FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

ZADÁNÍ DIPLOMOVÉ PRÁCE

Název:	Identifikace emočních stavů člověka na základě posloupnosti snímků jeho obličeje
Student:	Bc. Martin Endršt
Vedoucí:	doc. RNDr. Ing. Marcel Jiřina, Ph.D.
Studijní program:	Informatika
Studijní obor:	Znalostní inženýrství
Katedra:	Katedra aplikované matematiky
Platnost zadání:	Do konce letního semestru 2018/19

Pokyny pro vypracování

Cílem práce je analyzovat posloupnost snímků a na jejich základě identifikovat emoční stavy snímané osoby (čelní pohled). Vstupem pro analýzu je posloupnost snímků (získaná z video záznamu) a výstupem je odhadovaná míra jednotlivých emočních stavů.

1) Seznamte se klasifikací emočních stavů člověka a s problematikou detekce těchto emočních stavů.
2) Navrhněte algoritmus, resp. dílčí metody, které umožní detekovat jednotlivé emoční stavy a přiřadit jim míru jejich výskytu v daném snímku.
3) Navržený algoritmus (metody) implementujte ve vhodném programovacím jazyku s využitím volně dostupných knihoven.
4) Navržený a implementovaný algoritmus (metody) ověřte na reálných datech a vyhodnoťte dosažené výsledky. Diskutujte výhody a nevýhody zvoleného přístupu.

Seznam odborné literatury

Dodá vedoucí práce.

Ing. Karel Klouda, Ph.D.
vedoucí katedry

doc. RNDr. Ing. Marcel Jiřina, Ph.D.
děkan

V Praze dne 16. února 2018

FACULTY
OF INFORMATION
TECHNOLOGY
CTU IN PRAGUE

Master’s thesis

Identification of the human emotional states based on a sequence of images of his face

Bc. Martin Endršt

Department of Knowledge Engineering
Supervisor: doc. RNDr. Ing. Marcel Jiřina, Ph.D.

May 8, 2018

Acknowledgements

I would like to thank my supervisor, doc. RNDr. Ing. Marcel Jiřina, Ph.D., for the guidance and help with this thesis.

Declaration

I hereby declare that the presented thesis is my own work and that I have cited all sources of information in accordance with the Guideline for adhering to ethical principles when elaborating an academic final thesis.

I acknowledge that my thesis is subject to the rights and obligations stipulated by the Act No. 121/2000 Coll., the Copyright Act, as amended, in particular that the Czech Technical University in Prague has the right to conclude a license agreement on the utilization of this thesis as school work under the provisions of Article 60(1) of the Act.

In Prague on May 8, 2018 . . . .

Czech Technical University in Prague
Faculty of Information Technology

Citation of this thesis

Endršt, Martin. *Identification of the human emotional states based on a sequence of images of his face.* Master's thesis. Czech Technical University in Prague, Faculty of Information Technology, 2018.

Abstrakt

Důležitou součástí komunikace mezi lidmi je i exprese emoce. Pochopení emocionálního rozpoložení jedince pomáhá porozumět řečnickým formám jako je ironie, pochopit vážnost popisované situace a vnímat další informace, které často nejsou obsahem verbální komunikace. Vzhledem k rostoucí popularitě integrovaných rozhraní mezi člověkem a strojem má automatizované rozpoznání emoce potenciál zlepšit způsob jakým se stroji interagujeme. Díky přítomnosti kamerových senzorů téměř ve všech zařízeních je rozpoznání emoce na základě výrazu obličeje nejpřijatelnější formou vhodnou k masovému využití. V rámci této práce bylo navrženo a implementováno několik modelů rozpoznávajících emoci na základě sekvence obrázků obličeje v čelním pohledu. Jelikož je emoce dynamický psychický stav, byly prozkoumány a porovnány tři druhy časového kontextu. Pro zajištění využitelnosti vytvořených modelů s obrazovými toky v reálném čase byl vytvořen framework zapouzdřující funkcionalitu klasifikátoru. Zapouzdřenému celku jsou snímky předávány po jednom. Klasifikátory založené na metodách hlubokého učení i klasifikátory bežného typu byly využity v implementaci. Nejúspěšnější implementovaný model dosáhl přesnosti 95.1% na datové sadě CK+.

