University of West Bohemia
Faculty of Applied Sciences

Department of Cybernetics

Automated Lipreading with LipsID Features

DISSERTATION THESIS

submitted in partial fulfilment of the requirements
for the degree of
Doctor of Philosophy in the field of
Cybernetics

Ing. Miroslav Hlaváč

Advisor: Doc. Ing. Miloš Železný, Ph.D.

Pilsen, 2019

Západočeská Univerzita v Plzni
Fakulta Aplikovaných Věd

Katedra Kybernetiky

Automatické odezírání ze rtů pomocí příznaků LipsID

DISERTAČNÍ PRÁCE

k získání akdemického titulu doktor
v oboru **Kybernetika**

Ing. Miroslav Hlaváč

Školitel: Doc. Ing. Miloš Železný, Ph.D.

Plzeň, 2019

Acknowledgments

I would like to sincerely thank my supervisor Doc. Ing. Miloš Železný, Ph.D. for his guidance and support during my Ph.D. study. I would also like to express my gratitude to my advisor Ing. Marek Hrúz. Ph.D. for bringing new ideas, insights, and encouragements.

My sincere thanks goes to my colleagues for cooperation in research, writing papers together, and for creating a good atmosphere in our office. In particular, I am thankful to Ing. Ivan Gruber for stimulating discussions over new ideas.

Last but not least, I would like to thank my family for supporting me in the long years of my Ph.D. study.

Abstract

The aim of this is thesis is to create new visual features for the automatic lipreading systems. State-of-the-art methods (mainly machine learning methods) are not using any form of adaptation for a specific speaker during their training. There arises a question on how to adapt this method for the purposes of visual speech recognition and how to implement it into the current lipreading methods. Using the analysis of state-of-the-art methods for visual speech feature extraction we propose a new set of features LipsID and the method for extracting them. We show that by adapting the current system (based on neural networks) by the proposed LipsID features a higher recognition rate of the speech can be achieved. The significance of this work is in showing the importance of features based on the speaker identity for the task of lipreading.

Keywords

lipreading, machine learning, neural networks, visual features, speech recognition

Abstrakt

Cílem této práce je vytvoření nových vizuálních příznaků pro systémy automatického odezírání ze rtů. Metody současného stavu poznání (především metody strojového učení) nevyužívají při svém trénování možnosti adaptace na konkrétního řečníka. Vyvstává tedy otázka, jak tuto metodu adaptace přizpůsobit pro oblast rozpoznávání vizuální řeči a jak ji implementovat do současných algoritmů pro strojové odezírání ze rtů. Pomocí analýzy současného stavu poznání v oblasti vizuálních příznaků pro rozpoznávání řeči navrhujeme novou parametrizaci LipsID a metodu pro její získání. Ukazujeme, že adaptací současných systému (založených na neuronových sítích) navrženou metodou LipsID se dá dosáhnout vyšší přesnosti rozpoznávání vizuální řeči. Význam této práce spočívá v prokázání významnosti příznaků založených na identitě řečníka pro automatické metody odezírání ze rtů.

Klíčová slova

odezírání ze rtů, strojové učení, neuronové sítě, vizuální příznaky, rozpoznávání řeči

Аннотация

Целью данной работы является создание новых визуальных признаков для систем автоматического чтения речи по губам диктора. Современные модели и методы (прежде всего, машинного обучения) не используют в процессе своего обучения возможности адаптации к определённому диктору. Поэтому возникает вопрос, как этот метод адаптировать к процессу распознавания визуальной речи и как его включить в современные системы машинного чтения речи по губам. В результате анализа современных методов визуальной параметризации речи для систем распознавания мы предлагаем новый набор признаков LipsID и метод их вычисления. Мы показываем, что используя современные модели, основанных на нейронных сетях, в предлагаемом методе LipsID возможно достигнуть более высокой точности распознавания визуальной речи. Значимость этой работы заключается в демонстрации важности признаков, основанных на идентификации говорящего, для задачи автоматического чтения речи по губам.

Ключевые слова

чтение речи по губам, машинное обучение, нейронные сети, визуальные признаки, распознавание речи

Declaration

I hereby declare that this thesis is my own work and to my best knowledge it does not contain any previously published materials except for the ones acknowledged in the text.

. . . . . .

Ing. Miroslav Hlaváč

Contents

List of Tables

List of Figures

List of Abbreviations

AAM	Active Appearance Model
ASR	Automatic Speech Recognition
CER	Character Error Rate
CLR	Cascade of Linear Regressors
CNN	Convolutional Neural Network
CTC	Connectionist Temporal Classification
DCT	Discrete Cosine Transform
DNN	Deep Neural Network
HMM	Hidden Markov Model
HOG	Histogram of Oriented Gradients
LDA	Linear Discriminative Analysis
LRS	Lip Reading Sentences
LSTM	Long Short Term Memory
PCA	Principal Component Analysis
ROI	Region of Interest
Seq2Seq	Sequence to Sequence
SGD	Stochastic Gradient Descent
VGG	Visual Geometry Group
WER	Word Error Rate
WLAS	Watch Listen Attend and Spell

Part I

Introduction

Chapter 1

Introduction

The current state-of-the-art of Automatic speech recognition(ASR) can be divided into two different approaches based on physical attributes of speech. First approach is focused on the sound part of the speech. The other approach is focused on the visual part. This work is focused on speech recognition using visual information, which is called lipreading. The main idea of visual speech recognition comes from the experience with humans that are able to recognise spoken words from the observation of a speaker's face without or with limited access to the sound part of the voice. This ability is usually developed by people with some kind of hearing impairment. The important thing to mention here is that the knowledge of a language structure helps the lip reader to recognise the meaning in a broader context. The ability to recognise what is spoken without hearing the sound leads to an important fact. The movements of a speaker's face contain at least some information about the speech. Therefore, the whole area of automatic lipreading is established and a lot of new methods are being developed at the time of writing this work. Visual speech recognition can be used in many applications. For example surveillance, silent dictation, and to help people with speech impairments.

This thesis is divided into the following parts. The introduction continues with broader description of the problem of automated lipreading followed by the list of goals of this dissertation thesis. The next part is composed of methods for visual features extraction and for automated lipreading followed by a description of the state-of-the-art in these fields. The third part contains my own contribution to the field of automated lipreading. The LipsID features are proposed, discussed, and evaluated followed by their implementation to the state-of-the-art lipreading methods. The last part is conclusion where the results and contributions of this work are evaluated.

Chapter 2

Lipreading

This chapter introduces the human ability of lipreading from the historical point of view followed by the definition of the automated lip reading problem.

2.1 Human lipreading

The first documented cases of human developing methods how to teach deaf people to communicate with other people are from 16th century AD. The first known teacher of the deaf was the Benedictine monk Dom Pedro Ponce from Spain. The first actual lipreading school was established in Leipzig in 1787 by Samuel Heinicke. The first conference on the topic of lipreading was held in 1894 in USA [1].

There are different method described in the literature. They include Jena, Bruhn, Kinzie, Nitchie, and Muller-Walle methods. The Nitchie method was published in 1912. It was based on analytical approach at first but later on it was switched to synthetic approach to analyse the speech as a whole. Bruhn method is advocating focus on the movement of the lips. Muller-Walle method incorporates the ideas from Bruhn method but instead of focusing on movement of lips it is focusing on the position of the vocal organs. Kinzie method published in 1931 introduced division of the lipreading process according to the difficulty. These focus on the lips area of the speaker is common in all the above mentioned methods [1].

Modern time visual speech analysis provided interesting data for human ability to lipread. Potamianos at al. in [2] conducted experiments with humans aimed at speech recognition based on four videos with two male and two female speakers. The

participants could see each video three times and they knew that the speakers only say number from one to nine. The results shows average word recognition rate of 53%. There were differences between recognition rate based on the speaker sex. Female speaker videos had significantly better recognition rate. However, this experiment could not lead to strong conclusions since it was done on a small set of speakers.

2.2 Automated lipreading

The problem of automated lip reading can be divided into three consecutive tasks: (1) region of interest (ROI) detection; (2) visual speech features extraction; (3) speech recognition by processing of visual features. The process of selecting the ROI is usually done by general face detectors (Viola & Jones [3], HoG face detector [4]). In some works the whole face is used for visual speech recognition, in other works only the lower half of the face is used. According to previous research [5] the region between nose and chin is the most important for visual speech recognition. Because the detection and tracking of mouth region alone is non-trivial task, the face is detected as whole and the ROI around the mouth is extracted from the detected face image [6]. This work will focus on the second and third tasks of visual speech features extraction and visual speech recognition. Some of the most recent (mainly those based on neural networks) methods solve both tasks of features extraction and consecutive speech recognition.

2.2.1 Visual Features Extraction

Visual speech features can be interpreted as physical properties of a speaker. The speaker moves his face muscles during the production of speech, which can be captured by video recording devices. The resulting video provides a recording of the movements of the speaker's mouth and the surrounding regions. The position of lips, visibility of tongue, and teeth are all important information in the process of visual speech recognition. Visual features can be divided by different criteria: (1) static and dynamic features; (2) geometric and texture features; (3) features based on visibility of teeth and tongue; etc. The state-of-the-art methods are usually focusing on geometric and texture detection. The geometry of the face and mouth is represented by landmarks or key points, which represent the shape. The texture represents the visual properties of the recorded image (pixel intensity).

2.2.2 Extracted Features Processing

The task of speech recognition is usually implemented as Speech to Text (STT). This means that the input to the system is a recording of a spoken word (video, audio, or combination of both) and the output is in a form of transcribed text. The transcription is handled differently by different methods. For example, the recent work [7] implements per character transcription system, on the other hand, some methods [8, 9] use some form of classification into words. Audio speech recognition is currently providing better results and is often used in the process of training visual speech neural networks as additional input [7]. The classification in neural networks is done by the softmax function. Softmax outputs probabilities of classes, which sum to one. The other methods also use Support Vector Machine (SVM) classifier [10].

Figure 1: Automated lip reading pipeline example.

This output of characters or words is not precise enough by itself. The following step in the state-of-the-art systems is implementation of some type of language model. This can be done in few different ways. One way is to implement a Conectionist Temporal Classification(CTC) that will account for all the different ways a single word can be represented (more on that in the Methodology Part II). The second way is to implement a beam search decoder [11] which can also fallow the CTC.

Chapter 3

Dissertation Goals

This chapter represents the original goals of the dissertation thesis as they were set by the assignment. After evaluating state-of-the art methods for visual speech recognition, there is still enough space for improvement. The current aim of our research is focused on neural network development, which is currently the most viable solution in this field of research. We propose three steps for the future research based on utilisation of neural networks to produce deep features and utilise them in end-to-end systems for lipreading like LipNet network (6.2.5).

3.1 Visual Speech Features Representation

The problem of representing visual features is still open. Currently, the WLAS network(6.2.6) uses pre-trained VGG networks(6.1.5) for evaluating and extracting features from video sequences. Other methods as mentioned above use key-points, AAM features, DCT features etc.

The aim of this goal is to analyse the state-of-the-art visual speech features which can be used for the task of lipreading. The output should be a review of features and how they work.

3.2 New feature extraction method development

Development of a new set of visual features. Analysis and evaluation of the proposed visual features and their viability for the purpose of training a neural network for visual/audio-visual speech recognition. This is the main goal of the dissertation.

The aim of this goal is development of a method for extracting visual features suitable for training of visual speech recognition system as an additional input. These features should be extracted from the data used for the training of the above mentioned system.

3.3 DNN Based Visual Speech Recognition

Due to the unpredictability of neural network training process and its dependence on thousands of variables, this step is very dependent on the proposed features in the previous step and the results will be an extension of the previous goal.

The aim of this goal is to prove the viability of the new feature set by implementing it into a state-of-the-art system for visual speech recognition. The development of the system is not part of this work.

Part II

Methodology

Introduction

This part of the thesis is composed of methods and approaches used for the purpose of my research. They are followed by a review of state-of-the-art methods in the fields of feature extraction and lipreading. The first chapter describes methods based on statistical models. The second chapter describes Deep Neural Networks which are currently heavily used in the field of lipreading as they vastly improved recognition rate over previously used methods. The third chapter describes state-of-the-art methods for feature extraction and visual speech recognition. The last chapter is a review of the datasets used in tasks of feature extraction and lipreading.

Chapter 4

Statistical Models

This chapter is composed of methods based on statistical analysis of the shape and texture of face data. The methods are usually based on Active Shape Model (ASM) and Active Appearance Model (AAM) [12]. Both models use statistical approaches to create a model from the set of training data. ASM uses only shape information. AAM is built on the ASM and it adds information about texture to the model. Both models rely on dimension reduction by Principal Component analysis (PCA) and they can be combined into one model to reduce the model dimension even further. The basic units, landmarks (also called key-points), for representing shapes are described, along with the process of their selection. A review of state-of-the-art methods for feature extraction based on statistical models is included at the end of this chapter.

4.1 Statistical Models of Shape

Statistical models of shape are used to represent objects in images. The model is created by applying statistical analysis methods to a set of points. The aim of this method is to both analyse and to synthesise a new shape similar to those in a training set.

4.1.1 Landmarks

A landmark is a significant point on the boundary of an object, that is present in most of the training images. It can represent a corner, end of a line, or a significant

change in an object boundary. This set of significant points is often supplemented by points along the object boundary, that equally fill the space between well-defined ones. The shape of an object is then reconstructed by suitably connecting the landmarks. The coordinates are usually two or three-dimensional.

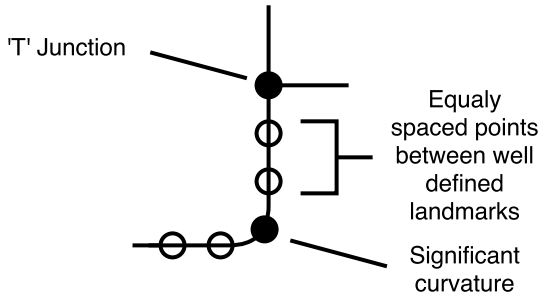


Figure 2: Well defined landmarks and points between them.

The simplest method of creating a training set is a for a human expert to manually choose landmarks in each of a series of images. This process is very time-consuming, and automatic and semi-automatic methods are developed to aid the human annotator.

Annotated shape with n selected landmarks in d dimensions is represented by nd element vector. This vector is constructed by concatenating the coordinates of each individual point in given image.

$$\mathbf{x} = (x_1, x_2, ..., x_n, y_1, y_2, ..., y_n)^T \tag{4.1}$$

The model representing the points dependencies is obtained by applying the PCA to the points coordinates. Before the statistical analysis can be performed it is important that shapes represented by these vectors are in the same coordinate frame. The training set needs to be aligned to remove variation in global position.

4.1.2 Aligning the Training Set

The most popular approach to aligning shapes into a common coordinate frame is Procrustes Analysis [13]. Each shape is aligned so that the sum of distances of each shape to the mean shape is minimised.

$$D = \sum |\mathbf{x}_i - \bar{\mathbf{x}}|^2 \tag{4.2}$$

Constraints have to be placed on the alignment of the mean to ensure good definition (for example, orientation, centring, and scaling). The algorithm can be represented by simple iterative approach[12]:

1. Translate each example so that its centre of gravity is at the origin.
2. Choose one example as an initial estimate of the shape mean shape and scale so that $|\bar{\mathbf{x}}| = 1$.
3. Record the first estimate as $\bar{\mathbf{x}}_0$ to define the default reference shape.
4. Align all the shapes with the current estimate of the mean shape.
5. Re-estimate mean from aligned shapes.
6. Apply constraints on the current estimate of the mean by aligning it with $\bar{\mathbf{x}}_0$ and scaling so that $|\bar{\mathbf{x}}| = 1$.
7. Check if the mean changed significantly from the previous iteration. If not, return to **4.**

The operation used to align the set will affect the final distribution. A common approach is to centre each shape on the origin, scale each to $|\mathbf{x}| = 1$ and then choose the orientation for each which minimises D.

4.1.3 Modelling the Shape Variance

Principal component analysis is used to convert an observation of possibly correlated variables to a set of linearly uncorrelated variables. It is often used to reduce the dimension of the feature space. The transformation is defined so that the first principal component would be the one with the largest variance. The resulting vectors are an uncorrelated orthogonal basis set [14].

The algorithm works as follows.

- Compute the mean of the data $\overline{\mathbf{x}}$:

$$\overline{\mathbf{x}} = \frac{1}{s}\sum_{i=1}^{s}\mathbf{x}_i, \tag{4.3}$$

- Compute the covariance matrix $\mathbf{S}$ of the data:

$$\mathbf{S} = \frac{1}{s-1} \sum_{i=1}^{s} (\mathbf{x_i} - \overline{\mathbf{x}}) (\mathbf{x_i} - \overline{\mathbf{x}})^T, \tag{4.4}$$

- Compute the eigenvectors ϕ_i and corresponding eigenvalues λ_i of $\mathbf{S}$ (sorted descending by value).

The percent value of eigenvalues is cumulatively summarised from the top until the sum is over the desired value (typically 99%) to reduce the dimensions of the modelled variance. The rest of the eigenvalues and their corresponding eigenvectors are then removed.

4.1.4 Model Generation and Constraints

An instance x of the model is generated by the following equation:

$$\mathbf{x} = \overline{\mathbf{x}} + \mathbf{\Phi b}, \tag{4.5}$$

where $\overline{\mathbf{x}}$ is the mean shape from the data, $\mathbf{\Phi}$ are the eigenvectors of the model and $\mathbf{b}$ is a vector with set of parameters of the deformable model. The constrain to the values of $\mathbf{b}$ was chosen by Cootes [12] to be $|b_i| \leq 3\sqrt{\lambda_i}$ to generate plausible models.

4.1.5 Fitting the Model to New Points

With a model instance obtained by equation 4.5, the next step is to add position (X_t, Y_t), rotation θ and scale s. The position of the model in the image is then given by:

$$\mathbf{x} = T_{X_t, Y_y, \theta, s}(\overline{\mathbf{x}} + \mathbf{\Phi b}), \tag{4.6}$$

which can be applied to single point with coordinates $[x, y]$ as follows:

$$T_{X_t, Y_y, \theta, s} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} X_t \\ Y_t \end{pmatrix} + \begin{pmatrix} s\cos\theta & s\sin\theta \\ -s\sin\theta & s\cos\theta \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}. \tag{4.7}$$

The sum of square distances is used to fit a generated model instance $\mathbf{x}$ to a new set of points in an image $\mathbf{Y}$, minimising the expression:

$$|\mathbf{Y} - T_{X_t,Y_y,\theta,s}(\overline{\mathbf{x}} + \mathbf{\Phi b})|^2. \tag{4.8}$$

A simple iterative approach is proposed by Cootes [12] to solve this problem.

1. Initialise parameters $\mathbf{b}$ to zero.
2. Generate model instance $\mathbf{x} = \overline{\mathbf{x}} + \mathbf{\Phi b}$.
3. Find the parameters (X_t, Y_y, θ, s) which best map $\mathbf{x}$ to $\mathbf{Y}$.
4. Invert the parameters (X_t, Y_y, θ, s) and use them to project $\mathbf{Y}$ to model co-ordinate frame:
$$\mathbf{y} = T^{-1}_{X_t,Y_y,\theta,s}(\mathbf{Y}). \tag{4.9}$$
5. Project $\mathbf{y}$ into the tangent plane to $\overline{\mathbf{x}}$ by scaling $\frac{1}{\mathbf{y}.\overline{\mathbf{x}}}$.
6. Update the model parameters $\mathbf{b}$ to match to $\mathbf{y}$:
$$\mathbf{b} = \mathbf{\Phi}^T(\mathbf{y} - \overline{\mathbf{x}}). \tag{4.10}$$
7. Apply constraints on $\mathbf{b}$ as described in 4.1.4.
8. If not converged, return to **2.**

The convergence is achieved when the model parameters $\mathbf{b}$ or the pose parameters (X_t, Y_y, θ, s) does not significantly change after the iteration.

4.2 Statistical Model of Appearance

Similarly to creating a statistical shape model, an appearance model can be created to generate the texture of an object. This complex model includes both shape variation and texture variation to generate the complete image of object. The model then generates the object in a shape-normalised frame.

4.2.1 Statistical Model of Texture

The mean shape of object from 4.1.3 is needed to create a set of equally sized vectors containing textures of different training images. Textures of different sizes are warped 4.2.3 to the mean shape. The same process as in 4.1.3 is then applied to the set of textures to create statistical model of texture. The effect of global lighting c variation is normalised by application of scaling α and offset β as follows:

$$\mathbf{g} = (\mathbf{g}_{im} - \beta\mathbf{1})/\alpha, \tag{4.11}$$

where $\mathbf{g}$ is the normalised texture and $\mathbf{g_{im}}$ is the texture sampled from the image warped in the mean shape. To calculate the values of α and β a mean texture $\overline{\mathbf{g}}$ needs to be defined as normalised data with zero sum and unity variance. The values of α and β are then given by:

$$\alpha = \mathbf{g}_{im}.\overline{\mathbf{g}}, \tag{4.12}$$

$$\beta = (\mathbf{g}_{im}.\mathbf{1})/n, \tag{4.13}$$

where n is the number of values in the texture vector. This process is recursive and can be done by choosing one example from the data as a mean and aligning the rest of the data to it (by applying 4.11, 4.12 and 4.13), then re-estimating the mean and iterating.

The PCA 4.1.3 is used on the normalised data to obtain a linear model of texture:

$$\mathbf{g} = \overline{\mathbf{g}} + \mathbf{P}_g\mathbf{b}_g, \tag{4.14}$$

where $\mathbf{g}$ is an instance of the model, $\overline{\mathbf{g}}$ is the mean normalised grey level vector, $\mathbf{P}_g$ is a set of eigenvectors (modes of variation), and $\mathbf{b}_g$ is a vector of parameters controlling the model.

4.2.2 Combined Appearance Model

The shape and the texture of an object can be represented by parameters of corresponding models $\mathbf{b}_s$ and $\mathbf{b}_g$. Correlations may exist between the variations of the models. A combined model is obtained by applying PCA 4.1.3 to the parameters of both models. For each corresponding training example the concatenated vector of

parameter is generated a follows:

$$\mathbf{b} = \begin{pmatrix} \mathbf{W}_s \mathbf{b}_s \\ \mathbf{b}_g \end{pmatrix} = \begin{pmatrix} \mathbf{W}_s \mathbf{P}_s^T (\mathbf{x} - \overline{\mathbf{x}}) \\ \mathbf{P}_g^T (\mathbf{g} - \overline{\mathbf{g}}) \end{pmatrix}, \tag{4.15}$$

because of the difference between units in both models (intensity vs. distance) the diagonal matrix $\mathbf{W}_s$ contains weights for each shape parameters. A simple approach to gain the weights is to set them to:

$$\mathbf{W}_s = r\mathbf{I}, \tag{4.16}$$

where r^2 is the ratio of the total intensity variation to the total shape variation in the normalised frames.

4.2.3 Image Warping

The simplest and the fastest method for warping image from one set of control points to a new set of control points is a piece-wise affine transformation. For the case of two dimensions a triangulation (Delanuy [15]) can be used to partition the convex hull to a set of triangles. The affine transformation can be then applied to each point within each triangle, which will uniquely map the corners of the triangle to their new position in the warped image. If $\mathbf{x}_1, \mathbf{x}_2$, and $\mathbf{x}_3$ are corners of a triangle then any internal point can be written as:

$$\mathbf{x} = \mathbf{x}_1 + \beta(\mathbf{x}_2 - \mathbf{x}_1) + \gamma(\mathbf{x}_3 - \mathbf{x}_1) = \alpha\mathbf{x}_1 + \beta\mathbf{x}_2 + \gamma\mathbf{x}_3, \tag{4.17}$$

where $\alpha = 1 - (\beta + \gamma)$. For $\mathbf{x}$ to be inside the triangle $0 \leq \alpha, \beta, \gamma \leq 1$. The point then maps to:

$$\mathbf{x}' = \alpha\mathbf{x}'_1 + \beta\mathbf{x}'_2 + \gamma\mathbf{x}'_3. \tag{4.18}$$

The main idea is to find a corresponding pixel in the original image for each pixel in the warped image. By applying this reverse search no holes will be present in the warped image.

