deeplearningbook.org. MIT Press.

Grønnum, N. (2005). *Fonetik og fonologi: almen og dansk*. Dansk. 3. Akademisk Forlag. ISBN: 87-500-3865-6.

Jalil, M., F. A. Butt, and A. Malik (2013). "Short-time energy, magnitude, zero crossing rate and autocorrelation measurement for discriminating voiced and unvoiced segments of speech signals". In: *2013 The International Conference on Technological Advances in Electrical, Electronics and Computer Engineering (TAEECE)*, pp. 208–212. DOI: 10.1109/TAEECE.2013.6557272.

King, D. E. (2009). "Dlib-ml: a machine learning toolkit". *Journal of Machine Learning Research* **10**, pp. 1755–1758.

King, D. E. (2015). *Max-margin object detection*. DOI: 10.48550/ARXIV.1502.00046. URL: https://arxiv.org/abs/1502.00046.

Krishnamurthi, R., D. Gopinathan, and A. Kumar (2022). "Chapter 10 - using wavelet transformation for acoustic signal processing in heavy vehicle detection and classification". In: Krishnamurthi, R. et al. (Eds.). *Autonomous and Connected Heavy Vehicle Technology*. Intelligent Data-Centric Systems. Academic Press, pp. 199–209. ISBN: 978-0-323-90592-3. DOI: https://doi.org/10.1016/B978-0-323-90592-3.00011-2. URL: https://www.sciencedirect.com/science/article/pii/B9780323905923000112.

Lee, A. K. C., E. Larson, and C. W. Miller (1, 2018). "Effects of hearing loss on maintaining and switching attention". *Acta Acustica united with Acustica* **104**:5, pp. 787–791. ISSN: 1610-1928. DOI: doi:10.3813/AAA.919224. URL: https://www.ingentaconnect.com/content/dav/aaua/2018/00000104/00000005/art00016.

Marinato, G. and D. Baldauf (2019). "Object-based attention in complex, naturalistic auditory streams". en. *Sci Rep* **9**:1, p. 2854.

Marius, H. (2020). *Multiclass classification with support vector machines (svm), dual problem and kernel functions*. [Accessed 2022-04-26]. URL: https://towardsdatascience.com/multiclass-classification-with-support-vector-machines-svm-kernel-trick-kernel-functions-f9d5377d6f02.

McDermott, J. H. (2009). "The cocktail party problem". *Current Biology* **19**:22, R1024–R1027. ISSN: 0960-9822. DOI: https://doi.org/10.1016/j.cub.2009.09.005. URL: https://www.sciencedirect.com/science/article/pii/S0960982209016807.

McGurk, H. and J. MacDonald (1976). "Hearing lips and seeing voices". *Nature* 264, pp. 746–746. DOI: 10.1038/264746a0.

Mitchell, T. M. (1997). In: *Machine Learning*. New York. ISBN: 9780070428072.

National Deaf Children's Society (n.d.). *Speech reading for deaf children: more information about lipreading*. [Accessed 2022-03-23]. URL: https://www.ndcs.org.uk/information-and-support/language-and-communication/spoken-language/supporting-speaking-and-listening/lip-reading/.

Noble, W. S. (2006). "What is a support vector machine?" *Nature Biotechnology* **24**, pp. 1565–1567. DOI: 10.1038/nbt1206-1565.

Oppenheim, A. V. (1999). *Discrete-time signal processing / Alan V. Oppenheim, Ronald W. Schafer, with John R. Buck.* eng. Second edition. Prentice-Hall signal processing series. Prentice Hall, Upper Saddle River, N.J. ISBN: 0137549202.

Palacio-Niño, J.-O. and F. Berzal (2019). *Evaluation metrics for unsupervised learning algorithms*. DOI: 10.48550/ARXIV.1905.05667. URL: https://arxiv.org/abs/1905.05667.

Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay (2011). "Scikit-learn: machine learning in Python". *Journal of Machine Learning Research* **12**, pp. 2825–2830.

Reiss, L. A. J. and M. R. Molis (2021). "An alternative explanation for difficulties with speech in background talkers: abnormal fusion of vowels across fundamental frequency and ears". en. *J Assoc Res Otolaryngol* **22**:4, pp. 443–461.

Rocchesso, D. (2003). *Introduction to Sound Processing*. Ass. Culturale Mondo Estremo. ISBN: 9788890112614. URL: https://books.google.se/books?id=F76OxogOaC0C.

Rosebrock, A. (2021). *Face detection with dlib (hog and cnn)*. [Accessed 2022-03-21]. URL: https://pyimagesearch.com/2021/04/19/face-detection-with-dlib-hog-and-cnn/.

Russ, J. C. (2008). *The Image Processing Handbook*. 5. CRC Press. ISBN: 0-203-88109-5.