Klíčová slova klasifikace video sekvence, rozpoznání emoce, rozpoznání výrazu obličeje, transfer learning, konvoluční rekurentní neuronová síť, support vector machine

Abstract

Emotion expression is an important aspect of human to human communication. Recognizing the emotional state of a person can help us better understand complex rethorical devices such as irony, understand the gravity of described situation and infer other information that is often not expressed as part of the verbal communication channel. With the growing popularity of integrated human-machine interfaces automatic emotion detection has a great potential to improve the way we interact with machines. Since camera sensors are being integrated into almost all devices, emotion recognition based on facial expression is one of the viable methods for widespread use. Several models performing emotion recognition based on sequence of frontal facial images were proposed and implemented in this thesis. Because emotion is a dynamic psychical state, three different types of temporal context information for recognition were examined and compared. To ensure usability with real-time streams a wrapper framework consuming one frame at the time is proposed. Both deep-learning based and conventional types of classifiers were implemented. The best performing model achieved accuracy of 95.1% on the CK+ dataset.

Keywords video classification, emotion recognition, facial expression recognition, transfer learning, convoluitonal recurrent neural net, support vector machine

Contents

List of Figures

List of Tables

Introduction

Emotion expression is an important aspect of human to human communication. Recognizing the emotional state of a person can help us better understand complex rethorical devices such as irony, understand the gravity of described situation and infer other information that is often not expressed as part of the verbal communication channel. The ability to recognize emotion traverses both cultural and language barriers [1] [2] and is therefore a vital part of communication between foreign individuals.

With the growing popularity of integrated human-machine interfaces automatic emotion detection has a great potential to improve the way we interact with machines. All kinds of devices including, but not limited to automated personal assistants, health robots and smart home hubs would benefit from such ability. For these reasons emotion recognition has become a popular topic of research in past few years.

There are several ways an emotion can be recognized such as voice intonation, body language, or complex methods like electroencephalography (EEG) [3]. Because camera chips are integrated into most of the devices today the visual examination of facial expression, which is the chosen method for this thesis, is probably the most practical method for widespread use.

The problem of facial expression recognition (FER) can be categorized as either static classification, where model is only classifying one frame at the time, or sequence classification, which takes temporal aspect of emotion into consideration. Large portion of related work examines static version of this problem. Emotion, however, is a dynamic state and the temporal properties of facial expression could be important in the recognition process.

This thesis aims to create and evaluate several models for FER in sequences of frontal facial images. Seven expressions of basic emotion — anger, disgust, fear, happiness, neutral, sadness and surprise — were selected. Approaches based on deep-learning as well as conventional approaches will be examined. Comparsion between temporaly-aware models and per-frame models will also be performed.

Chapter 1

Emotion and its expression

Before diving into the problem of automatic FER this chapter aims to summarize psychological background of emotion and its manifestation in facial expression.

1.1 Emotion

Even though the science of emotion is an active field there is no universal definition of what emotion is and how to distinguish it from other psychological states. According to Dr. Paul Ekman emotion has following characteristics:

1. **There is a distinctive pan-cultural signal for each emotion**
   There is a distinctive universal expression associated with given state which functions as a signal, even though it might be very subtle. Presence of expression is not a sufficient evidence of presence of an emotion, as the expression can be simulated.

2. **Distinctive universal expressions of emotion can be traced phylogenetically**
   While this characteristic does not help to clarify boundaries of emotion it is important to note as an explanation of universality.

3. **Emotional expressions involve multiple signals**

4. **There are limits on the duration of emotion**

5. **The timing of an emotional expression reflects the specifics of particular emotinal experience**
   The duration of emotional expression is correlated to the strength of emotional experience (possibly modified by an attempt to manage said expression).