4.3 Active Shape Model

The search for the model fit starts with a rough starting approximation of the object position and the mean shape of the shape model. The instances of the model are generated by changing the shape parameters $\mathbf{b}$. The fitting algorithm is trying to bring the starting parameters $\mathbf{b}_0$ to $\mathbf{b}_{obj}$, where the difference between the generated shape and the shape in the image is minimal. An iterative approach proposed by Cootes [12] to fit the instance of the model $\mathbf{x}$ to and image follows:

1. Examine a region of the image around each point $\mathbf{x}_i$, to find best nearby match for $\mathbf{x'}_i$.
2. Update parameters $(X_t, Y_t, s, \theta, \mathbf{b})$ to best fit $\mathbf{X}$ to the new found points $\mathbf{X'}$.
3. Repeat until converged.

The most used approach to locate new points $\mathbf{x'}_i$ is to look along profile normals as shown in 3. If the model boundary is an edge, the strongest edge is located along the profile. Its position then gives the new location $\mathbf{x'}_i$.

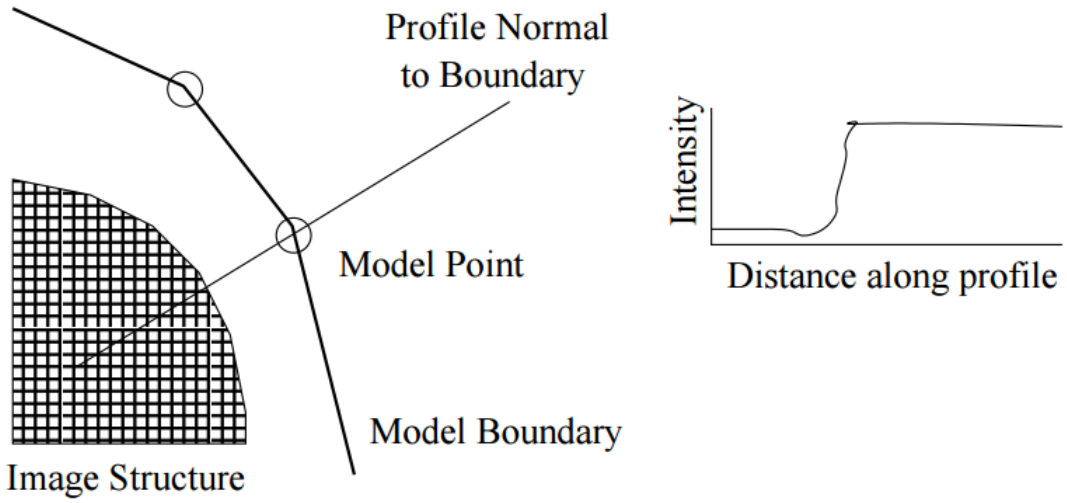


Figure 3: Sampling along point profile normal [12].

Model point, in general, does not always represent the strongest edge in the region but it also represents structures and secondary edges. The local structure has to be learned from the training data to fit the points to the desired coordinates. This problem is usually solved by two approaches. The first is to build a statistical model of the local structure 4.3.1 and the second is to solve the problem as a classification task. The classification was used by authors of [16] to separate lung fields in radiograph images.

4.3.1 Modelling Local Structure

The local structure is modelled by sampling k pixel along both directions of the profile for each point in each training image. This procedure produces $2k + 1$ samples for i^{th} point in j^{th} image. Only the derivatives are sampled to reduce the effect of global intensity and normalisation by the sum of absolute values is applied. This process is repeated for each image to produce a set of normalised samples $\{\mathbf{g_i}\}$. Their mean value $\overline{\mathbf{g}}$ and covariance $\mathbf{S}_g$ are estimated and used to compute the quality of fit of a new sample by:

$$f(\mathbf{g}_s) = (\mathbf{g}_s - \overline{\mathbf{g}})^T \mathbf{S}_g^{-1} (\mathbf{g}_s - \overline{\mathbf{g}}), \tag{4.19}$$

which is Mahalanobis distance of the sample $\mathbf{g}_s$ from the model mean.

During the search, a profile of m pixels is sampled from either side of the current point ($m > K$) and the quality of the fit is assessed at each of the $2(m-k)+1$ possible positions. The best match is then selected as the new position for the current point.

4.4 Active Appearance Model

The main advantage of Active Appearance Model (AAM) [12] over previously discussed Active Shape Model 4.3 is adding the information about the texture of the target object. The texture can be modelled by Appearance Model described in 4.2.1.

4.4.1 AAM Search

Assuming a reasonable starting approximation, a synthetic picture which matches the target object can be generated by AAM search algorithm. The algorithm solves an optimisation problem in which it minimises the difference between the image generated by the model and the actual image of the object. The difference is defined as vector $\delta\mathbf{I}$:

$$\delta\mathbf{I} = \mathbf{I}_i - \mathbf{I}_m, \tag{4.20}$$

where $\mathbf{I}_i$ is the vector of intensity values from the image and $\mathbf{I}_m$ is the vector of intensity values generated by current model parameters.

The proposed algorithm minimises the difference $\Delta = |\delta\mathbf{I}|^2$ by varying model's parameter $\mathbf{c}$. Each attempt to fit the model to the image is a similar optimisation

problem. This allows some of the information about the fitting to be learned in advance. This a priory information is the knowledge about adjusting the model parameters in dependence to the difference vector. This represents two parts of the problem to find the best fit. The first part is learning the relation between the difference vector $\delta\mathbf{I}$ and the error in model parameters $\delta\mathbf{c}$ and the second part is using the previous information to iteratively solve the minimisation problem.

4.4.2 Learning the Relation between δc and δI

The parameters $\mathbf{c}$ of the combined appearance model 4.2.2 are controlling the shape and the texture of the instances by:

$$\mathbf{x} = \overline{\mathbf{x}} + \mathbf{Q}_s\mathbf{c}, \tag{4.21}$$

$$\mathbf{g} = \overline{\mathbf{g}} + \mathbf{Q}_g\mathbf{c}, \tag{4.22}$$

where $\overline{\mathbf{x}}$ is the mean shape and $\overline{\mathbf{g}}$ is the mean texture in a mean shaped patch. $\mathbf{Q}_s$ and $\mathbf{Q}_g$ are matrices of the eigenvectors describing the modes of variations computed from the training set.

The shape of the object in the image frame is represented by $\mathbf{X}$, which is gained by applying appropriate transformations to the points $\mathbf{x} : \mathbf{X} = S_\mathbf{t}(\mathbf{x})$. $S_\mathbf{t}$ is a similarity transformation consisting of scale s, in-plane rotation θ, and a translation (t_x, t_y). To keep the problem linear, scaling and rotation are represented as (s_x, s_y), where $s_x = (s\cos\theta - 1)$, $s_y = s\sin\theta$. The pose parameter vector is then $\mathbf{t} = (s_x, s_y, t_x, t_y)^T$. The texture in the image frame is generated by $\mathbf{g}_{im} = T_\mathbf{u}(\mathbf{g}) = (u_1+1)\mathbf{g}_{im} + u_2\mathbf{1}$, where $\mathbf{u} = (\alpha - 1, \beta)$ described in 4.2.1.

The difference between the model instance and the image is:

$$\mathbf{r}(\mathbf{p}) = \mathbf{g}_s - \mathbf{g}_m, \tag{4.23}$$

where model parameters $\mathbf{p} = (\mathbf{c}^T|\mathbf{t}^T|\mathbf{u}^T)$.

A first order Taylor expansion of 4.23 gives:

$$\mathbf{r}(\mathbf{p} + \delta\mathbf{p}) = \mathbf{r}(\mathbf{p}) + \frac{\partial\mathbf{r}}{\partial\mathbf{p}}\delta\mathbf{p}, \tag{4.24}$$

where the ij^{th} element of matrix $\frac{\partial\mathbf{r}}{\partial\mathbf{p}}$ is defined as $\frac{dr_i}{dp_j}$.

To minimise $|\mathbf{r}(\mathbf{p}+\delta\mathbf{p})|^2$ by choosing $\delta\mathbf{p}$ the equation 4.24 is equated to 0 to obtain the RMS solution:

$$\delta\mathbf{p} = -\mathbf{R}\mathbf{r}(\mathbf{p}) \quad where \quad \mathbf{R} = \left(\frac{\partial\mathbf{r}}{\partial\mathbf{p}}^T \frac{\partial\mathbf{r}}{\partial\mathbf{p}}\right)^{-1} \frac{\partial\mathbf{r}}{\partial\mathbf{p}}^T. \tag{4.25}$$

Since the model is being computed in a normalised reference frame, $\frac{\partial\mathbf{r}}{\partial\mathbf{p}}$ is considered as fixed and thus can be pre-calculated. This is done by numeric differentiation, systematically displacing each parameter from the optimal value known by back-projection of the model to the training data. Residuals at displacements are a value for up to 0.5 of standard deviation of each parameter and combined with Gaussian kernel to smooth them:

$$\frac{dr_i}{dp_j} = \sum_k w(\delta c_{jk})(r_i(\mathbf{p} + \delta c_{jk}) - r_i(\mathbf{p})), \tag{4.26}$$

where $w(x)$ is a suitably normalised Gaussian weight function.

Cootes [12] proposed that optimal perturbation of scale s is 10% and 3 pixels for translation.

4.4.3 Iterative Model Refinement

An iterative procedure proposed by Cootes [12] based on predicting the correction of the parameters $\mathbf{c}$ is solving the optimisation problem. Starting from the current estimate of model parameters $\mathbf{c}_0$ and the normalised image sample $\mathbf{g}_s$ one iteration of the algorithm is:

1. Evaluate the error $\delta\mathbf{g}_0 = \mathbf{g}_s - \mathbf{g}_m$
2. Evaluate the current error $E_0 = |\delta\mathbf{g}_0|^2$
3. Compute the predicted correction $\delta\mathbf{c} = \mathbf{A}\delta\mathbf{g}_0$
4. Set $k = 1$
5. Let $\mathbf{c}_1 = \mathbf{c}_0 - k\delta\mathbf{c}$
6. Sample the image at the new prediction, and calculate a new Error vector $\delta\mathbf{g}_1$
7. If $|\delta\mathbf{g}_1| < E_0$ then accept the new estimate $\mathbf{c}_1$
8. Otherwise try at $k = 1.5, k = 0.5, k = 0.25$ etc.

Chapter 5

Neural Networks

During the last few years, a great progress has been done in the field of machine learning. Evolving hardware enabled an easy access to deep neural networks toolboxes and pushed down training times to manageable levels. Training times are improved mainly by the use of modern graphical acceleration units(GPUs). In the time of writing this work, various companies begin to sell specialised accelerators for deep learning algorithms.

Deep neural networks are based on the idea of simulating the function of the human brain. Its complex structure is not yet fully understood, but the attempts to understand its principal function led the scientists to the definition of a basic unit called neuron. The model of an artificial neuron is changing since the time of its creation and the latest used definition is described in this chapter. To simulate the function of the human brain, the neurons are organised into structures called networks. The structure is derived from the connections between biological neurons in the brain. A biological neuron is an electrically excitable cell that processes and transmits information in the form of electrical and chemical signals. The connections between neurons are called synapses. Each neuron may be connected to up to 10,000 other neurons. By simulating these connections and their biological function an artificial neural network is formed. The number of connections between neurons in artificial networks is much smaller than in human brain to accommodate for the limited resources the computer has to process the network. There is also an important difference between artificial and biological neural networks. The biological network and its neurons can work asynchronously but the artificial network only works synchronously. The asynchronous process in artificial networks can be simulated by using recurrent layers, which implements time-dependent memory.

The most important neural network progress towards image processing was the introduction of Convolutional Neural Networks[17]. The concept was later used in image classification by Krizhevsky et al. in [18]. This chapter includes a description of the basic functions, the most used layers, and the process of training the network. A short review of the currently available frameworks for neural networks training and simulation is also included.

5.1 Artificial Neuron

The artificial neuron is defined as a mathematical function that models the function of a biological neuron. It has one or more inputs (representing dendrites) and one output (axon) which is represented by a weighted sum of all inputs. An activation function is usually applied to the weighted sum.

5.1.1 Activation Functions

The activation function of a neuron can be linear or non-linear. A short list of the most used functions follows. x is the weighted sum of inputs.

- Linear activation function - no function is applied on the weighted sum of inputs, only bias is added.

$$f(x) = x \tag{5.1}$$

- Sigmoid activation function

$$f(x) = \frac{1}{(1 + e^{-x})} \tag{5.2}$$

- Rectified Linear Unit(ReLU)

$$f(x) = \max(0, x) \tag{5.3}$$

- Softplus - a smooth approximation of ReLU

$$f(x) = \ln(1 + e^{x}) \tag{5.4}$$

- Hyperbolic tangent activation function

$$f(x) = \tanh(x) \tag{5.5}$$

5.2 Neural Network Topology

Neural networks are composed of different types of hidden layers. Each layer can perform various tasks depending on its type. A brief overview of basic layers is included in this section.

5.2.1 Fully Connected Layer

The fully connected layer is the most commonly used layer in neural networks. It is also called dense layer in some works. It is composed of n neurons. Each neuron is connected to each output of previous layer or each input value. Each of the connections has its own weight $w_{i,n}$. The number of outputs is equal to the number of n. Each output has its own value of bias b_n. There is one activation function f for all neurons in this layer. The output is mathematically represented by equation 5.6. A basic representation is shown on picture 4.

$$f(\mathbf{x}) = \varphi(\mathbf{x}\mathbf{W} + \mathbf{b}) \tag{5.6}$$

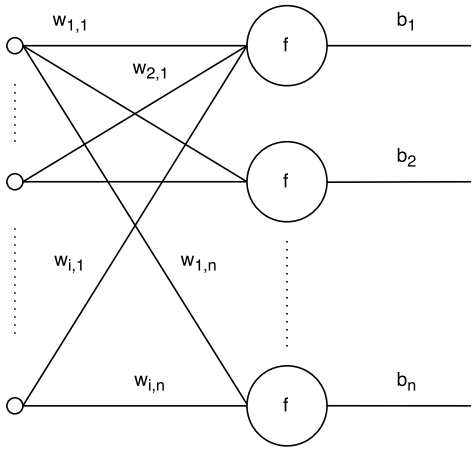


Figure 4: Fully connected layer.

5.2.2 Convolutional Layer

The convolutional layer is the core block of Convolutional Networks. It is built from a set of learnable filters. Each filter has height h, width w and it extends through all input channels d (usually 3 for coloured images). During a forward pass, each filter is slid(convolved) across height and width of the input and a dot product is computed between the input and filter weights. The stride of in each dimension can be defined separately. The output of Convolutional layer is a stack of activation maps produced by all the filters. A graphical representation of the principle is shown in picture 5 .

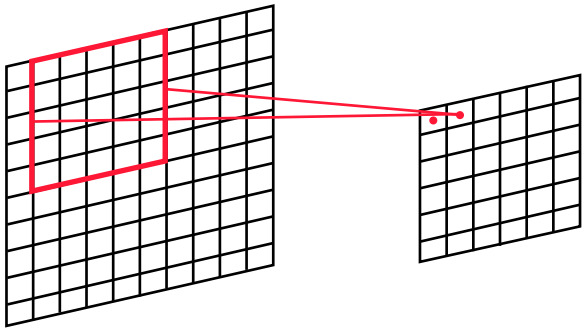


Figure 5: Convolutional layer.

5.2.3 Response Normalisation Layers

This type of layers simulates a biological concept called lateral inhibition. Lateral inhibition is the capacity of an excited neuron to outweigh the activity of its neighbours It disables the spreading of information from excited neurons in the lateral direction. This effect creates a significant peak in the form of local maxima which increase the sensory perception. Three commonly used algorithms for response normalisation follow in this chapter.

Batch normalisation

Ioffe and Szegedy proposed an algorithm called batch normalisation [19]. This algorithm performs normalisation as part of the network model architecture for each training mini-batch. It allows training of networks with higher learning rates and to be less careful about layers initialisation. The batch normalisation acts as a regulariser, in some cases eliminating the need for Dropout layer.

Dropout Layer

The dropout layer prevents a neural network to overfit. With limited training data, many of the relations learned by the network may be the result of the sampling noise. The dropout layer uses a variable to set a probability of setting a neuron output to zero [20]. This probability is applied to every neuron in the affected layer.

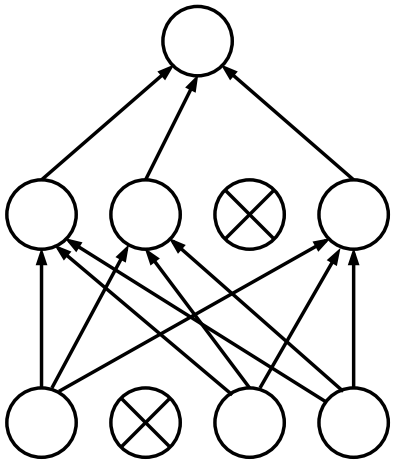


Figure 6: Dropout principle with 25% drop probability.

Local Response Normalisation

Krizhevsky et al. in their article [18] suggested a method to simulate lateral inhibition for ReLU activations. This method applyies the expression 5.7 to output of neuron $a^i_{x,y}$ to produce the response-normalised activity $b^i_{x,y}$.

$$b^i_{x,y} = a^i_{x,y} / \left(k + \alpha \sum_{j=\max(0,i-n/2)}^{\min(N-1,i+n/2)} (a^j_{x,y})^2 \right)^{\beta} \tag{5.7}$$

N is the total number of kernels in the layer, n is the number of adjacent kernel maps at the same spatial position. The constants k, n, α, and β are hyper-parameters with values determined by the authors using a validation set ($k = 2, n = 5, \alpha = 10^{-4}$, and $\beta = 0.75$).

L2 Regularisation

It is one of the most commonly used regularisation techniques, also called weight decay. The main idea is to add an extra part to the cost function called the regularisation term. The term looks like:

$$\frac{\lambda}{2n}\sum_{w} w^2, \tag{5.8}$$

where w are all weights in the network. The sum of squared weights is then scaled by a factor $\frac{\lambda}{2n}$, where $\lambda > 0$ is the regularisation parameter and n is the size of the training set. The regularised cost function then looks like:

$$C = C_0 + \frac{\lambda}{2n}\sum_{w} w^2, \tag{5.9}$$

where C is the regularised cost function and C_0 is the original cost function.

5.2.4 Pooling Layers

Pooling is a process, where a rectangular window is slid over the data to compute data reduction by averaging or replacing by the maximum. This type of layer is usually placed between two consecutive convolutional layers. The layer takes as an input the size of the sliding window and the stride of the window in each direction. The operation is mostly used on each channel separately, but a 3D pooling is also possible.

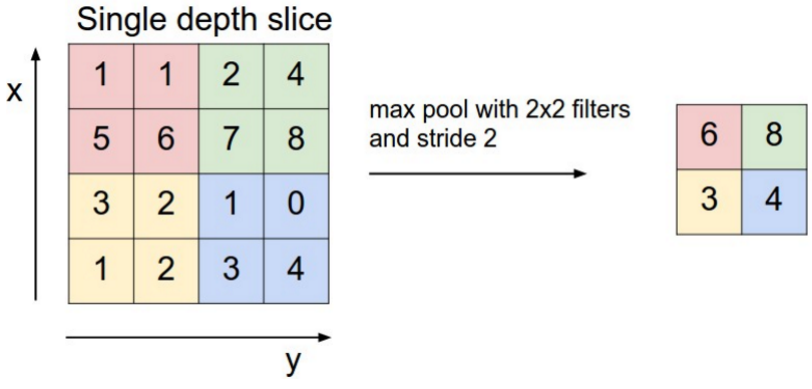


Figure 7: Max pooling [21].

5.2.5 Recurrent Layers

This type of layer contains an internal memory which allows the layer to create an internal state and allows it to exhibit dynamic temporal behaviour. It is often used to process sequences of input data. It is currently used to process speech recognition tasks, image sequence description etc.

Simple Recurrent Layer

This type of network was first proposed by Elman [22] in 1990. It was created as simple three-layer feed-forward back propagation network. The only difference is that the input in composed of two parts. One part is the input data and the other part is the pattern of activation over the network's own hidden units. The process is illustrated on picture 8. The output y of at time t is defined as:

$$\begin{aligned} \mathbf{h}_t &= \varphi\left(\mathbf{W}_h \mathbf{x}_{t-1} + \mathbf{U}_h \mathbf{h}_{t-1} + \mathbf{b}_h\right) \\ \mathbf{y}_t &= \varphi\left(\mathbf{W}_y \mathbf{h}_t + \mathbf{b}_y\right) \end{aligned} \tag{5.10}$$

where $\mathbf{h}_t$ is hidden layer state at time t, $\mathbf{x}_t$ is input vector and $\mathbf{W}, \mathbf{U}$ and $\mathbf{b}$ are weights and biases of the network.

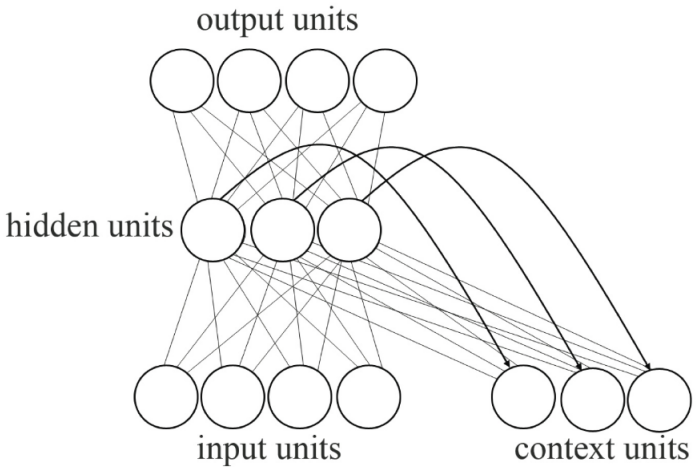


Figure 8: Simple recurrent network [23].

Long Short-term Memory

Long short-term memory(LSTM) is a type of recurrent neural network proposed by Hochreiter and Schmidhuber [24]. It consists of LSTM units. Each unit has its own input, output and forget gates. The basic scheme is shown on picture 9. There is no activation function within the recurrent components of the unit. Thus the value in the memory block in not changed iteratively and the gradient does not vanish when backpropagation through time is applied.

$$\begin{aligned}
\mathbf{f}_t &= \varphi(\mathbf{W}_f \mathbf{x}_t + \mathbf{U}_f \mathbf{h}_{t-1} + \mathbf{b}_f) \\
\mathbf{i}_t &= \varphi(\mathbf{W}_i \mathbf{x}_t + \mathbf{U}_i \mathbf{h}_{t-1} + \mathbf{b}_i) \\
\mathbf{o}_t &= \varphi(\mathbf{W}_o \mathbf{x}_t + \mathbf{U}_o \mathbf{h}_{t-1} + \mathbf{b}_o) \\
\mathbf{c}_t &= \mathbf{f}_t \circ \mathbf{c}_{t-1} + \mathbf{i}_t \circ \varphi(\mathbf{W}_c \mathbf{x}_t + \mathbf{U}_c \mathbf{h}_{t-1} + \mathbf{b}_c) \\
\mathbf{h}_t &= \mathbf{o}_t \circ \varphi(\mathbf{c}_t)
\end{aligned} \tag{5.11}$$

$\mathbf{c}_t$ is internal state of the unit. $\mathbf{W}, \mathbf{U}$ and $\mathbf{b}$ are weights and biases of internal gates. Activation functions are sigmoids for gates and hyperbolic tangent for the memory unit.