Sairamya, N., L. Susmitha, S. Thomas George, and M. Subathra (2019). "Chapter 12 - hybrid approach for classification of electroencephalographic signals using time–frequency images with wavelets and texture features". In: Hemanth, D. J. et al. (Eds.). *Intelligent Data Analysis for Biomedical Applications*. Intelligent Data-Centric Systems. Academic Press, pp. 253–273. ISBN: 978-0-12-815553-0. DOI: https://doi.org/10.1016/B978-0-12-815553-0.00013-6. URL: https://www.sciencedirect.com/science/article/pii/B9780128155530000136.

Sharma, S., K. Shanmugasundaram, and S. K. Ramasamy (2016). "Farec — cnn based efficient face recognition technique using dlib". In: *2016 International Conference on Advanced Communication Control and Computing Technologies (ICACCCT)*, pp. 192–195. DOI: 10.1109/ICACCCT.2016.7831628.

Shinn-Cunningham, B. G. and V. Best (2008). "Selective attention in normal and impaired hearing". *Trends in Amplification* **12**:4. PMID: 18974202, pp. 283–299. DOI: 10.1177/1084713808325306. eprint: https://doi.org/10.1177/1084713808325306. URL: https://doi.org/10.1177/1084713808325306.

Suthaharan, S. (2016). "Support vector machine". In: *Machine Learning Models and Algorithms for Big Data Classification: Thinking with Examples for Effective Learning*. Vol. 36. Springer US, Boston, MA, pp. 207–235. ISBN: 978-1-4899-7641-3. DOI: 10.1007/978-1-4899-7641-3_9. URL: https://doi.org/10.1007/978-1-4899-7641-3_9.

Szeliski, R. (2021). *Computer Vision: Algorithms and Applications*. 2. Springer. URL: https://szeliski.org/Book/.

The MathWorks, I. (2020). *detectSpeech*. Natick, Massachusetts, United State. URL: https://se.mathworks.com/help/audio/ref/detectspeech.html.

Tris Atmaja, B. (2019). "The mechanism on how auditory system solves the cocktail party problem".

World report on hearing: executive summary (2021). World Health Organization, Geneva. ISBN: 978-92-4-00257-0.

Wu, Y. and Q. Ji (2018). "Facial landmark detection: a literature survey". *International Journal of Computer Vision* **127**:2, pp. 115–142. DOI: 10.1007/s11263-018-1097-z. URL: https://doi.org/10.1007%5C%2Fs11263-018-1097-z.

Zhou, Z.-H. (2021). In: *Machine Learning*. Singapore, pp. 25–31, 129–148. ISBN: 978-981-15-1967-3. DOI: 10.1007/978-981-15-1967-3.

Lund University
Department of Automatic Control
Box 118
SE-221 00 Lund Sweden

Document name
MASTER'S THESIS

Date of issue
June 2022

Document Number
TFRT-6171

Author(s)
Viktor Andersson
Nelly Ostréus

Supervisor
Emina Alickovic, Oticon A/S
Martin Skoglund, Oticon A/S
Johannes Zaar, Oticon A/S
Bo Bernhardsson, Dept. of Automatic Control, Lund University, Sweden
Kristian Soltesz, Dept. of Automatic Control, Lund University, Sweden (examiner)

Title and subtitle
Speech activity detection in videos

Abstract

Speech is an important way of communication all over the world. The speech information is encoded both aural and visual. More than 1.5 billion people have hearing loss and for those the visual information is even more important than for people with normal hearing. Lip reading is therefore an important research topic.

In this master thesis, machine learning algorithms were used to identify speech activity in realistic video with monologues and dialogues. Each video contained three persons speaking: one performing a monologue and two performing a dialogue. Support vector machines for linear, radial basis function, sigmoid and polynomial kernels were used to classify the audio as either speech or non-speech based on faces from realistic videos. A speech envelope was calculated and resampled to four Hertz. Based on a threshold of the envelope, the ground truth was created and each audio data point was selected to be either speech or non-speech. Convolutional neural networks using max-margin object detection were used to extract facial landmarks from the videos. Six different video features were calculated and used: the mouth opening distances, the variance of the mouth opening distances and the difference of mouth opening distances between several frames, the mouth area, the variance of the area and the difference of area between several frames.

The mean accuracy for the speech activity in the monologues were low. This was probably due to the unbalanced data in the monologues, since most data in the ground truth were classified as speech. For the dialogues, the accuracy were slightly higher than classifying everything as the most frequent class. The variance of the mouth area was the best performing feature. The performance varies between the videos and combining the best mouth opening distances feature with the best mouth area feature for the two best kernels, increased the accuracy for the best performing videos.

Keywords

Classification system and/or index terms (if any)

Supplementary bibliographical information

ISSN and key title
0280-5316

ISBN

Language
English

Number of pages
1-56

Recipient's notes

Security classification

http://www.control.lth.se/publications/