6. **Emotions are graded in intensity reflecting variations in the strength of felt experience**
7. **Emotional expression can be totally inhibited**
8. **Emotional expressions can be convincingly simulated**
9. **There are pan-human commonalities in the elicitors for each emotion**
10. **There is a pan-human, distinctive pattern of changes in the autonomos nervous system for each emotion**

Based on his cross-cultural experiments Ekman identified seven basic emotions: anger, contempt, disgust, enjoyment, fear, sadness, surprise. [4] [5]

1.2 Universality of emotion expression

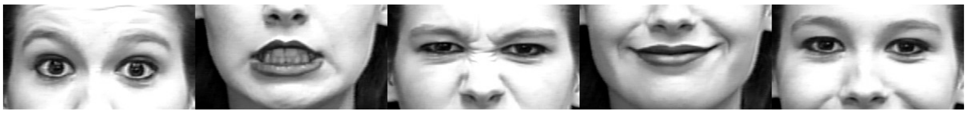
The question of universality of expression is an important one in order to establish whether emotion recognition based on facial expression is a viable general method. Until the second half of 20th century most academics believed that expresions of emotion are culturally bound and that only members of same or similiar culture express emotions in same way. Charles Darwin, however, thought otherwise. In [6] he argued that expressions of emotion were universal as they were a product of evolution. To support this claim he proposed three principles.

Principle of serviceable habits describes some expression habits as helpful and therefore reinforced by natural selection. An example would be raising the eyebrows to increase field of view in an event of danger (correlated with fear emotion). Antithesis principle states that some expressions, such as shoulder shrugging, exist merely because of their opposite nature to a serviceable habit. Some expressions, as proposed by the expressive habit principle, are a result of discharge of excitement in the nervous system. Vocal roar of anger would be an example of such expression. [1] [2]

In mid-1960s Paul Ekman took an interest in this issue. Based on hundreds of hours of film capturing isolated cultures in New Guinea highlands, taken by Carleton Gajdusek and Richard Sorenson, Ekman found that in response to given stimuli the face expressions observed were in accordance with his expectations. No culturally unique expressions were observed either. Even though he was leaned towards the culturally relativistic viewpoint at first, this experience swayed him that Darwin might be right and inspired him to travel to the New Guinea highlands.

After conducting his own experiments and collecting supporting evidence for universality of expression, Ekman came up with the idea of ”display rules” (a set of socially learned, culturally unique behaviors that are used to mask,

Figure 1.1: AU examples. AU1 — inner brow raiser, AU25 — lips part, AU9 — nose wrinklerer, AU12 — lip corner puller, AU6 — cheek raiser



exaggerate, diminish or exhibit expressions in specific cultural contexts) that would explain culture-based differences in expressions. In late 1960s he gathered evidence supporting this explanation by conducting a study of students in Tokyo, Japan and Berkeley, California. He found that both Japanese and American students reacted the same way to emotion inducing clips as long as they were filmed alone by a hidden camera. However, when a scientist entered the room, Japanese students masked negative expressions with positive ones. [7]

Expression universality has been widely accepted as a number of cross-cultural studies yielded supporting results. In [8] Lisa Feldman Barett and Maria Gendron argue that only a few of these studies were truly cross-cultural. They claim that cultures that have been exposed to the western culture have adapted their emotion expressions and concepts. Furthermore, in the studies that were truly cross-cultural (such as Ekman's experiments in New Guinea), an emotion conecptual context was included in the experimental method by asking the subjects to assign facial expression to word or description. Their free label experiment with participants from Himba ethnic group and America did not find supporting evidence for expression universality and authors are suggesting that emotion expressions are actually culture based to some degree.

Whether the emotion expression is truly universal or not, the findings of universality between cultures exposed to western culture is sufficient for vast majority of potential FER applications.

1.3 Expression measurement

In order to be able to measure and describe facial expression Dr. Paul Ekman and Dr. Wallace Friesen developed an anatomically-based system designed to measure human facial movements called Facial Action Coding System (FACS). The system uses Action Units (AUs) to describe muscular activities that produce changes in facial appearence. Action unit is a numeric code that represents muscle activity of certain facial muscles or muscle groups. FACS distinguishes 46 different AUs (e.g. AU1 - Inner brow raiser, AU23 - Lip tightener). Resulting FACS code is a string of present AUs. Presence of emotion is decided based on rules of presence of certain AUs. Even though FACS was primarily developed to help describing facial expressions while studying emotion it is a robust system that can be used in other areas as well. [9]

Chapter 2

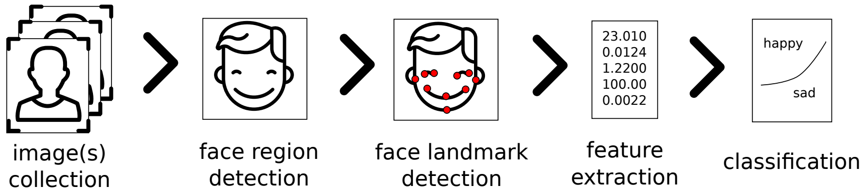
State-of-the-art

In addition to aforementioned distinction between static FER and FER on sequences, approaches to FER can also be categorized by the features used for classification. Conventional FER approaches use handcrafted features inferred from face in the facial extraction step of FER process. Deep-learning approaches often use convolutional neural network (CNN) to extract features directly from images during training process.