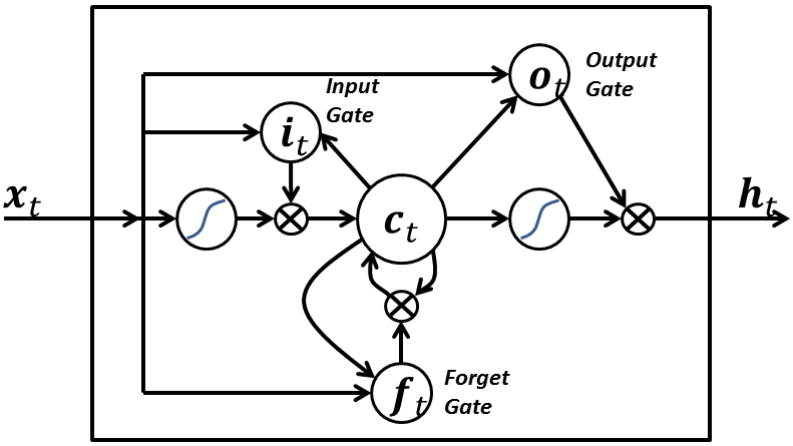


Figure 9: LSTM unit structure (from Wikipedia).

5.2.6 Softmax Layer

Softmax layer is commonly used in classification task as the last layer in the network. It applies the softmax function to the output of the previous layer. The number of

outputs corresponds to the number of classes in the training data. The softmax function is defined as:

$$\sigma(\mathbf{z})_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}} \quad \text{for } j = 1, ..., K. \tag{5.12}$$

In classification, the output is used to represent a categorical distribution (the probability distribution over K classes with the sum of one).

5.3 Training the Network

The learning process of neural networks is based on an algorithm called backpropagation [25]. The training is implemented as a supervised learning algorithm. Training data are continuously input to the network while its output is compared to the information from the supervisor. Data are then provided as a pair $[input, label]$, where the *label* is desired output of network after forward pass when input is provided.

5.3.1 Cost Functions

The goal of the training algorithm is to minimise the error between the demanded data and the output of the neural network. This error is represented by a cost function. These are the most commonly used cost functions:

- **Mean squared error** - MSE is one of the simplest cost functions that can be used for training. It is defined as:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \overline{y}_i)^2, \tag{5.13}$$

  where y_i is the i-th desired output and $\overline{y}_i$ is the corresponding output of the network. N is the number of outputs. This function is commonly used in tasks of regression.

- **Cross-entropy** is based on information theory. It is a difference between two

probability distributions. The function is defined as follows:

$$H(x) = -\sum_{i=1}^{N} p(x) \log q(x), \tag{5.14}$$

where $p(x)$ is the the desired output and $q(x)$ is the output of the network. Cross entropy works best when the data are normalised between values 0 and 1. This cost function is most commonly used for classification tasks.

- **Kullback-Leibler Divergence** is very similar to cross-entropy, but in information theory it focuses on the extra number of bits needed to encode the data. It is defined as:

$$KL(P||Q) = \sum_{i=1}^{N} p(x) \log \frac{p(x)}{q(x)}, \tag{5.15}$$

this mean that when $p = q$ the KL divergence is equal to 0. This function is not a distance in contrary to cross entropy.

5.3.2 Optimisation Algorithms

The optimisation algorithms are used to minimise the objective function of neural networks. The objective function we want to minimise is:

$$E(\mathbf{w}) = \frac{1}{n} \sum_{i=1}^{n} E_i(\mathbf{w}), \tag{5.16}$$

where $\mathbf{w}$ are weights of the network.

Stochastic Gradient Descent

Stochastic gradient descent is the most commonly used algorithm for neural network weights update. It is a stochastic approximation of the gradient descent optimisation method. The weights start at the initialisation values w_0 and then we apply the update by Gradient Descent method as follows:

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta_t \nabla E(\mathbf{w}_t), \tag{5.17}$$

where η is learning rate. The problem is that this method needs all the available data to be efficient. Since computer memory is limited the training data have to be divided

into batches. The update is then:

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta_t \sum_{i=1}^{n} \nabla E(\mathbf{w}_t), \tag{5.18}$$

where if n is the size of the training dataset, the method is Gradient Descent. If $n = 1$, then this method is called Stochastic Gradient Descent(SGD), where the sample is chosen randomly. If n is between one and the size of the training dataset, the method is called mini-batch SGD. The data are usually shuffled before training the network instead of choosing them randomly. The following methods are based on SGD.

Adagrad

Adagrad adapts the learning rates to each parameter [26]. It performs larger updates for infrequent and smaller updates for frequent parameters [27]. This approach improves performance for sparse data and where sparse parameters are more informative. This includes tasks of natural language processing and image recognition. It still has a global learning rate η, but this learning rate is multiplied with elements of a vector $\mathbf{G}_{j,j}$ as follows:

$$\mathbf{G} = \sum_{\tau=1}^{t} \mathbf{g}_\tau \mathbf{g}_\tau^T, \tag{5.19}$$

where $\mathbf{g}_\tau = \nabla Q_i(\mathbf{w})$ is the gradient at iteration τ. The diagonal $\mathbf{G}_{j,j}$ is then given by:

$$\mathbf{G}_{j,j} = \sum_{\tau=1}^{t} \mathbf{g}_{\tau,j}^2. \tag{5.20}$$

The per parameter update is then computed as follows:

$$w_{j_{t+1}} = w_{j_t} - \frac{\eta}{\sqrt{G_{j,j_t} + \epsilon}} g_{j_t}, \tag{5.21}$$

where ϵ is a small constant to eliminate a division by zero.

Adadelta

Adadelta [28] is based on Adagrad. It is designed to reduce the monotonic decrease of Adargrad's learning rate. Instead of accumulating all past square gradients, it only accumulates square gradients in a fixed size window s. The sum of gradients is recursively defined as a decaying average of all past gradients [27]. The running average $E[g^2]_t$ at time step t is defined as:

$$E[g^2]_t = \gamma E[g^2]_{t-1} + (1-\gamma)g_t^2, \tag{5.22}$$

where γ is defined similarly as the Momentum (typically 0.9). The parameter update changes to:

$$\Delta w_t = -\frac{\eta}{\sqrt{E[g^2]_t + \epsilon}} g_t, \tag{5.23}$$

where the denominator is the root mean squared error (RMS) of the gradient. The authors of [28] then define a squared gradient of squared parameter updates:

$$E[\Delta\theta^2]_t = \gamma E[\Delta\theta^2]_{t-1} + (1-\gamma)\Delta\theta_t^2, \tag{5.24}$$

with its RMS error defined as:

$$RMS[\Delta\theta]_t = \sqrt{E[\Delta\theta^2]_t + \epsilon} \tag{5.25}$$

Since $RMS[\Delta\theta]_t$ is unknown, it is approximated from the previous step by $RMS[\Delta\theta]_{t-1}$ replacing the learning rate η. Finally the Adadelta update rule can be rewritten as:

$$w_{j_{t+1}} = w_{j_t} - \frac{RMS[\Delta\theta]_{t-1}}{RMS[g]_t} g_t. \tag{5.26}$$

Root Mean Square Propagation

RMSProp is another method proposed to solve the problem of Adagrad's radically diminishing learning rates. It was proposed by Geoff Hinton in his lectures on Coursera and not published. The method was developed at the same time as Adadelta and in fact, the first update vector of both methods is identical. The update rule is defined by Hinton as:

$$w_{j_{t+1}} = w_{j_t} - \frac{\eta}{RMS[g]_t} g_t, \tag{5.27}$$

where the initial learning rate η is suggested as 0.001 and γ as 0.9.

Adaptive Moment Estimation

Adaptive moment Estimation(Adam) [29] is another method that computes adaptive learning rates for each parameter of the network. In addition to storing an exponentially decaying average of previously squared gradients v_t like Adadelta and RMSProp, Adam also keeps and the exponentially decaying average of previous gradients m_t, like momentum [27]:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t, \tag{5.28}$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2, \tag{5.29}$$

where m_t is an estimate of the first moment (the mean), and v_t is an estimate of the second moment (the uncentered variance) of the gradients. The author of [29] observed that m_t and v_t are biased towards zero and proposed bias-corrected first and second moments:

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t}, \tag{5.30}$$

$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t}. \tag{5.31}$$

The Adam update rule is then proposed by authors as:

$$w_{t+1} = w_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon} \hat{m}_t \tag{5.32}$$

AdaBound

The research in the task of finding a better optimisation algorithm still continues and in the last year a paper was published that claims to bring an optimiser as fast as Adam mentioned above and as good as SGD. This algorithm was named AdaBound

[30]. The main idea of the paper is introducing adaptive dynamic bounds on learning rates to get a gradual transition from adaptive methods to SGD.

5.4 Deep Learning Frameworks

This section provides a short overview of the most used frameworks, which are publicly available for DNN development and training. The number of the frameworks is large and different teams are focusing on different goals. The common goal is to provide a framework to implement neural network topologies and deep learning algorithms. The frameworks are usually supporting multiple programming languages where Python is the most common. The differences are mainly in speed, supported GPUs, and supported operating systems.

5.4.1 Caffe

Caffe [31] is a DNN framework made primarily for image processing. It was developed by Berkeley Vision and Learning Center and extended by community contributions. It is distributed as BSD-licensed C++ library with Python and Matlab support. The model definition is separated from actual implementation, which allows platforms switching and easy deployment. CUDA GPUs are supported for faster processing.

5.4.2 Theano

Theano [32] is a Python library for computing mathematical expressions including multidimensional arrays efficiently. It is highly optimised for CUDA GPUs. Theano can compute symbolic differentiations, reuse partial results to save system resources and compute in place operations. It can be easily extended by adding user created functions written in Python, C++ or CUDA. User defined functions are stored as a graph of variables and operations, that is optimised at compilation time.

5.4.3 Tensorflow

Tensorflow [33] is a machine learning interface and toolbox developed by Google as a second generation system for deployment of machine learning following DistBelief

[34]. It implements computations as a dataflow-like model and maps them on a wide variety of hardware platforms. Its support ranges from Android and IOs to large scale GPU cluster. Main supported languages are Python and C++. It is worth mentioning that Tensorflow recently started support Microsoft Windows with GPU support.

5.4.4 Torch7

Torch7 [35] is a framework directed on providing three main advantages over other frameworks: (1) easy development of numerical algorithms, (2) easy extension by user defined libraries, and (3) speed. It is developed in Lua [36] scripting language, which is written in clean C. Torch7 implements CUDA GPU support.

5.4.5 CNTK

Computational network toolkit (CNTK) [37] is a toolkit developed by Microsoft for neural network development. It is written in C++ with support for CUDA GPUs. It implements efficient memory management by removing duplicated computations in forward and backward passes and by reusing them it reduces memory reallocations. The models are written in network description language or simple network builder further described in [37].

Chapter 6

State-of-the-art methods for feature extraction and visual speech recognition

This chapter contains an overview of the currently used methods in the fields of feature extraction and visual (audio-visual) speech recognition. It is divided into two sections accordingly. The first section provides a review of state-of-the-art methods used for feature extraction focused on lipreading with methods based on statistical models, neural networks, and regression trees. The second chapter provides an overview of state-of-the-art methods used for visual speech recognition. These methods are mainly based on neural networks.

6.1 State-of-the-Art Methods for feature extraction

The process of visual speech feature extraction depends on the analysis of the speaker's face region. The analysis was mainly focused on the mouth region in the past but current methods usually take into account the whole face. By analysing the face as a whole, some additional information can be gained from the whole expression, eye movement, etc. The methods in this chapter are focused on landmark detection by statistical models and decision trees. Then follows a method proposing the features analysis and augmentation to improve their usability in the task of lipreading. At the end of the chapter, there is a network used mainly for object detection classification which is currently the most used for feature extraction by separation the last classifica-

tion layer and using the raw output of the last fully connected layer. An interesting new method for training very deep networks is also briefly mentioned for it could provide a relevant improvement to the current state-of-the-art approaches to neural network training.

6.1.1 Chehra

A framework named Chehra implements Parallel Cascade of Linear Regression method. The authors of Incremental Face Alignment in the Wild [38] proposed a method for face shape detection based on a discriminative model. *In-the-wild* means that the method is successful on data created in uncontrolled environments. The paper presents a study about the problem of applying Incremental training to the discriminative facial deformable model. The incremental learning is mostly used to train AAM [12, 39] by the means of Incremental Principal Component Analysis [40]. A cascade of linear regressors is used to learn the mapping from facial texture to the shape. This is achieved by application of Monte-Carlo sampling methodology [41, 42]. The proposed method presents an algorithm that is capable of adding new samples and updating the models without retraining and that can automatically adapt to the currently tracked subject and ambient conditions.

Sequential Cascade of Linear Regression

This method was published by Xiong at al. [41]. The method uses a function $\mathbf{f}(\mathbf{I}, \mathbf{s})$ for representing the features around each of the landmark, where $\mathbf{I}$ is an image from the training set with corresponding shape $\mathbf{s}$. This function could return a vector of concatenated SIFT [43] or Histogram of Oriented Gradient (HoG) [4] from the training image. The training procedure of a discriminative model can be written as: Find a function g that maps the initial shape $\mathbf{s}^a$ of image $\mathbf{I}_i$ to the ground truth shape $\mathbf{s}_\mathbf{i}^*$, as $g(\mathbf{s}^a, \mathbf{I}_t, \mathbf{F}) = \mathbf{s}_\mathbf{i}^*$. The initial shape can be a mean shape from the training set initialised in a bounding box from a face detector.

The function g is learned iteratively using a cascade of regression functions in [41]. The authors of [38] used a parametric 3D shape model [44] described as:

$$\mathbf{s}(\mathbf{p}) = s\mathbf{R}(\bar{\mathbf{s}} + \mathbf{\Phi}_s\mathbf{g}) + \mathbf{t}, \tag{6.1}$$

where $\mathbf{R}$ is computed via pith r_x, yaw r_y and roll r_z, s is scale and $\mathbf{t} = [t_x; t_y; 0]$.

These parameters control the rigid 3D rotation, scale and translation. The expression $\bar{\mathbf{s}} + \mathbf{\Phi}_s\mathbf{g}$ represents a shape model as described in 4.1.4, where $\mathbf{g}$ control the non-rigid variations of the shape. The parameters of the 3D shape are then represented as $\mathbf{p} = [s; r_x, r_y, r_z; t_x; t_y, \mathbf{g}]^T$. This set replaces the ground truth shapes $\mathbf{s}_i^*$ from the training set with ground truth parameters $\mathbf{p}_i^*$. The Monte-Carlo algorithm works as follows. For each training shape $\mathbf{s}_i^*$, the shape model parameters subspace is sampled within a pre-defined range around ground-truth parameters $\mathbf{p}_i^*$ and an initial set of L perturbed shapes is sampled which provides a set of L perturbed shape parameters $\{\mathbf{p}_j^{(1)}\}_{j=1}^L$. The linear rule can be learned from the perturbed parameters such that [38]:

$$\begin{aligned} \mathbf{p}^* &= \mathbf{p}^{(1)} + \mathbf{f}(\mathbf{I}, \mathbf{s}(\mathbf{p}^{(1)}))\mathbf{W} + \mathbf{b} \\ &= \mathbf{p}^{(1)} + [\mathbf{f}(\mathbf{I}, \mathbf{s}(\mathbf{p}^{(1)}))1]\tilde{\mathbf{W}} \\ &= \mathbf{p}^{(1)} + \tilde{\mathbf{f}}(\mathbf{I}, \mathbf{s}(\mathbf{p}^{(1)}))\tilde{\mathbf{W}}, \end{aligned} \tag{6.2}$$

where $\tilde{\mathbf{W}} = [\mathbf{W}; \mathbf{b}]$ and $\tilde{\mathbf{f}}(\mathbf{I}, \mathbf{s}(\mathbf{p}^{(1)})) = [\mathbf{f}(\mathbf{I}, \mathbf{s}(\mathbf{p}^{(1)}))1]$. Learning only single $\tilde{\mathbf{W}}$ would be difficult and thus a cascade of regression functions is trained in sequential manner. The first $\tilde{\mathbf{W}}^{(1)}$ is estimated by [38]:

$$\tilde{\mathbf{W}}^{(1)} = \left[(\mathbf{X}^{(1)})^T\mathbf{X}^{(1)} + \lambda\mathbf{E}\right]^{-1}(\mathbf{X}^{(1)})^T\mathbf{Y}^{(1)}, \tag{6.3}$$

where $\mathbf{X}^{(1)} = [\tilde{\mathbf{f}}_{i,j}] = [\tilde{\mathbf{f}}(\mathbf{I}_i, \mathbf{p}_{i,j})]$, $\mathbf{Y}^{(1)} = [\Delta\mathbf{p}_{i,j}^{(1)}] = [\mathbf{p}_i^* - \mathbf{p}_{i,j}^{(1)}]$, and $\mathbf{E}$ is the identity matrix. The term $\lambda\mathbf{E}$ is included for case that $(\mathbf{X}^{(1)})^T\mathbf{X}^{(1)}$ is singular, j counts the perturbations. This approach is known as Ridge Regression [45].

The update fo k-th step can be generalised from the update rule $\mathbf{p}_{i,j}^{(2)} = \mathbf{p}_{i,j}^{(1)} + \tilde{\mathbf{f}}(\mathbf{I}_i, \mathbf{s}(\mathbf{p}_{i,j}^{(1)}))\tilde{\mathbf{W}}^{(1)}$ as:

$$\begin{aligned} \tilde{\mathbf{W}}^{(k)} &= \left[(\mathbf{X}^{(k)})^T\mathbf{X}^{(k)} + \lambda\mathbf{E}\right]^{-1}(\mathbf{X}^{(k)})^T\mathbf{Y}^{(k)} \\ \mathbf{p}_{i,j}^{(k+1)} &= \mathbf{p}_{i,j}^{(k)} + \tilde{\mathbf{f}}(\mathbf{I}_i, \mathbf{s}(\mathbf{p}_{i,j}^{(k)}))\tilde{\mathbf{W}}^{(k)} \end{aligned} \tag{6.4}$$

The problem of adapting the Sequential Cascade of Linear Regression (Seq-CLR) to a set of new training samples $\mathbf{p}_{new}$ is shown in the picture 10. Since the initial regression function is updated to $\mathbf{W}_{new}^{(1)}$ every subsequent regression function have to be updated according to equation 6.4. This is computationally extremely intensive and

time-consuming.

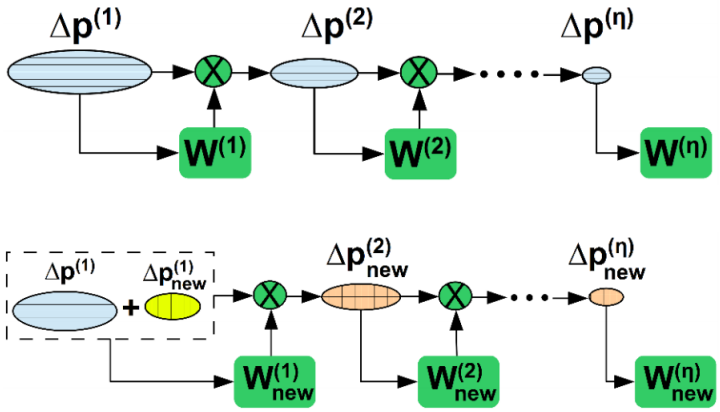


Figure 10: Updating Seq-CLR after adding new samples [38].

Parallel Cascade of Linear Regression

The authors of [38] proposed a new method to solve the problem of adapting the Seq-CLR model to a new set of training samples. The method was developed to be as accurate as Seq-CLR, the perturbations required for training or updating the cascade regression not relying on the previous iteration, and that the method is suited for incremental formulation thus allowing parallelisation of the update process.

In the original Seq-CLR the Monte-Carlo sampling procedure [41] is used to obtain the initial set of perturbed parameters $\Delta\mathbf{p}^{(1)}$. The aim of the cascade is to reduce the variance of the perturbations at each level. Based on this the authors of [38] argued that the regression functions at all levels in a cascade can be trained and updated independently using only the statistics of the previous level. This approach eliminates the need to propagate the samples through the whole cascade in sequence. The proposed method was named Parallel Cascade of Linear Regression (Par-CLR).

To implement this method, the variance of the shape is computed in each iteration of the regression, while training the regression functions by Seq-CLR on an off-line training set. In Par-CLR the perturbations for the training are drawn directly from the distributions described by the previously sampled variations (represented on picture 11). This makes the individual steps independent of each other and allows the update to be computed in parallel.

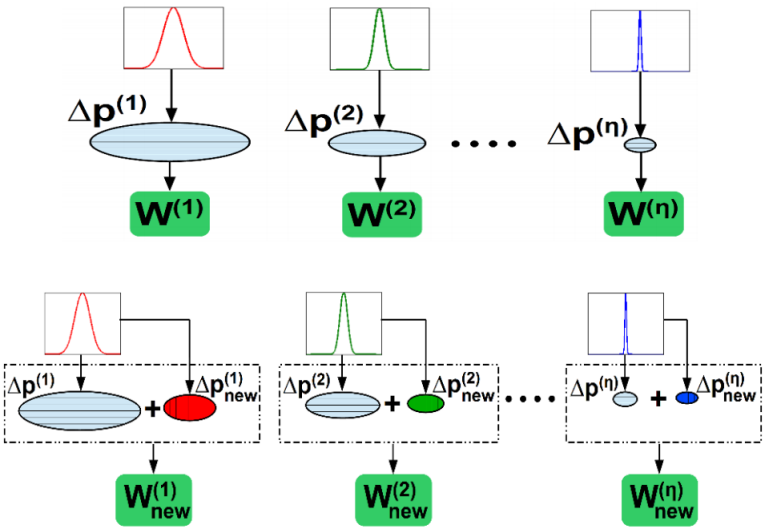


Figure 11: Updating par-CLR after adding new samples [38].

For reference, this approach can be compared to the search problem of the original AAM 4.4.1. The method was trained and tested on Helen (7.1.1) and LFPW (7.1.2) datasets. The detailed analysis of the fitting and adaptation process accuracy can be found in the original article [38].

6.1.2 Ensemble of Regression Trees

A decision tree is a binary tree. Binary tree means that every parent node has two child nodes. The decision trees are used mainly for regression or classification. Methods described bellow use the regression trees. Regression tree predicted output is usually a real number. Trees can be organised into structures called ensembles, which are composed of multiple trees with different decision criteria. The general structure of classification and regression trees was described by Breiman in [46].

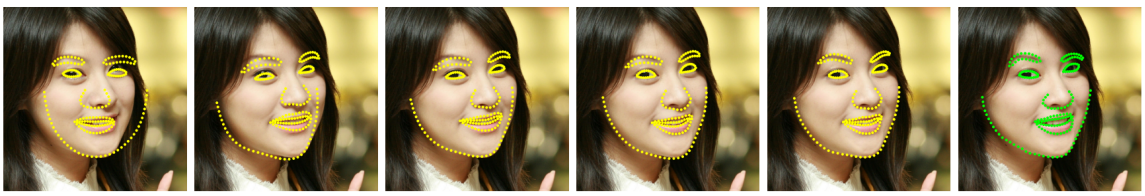
Kazemi et al. in paper One Millisecond Face Alignment with an Ensemble of Regression Trees [47] proposed a method for key-point detection based on regression trees. The ensemble of regression trees directly estimates the position of face landmarks from a sparse subset of pixel intensities.

Cascade of Regressors

The proposed algorithm is based on the cascade of regressors, previously used in [48, 49]. Each regressor in the cascade predicts an update for the current shape estimate to improve its fit. The regressor makes a prediction based on pixel intensities values from image I. The initial shape is the mean shape computed from the training dataset, centred and scaled to the bounding box of a face detector. During training, each regression function takes three parameters as input (face image, initial shape estimate, and the targeted update step). The regressors are trained by gradient tree boosting algorithm with a sum of square error loss. After learning the first regression function, the input parameters are updated to provide data for the next regression function. This process is repeated until a sufficient accuracy of the updated shape is achieved.

Tree Based Regressors

Each node in the regression tree contains a function thresholding the difference between intensity values of two pixels. The pixels positions are defined in the coordinate frame of the mean shape. To achieve this, the image is warped to the mean shape based on the current shape estimate (4.2.3). To improve efficiency, only the locations of points are warped and only an approximation of warping is done by global similarity transform suggested in [48]. The pairs are chosen by randomly generating candidates and then choosing the best ones by minimising the sum of squared errors based on the residuals computed from the image. The training is described in more detail in [47]. The process of fitting the mean shape by the cascade is depicted in the following figure.



(a) $T = 0$ (b) $T = 1$ (c) $T = 2$ (d) $T = 3$ (e) $T = 10$ (f) Ground truth

Figure 12: Landmark estimates at different levels of the cascade [47]. Initialisation is done by mean shape, centred at Viola Jones face detector bounding box.

6.1.3 Improving Visual Features for Lip-reading

Lan et al. in [50] proposed methods to improve the selection of features important for visual speech recognition. The appearance model inherently models the speaker identity instead of the visual features. A z-score normalisation per speaker [51] and Hi-LDA [52] methods are used for feature improvement.

6.1.4 Per-speaker z-score Normalisation

The main idea is to minimise the distance between the feature spaces of different speakers. The effect on separation of two different visemes is illustrated in the following figure.

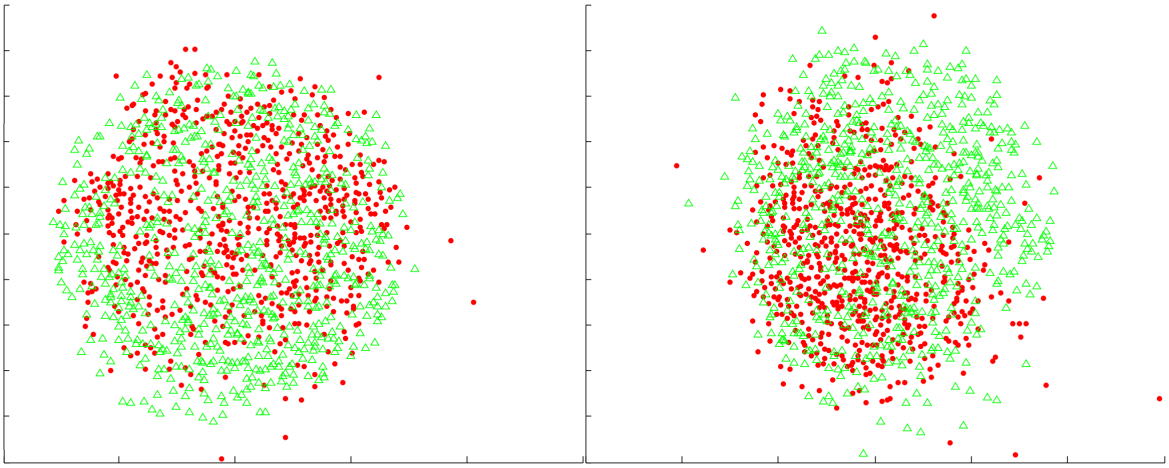


Figure 13: Sammon projection of AAM features for 12 different speakers for visemes /f/(red) and /u/(green). The graph on the left represents the global z-score normalisation vs, the picture on the right represents the per-speaker z-score representation [50].

The method improved the results of the global application, but not significantly enough to be separable.

Hi-LDA

The AAM and DCT features are static. They do not encode the dynamic features of visemes spanning over several frames of the video. The Hi-LDA method considers successive features over several frames as one feature vector. The illustration of constructing the Hi-LDA features is visible in the figure bellow.

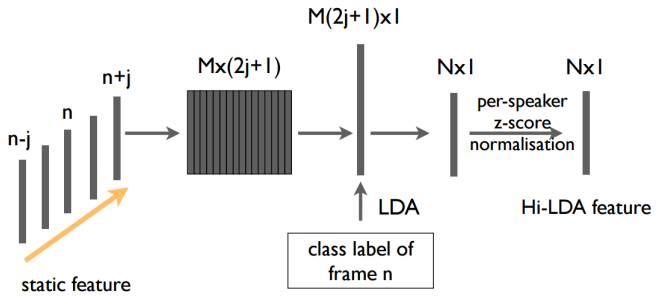


Figure 14: Hi-LDA features construction [50]. $2j+1$ features centred on the current frame are stacked to a hyper-vector. LDA and z-score normalisation are then applied on the hyper-vector.

The Linear Discriminative Analysis (LDA) learns a set of orthogonal projections that maximises the distances between classes during training. The distance within classes is minimised at the same time. The label can be phoneme, viseme, or HMM state sequence etc.

The authors applied the previously described method to improve the separability of the viseme classes. The results are clearly visible in the following figure. The Hi-LDA AAM features show significant improvement over the previous results. The authors also suggest that more work has to be focused on the back-end of the visual recogniser(language model) than on the feature extraction methods.

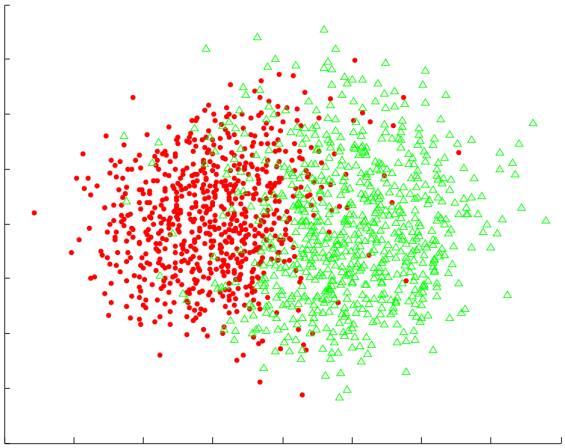


Figure 15: Sammon projection of Hi-LDA AAM features for 12 different speakers for visemes /f/ and /u/ [50].

6.1.5 VGG

Very Deep Convolutional Networks for Large-scale Image Recognition [53] denoted as "VGG" is a set of deep convolutional neural network (CNN) implementations. The authors focused on making an accurate CNN (5.2.2) which will provide state-of-the-art results in tasks of classification and localisation. The networks are also applicable to the task of image classification, where they provide excellent results. The final release contains two of the best performing models developed by the "VGG" team. The networks are provided as network specification for Caffe framework (5.4.1) with pre-trained weights.

VGG Architecture

The input of the network is 224 × 224 RGB image. The image is passed through the stack of convolutional layers with filter size 3 × 3 and fixed stride of 1. The padding of convolutional layers is set to preserve the spatial resolution of the input. Each stack of convolutional layers(but not every layer) is followed by a 2 × 2 max pooling (5.2.4) layer with stride 2 and no padding. The last three layers are fully connected layers (5.2.1) with sizes of 4096, 4096, and 1000. Activation functions used in all layers are ReLU (5.3). The architecture of "VGG16" can be seen in figure 17.

VGG training

The network was trained using procedures proposed by Krizhevsky et al. [18]. The dropout (5.2.3) was applied to the first two fully connected layers (0.5 probability). The deep networks were initialised by pre-trained weights from a smaller network trained separately. Training images of size 224 × 224 × 3 were randomly cropped from training data. To augment the data further, horizontal flipping and a random colour shift were applied [18].

Results

The "VGG" network was evaluated on ILSVRC-2012 dataset (7.1.3). The comparison was done against other state-of-the-art methods in the ILSVRC-2014 challenge. The network achieved second place with error rate 7.3 % which was further decreased

after submission to 6.8 % by using an ensemble of two models. The network is released as "VGG16" with 16 weight layers and "VGG19" with 19 weight layers.

6.1.6 ResNet

The authors of Deep Residual Learning for Image recognition [54] proposed an innovative method of training very deep neural networks. The proposed network was tested with the maximum of 152 weight layers, which was implemented with lower complexity than 8× smaller "VGG19" (6.1.5). The network architecture implements residual learning block to overcome the degradation of training accuracy.

Residual Learning

Instead of directly fitting the desired underlying mapping by stacking layers, the authors proposed a method of fitting only the residual mapping. The desired mapping $\mathcal{H}(\mathbf{x})$, the stacked non-linear layers fit residual mapping $\mathcal{F}(\mathbf{x}) := \mathcal{H}(\mathbf{x}) - \mathbf{x}$. The original mapping is then recast as $\mathcal{F}(\mathbf{x}) + \mathbf{x}$. The authors hypothesised that is easier to optimise the residual mapping then the original one. The realisation of the recast mapping in a feed-forward network is shown in fig 16.

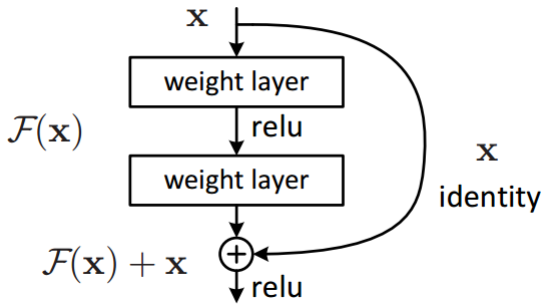


Figure 16: Residual learning: building block [54].

ResNet Architecture

Two models of ImageNet [55] were implemented for the purpose of testing the effect of residual learning. Both architectures are depicted in figure 17. The architecture was inspired by stacked convolutions of "VGG" network. The convolutions then follow two simple rules (1) for the same output feature map, the layer has the same number of

filters; (2) if the feature map size is halved, the number of filters is doubled so the time complexity of the layer is preserved [54].

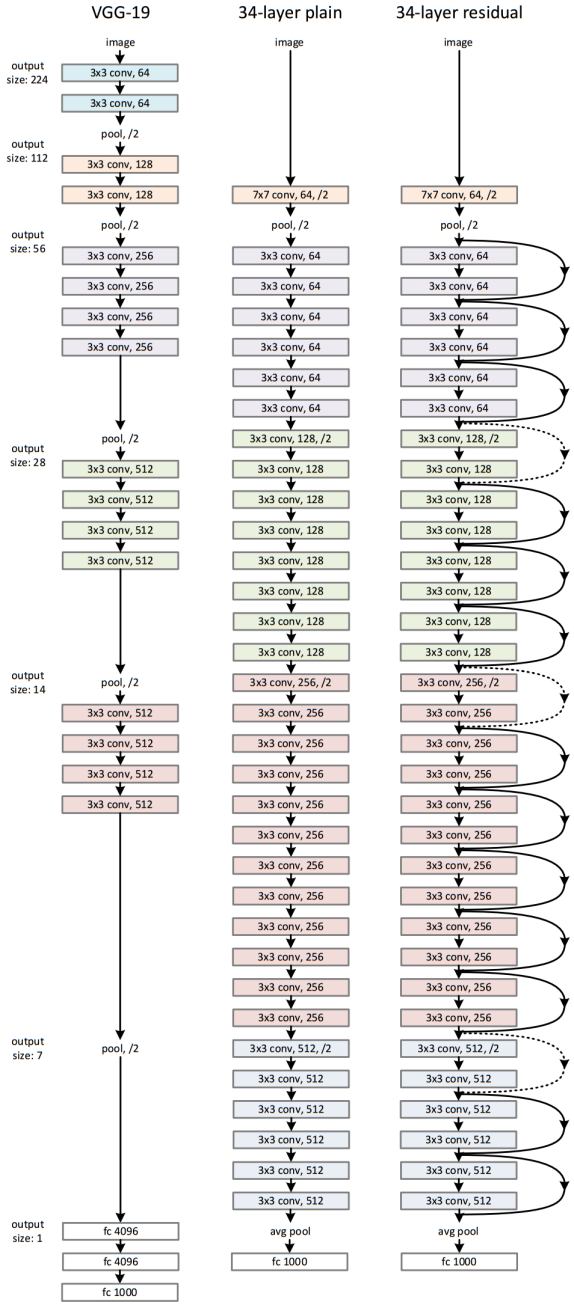


Figure 17: VGG16 and ResNet network architectures [54].

6.2 Visual Speech Recognition

The task of visual speech recognition is based on analysis of the extracted visual features. This process is based on video sequences processing. Extracted features are

usually analysed by some form of classification to provide the text of the actual speech captured in the video sequences. This chapter contains a review of current methods used for the purpose of lipreading. The current state-of-the-art approaches are using mainly complex end-to-end systems based on neural networks which take the video sequences as an input and provide output in the form of characters or words. The other methods mentioned in this chapter use features extracted by statistical model analysis and landmark detection to provide information for following speech recognition by classification. The methods also utilise audio signals of the speech to improve the training process of the recognition system which can then be used with video input only.

6.2.1 View Independent Computer Lip-reading

La et al. in paper [56] developed a view independent lipreading system based on AAM (4.4) features. The LiLIR dataset (7.2.1) is used to determine the visual speech recognition dependence on the speaker's head angle versus a camera. The paper builds on the previously obtained result with Hi-LDA AAM features (6.1.4). The system is based on a set of HMMs built and manipulated by HTK [57]. Fourteen HMMs were trained as viseme level models. Two systems were trained for comparison. The first system was separately trained and tested for each of the 5 angles provided by the dataset. From the results of the first system, the 30° angle was chosen as the best for visual speech recognition. The second system was trained on the angle of 30° and tested on other angles. Both systems were trained with different visual features for comparison of feature dependence on different angles. The features were *cat* (concatenated shape and appearance), *csam* (combined shape and appearance 4.2.2), *hilda*, *shape* (shape model parameters), *app* (appearance model parameters), and their second derivatives denoted as $\_\Delta\Delta$. The results comparing both systems and their performance in different angles with different features are depicted in the following figure.

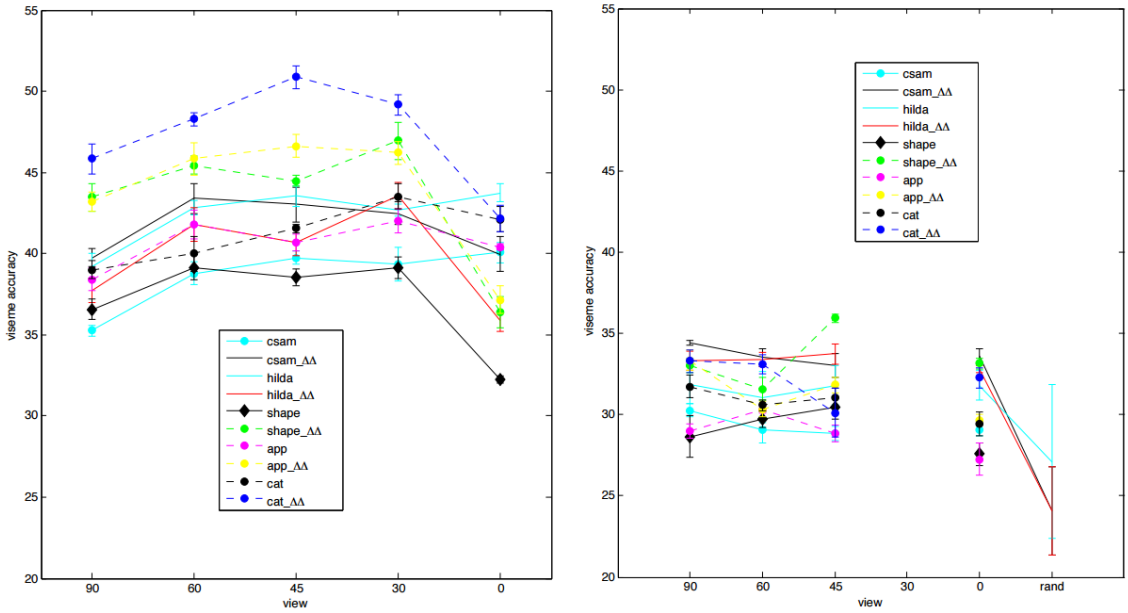


Figure 18: Results of view dependent (left) and view independent (right) systems with different features used. The x-axis shows the view angle, the y-axis the percent correct viseme accuracy [56]. The left graph also includes viseme accuracy on random feature vectors denoted as **rand** on the x-axis.

The paper also provides the results of feature mapping to increase the correlation between two sets of features from different views. A rigid regression is applied to map the features of one view to another one. This increased the performance of the proposed system.

6.2.2 Adaptive Multimodal Fusion by Uncertainty Compensation

Papandreou et al. in [6] proposed a method for audio-visual speech recognition based on AMM (4.4) model. The paper deals with the means of combining multiple modalities to achieve better results in the recognition task. The audio and the video streams are combined by using Multi-stream Hidden Markov models (HMM). The AAM is used as a face tracker and for extracting visual speech features. The novel approach is in using a cascade of two AAMs. The first AAM models the whole face and can reliably track the speaker in long videos. The second ROI-AAM (region of interest) is used for the bottom part of the speaker's face, which contains the most important speech-related visual information [5]. The first AAM is also used for the initial fit of the second AAM because the region is too small to be reliably tracked by

the second AAM alone. The feature vector is then composed of the second AAM control parameters and their uncertainty estimates. Uncertainty estimates are calculated as Gaussian distribution with covariance matrix $\mathbf{\Sigma}_{\tilde{\mathbf{p}}}$.

The Uncertainty Estimation

The shape $\mathbf{s}$ is generated by deviating the mean shape $\mathbf{s}_0$ by letting it lie in n-dimensional subspace:

$$\mathbf{s} = \mathbf{s}_0 + \sum_{i=1}^{n} p_i \mathbf{s}_i. \tag{6.5}$$

The mapping defined by this deformation is then $\mathbf{z} \mapsto \mathbf{W}(\mathbf{z}; \mathbf{p})$, where $\mathbf{z}$ represents any point inside the mean shape. Corresponding texture in the image frame I registered with the mean shape can be modelled as a weighted sum of m eigen-textures A_i:

$$I(\mathbf{W}(\mathbf{z}; \mathbf{p})) = A_0(\mathbf{z}) + \sum_{i}^{m} \lambda_i A_i(\mathbf{z}), \tag{6.6}$$

where A_0 is the mean face texture. The mean values are learned during training of the AAM model (4.4). The fitting amounts of parameters $\tilde{\mathbf{p}} \equiv \{\mathbf{p}, \lambda\}$ in each image I, which fit the model to the object and minimise the penalised error functional are defined in the following equation:

$$f(\tilde{\mathbf{p}}) = \frac{\kappa}{2\sigma^2} ||E(\tilde{\mathbf{p}})||^2 + Q(\tilde{\mathbf{p}}), \tag{6.7}$$

where $E(\tilde{\mathbf{p}}) \equiv I(\mathbf{W}(\mathbf{z}; \mathbf{p})) - A_0 - \sum_i^m \lambda_i A_i$ is the model's texture reconstruction error image, σ^2 is the reconstruction error variance, $Q(\tilde{\mathbf{p}}) = \frac{1}{2}(\tilde{\mathbf{p}} - \tilde{\mathbf{p}}_0)^T \mathbf{\Sigma}_{\tilde{\mathbf{p}},0}^{-1} (\tilde{\mathbf{p}} - \tilde{\mathbf{p}}_0)$ is quadratic penalty corresponding to a Gaussian coefficient prior with the mean $\tilde{\mathbf{p}}_0$ and covariance matrix $\mathbf{\Sigma}_{\tilde{\mathbf{p}},0}$, κ is a positive parameter adjusting the share of the prior and reconstruction terms in the fitting criterion [6].

Audio-visual Speech Recognition

To address the problem of visual features dependence on a speaker, the authors allowed speaker dependent A_0 mean texture and s_0 mean shape in the face tracking AAM. This allowed the model to focus on the speech features and not on visual identity features. The speaker dependent mean values are then subtracted before analysing the mouth region with ROI-AMM model in the training phase.

The audio features correspond to log-filter energies of a Mel-scale filter-bank [58]. The uncertainty is also applied to the audio features.

The combination of the audio a the video models is done by Multi-stream Hidden Markov models [59]. The uncertainty principle is also applied to the combination of the two models, which is extensively discussed in the original paper [6].

6.2.3 LSTM Lipreading

The authors of Lipreading with long short-term memory [9] proposed a neural network architecture based on LSTM units (5.2.5). The network was trained using the GRID dataset (7.2.6). The LSTM lipreader was composed of one feed-forward layer followed by two recurrent LSTM layers (128 LSTM cells each), and a softmax layer (5.2.6) with 51 units corresponding to the number of different words in the dataset. The results were compared to the SVM-based classifier based on two different feature extraction method (HoG [4], and Eigenlips [60]). The method outperformed both classifiers and the total accuracy of this method was 82%.

6.2.4 Lip Reading in the Wild

Chung et al. in [61] proposed a neural network architecture and a pipeline for large-scale data collection from the TV broadcast. The LRW (7.2.7) dataset was created using this pipeline and it was used to train the proposed networks.

Network architectures

The base model relies on VGG-M model [62]. It is a modified version of the VGG (6.1.5) network. The VGG-M architecture is depicted in fig 19 along with the developed network architectures. Four different network architectures were developed by Chung et al. [61]. The models differ in the way the input data T (T = 25 images for one second of video) are fed into the networks. These architectures were inspired by previous work on human action classification. The comparison of these architectures performance is possible due to shared VGG-M core. The architectures can be divided into two groups: 2D vs. 3D convolutions and Early Fusion vs Multiple Towers. A brief overview of the architectures follows:

- **3D Convolution with Early Fusion(EF-3):** This architecture is inspired by a network for human action recognition [63]. The structure is ordinary CNN (5.2.2), but instead of H × W × 3 input, it takes H × W × T × 3. Convolutions and pooling filters operate along all three dimensions.

- **3D Convolution with Multiple Towers(MT-3):** This model shares the basic design with **EF-3**, but there is no time-domain connectivity between frames before *conv2* layer. There are T = 25 towers with *conv1* layers (sharing weights), each takes one frame as input. The output of *pool1* is concatenated and 3D convolutions are then performed as in **EF-3**.

- **Early Fusion (EF):** The network input is a T-channel image, where each channel is a single grayscale frame. The architecture is similar to the VGG-M network. This method is inspired by a model proposed in [64]. Grayscale image representation was chosen for the purpose of reducing the number of parameter in *conv1* in comparison to adding 2 more colour channels.

- **Multiple Towers (MT):** The input layer consists of T = 25 towers with *conv1* layers (sharing weights), each takes one frame as input. The outputs of *pool1* are concatenated to H × W × 1200 matrix. The convolution of size 1 × 1 is performed to reduce the number of parameters in a subsequent layer to manageable levels.

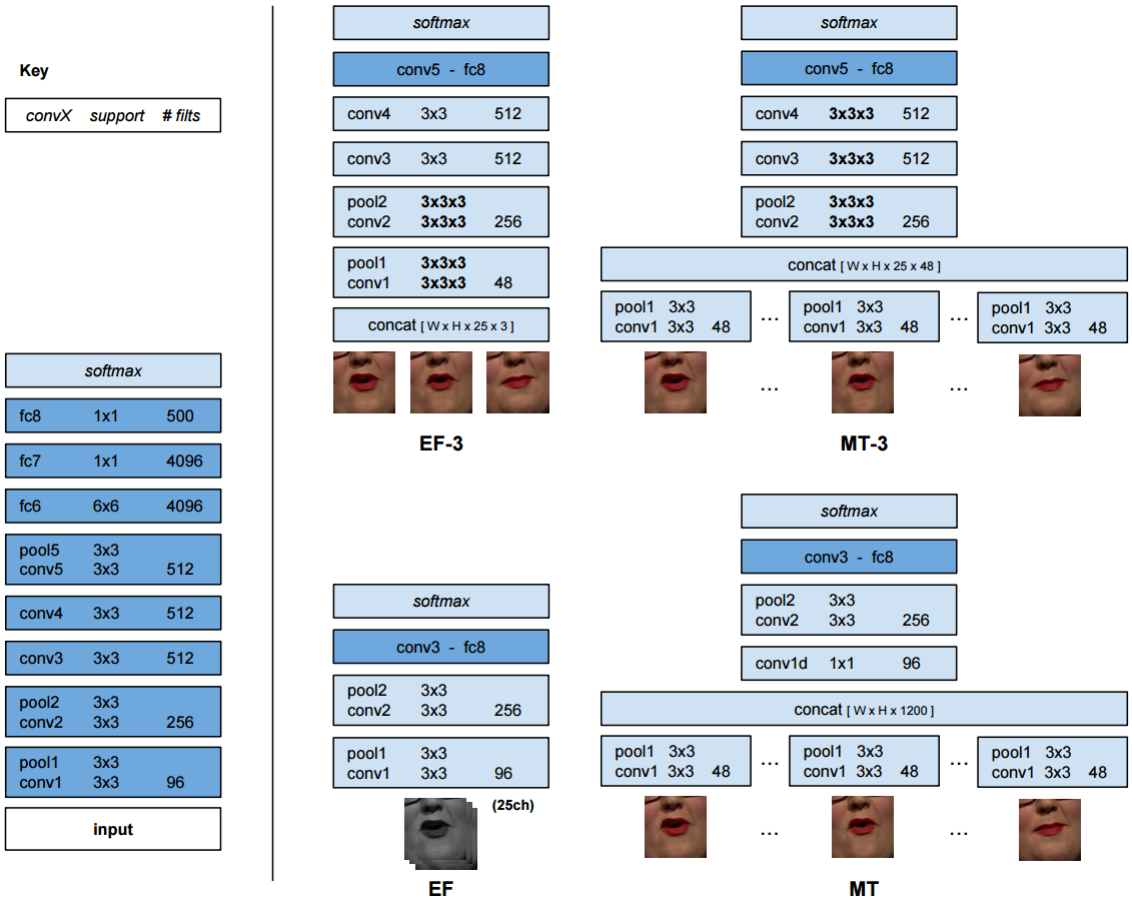


Figure 19: VGG-M and MT/EF network architectures [61]. **EF-3:** 3D Convolution with Early Fusion; **TF-3:** 3D Convolution with Multiple Towers; **EF:** Early Fusion; **MT:** Multiple Towers.

Results

Proposed networks were evaluated on OuluVS dataset 7.2.2. The best result was achieved by **MT** architecture. It achieved *top1* accuracy of 65.4 % on 333-word test set and *top-10* accuracy of 92.3 %. The difference in the results is given by ambiguities in lipreading, where different words can have very similar visual representation.

6.2.5 LipNet

LipNet [8] is a neural network architecture developed by Oxford University, Google and CIFAR. It is composed to map variable-length sequences of video frames to text sequences. The network is composed of Spatio-temporal Convolutions, Gated Recurrent Units and Connectionist Temporal Classification.

Spatiotemporal Convolution

Convolutional neural networks and their function as described in 5.2.2 convolve 2D data. Adding a time dimension is needed to process image sequences. This method used in [64, 63] adds a third dimension to the convolution.

Gated Recurrent Unit

Gated Recurrent Unit (GRU) [65] is a type of recurrent neural network (5.2.5) that adds cells and gates for propagating information over more time steps and learning to control the information flow. It is similar to the LSTM cell described in 5.2.5. The bidirectional variant of the GRU was used as introduced in [66].

Connectionist Temporal Classification

The Connectionist Temporal Classification (CTC) was proposed by Graves in [67]. It is a widely used method in modern speech recognition works because it eliminates the need for training data that aligns inputs to target outputs [68]. CTC computes probabilities of a sequence by marginalising over all sequences that are defined as equivalent to this sequence. This not only removes the need for alignments but also addresses the variable-length sequences.

LipNet Architecture

The figure 20 illustrates the network architecture. The first three layers are spatiotemporal convolutions (with channel-wise dropout (5.2.3) and spatial max-poling (5.2.4)). The features extracted are then processed by two bidirectional GRUs.The Bi-GRUs are important for the further aggregation of the STCNN output. A linear transformation is applied at each time step. The output is softmax (5.2.6) over a vocabulary with CTC blank augmentation and CTC loss. All layers use ReLU activation function (Eq. 5.3).

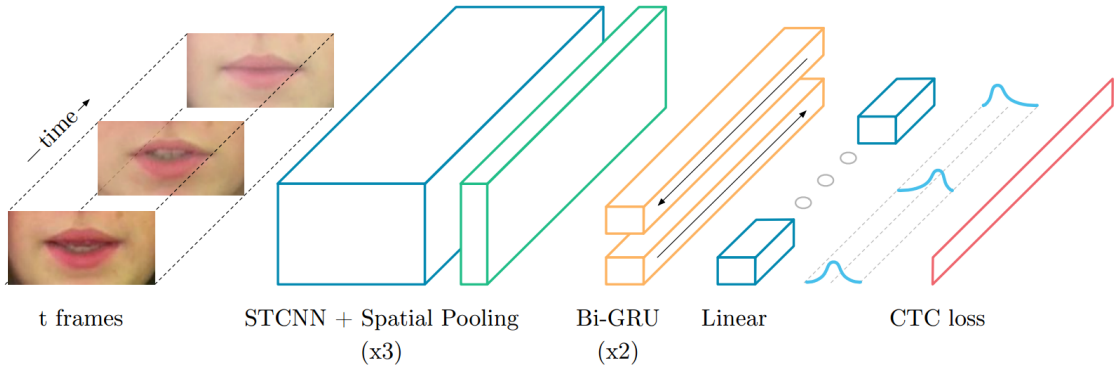


Figure 20: LipNet network architecture [8].

Results

LipNet was tested on the GRID dataset (7.2.6). The results show 2.8× better performance than the state-of-the-art word-level method [69]. The Word Error Rate (WER) is 4.8%. The future development suggests combination with audio signals and training on sentence level datasets.

6.2.6 WLAS network

Chung et al. in [7] created a network called "Watch, Listen, Attend and Spell". The network is trained to transcribe videos of mouth motion to characters. The authors also proposed a curriculum learning strategy to accelerate training a to reduce overfitting. For the purpose of training the network, LRS (7.2.8) dataset was created.

WLAS architecture

The network is designed to predict characters from videos of a talking face, with or without an audio signal. It is composed of parts performing different tasks to achieve the goal of character prediction. The Watch part is an image encoder consisting of a convolutional module that generates image features and a recurrent module encoder that creates fixed-dimensional state vectors from the features. The convolutional module is based on VGG network (6.1.5). The encoder is composed of LSTM (5.2.5) units. The Listen part is an audio encoder based on LSTM unit similar to the Watch part but without the convolutional module. Its inputs are 13-dimensional MFCC (Mel-frequency cepstrum) features in reverse time order. The Spell part is a LSTM based

transducer. The transducer uses dual attention mechanism consisting of attention vectors from video and audio features. The mechanism of attention vectors is described by Bahdanau et al. in [70]. Two independent attention mechanisms represent asynchronous inputs of video and audio features with different sampling frequencies. The probability output of the character is computed by multi-layer perceptron with softmax (5.2.6) activation function. The main advantage of the dual attention system is the ability to combine both features of audio and video, but the system is still usable even when one of the streams is missing. The full architecture can be seen in fig 21.

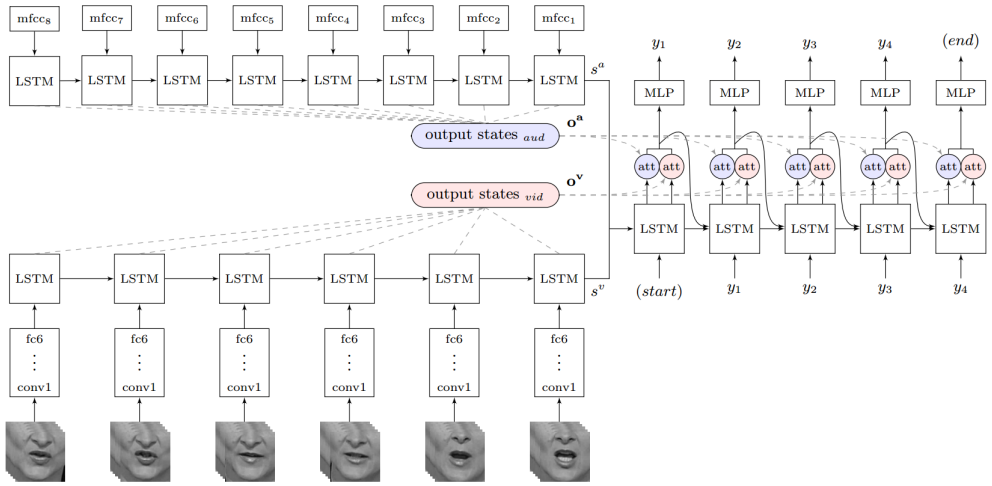


Figure 21: WLAS network architecture [7].

WLAS Training Procedure

The concept of training proposed by Chung et al. [7] is based on three steps. The first step is the Curriculum learning, where the baseline is the network trained from scratch with full sentences from LRS dataset (7.2.8). The LSTM units are known for slow convergence with very large time steps because the decoder does not initially know which features are relevant to extract from all the input steps. The authors started the training process on single word examples and slowly increased the sequence lengths as the network continued to train. This process severally reduced training time and overfitting. The next step is Scheduled sampling method proposed by Boendio et al. [71]. This method randomly samples from previous input instead of using ground truth during training. The third step is Multi-modal training. To prevent one of the two inputs to dominate over the other the inputs are randomly selected as (1) audio only, (2) lips only, and (3) audio and video. The network is trained with clear audio recording at first, but to improve the tolerance to noise an additive Gaussian noise is added during the training.

The input images are 120x120 sampled at 25 fps. The images only contain the speakers' mouth region. The convolution net takes 5-frame sliding windows input, moving 1 frame at a time. The MFCC features are calculated over 25ms window at 100Hz. The network was implemented and trained in Tensorflow (5.4.3) using SGD (5.3.2).

Results

The proposed network outperformed current state-of-the-art methods on GRID dataset (7.2.6), where it achieved 3.0% WER. The testing was also done on LRW dataset (7.2.7) where it achieved 23.8% WER, which was a significant improvement over other methods.

6.2.7 Transformer network

Afouras et al. published in [72] a follow-up study to the WLAS network. The paper introduces an audio-visual speech recognition system based on the transformer self attention network published in [73]. The authors compare two different approaches to training the output of the network and computing the loss function. The first type is CTC 6.2.5, the second type is Sequence to sequence(seq2seq) [74].

Sequence to sequence versus CTC

The Figure 22 illustrates the differences in CTC versus the sequence to sequence approach. The CTC model concatenates audio and video feature vectors and uses a stack of self-attention and feedforward blocks to analyse them. The output is represented by posterior probabilities for every input frame. The seq2seq model is implemented as separate attention mechanism. The output produces character probabilities.

Results

The authors of the paper [72] proposed that the seq2seq model is better for situations where the audio signal is missing and thus it works better for lip reading tasks. On the other hand, the CTC model is able to handle the interference and noise in a better way. The seq2seq model is also harder to train.

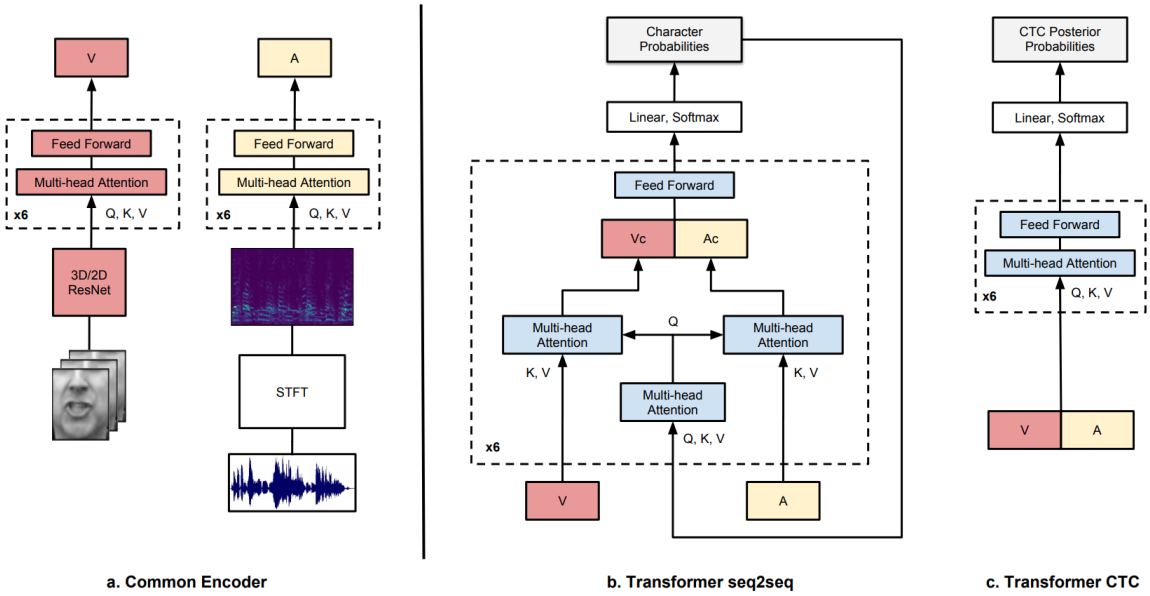


Figure 22: CTC vs Seq2Seq network architecture [72]. **Common Encoder:** Video features are extracted by ResNet 6.1.6 and audio features are spectrograms from Short Time Fourier Transform. **TM-seq2seq:** Transformer sequence to sequence. Every input channel is attended separately. Conetext vectors are concatenated channel-wise. K, V, and Q are Key, Value, and Query tensors of the multiheaded attention block. **TM-CTC:** Transformer CTC.

Chapter 7

Datasets

This chapter includes a short review of the most commonly used datasets. Datasets are divided into two groups: (1) the landmark and object detection datasets [75], and (2) the audio-visual speech recognition datasets. The landmark datasets are included for the purpose of geometric and visual feature extraction, they contain images of human faces in different conditions, orientations, and resolutions. The object detection dataset are often used for training neural networks for classification tasks. The audio-visual dataset are designed for the complex problem of speech recognition from audio, video, and their combination.

7.1 Landmark and Object Detection Datasets

The section provides a short review of currently used datasets for the task of feature extraction. The datasets are usually provided as a collection of annotated images. The annotations are in the form of landmark positions or object labels. There is no united approach to choosing the number of landmarks for annotation, so the provided annotations are based on the decision of the creators. The annotations are made by hand or by using a semi automated pipeline.

7.1.1 Helen

Helen [76] dataset was created to provide dataset with high resolution images with broad range of appearance variations, including pose, expression, illumination, and

occlusion. Images are originally from Flicker. Face images with smaller area than 500 pixels were filtered using a face detector at first pass. The second filtering was done manually. The dataset consists of 2330 images in variable conditions. Images are divided into training set (2000) and testing set (330). The dataset provides 194 manually annotated keypoints from Amazon Mechanical Turk.

7.1.2 LFPW

Labelled Face Parts in the Wild [77] dataset is one of the most commonly used dataset for landmark localisation benchmarks. It was created to provide more challenging dataset with variable pose and conditions. The original release contained 1432 images divided to 1132 training set images and 300 test images set. For the reasons of copyright only the url addresses of images are provided, that means not all images are available. The dataset of 35 keypoints was manually annotated by Amazon Mechanical Turk.

7.1.3 ILSVRC2012

The dataset is provided for the participants of the ImageNet Large Scale Visual Recognition Challenge [78]. The 2012 version contains $150,000$ photographs labelled with 1000 object categories. A random sample of $50,000$ data was released with annotations for the purpose of validation. The remaining images were available without labels during the challenge.

7.2 Audio-visual Speech Recognition Datasets

This section provides a short review of the state-of-the-art datasets containing audio-visual data for the purpose of speech recognition. The data are provided in the form of videos focused on the speaker, with included audio stream. The audio is usually provided as a separate file, which should be synchronised with the video stream. Annotations in the form of textual representation of the spoken word and a label for different speakers are also provided.

7.2.1 LiLIR

The dataset was created for the purpose of improving the visual features for lip-reading [50]. The dataset contains recordings of 12 speakers (7 male and 5 female). Each speaker recorded 200 sentences selected from Resource Management Corpus [79]. Each recording was done in one sitting to ensure constant illumination. The dataset contains two HD camera recordings from angles 0° and 90°. In addition three SD cameras were used to record the speech from angles 30°, 45°, and 60°. All cameras were sync locked during recording.

7.2.2 OuluVS

The OuluVS dataset [80] consist of 20 speakers captured with resolution 720 × 576 pixels at 25 fps. The distance of the speaker from the camera is constant 160 cm. The dataset contains utterances of ten everyday' greetings. Seventeen males and three females are included, nine of whom wear glasses.

7.2.3 AV-TIMIT

Audio-Visual TIMIT dataset [81] was developed at MIT for the purpose of creating an audio-visual speech recognition system. It contains read speech based on TIMIT sentences [82] and was recorded in controlled light and noise conditions. The total time is 4 hours of speech from 223 different speakers (117 male and 106 female). Video resolution is 720x480. The audio is provided both in video files and separately recorded WAV files. Audio annotations are created by automatic force-path alignment by word recognition system. Video is processed by face detector and mouth detector to locate the appropriate regions for speech recognition.

7.2.4 TCD-TIMIT

TCD-TIMIT dataset [83] was developed at Trinity College Dublin to provide an accessible audio-visual speech recognition dataset. The dataset is based on TIMIT sentences [82]. It contains audio and video recording (from two angles) of 62 speakers (3 professional lip-speakers and 59 volunteers). Each volunteer recorded 98 sentences and each professional lip-speakers recorded 377 sentences. The video is recorded in

FullHD resolution at 30fps. The dataset provides phoneme level labels with timing of start and end.

7.2.5 AVICAR

The Audio-Visual Speech Corpus in a Car Environment dataset [84] was recorded in car environment. The script for the corpus consists of four categories: isolated letters, isolated digits, phone numbers, and sentences. Speakers are evenly distributed (50 males and 50 females) and come from various language backgrounds (60 % native speakers). The dataset is recorded at five different noise conditions: idle, driving at 35 mph (windows opened and closed), and driving at 55mph (windows opened and closed). The total number of utterances is 59 000 recorded in eight audio channels and four video channels.

7.2.6 GRID

The GRID dataset [85] contains 1000 sentences from 34 different speakers. The total number of usable videos is 32746. The sentences spoken are specifically designed to be composed of six words (command(4), colour(4), preposition(4), letter(25), digit(10), and adverb(4)). The dataset contains 51 different words in total. Sentences have a fixed length of 3 seconds with 25fps. The speakers are 18 males and 16 females, and all are native English speakers with a range of English accents.

7.2.7 LRW

The Lip Reading in the Wild dataset [61] contains up to 1000 utterances of 500 different words. The size of the videos is exactly 29 frames. The timing is aligned so that the word appear in the middle of the video. Metadata are provided with exact frame of the start and the end of the word in the video. The dataset was created by an automatic pipeline from BBC News and some other shows. The pipeline is shown in figure 23

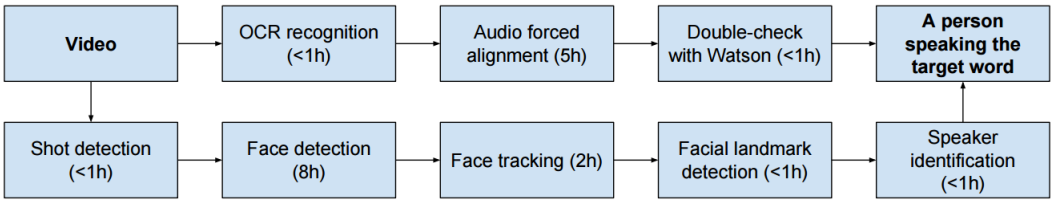


Figure 23: Pipeline used to generate LRW dataset [61]. Timings are for a one-hour video.

The programs were selected so the speakers are changing. The poses of speakers are changing depending on the number of speakers. The subtitles are extracted by OCR methods [86] because they are broadcast as bitmaps on BBC. The Penn Phonetics Lab Forced Aligner [87] was used to force-align the subtitle to the audio. The alignment is then checked against the IBM Watson Speech to Text. The face of the speaker is detected the same way as in LRS (7.2.8). The dataset provides training, validation, and testing data. The lexicon is created by selecting the 500 most occurring words with lengths from 5 to 10 characters.

7.2.8 LRS

The Lip Reading Sentences dataset was created for the purpose of training lipreading neural networks in [7]. An automatic pipeline was created to generate large-scale datasets for audiovisual speech recognition. The pipeline uses a variety of BBC programs form years 2010-2016. The authors collected thousand of hours of spoken sentences and phrases with corresponding face tracks. The boundaries of the shot in the video are detected at first by comparing colour histograms, then HOG-based [4] face detection is performed on every frame of the video. The face detections of the same person are then grouped together by KLT tracker [88]. Landmark detection is performed on extracted faces by ensemble of regression trees (6.1.2). The subtitles are not broadcast in sync with speakers lips, so to create audio annotation the Penn Phonetics Lab Forced Aligner [87] was used to force-align the subtitles to audio signal. The alignment is then filtered by IBM Watson Speech to Text service. The audio-video sync was achieved by using two stream network described in [89], the network is also used for speaker identification and to reject voice-over videos. The dataset contains thousands of different speakers, 807 375 words, and 118 116 utterances. There is also audio-only part of the dataset available, which contains different audio data, then the main dataset.

In the following paper [90] the dataset was extended and videos with different non-frontal angles of view were added. This makes the LRS2 dataset more challenging then the original LRS. Obtaining this dataset requires signing an agreement with BBC.

Summary

This Part II of the thesis presented an overview of the state-of-the-art methods required for developing experiments presented in the next Part III. The methods for statistical analysis (Chapter 4) of shape and texture are important for feature extraction. Chapter 5 presented the basic ideas and algorithms behind neural networks. A review of methods which are based on these chapters follows in Chapter 6. This chapter deals with feature extraction methods and visual speech recognition methods. The last Chapter 7 contains a brief overview of the datasets that are available for feature extraction and visual speech recognition.

Part III

Contribution to the state-of-the-art

Introduction

This part of the thesis is written in the first-person plural for the purpose of consistency. The research was carried out solely by the author of this thesis, unless otherwise specified, for example by citing the source. We will introduce our contribution to the field of automated lipreading in this part of the thesis. The contribution follows the goals set in the Chapter 3. The first chapter is composed of an analysis of different types of visual features used in the task of visual speech recognition. The features are divided into two groups by the way they are obtained. The first group deals with geometrical and appearance features. The second group deals with features obtained from neural networks. A set of deep features is introduced in the second chapter. We have named these features LipsID and they are produced by a neural network. The third chapters consists of experiments focused on implementing the LipsID features to the state-of-the-art networks currently used in the field of automated lipreading.

Chapter 8

Visual speech features analysis

We will discuss and compare various types of visual speech features which can be extracted from images of human face. The features can be divided into two groups. The first group is composed of geometric features and appearance features. Geometric features are various properties which are measurable on human face. Appearance features are based on analysis of the texture around and also between the lips. The second group are combined and deep features which are produced as results of computer analysis or byproduct of a trained neural network. We have obtained the information contained in this chapter by studying various publications about visual speech recognition like [1, 91]. The source are cited in the following text or in the Part II.

8.1 Geometric features

Geometric features can be divided into two groups. The first group contains properties that are unique for each speaker. These properties are usually obtained by keypoint detection methods described in Chapter 4. Detected keypoints are then used as features themselves or as the means to obtain other information like: position of eyes, width of the face, height of the face, distance from the upper lip to the lower lip, distance from the nose to the chin, etc. Some of these properties are static and they do not change during the speech. The properties that change in the process of creating the speech can be assigned to the second group. The second group then represents the dynamic properties that are changing in time. The usual recording speed of a common camera is 25 frames per second. This produces one picture (frame) in $40ms$. In this short time period a lot can change on the face of the speaking person. The dynamic

geometric features represent specific movements of individual speakers by measuring distances between important objects in the lips area. A good example is the distance between the upper lip and the lower lip, distance between corners of the mouth, and other distances that are changing in time during speech. These features have been historically used not only for the task of face identification but also in early attempts of visual speech recognition. Thanks to the development of better computers the focus has been placed on the texture of the face.

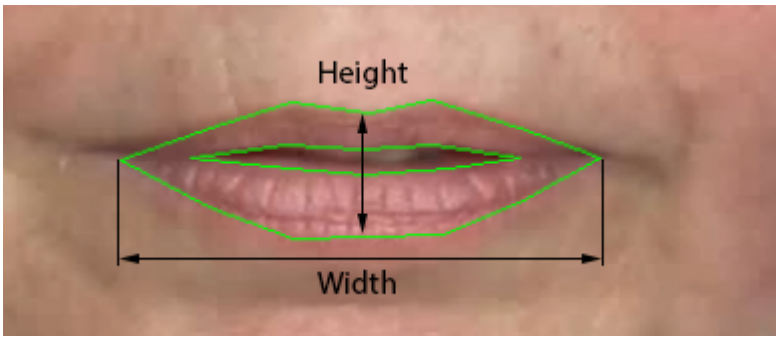


Figure 24: Geometric features of lips.

8.2 Appearance features

This section is dedicated to the analysis of the texture of a speaker's face. Appearance features can also be divided into two groups. The first group is a collection of texture specific properties like a colour of the skin, any irregularities like moles and freckles, a colour of eyes, and the colour of lips. There is also a global variable in the form of lighting. Some special cases like occlusion can also be part of appearance features, but they are not relevant in the topic of visual speech features. The second group of features is more focused on visual speech recognition. This group collects information about the visibility of certain features that are important in visual speech recognition. These features consists of visibility of upper and lower teeth, visibility of the tongue, and position of the tongue. There is also a possibility to analyse the visible volume of the upper and lower lip from the corresponding texture.

This type of features can be further analysed using statistical methods as in AAM (4.4). The model applies PCA (4.1.3) on the feature space of texture and shape of lips and uses it to get a set of controlling parameters. These parameters can then be used as a set of features for the purpose of visual speech recognition. As the model combines

both texture and shape of the ROI, the features are suitable for the task of lipreading as published in [52].

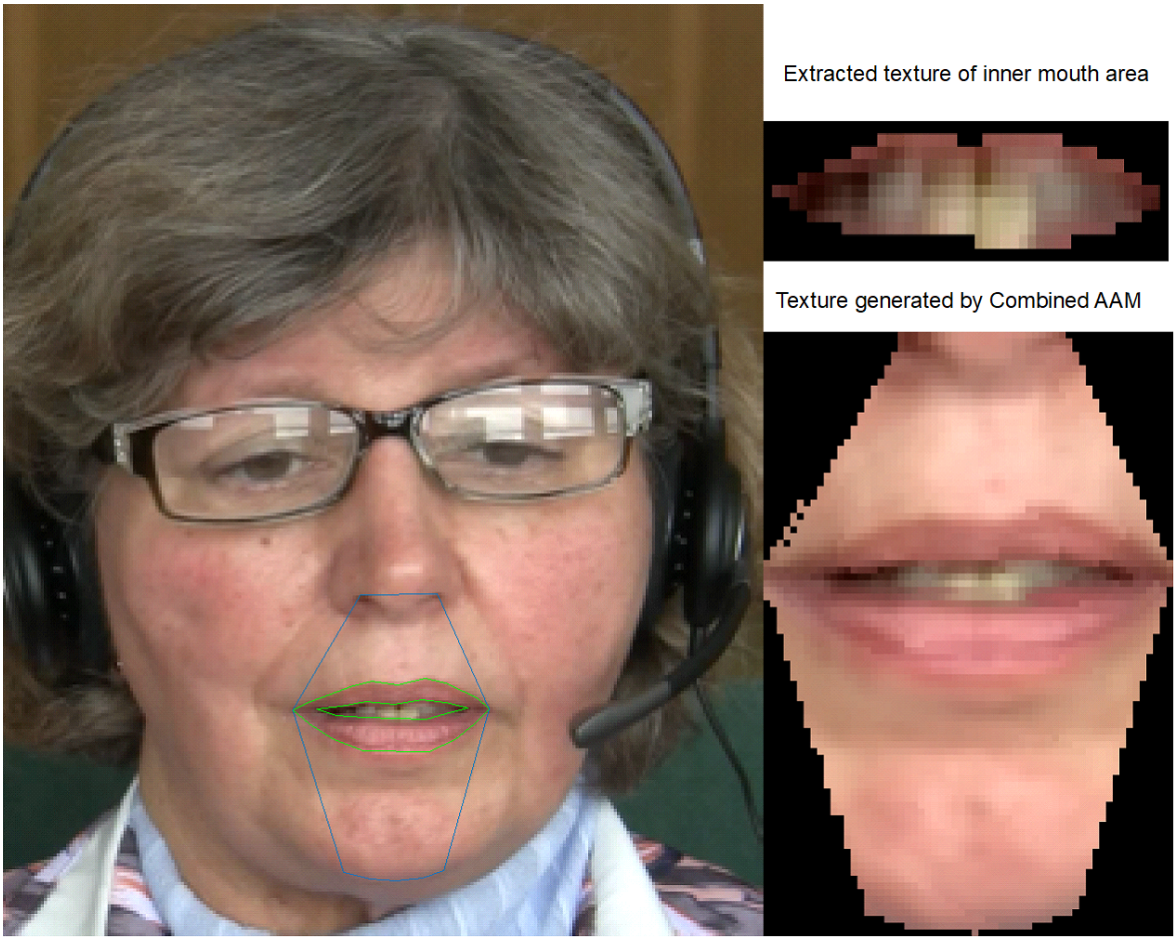


Figure 25: Appearance features. The figure shows shape detection from Combined AAM. The texture generated by the model is depicted in the lower right. The upper right image represents the inner area between lips.

8.3 Deep features

This type of features is a special case. It cannot be precisely describer in a manner of their meaning. Deep features are produced as a byproduct of a neural network trained in a specific task, for example, if the network is trained in a task of face recognition it processes the input image in a way that it extracts the features that are the most discriminating in the given task. The features extracted by using convolutional kernels (5.2.2) usually correspond to the features proposed by human experts. Kernel in the first layers mainly contain colour and edge detectors. The deeper the network, the

more specialised kernels are employed and their exact meaning is not obvious to a human observer. They usually contain sets of very specific feature detectors that are dependent on the training data. By connecting the output of a convolutional cascade with a fully connected layer (5.2.1) the features are getting connected with each other. The output of such fully connected layer can then be used as a feature vector. This is the way we have chosen to follow in this work and our features are based on the approach described above.

8.4 Feature use analysis

This section is dedicated to comparison of different types of features discussed above and analysis of their use in the task of lipreading. The task of recognising the visual speech can be divided into two directions. The first direction is the viseme approach where the algorithm tries to analyse the visual speech segments that creates words. The second direction is analysis of each word as whole. This part of the thesis considers the second direction of finding the boundaries between words. The task of recognising the word requires a good set of relevant visual features.

8.4.1 Height and width

The first most obvious feature is a combination of two geometric properties of the speaker's mouth. The lips are moving during the speech in such way that the ration between the height and width is constantly changing. The exact measurements of both variables can serve as features themselves. To properly create a sense of generalisation the features have to be made independent on the individual characteristics of the speaker and also on the recording device. This can be done in two steps. The first step is to resize the ROI to a constant size. The second step is to fit an ellipse in the area and to read the values of height and width of the ellipse. This approach is further developed by fitting the largest ellipse in the minimal bounding box [1]. The ellipse forces a symmetry to the shape and thus eliminate the differences in the shapes of lips between different speakers. The features are illustrated in Figure 26.

There is another way of obtaining the height and the width of the lips. I have made an AAM as a part of my masters thesis that can be fitted to the area of lips. The model then provides the positions of keypoint around the outer edge of lips. This keypoints can be used to obtain the information about the height and the width. This

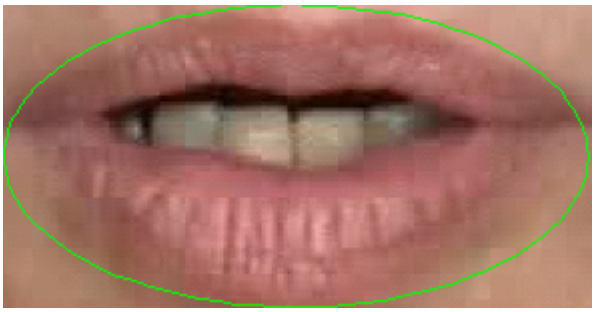


Figure 26: Lips ROI with fitted ellipse.

measurement is illustrated in the Figure 24.

8.4.2 Mutual information

The difference between two consecutive frames of the ROI can be expressed as temporal dependency. The information in the ROI is changing during speech as the frames capture the change between different phonemes(visemes). This information is important to differentiate between words. Since the appearance in the image is very dependent on lighting the visual speech methods use analysis in the frequency spectrum. The method used is Discrete wavelet transform(DWT) since it captures both frequency and location information. This approach produces the information about the dependency(similarity) between consecutive frames. The mutual information M between X and Y is computed in Eq. 8.1 [1].

$$M\left(X;Y\right)=\sum_{x}\sum_{y}p\left(x,y\right)\log\left(\frac{p\left(x,y\right)}{p\left(x\right)p\left(y\right)}\right) \tag{8.1}$$

8.4.3 Image quality

The quality measure is the opposite of the mutual information. This feature expresses the difference between two consecutive images. The amount of the distortion occurring during the production of a phoneme can be measured as a difference in the ROI. The authors of [1] proposed an universal image quality index taken from [92]. The measure Q is computed by Eq. 8.2 [1].

$$Q=\frac{4\sigma_{xy}\bar{x}\bar{y}}{\left(\sigma_{x}^{2}+\sigma_{y}^{2}\right)\left[\left(\bar{x}\right)^{2}+\left(\bar{y}\right)^{2}\right]} \tag{8.2}$$

where $Q\epsilon[-1,1]$, $\bar{x}$ is the estimate of mean value of x, $\bar{y}$ is the estimate of mean value of y, σ_x^2 and σ_y^2 are the estimates of variances of x and y, and σ_{xy}^2 is the estimate of covariance between x and y.

8.4.4 Appearance of tongue and teeth

The appearance of the tongue can be associated with specific phonemes. The movement of the tongue is important during vocalisation of certain sounds. This information can reduce the number of possible results of the classification and thus is considered important for the task of lipreading. The detection of the tongue is a difficult task because its only available characteristics is its red colour. The authors of [1] proposed a method based on the amount of red colour contained in the lips ROI. In the work [5] the author created a model of the inside area between the lips that can be better analysed for the presence of the tongue. The analysis can be seen in the Figure 27.

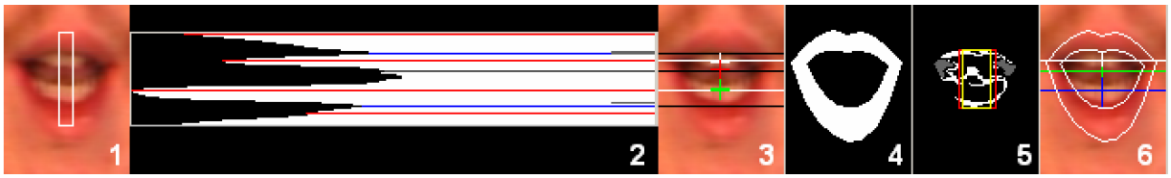


Figure 27: Analysis of insides of the mouth area [5]. 1 - selected area for analysis, 2 - sum of the individual rows in the selected area(values from grayscale image) with detected extrema, 3 - detected extrema projected into the image(green cross = estimated lower teeth position, red cross = estimated upper teeth position, white cross = new estimate of the upper teeth position), 4 - binary image of the lips from the model, 5 - thresholded image of the area between lips, selection of areas to search for the tongue (red) and the gap (yellow), 6 - detected centres and sizes of the objects (white = upper teeth, green = tongue, blue = lower teeth).

The importance of mutual position of lips, the tongue, and teeth is also emphasised in [5]. This includes cases of the lower lip touching upper teeth, the tongue between between teeth, etc. The percentage of tongue appearing in the gap between lips can also help to differentiate the phonemes spoken.

The appearance of teeth is important for detecting some phonemes - e.g. phoneme /s/. The authors of [1] propose a method based on the properties of teeth texture. Teeth usually have low saturation and high intensity value that separates them from the red surrounding of the mouth. This can be utilised for conversion of the image to

a different colour spaces (like CIELAB and CIELUV) where these properties allow us to separate the teeth from the lip region. The other available approach is to utilise the approach [5] illustrated in Figure 27 where the teeth and the tongue are extracted together.

8.4.5 DCT features

The Discrete Cosine Transform (DCT) is a method similar to the Discrete Fourier Transform. It has 8 different forms, but only four are commonly used. The one usually called DCT is DCT type II. The DCT is applied to the whole ROI to produce a set of coefficients. The coefficients represent low, middle, and high frequencies. Low frequencies represents illumination. High frequencies represents noise. The important information for feature extraction lies in middle frequencies. The DCT features are then produced by the selection of important coefficients [93]. The process of selecting the right coefficients is still an open task. The Fig. 28 illustrates the DCT transformation of face image.

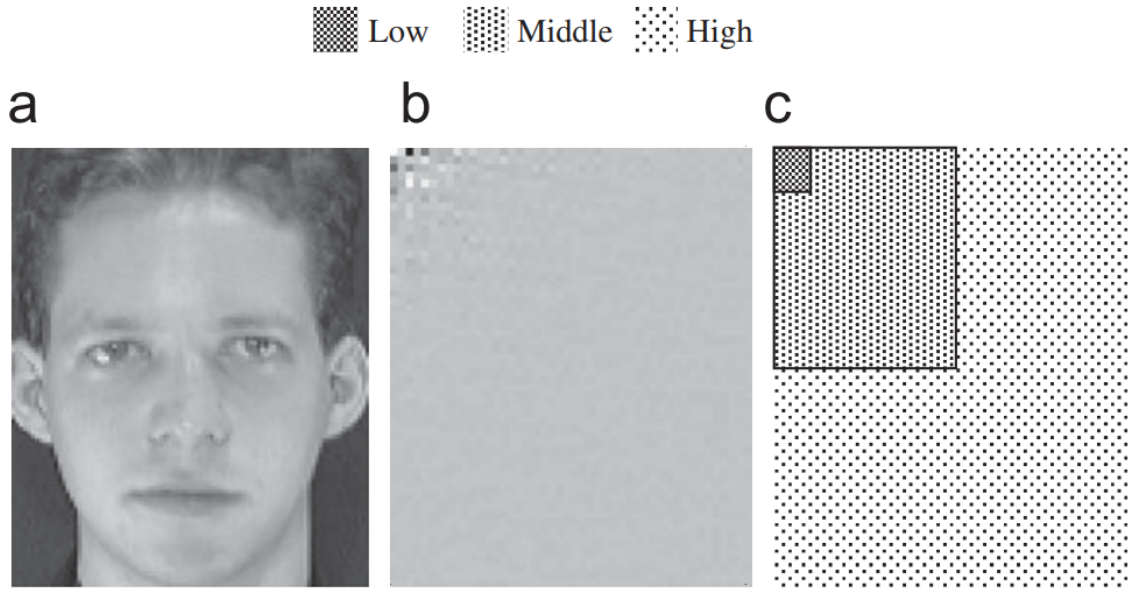


Figure 28: DCT applied to a face image [93].

8.5 UWB-HSCAVC dataset extension

During the development of our feature extraction method we have created a system for semiautomatic labelling of keypoints in the ROI of lips. The system was published in our paper Semi-automatic facial key-point dataset creation [94]. All the information and figures following in this section are from this paper. The neural networks require a lot of training data and the manual annotation of the facial keypoints is a demanding task. We have created a method based on the AMM 4.4 which can be trained from hundreds of annotated pictures and it can produce annotations by itself after training. The AAM has its own measure of fitting accuracy that can be used to filter annotations that are considered not good enough automatically.

We have created a labelling application in Matlab for the purpose of annotating the data in a consistent way by different annotators. The main GUI of the application can be seen in Fig. 29. The picture in the figure is from another recording and it is not part of the UWB-HSCAVC dataset. The model of shape is composed of 35 points. Each point is represented by (x, y) coordinates giving us a label composed of 70 floating-point numbers. The AAM model was trained from 350 manually labelled images. The model was then used to create training data for a neural network. The network with the structure described in Tab. 1 was trained in the task of regression. The network achieved sub pixel accuracy compared to the manual annotations and to the AAM detections. The main advantage over AAM is the speed where the network is roughly 10 times faster as is able to produce real time annotations for 25 fps videos. The comparison of results between the detection network and AAM can be seen in Fig. 30.

Table 1: Parameters of deep neural network architecture used to detect mouth region key-points. All strides were set to 1. BN means batch normalisation layer. MP max-poling with kernel size 2×2. Dropout probability was set to 0.5. [94]

conv1	**conv2**	**conv3**	**conv4**	**conv5**	**conv6**	**dense1**	**dense2**	**dense3**
64x3x3	64x3x3	128x3x3	128x3x3	256x3x3	256x3x3	4096	4096	70
	MP 2x2		MP 2x2			drop-out	drop-out	
	BN		BN		BN	BN	BN	

We have used the network to annotate the UWB-HSCAVC dataset for the purpose of having the exact coordinates for ROI extraction. This data were then used in the following experiments with LipsID.

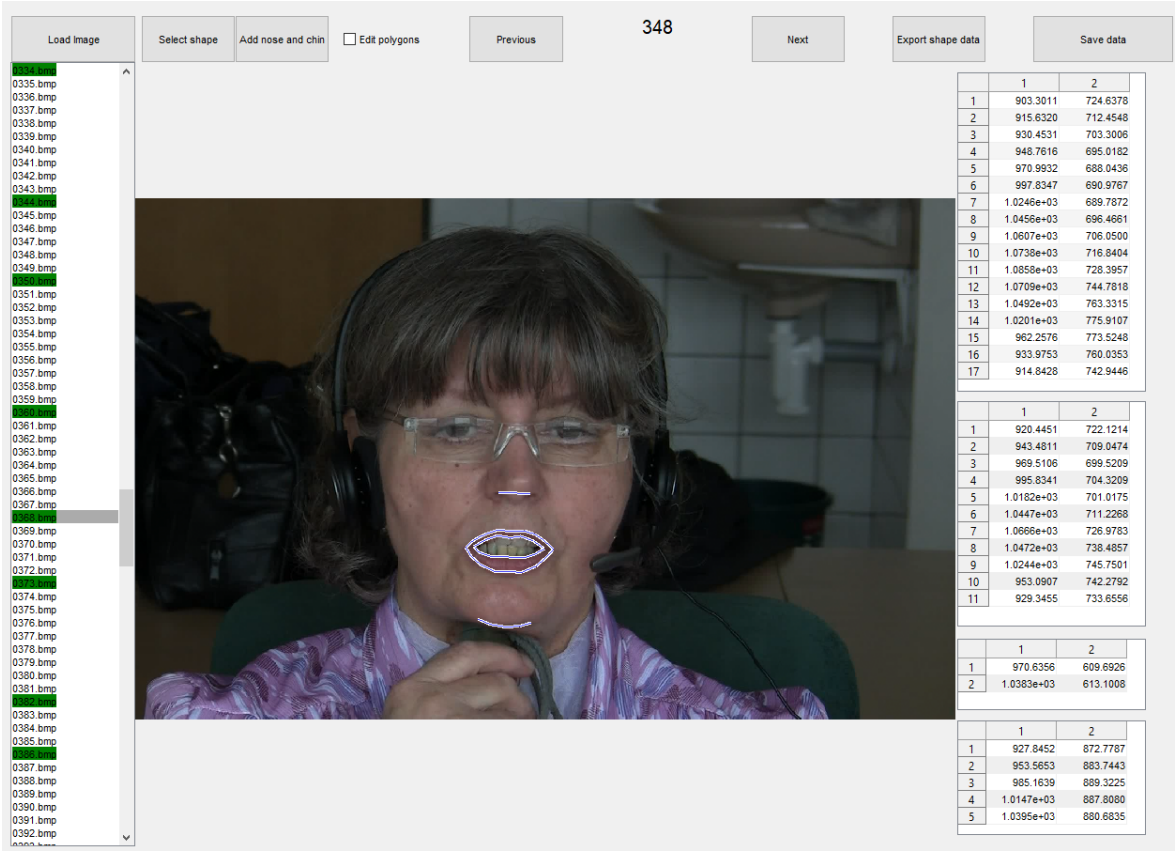


Figure 29: GUI of the keypoint labelling application.

Figure 30: Comparison of keypoint detection accuracy of the network(top row) and the AAM(bottom row). The last column represents misdetections [94].

Chapter 9

LipsID

New set of deep features which we named LipsID is presented in this chapter. The name was chosen because the features are based on a classification task with images of speaker's lips. The ideas behind the development of these features are presented. A method from audio speech recognition named i-vectors was the inspiration for our visual features. The process of training the features is explained with different implementations and different training data. As the usual input to the lipreading systems is a sequence of images (the lips area or the whole face), the LipsID features are firstly trained from images of lips and from images of whole faces in the later experiments.

9.1 Development of new deep visual features

During the analysis of state-of-the-art systems we have discovered that the main visual features used are based on the convolutional processing of the sequence of input images. The nature of deep features can be analysed by visualising convolutional kernels from the networks trained in the task of lipreading. The general idea is that the convolutional features are focused on the dynamic changes of the lips region during the image sequence. These changes are then important for the visual speech recognition in the same way as the frequency analysis for audio speech recognition.

We have also briefly reviewed the field of audio speech features and discover a method called i-vectors [95]. This method was originally developed for the purpose of speaker verification. The original paper develops a method for modelling the speaker (characteristics of the voice) and the channel (all other characteristics like the envi-

ronment, the recording device, etc.). The authors presented this research as an improvement for the task of speaker verification. The following research then proved that the i-vectors can be used to improve the recognition rate in the task of audio speech recognition [96]. This research has inspired us to design visual features based on the speaker's identity which can also capture a specific dynamic properties of individual speaker.

The process of developing the LipsID features started as a classification task. The goal was to try to do a speaker classification based on a single image of lips area. The initial experiments were done with our internal UWB-HSCAVC dataset [97]. The proof of concept was successful. We have then moved to a more complex task of classification based on sequences of consecutive frames. The idea was to create a network that can take the same input as the lipreading system (sequences of lips or face area) and provide a different set of features which will be speaker dependent.

9.2 LipsID using 3D convolutions

We have developed a network for the task of speaker classification based on 3D convolutions [63]. This research was published at SPECOM 2018 conference in the paper LipsID using 3D convolutions [98].

The data used for training the network came from the UWB-HSCAVC dataset [97]. The dataset was created at the University of West Bohemia. It is recorded in Czech language in laboratory conditions. Both audio and video recording were made and synchronised for the purpose of audio-visual speech recognition. We have only utilised the video data for my research. The video part is recorded in the resolution of 720×576 pixels at 25 fps.

We have used data of 64 speakers to train the LipsID network. The lips area was extracted from keypoints detection utilising Chehra tracker described in 6.1.1. Region obtained from the tracker were then resized to uniform resolution of 40×60 pixels. Sequences of visual speech were created to simulate the real input data of lipreading system. The sequence length was chosen to be 15 frames, but the network can be adapted to a different size. The structure of the data from the dataset is following: First the speaker recorded 50 sentences that are common for everyone. Then 150 sentence were recorded that were different for each speaker. The training data for the network were selected from 50 common sentence and the rest were used as development

Figure 31: Recording conditions of the UWB-HSCAVC dataset [97].

and testing data. This approach produced 20740 training sequences and 61709 testing sequences. Grayscale images were used for the purpose of the paper, but the network architecture can handle images in colour with little changes. The examples of training data are included in the Figure 32.

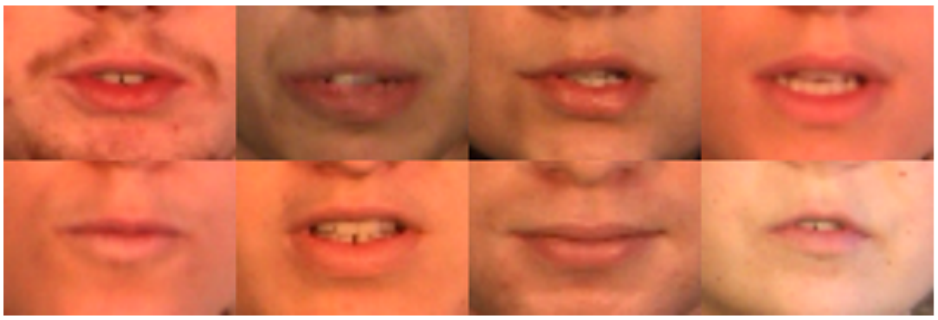


Figure 32: Example of the data used for training the networks.

Experiments

In the first experiment the network is classifying speakers based on a single image of the lips area. The topology for this network utilised 2D convolutions. The training data were extracted from the sequences that were prepared for the following experiments.

The training set was composed of 83327 images of 64 different speakers. The testing set contained 245 398 images. The topology of this network is described in the Table 2. It is a VGG 6.1.5 like structure.

Conv2D(64,3x3)	Conv2D(128,3x3)	Conv2D(256,3x3)	FC(4096)
Conv2D(64,3x3)	Conv2D(128,3x3)	Conv2D(256,3x3)	Dropout(0.5)
Batch Normalization	Batch Normalization	Batch Normalization	FC(4096)
Maxpooling(2x2)	Maxpooling(2x2)	Maxpooling(2x2)	FC(64,softmax)

Table 2: LipsID - single image topology, where Conv2D is a 2D convolution layer and FC is a fully connected layer.

The parameters for training this network were selected as follows: SGD optimiser with learning rate = 0.01, momentum = 0.9, weight decay = $1e^{-6}$, batch size 32 images, and 15 epochs of training. The layers used RELU 5.3 as the activation function. The last fully connected layer used softmax (5.2.6). The stride in the convolutions was set to one. The stride in maxpooling layers was set to two. The trained network achieved accuracy of 99.1% on the test set.

The second experiment was designed as the desired sequence classification of the speaker. The network takes the above mentioned sequences and classifies them according to the speaker label. The last but one fully connected layer then produces the LipsID feature vector of size 256. The training was done by the same SGD optimiser with cross-entropy loss as in the previous experiment. The network achieved accuracy 99.98% on training data and 99.29% on testing data. The topology is described in Table 3.

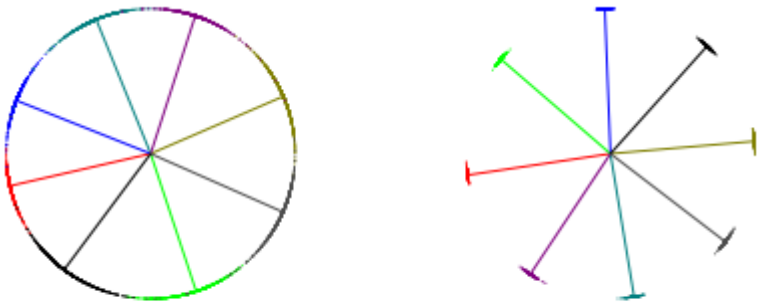
Conv3D(32,3x3x3,ReLU)	Conv3D(64,3x3x3,ReLU)	Conv3D(128,3x3x3,ReLU)	FC(256)
Conv3D(32,3x3x3,ReLU)	Conv3D(64,3x3x3,ReLU)	Conv3D(128,3x3x3,ReLU)	FC(64,softmax)
Batch Normalization	Batch Normalization	Batch Normalization	
MaxPool3D(3x3x2)	MaxPool3D(3x3x2)		

Table 3: LipsID - sequences topology, where Conv3D is a 3D convolution layer, MaxPool3D is three-dimensional maxpooling, and FC is a fully connected layer.

These experiments served as a first step and also as a proof of concept to creating the LipsID features. The network can be successfully trained to recognise the speaker based on the sequence of consecutive frames extracted from a video.

9.3 LipsID using ArcFace

Since the development of our original LipsID network some new method of training the neural networks appeared. There is a special type of training named by the authors ArcFace [99]. In their paper the authors propose a method that separates individual classes in a better way then the softmax. The proposed loss is named Additive Angular Margin Loss. The loss tries to make the gap between neighbours from different classes as big as possible. The difference between softmax and ArcFace is illustrated in the Fig.33. This brought an interesting novelty into my work. The idea of LipsID is to provide feature vector that is specific for each speaker. The method that ensures good separation of classes is very useful for the feature extraction because the analysis in the big space of hundreds or thousands of dimensions is very difficult. We have augmented our network with the proposed loss function to improve the potential of the LipsID features to be different for each individual speaker.



(a) Softmax (b) ArcFace

Figure 33: Softmax versus ArcFace [99].

9.4 Final form of LipsID features

The research in the field of visual speech recognition continuously evolves in a fast pace. New methods and improvements are emerging every few months. As we developed the LipsID features for lipreading, the state-of-the-art methods and networks changed significantly. We have started the development when the LipNet paper[8] was first published in 2016 and since then structures of networks used for lipreading have changed completely. The field is now split between pure lipreading systems and systems which utilise the video information together with the video. The second approach has

the advantage of having more training data from two different modalities. The learning process can use information about dependency between the two signals to improve recognition rates. Various types of training scenarios can prepare the system for the occasion where one of the inputs is missing like in WLAS6.2.6. We have decided to try both of these types to test our features.

The final form of the LipsID network depends on the system where its supposed to be implemented in. the output dimension of the feature layer has to be compatible with the dimension of the features in the lipreading system which they are supposed to be concatenated to. The individual changes to LipsID network are discussed in the chapter where the features are implemented in the state-of-the-art systems.

Chapter 10

Lipreading Experiments

Our proposed LipsID features needed to be tested with the state of the art lipreading systems. The general availability of the code needed to replicate the original experiments is low. We have contacted the authors of Transformer network 6.2.7 and asked them to provide the training code they have promised to release in their article [72]. They only provided trained models for evaluation and no training scripts or any other code to replicate their experiments. Since the newest networks could not be obtained, we had only two options, to use the older systems like LipNet 6.2.5 which code was available online to test the proposed features or to program the newest network ourselves. After careful research of the original papers [7, 72] we have decided to use the LipNet, because the information in the papers is not complete enough to precisely replicate their experiments and results. During the development of our experiments we have discovered a paper [100] which was dealing with audio visual speech recognition and provided their own implementation of WLAS-like network. Our final decision led to experiments with LipNet, and AVSR network provided from authors of [100].

10.1 The problem of feature normalisation

The scale of the output of the LipsID feature network depends on the activation function of the last layer. This can cause a problem in matching the common scale in case of concatenation of LipsID features whose internal features of a lipreading network. There are multiple solution to this problem. One solution is to match the activation function of the LipsID layer to the activation function of the layer after which the LipsID features will be concatenated to the lipreading network. This is possible in the

case of activation functions which minimum and maximum values are defined. Those include *tanh*, sigmoidal function, limited ReLU, etc. The problem arises in the case of ReLU that is limited only from one side. This problem can be solved by adding a constraint to the ReLU function which will clip its maximum value. Second solution is to implement an adjustment function that will scale the values of the two sets of features that are going to be concatenated to the same minimum and maximum value. Since the first solution implementation is less intrusive to both networks we have chosen to proceed with it. We have chosen the right place and activation function to produce two sets of features (LipsID + the original set the lip reading network is generating) which are compatible and can be concatenated together.

10.2 LipNet with LipsID

The implementation of LipsID features into the LipNet(6.2.5) was done in the following way. The LipsID Resnet network was connected after the Bi-directional Gated Recurrent Units (Bi-GRU). The architecture of LipNet can be found in Section 6.2.5. Since the output of Bi-GRUs has ReLU activation function we have done multiple experiments with the concatenation of LipsID features. The diagram of connection of the LipsID network to the LipNet can be seen in Fig. 34. As the LipNet was designed and tested with GRID(7.2.6) dataset we have replicated the experiment to achieve comparable results.

The training scenario was following:

- Pretraining LipNet with the original paramaters from the paper [8] with GRID.
- Pretraining LipsID with speakers from GRID dataset.
- Creating a new model as described in Fig. 34.
- Locking weights of layers preceding the concatenation of LipsID and LipNet.
- Traning the new model until the loss stabilizes.
- Unlocking all the weights and fine tuning the whole network.

We have left the same input to the LipNet as in the original paper [8] and trained LipsID network with the same input images that are used in LipNet. The CER and WER dependency on the number of training epochs visualisation is in Fig. 35.

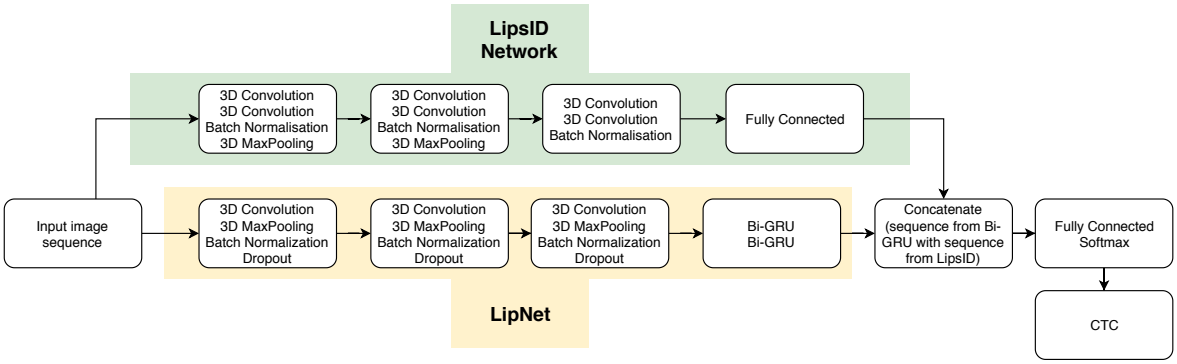


Figure 34: LipNet with LipsID features.

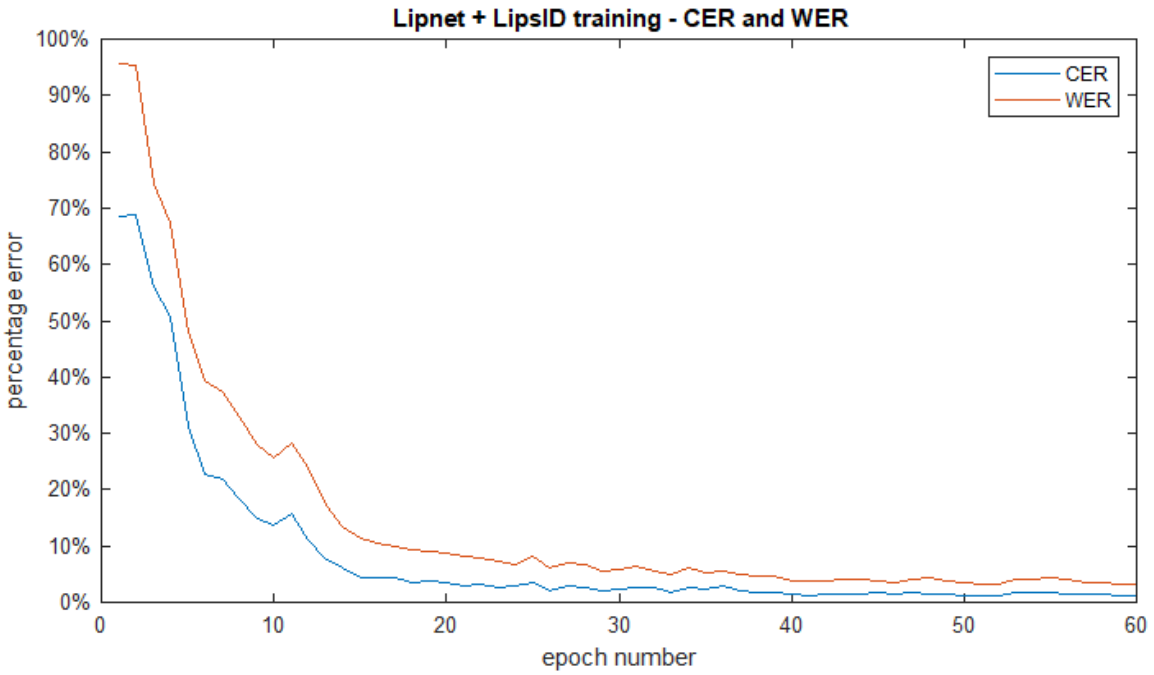


Figure 35: LipNet + LipsID CER and WER visualisation.

The parameters of the LipNet network are specified in the following text. The optimiser was originally Adam(5.3.2) but we have changed it to AdaBound(5.3.2). This change was done to make the training algorithm to converge faster to a stable solution. It has improved the total time of training from 3 days for 70 epochs to 1 day with the comparable results. The time improvement did not came from faster computation but from less epochs needed to reach the same value of loss. The parameters of the AdaBound are set as: learning rate = $1e^{-4}$ for the Adam-like part of the optimiser and final learning rate = 0.01 for the SGD(5.3.2) part. The structure of the network is in the Tab. 4. The visual representation of LipNet is shown in Fig. 20.

Conv3D($5 \times 5 \times 3$)	Conv3D($5 \times 5 \times 3$)	Conv3D($5 \times 5 \times 3$)	Bi-GRU(256)
BN	BN	BN	Bi-GRU(256)
CW-DO(0.5)	CW-DO(0.5)	CW-DO(0.5)	FC(28)
3D-MP($2 \times 2 \times 1$)	3D-MP($2 \times 2 \times 1$)	3D-MP($2 \times 2 \times 1$)	CTC

Table 4: The topology LipNet used for my experiments with LipsID. Conv3D are spatiotemporal convolutions (6.2.5). BN are batch normalisation layers (5.2.3). CW-DO are channel-wise dropouts(5.2.3). 3D-MP are three dimensional maxpooling layers (6.2.5). FC is a fully connected layer. CTC is Connections temporal classification (6.2.5). Activation functions are ReLU for convolutions and linear for the FC layer.

10.2.1 Results

Results compare the original accuracy reported in paper [8] with added LipsID features. The comparison in word error rate and character error rate is presented in Tab. 5. The absolute improvement in the recognition rate for overlapped speakers is 0.7% in CER and 1.5% in WER. The relative improvement is 37% in CER and 31% in WER. The absolute improvement in the recognition rate for unseen speakers is 1.2% in CER and 1.5% in WER. The relative improvement is 19% in CER and 13% in WER. This result show that the LipsID features bring significant improvement to the accuracy of lipreading by LipNet.

Scenario	**Unseen speakers**		**Overlapped speakers**	
Error type	**CER**	**WER**	**CER**	**WER**
LipNet[8]	6.4%	11.4%	1.9%	4.8%
LipNet + LipsID	5.2%	9.9%	**1.2%**	**3.3%**

Table 5: Comparison of LipNet vs LipNet with LipsID. WER is word error rate and CER is character error rate. These values are measured on a separated test data that are not part of the training set.

10.3 AVSR with LipsID

The implementation of LipsID features into the AVSR([100]) network was done as a second experiment. The previously tested LipNet represented a pure lipreading system. The AVSR network is the representative of the second approach where the system is trained with both audio and video signals. This network was trained with TCD-TIMIT dataset (7.2.4) and LRS2 dataset (7.2.8). The authors provide support for audio only, video only, and audio visual fusion inputs. We did some preliminary experiments with the code that is provided on GitHub from the authors and discovered that the training

of this type of the network is very time consuming. The training took several weeks until it started producing meaningful results with the prospect of months to get the desired accuracy in case of training the network with visual only input . After this discovery we have changed the training scenario to utilise the audio video fusion as an input which cut the time of training to three weeks. This was still very time demanding but manageable for few experiments. The source code for this network in its current form was released in April 2019 which did not give us much time to implement and test our training scenario. We started by training the network as it was provided to achieve a base line accuracy for comparison with the network augmented with LipsID. The first round of experiments was done with TCD-TIMIT (7.2.4) dataset. The network was already prepared to accept input from this dataset and even provided a script for data preparation, augmentation, and a list for splitting the dataset to training and testing parts.

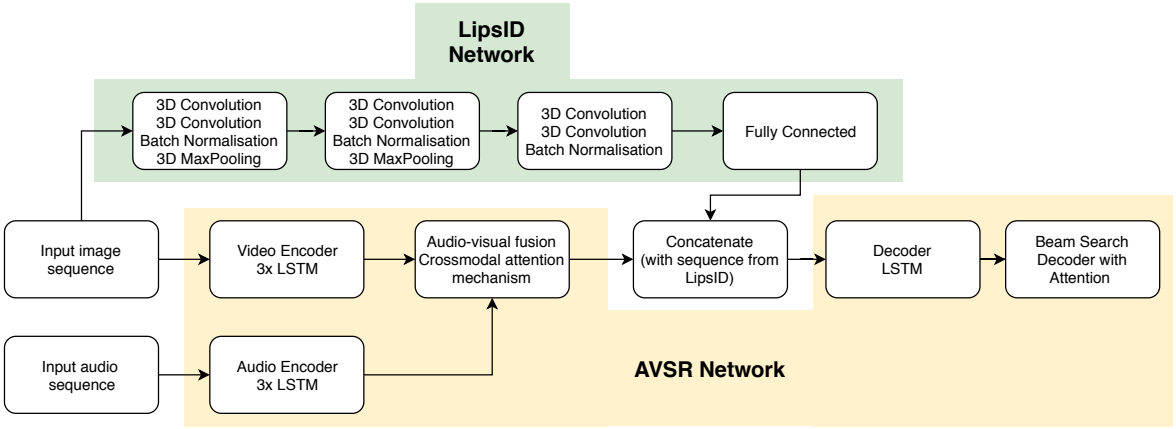


Figure 36: AVSR with LipsID structure.

10.3.1 Testing with TCD-TIMIT dataset

The TCD-TIMIT dataset is described in 7.2.4. This dataset is relatively compact and thus suitable for the first round of experiments. The average time for training the AVSR network was 21 days for 6100 epochs. This resulted in a recognition rate of 30.1% CER and 60.5% WER. After 6100 epochs the loss stopped to decrease. The original results published in [100] we 17.7% CER and 41.9% WER. We were not able to produce better results since not all of the parameters of the network used to produce the results in the original paper were disclosed by the authors. The authors acknowledge this problem in the discussion thread on GitHub where the code for their network is available. Also the optimiser they used for the training is presented as flawed there.

Because of these reasons we have decided to take the result we were able to achieve as the baseline to test our method. Since the training process is very time demanding we were not able to run multiple experiments. After establishing the baseline we have started the experiments with LipsID implementations into this network. The progress of CER and WER during training with LipsID is depicted in Fig. 37. The results are shown in the following Tab. 6.

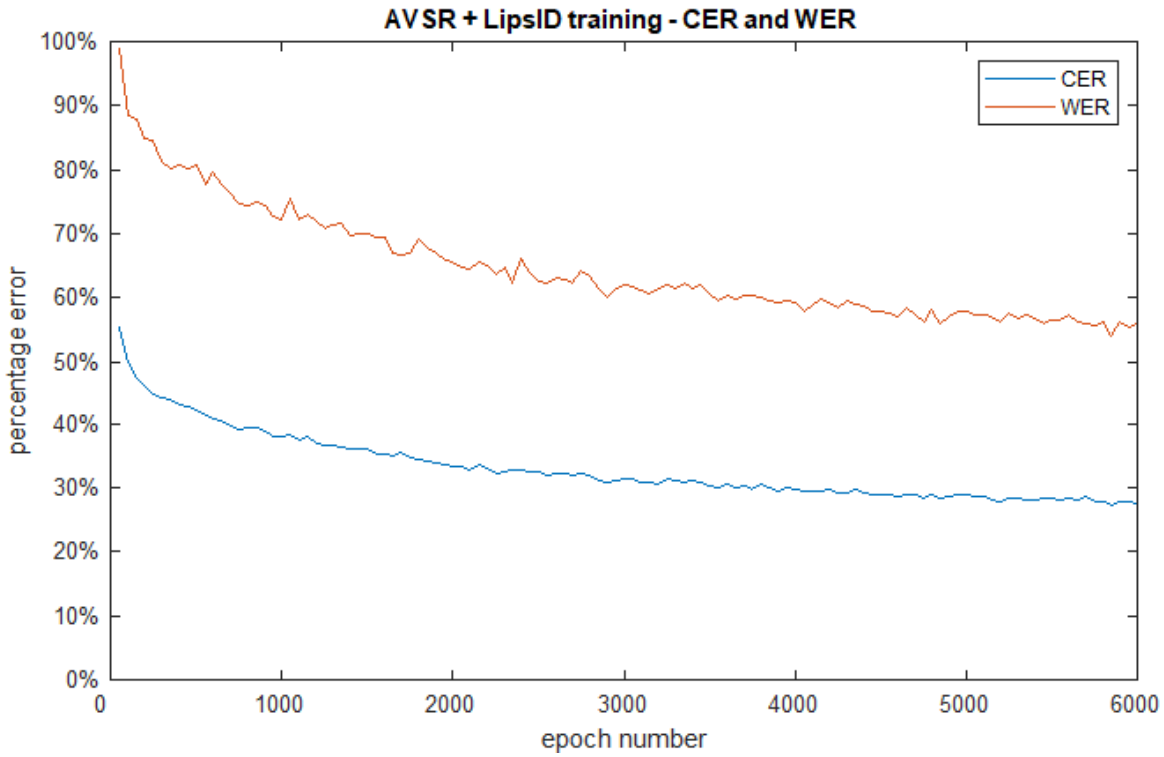


Figure 37: AVSR + LipsID CER and WER visualisation.

Error type	**WER**	**CER**
AVSR	30.1%	60.5%
AVSR + LipsID	**27.8%**	**55.7%**

Table 6: Comparison of AVSR network vs AVSR with LipsID. WER is word error rate and CER is character error rate. These values are measured on separated test data that are not part of the training set.

The results showed some improvement over the original network but it was not as good as the results with LipNet (5). The absolute improvement recorded is 2.3% in CER and 4.8% in WER. The relative improvement in CER is 7.6% and 7.9% in WER. The results of this experiment bring limited value because they cannot be directly compared to the original results published in [100]. The algorithm used for training the AVSR network suffered from two facts. First it was not the exact code used to

produce the results in [100] as the authors themselves admit in the documentation for the release. Second the training process was very long on our available hardware which limited the number and the length of the experiments with this implementation.

Summary

In this part we have presented the contribution to the state-of-the-art of lipreading. In the first chapter we analyse the various types of features that are considered important for visual speech recognition. The review is followed by brief evaluation of datasets useful for our research. The next chapter introduces our new feature vector named LipsID. It is obtained as an output of neural network trained in the task of classification based on the pictures of lips ROI or whole face of a speaker. The last chapter presents experiments that are designed to test if our proposed features bring any improvement in recognition rates in two selected state-of-the-art systems.

Part IV

Conclusion

Chapter 11

Conclusion

This part of the thesis is written in the first-person plural for the purpose of consistency. The research was carried out solely by the author of this thesis, unless otherwise specified, for example by citing the source. This part presents a brief summary of the thesis followed by a discussion about the fulfilment of the goals. The last section is dedicated to problems that were encountered during our experiments and to future research.

11.1 Thesis summary

We present a new feature extraction method in this thesis. Our features are based on the analysis of the state-of-the-art methods used to obtain visual features for lipreading. The features listed in Chapter 8 are generally used in systems that are not based on neural networks. These features needed to be updated because the state-of-the-art systems that were developed in recent years are generally based on neural networks. We have developed our own LipsID (Chapter 9) feature vector that is produced by a neural network and can be easily incorporated into existing system. We have tested the contribution of our method by implementing LipsID features into two state-of-the-art networks for visual speech recognition which improved the speech recognition rate. The results of our experiments are presented in the Chapter 10. The process of developing this thesis included a review of the state-of-the-art in the field of lipreading and feature extraction. The Part II is dedicated to describing the methods used to obtain visual features, methods used for lipreading, and review of datasets that are available for the research in those fields. Results and contributions of this work are further discussed in

the following section.

11.2 Dissertation goals

This section is dedicated to the evaluation of the goals set in the Chapter 3. Each goal definition is repeated and followed by the evaluation of the work done in this thesis and the discussion of the results.

11.2.1 Visual Speech Features Representation

The problem of representing visual features is still open. Currently, the WLAS network(6.2.6) uses pre-trained VGG networks(6.1.5) for evaluating and extracting features from video sequences. Other methods as mentioned above use key-points, AAM features, DCT features etc.

The aim of this goal is to analyse the state-of-the-art visual speech features which can be used for the task of lipreading. The output should be a review of features and how they work.

We have done an analysis of visual features used in the task of visual speech recognition in the Chapter 8. We present an overview of features used in the task of visual speech recognition. We have discovered that these features are usually extracted from the region of speaker's face. The features represent various characteristics of the speaker, dynamic properties of visual speech, and visibility of certain parts of the mouth ROI. There is also a group of features that are extracted by statistical methods which combine geometric and appearance features together to create a complex representation of the ROI. In this chapter we also present results of our paper where we have developed a system for semiautomatic dataset creation. This system is used to create annotations for facial keypoints. The research done in the Chapter 8 has led us to the realisation that a new set of visual features is needed for the systems based on the neural networks. The analysis also revealed an area which have not yet received much attention by researchers. This area is the adaption of the neural neural network speech recognition system based on the speaker identity. We have created an overview of visual features with the description of how they function. Our LipsID features filled a gap discovered by the analysis of the field. This allows us to consider this goal as **completed**.

11.2.2 New Feature Extraction Method Development

Development of a new set of visual features. Analysis and evaluation of the proposed visual features and their viability for the purpose of training a neural network for visual/audio-visual speech recognition. This is the main goal of the dissertation.

The aim of this goal is development of a method for extracting visual features suitable for training of visual speech recognition system as an additional input. These features should be extracted from the data used for the training of the above mentioned system.

We present a new set of visual features named LipsID in the Chapter 9. The idea of these features is based on the method of i-vectors used in the field of audio speech recognition. These features are based on the separation of personal characteristics of the speaker and characteristics of the channel. We have developed a similar method for extraction of visual features based on the speaker's identity from the input that is commonly used as an input to the lipreading system (ROI of lips or ROI of face). We present our neural network topology which is created for the task of classification of the speaker images. The last classification layer of the neural network is cut after training. The output of the penultimate layer is then used as our new feature vector. This network can be trained in a matter of days so it can be adapted to a specific dataset. We have developed new features based on the output of our neural network. These features can be implemented into state-of-the-art systems based on neural networks (these systems are further described in Section 6.2). This allows us to consider the goal as **completed**.

11.2.3 DNN Based Visual Speech Recognition

Due to the unpredictability of neural network training process and its dependence on thousands of variables, this step is very dependent on the proposed features in the previous step and the results will be an extension of the previous goal.

The aim of this goal is to prove the viability of the new feature set by implementing it into a state-of-the-art system for visual speech recognition. The development of the system is not part of this work. This goal is very dependent on the availability of the source code of systems published in recent papers.

We present the implementation of our LipsID features into state-of-the-art systems in Chapter 10. We have chosen LipNet 6.2.5 and AVSR network(based on 6.2.6). Implementations of these systems has been published online by the authors of the original papers [8, 100]. We have described the means of implementation of the LipsID network into the systems. Then we have showed that the systems trained with our features achieve higher accuracy in the form of character error rate and word error rate. Since the tests proved improvement in the recognition rate of these state-of-the-art systems we consider this goal as **completed**.

11.3 Future work

During the development of our experiments we have discovered several problems. The first and the biggest problem is the general availability of state-of-the-art systems for visual speech recognition. The papers usually do not publish the exact information about training processes of neural networks. This leads to a state where the results can not be precisely replicated. It is mostly problem in the case of very complex systems that are not easily implemented just from a picture of topography. In the future we would like to develop a lipreading end-to-end system based on neural networks which will be available on GitHub so everyone can replicate the experiments. The second problem we discovered is the availability of datasets that are in other languages than English. Neural networks need a vast amount of data for training which leads scientists to produce only datasets for the most used languages. We would like to create an audio-visual dataset for Czech language that would be suitable for lipreading experiments. The third problem is in the computer hardware. The hardware for processing the vast amounts of image data still needs some development. Training times of networks in our experiments were in weeks which severely limited the number of experiments we were able to accomplish. In the future it may be possible to use better hardware to accelerate the development of complex systems with lots of image data.

Bibliography

[1] A. Hassanat, *Visual Speech Recognition*, 06 2011, pp. 279–303.

[2] G. Potamianos, C. Neti, G. Iyengar, and E. Helmuth, "Large-vocabulary audio-visual speech recognition by machines and humans," in *Seventh European Conference on Speech Communication and Technology*, 2001.

[3] P. Viola and M. Jones, "Robust real-time object detection," in *International Journal of Computer Vision*, 2001.

[4] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 1. IEEE, 2005, pp. 886–893.

[5] P. Císař, "Using of lipreading as a supplement of speech recognition," Ph.D. dissertation, University of West Bohemia, 2007.

[6] G. Papandreou, A. Katsamanis, V. Pitsikalis, and P. Maragos, "Adaptive multimodal fusion by uncertainty compensation with application to audiovisual speech recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 17, no. 3, pp. 423–435, 2009.

[7] J. S. Chung, A. Senior, O. Vinyals, and A. Zisserman, "Lip reading sentences in the wild," *arXiv preprint arXiv:1611.05358*, 2016.

[8] Y. M. Assael, B. Shillingford, S. Whiteson, and N. de Freitas, "Lipnet: Sentence-level lipreading," *arXiv preprint arXiv:1611.01599*, 2016.

[9] M. Wand, J. Koutník, and J. Schmidhuber, "Lipreading with long short-term memory," in *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*. IEEE, 2016, pp. 6115–6119.

[10] J. He and H. Zhang, "Lipreading recognition based on svm and dtak," in *2010 4th International Conference on Bioinformatics and Biomedical Engineering*, June 2010, pp. 1–4.

[11] D. R. Reddy *et al.*, "Speech understanding systems: A summary of results of the five-year research effort," *Department of Computer Science. Camegie-Mell University, Pittsburgh, PA*, vol. 17, 1977.

[12] T. F. Cootes, C. J. Taylor *et al.*, "Statistical models of appearance for computer vision," University of Manchester, Tech. Rep., 2004.

[13] C. Goodall, "Procrustes methods in the statistical analysis of shape," *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 285–339, 1991.

[14] H. Hotelling, "Analysis of a complex of statistical variables into principal components." *Journal of Educational Psychology*, vol. 24, p. 417–441 and 498–520, 1933.

[15] B. Delaunay, "Sur la sphère vide." *Bull. Acad. Sci. URSS*, vol. 1934, no. 6, pp. 793–800, 1934.

[16] B. Van Ginneken, B. T. H. Romeny, and M. A. Viergever, "Computer-aided diagnosis in chest radiography: a survey," *IEEE Transactions on medical imaging*, vol. 20, no. 12, pp. 1228–1241, 2001.

[17] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[18] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105.

[19] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[20] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting." *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[21] A. Karpathy, *Cs231n: Convolutional neural networks for visual recognition*, 2016. [Online]. Available: http://cs231n.github.io/

[22] J. L. Elman, "Finding structure in time," *Cognitive science*, vol. 14, no. 2, pp. 179–211, 1990.

[23] M. Krause, "Not quite a book review: Christiansen's" infinite languages, finite minds" as an introduction to connectionist modeling, recursion, and language evolution," 2009. [Online]. Available: http://www.cogcrit.umn.edu/docs/Krause_09.shtml

[24] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[25] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Cognitive modeling*, vol. 5, no. 3, p. 1, 1988.

[26] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of Machine Learning Research*, vol. 12, no. Jul, pp. 2121–2159, 2011.

[27] S. Ruder, "An overview of gradient descent optimization algorithms," *CoRR*, vol. abs/1609.04747, 2016. [Online]. Available: http://arxiv.org/abs/1609.04747

[28] M. D. Zeiler, "Adadelta: an adaptive learning rate method," *arXiv preprint arXiv:1212.5701*, 2012.

[29] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[30] L. Luo, Y. Xiong, Y. Liu, and X. Sun, "Adaptive gradient methods with dynamic bound of learning rate," 2019.

[31] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," *arXiv preprint arXiv:1408.5093*, 2014.

[32] Theano Development Team, "Theano: A Python framework for fast computation of mathematical expressions," *arXiv e-prints*, vol. abs/1605.02688, May 2016. [Online]. Available: http://arxiv.org/abs/1605.02688

[33] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens,

B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: http://tensorflow.org/

[34] J. Dean, G. Corrado, R. Monga, K. Chen, M. Devin, M. Mao, A. Senior, P. Tucker, K. Yang, Q. V. Le *et al.*, "Large scale distributed deep networks," in *Advances in neural information processing systems*, 2012, pp. 1223–1231.

[35] R. Collobert, K. Kavukcuoglu, and C. Farabet, "Torch7: A matlab-like environment for machine learning," in *BigLearn, NIPS Workshop*, no. EPFL-CONF-192376, 2011.

[36] R. Ierusalimschy, L. H. De Figueiredo, and W. Celes Filho, "Lua-an extensible extension language," *Softw., Pract. Exper.*, vol. 26, no. 6, pp. 635–652, 1996.

[37] D. Yu, A. Eversole, M. Seltzer, K. Yao, O. Kuchaiev, Y. Zhang, F. Seide, Z. Huang, B. Guenter, H. Wang, J. Droppo, G. Zweig, C. Rossbach, J. Gao, A. Stolcke, J. Currey, M. Slaney, G. Chen, A. Agarwal, C. Basoglu, M. Padmilac, A. Kamenev, V. Ivanov, S. Cypher, H. Parthasarathi, B. Mitra, B. Peng, and X. Huang, "An introduction to computational networks and the computational network toolkit," Microsoft Technical Report MSR-TR-2014–112, Tech. Rep., October 2014. [Online]. Available: https://www.microsoft.com/en-us/research/publication/an-introduction-to-computational-networks-and-the-computational-network-toolkit/

[38] A. Asthana, S. Zafeiriou, S. Cheng, and M. Pantic, "Incremental face alignment in the wild," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1859–1866.

[39] L. Matthews, T. Ishikawa, and S. Baker, "The template update problem," *IEEE transactions on pattern analysis and machine intelligence*, vol. 26, no. 6, pp. 810–815, 2004.

[40] A. Levey and M. Lindenbaum, "Sequential karhunen-loeve basis extraction and its application to images," *IEEE Transactions on Image processing*, vol. 9, no. 8, pp. 1371–1374, 2000.

[41] X. Xiong and F. De la Torre, "Supervised descent method and its applications to face alignment," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2013, pp. 532–539.

[42] A. Asthana, S. Zafeiriou, S. Cheng, and M. Pantic, "Robust discriminative response map fitting with constrained local models," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 3444–3451.

[43] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[44] J. M. Saragih, S. Lucey, and J. F. Cohn, "Deformable model fitting by regularized landmark mean-shift," *International Journal of Computer Vision*, vol. 91, no. 2, pp. 200–215, 2011.

[45] A. E. Hoerl and R. W. Kennard, "Ridge regression: Biased estimation for nonorthogonal problems," *Technometrics*, vol. 12, no. 1, pp. 55–67, 1970.

[46] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen, *Classification and regression trees*. CRC press, 1984.

[47] V. Kazemi and J. Sullivan, "One millisecond face alignment with an ensemble of regression trees," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1867–1874.

[48] X. Cao, Y. Wei, F. Wen, and J. Sun, "Face alignment by explicit shape regression," *International Journal of Computer Vision*, vol. 107, no. 2, pp. 177–190, 2014.

[49] P. Dollár, P. Welinder, and P. Perona, "Cascaded pose regression," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. IEEE, 2010, pp. 1078–1085.

[50] Y. Lan, B.-J. Theobald, R. W. Harvey, E.-J. Ong, and R. Bowden, "Improving visual features for lip-reading." in *AVSP*, 2010, pp. 7–3.

[51] J. L. Newman and S. J. Cox, "Automatic visual-only language identification: A preliminary study," in *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*. IEEE, 2009, pp. 4345–4348.

[52] G. Potamianos, C. Neti, J. Luettin, and I. Matthews, "Audio-visual automatic speech recognition: An overview," *Issues in visual and audio-visual speech processing*, vol. 22, p. 23, 2004.

[53] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2014.

[54] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[55] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE, 2009, pp. 248–255.

[56] Y. Lan, B.-J. Theobald, and R. Harvey, "View independent computer lip-reading," in *Multimedia and Expo (ICME), 2012 IEEE International Conference on*. IEEE, 2012, pp. 432–437.

[57] S. Young, G. Evermann, M. Gales, T. Hain, D. Kershaw, X. Liu, G. Moore, J. Odell, D. Ollason, D. Povey *et al.*, "The htk book," *Cambridge university engineering department*, vol. 3, p. 175, 2002.

[58] L. Deng, J. Droppo, and A. Acero, "Dynamic compensation of hmm variances using the feature enhancement uncertainty computed from a parametric model of speech distortion," *IEEE Transactions on Speech and Audio Processing*, vol. 13, no. 3, pp. 412–421, 2005.

[59] S. Dupont and J. Luettin, "Audio-visual speech modeling for continuous speech recognition," *IEEE transactions on multimedia*, vol. 2, no. 3, pp. 141–151, 2000.

[60] C. Bregler and Y. Konig, "ldquo;eigenlips rdquo; for robust speech recognition," in *Acoustics, Speech, and Signal Processing, 1994. ICASSP-94., 1994 IEEE International Conference on*, vol. ii, Apr 1994, pp. II/669–II/672 vol.2.

[61] J. Chung and A. Zisserman, "Lip reading in the wild," in *Asian Conference on Computer Vision*, 2016.

[62] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, "Return of the devil in the details: Delving deep into convolutional nets," *arXiv preprint arXiv:1405.3531*, 2014.

[63] S. Ji, W. Xu, M. Yang, and K. Yu, "3d convolutional neural networks for human action recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 1, pp. 221–231, 2013.

[64] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2014, pp. 1725–1732.

[65] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.

[66] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional lstm and other neural network architectures," *Neural Networks*, vol. 18, no. 5, pp. 602–610, 2005.

[67] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of the 23rd international conference on Machine learning*. ACM, 2006, pp. 369–376.

[68] D. Amodei, R. Anubhai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, J. Chen, M. Chrzanowski, A. Coates, G. Diamos *et al.*, "Deep speech 2: End-to-end speech recognition in english and mandarin," *arXiv preprint arXiv:1512.02595*, 2015.

[69] S. Gergen, S. Zeiler, A. H. Abdelaziz, R. Nickel, and D. Kolossa, "Dynamic stream weighting for turbodecoding-based audiovisual asr," *submitted to Interspeech*, 2016.

[70] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[71] S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer, "Scheduled sampling for sequence prediction with recurrent neural networks," in *Advances in Neural Information Processing Systems*, 2015, pp. 1171–1179.

[72] T. Afouras, J. S. Chung, A. Senior, O. Vinyals, and A. Zisserman, "Deep audio-visual speech recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.

[73] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[74] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, 2014, pp. 3104–3112.

[75] I. Gruber, M. Hlaváč, M. Hrúz, M. Železný, and A. Karpov, "An analysis of visual faces datasets," in *International Conference on Interactive Collaborative Robotics*. Springer, 2016, pp. 18–26.

[76] V. Le, J. Brandt, Z. Lin, L. Bourdev, and T. S. Huang, "Interactive facial feature localization," in *European Conference on Computer Vision*. Springer, 2012, pp. 679–692.

[77] P. N. Belhumeur, D. W. Jacobs, D. J. Kriegman, and N. Kumar, "Localizing parts of faces using a consensus of exemplars," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 12, pp. 2930–2940, 2013.

[78] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015.

[79] W. M. Fisher, G. R. Doddington, and K. M. Goudie-Marshall, "The darpa speech recognition research database: specifications and status," in *Proc. DARPA Workshop on speech recognition*, 1986, pp. 93–99.

[80] G. Zhao, M. Barnard, and M. Pietikainen, "Lipreading with local spatiotemporal descriptors," *IEEE Transactions on Multimedia*, vol. 11, no. 7, pp. 1254–1265, 2009.

[81] T. J. Hazen, K. Saenko, C.-H. La, and J. R. Glass, "A segment-based audio-visual speech recognizer: Data collection, development, and initial experiments," in *Proceedings of the 6th international conference on Multimodal interfaces*. ACM, 2004, pp. 235–242.

[82] J. S. Garofolo, L. F. Lamel, W. M. Fisher, J. G. Fiscus, and D. S. Pallett, "Darpa timit acoustic-phonetic continous speech corpus cd-rom. nist speech disc 1-1.1," *NASA STI/Recon technical report n*, vol. 93, 1993.

[83] N. Harte and E. Gillen, "Tcd-timit: An audio-visual corpus of continuous speech," *IEEE Transactions on Multimedia*, vol. 17, no. 5, pp. 603–615, May 2015.

[84] B. Lee, M. Hasegawa-Johnson, C. Goudeseune, S. Kamdar, S. Borys, M. Liu, and T. S. Huang, "Avicar: audio-visual speech corpus in a car environment." in *INTERSPEECH*, 2004, pp. 2489–2492.

[85] M. Cooke, J. Barker, S. Cunningham, and X. Shao, "An audio-visual corpus for speech perception and automatic speech recognition," *The Journal of the Acoustical Society of America*, vol. 120, no. 5, pp. 2421–2424, 2006. [Online]. Available: http://dx.doi.org/10.1121/1.2229005

[86] M. Everingham, J. Sivic, and A. Zisserman, "Hello! my name is... buffy"–automatic naming of characters in tv video." in *BMVC*, vol. 2, no. 4, 2006, p. 6.

[87] J. Yuan and M. Liberman, "Speaker identification on the scotus corpus," *Journal of the Acoustical Society of America*, vol. 123, no. 5, p. 3878, 2008.

[88] C. Tomasi and T. Kanade, "Selecting and tracking features for image sequence analysis," *submitted to Robotics and Automation*, 1992.

[89] J. S. Chung and A. Zisserman, "Out of time: automated lip sync in the wild," in *Workshop on Multi-view Lip-reading, ACCV*, 2016.

[90] T. Afouras, J. S. Chung, A. Senior, O. Vinyals, and A. Zisserman, "Deep audio-visual speech recognition," in *arXiv:1809.02108*, 2018.

[91] V. Strnadová, *Hádej, co říkám, aneb, Odezírání je nejisté umění.* Ministerstvo zdravotnictví České republiky, 1998.

[92] Z. Wang and A. C. Bovik, "A universal image quality index," *IEEE signal processing letters*, vol. 9, no. 3, pp. 81–84, 2002.

[93] S. Dabbaghchian, M. P. Ghaemmaghami, and A. Aghagolzadeh, "Feature extraction using discrete cosine transform and discrimination power analysis with a face recognition technology," *Pattern Recognition*, vol. 43, no. 4, pp. 1431–1440, 2010.

[94] M. Hlaváč, I. Gruber, M. Železnỳ, and A. Karpov, "Semi-automatic facial keypoint dataset creation," in *International Conference on Speech and Computer*. Springer, Cham, 2017, pp. 662–668.

[95] N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 788–798, May 2011.

[96] G. Saon, H. Soltau, D. Nahamoo, and M. Picheny, "Speaker adaptation of neural network acoustic models using i-vectors," in *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*. IEEE, 2013, pp. 55–59.

[97] P. Císař, M. Železnỳ, Z. Krňoul, J. Kanis, J. Zelinka, and L. Müller, "Design and recording of czech speech corpus for audio-visual continuous speech recognition," in *Proceedings of the Auditory-Visual Speech Processing International Conference 2005*, 2005.

[98] M. Hlaváč, I. Gruber, M. Železnỳ, and A. Karpov, "Lipsid using 3d convolutional neural networks," in *International Conference on Speech and Computer*. Springer, Cham, 2018, pp. 209–214.

[99] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "Arcface: Additive angular margin loss for deep face recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4690–4699.

[100] G. Sterpu, C. Saam, and N. Harte, "Attention-based audio-visual fusion for robust automatic speech recognition," in *2018 International Conference on Multimodal Interaction (ICMI '18), October 16–20, 2018, Boulder, CO, USA*. ACM, New York, NY, USA, 2018. [Online]. Available: http://doi.acm.org/10.1145/3242969.3243014

Authored and Co-authored Works

Miroslav Hlaváč. Detection of lips in video sequences. Master Thesis. 2012.

Miroslav Hlaváč. Detekce rtů ve videozáznamech. SVK FAV. 2012.

Miroslav Hlaváč. Lips tracking using AAM. SVK FAV. 2013.

Miroslav Hlaváč. Sledování rtů v reálném čase pomocí aktivních kontur. SVK FAV. 2014.

Miroslav Hlaváč. Lips landmark detection using CNN. SVK FAV. 2016.

Ivan Gruber, Miroslav Hlaváč, Marek Hrúz, Miloš Železný, and Alexey Karpov. An analysis of visual faces datasets. In *International Conference on Interactive Collaborative Robotics*, pages 18–26. Springer, Budapest, 2016.

Jan Hrdlička and Miroslav Hlaváč. Poker cards recognition using neural networks. SVK FAV. 2017.

Ivan Gruber, Miroslav Hlaváč, Miloš Železný, and Alexey Karpov. Facing face recognition with resnet: Round one. In *International Conference on Interactive Collaborative Robotics*, pages 67–74. Springer, London, 2017.

Miroslav Hlaváč, Ivan Gruber, Miloš Železný, and Alexey Karpov. Semi-automatic facial key-point dataset creation. In *International Conference on Speech and Computer*, pages 662–668. Springer, London, 2017.

Lukáš Bureš, Petr Neduchal, Miroslav Hlaváč, and Marek Hrúz. Generation of synthetic images of full-text documents. In *International Conference on Speech and Computer*, pages 68–75. Springer, Leipzig, 2018.

Miroslav Hlaváč. LipsID. SVK FAV. 2018.

Miroslav Hlaváč and Alexey Karpov. LipsID detection with CNN. Almanac of Scientific Works. ITMO University. 2018.

Miroslav Hlaváč, Ivan Gruber, Miloš Železný, and Alexey Karpov. LipsID using 3D convolutional neural networks. In *International Conference on Speech and Computer*, pages 209–214. Springer, Leipzig, 2018.

Marek Hrúz and Miroslav Hlaváč. Lstm neural network for speaker change detection in telephone conversations. In *International Conference on Speech and Computer*, pages 226–233. Springer, Leipzig, 2018.

Ivan Gruber, Miroslav Hlaváč, Marek Hrúz and Miloš Železný. Semantic Segmentation of Historical Documents via Fully-Convolutional Neural Network. In *International Conference on Speech and Computer*. Springer, Istanbul, 2019.

Marek Hrúz, Petr Salajka, Ivan Gruber and Miroslav Hlaváč. Identity Extraction From Clusters of Multi-modal Observations. In *International Conference on Speech and Computer*. Springer, Istanbul, 2019.