2.1 Conventional approaches

Approaches in this category usually adhere to following FER process schema:

Figure 2.1: Conventional FER process



Facial images are first collected and preprocessed (histogram equalization, noise reduction, etc.). FER is usually performed on grayscale images as color does not carry significant information about the expression. Next step is face region detection. It is important to regionalize face in the image before attempting to localize facial landmarks to avoid false-positives.

Multiple approaches to face region detection have been proposed over past few decades. Haar cascade classifier is one of the more popular approaches. Localization is performed via AdaBoost method using Haar-like features (descriptors of contrast change between adjecent rectangular groups of pixels). [10] An-

other popular method is based on Histogram of Oriented Gradients (HOG) features and uses Support Vector Machine (SVM) to detect a region with face. [11]
Detected face region is then used as a region of interest for face landmark estimation (face alignment). Many face alignment approaches use cascade of regressors. Each regressor is improving on landmark position estimate based on image features relative to the previous landmark position estimate. In [12] Kazemi and Sullivan use ensmeble of regression trees learned by gradient boosting to achieve super-realtime performance while maintaining state-of-the-art accuracy on face alignment problem.
Feature extraction step uses face landmarks to produce feature vector for training. Temporal and appearence features are also often extracted in addition to geometric landmark features.
SVMs are dominant classification method in conventional FER approaches. Radial Basis Function (RBF) kernel SVM seems to usually outperform linear SVM in FER.

In [13] Suk and Prabhakaran present a real-time mobile application for FER using a set of SVMs to recognize 7 basic emotions. Active Shape Model (ASM) [14] is used to locate 77 face landmarks which are then used to generate 13 high-level distance features. The model performs classification based on displacements relative to the neutral feature set. During classification process each frame is first classified by binary classifier detecting neutral emotion state. Extracted features from neutral frames are then used to update the current neutral feature set. A CK+ dataset was used for training. Reported accuracy on CK+ dataset is 87.9%

In [15] Ghimire and Lee used Elastic Bunch Graph (EBG) [16] to initialize 52 landmark positions which are then tracked in rest of the frames in sequence using Gabor jets. The classification is performed by SVM using features of two types. First type is x and y displacement of 52 landmarks relative to neutral features. Second type is euclidean distance and angle change between all pairs of landmarks relative to the distances and angles in neutral features. Neutral frames are not being recognized in-process but rather an assumption is made that neutral frame is always the first frame in sequence. Final feature vector is selected from a feature pool consisting of the two aforementioned types of features using AdaBoost with Dynamic Time Warping (DTW) similarity. CK+ dataset was used for training and reprted accuracy on this dataset is 97.2%

In [17] Happy et. al. present real-time FER system using multi-block

Local Binary Pattern (LBP) appearance features and Principal Component Analysis (PCA) to classify 6 basic emotions (neutral emotion is not being classified). In proposed model Haar cascade is first used to detect face region in source image. Face region is then divided into small subsections and the LBP histogram is calculated for each block. Final feature vector is a concatenation of individual LBP histograms. The classification is done using PCA eigen values for each emotion. Reported accuracy on custom dataset is 97%

Unlike appearance features extracted from the global face region as done in [17], Ghimire et. al. [18] extracted region specific appearance LBP features by dividing face region into 29 domain specific local regions. Incremental search approach was employed to localize important local regions in order to reduce dimensionality. In addition to appearance LBP features, geometrical landmark features were also extracted using implementation of [12]. Final feature vector is presented to linear SVM classifier. Model was validated against CK+ dataset with reported accuracy of 91.8% when classifying 7 basic emotions.

Table 2.1: Conventional FER approaches

Reference	Emotions classified	Classification method	Validation dataset	Reported accuracy
[13]	7 basic	RBF SVM	CK+	87.9%
[15]	7 basic	RBF SVM	CK+	97.2%
[17]	6 basic	PCA	custom	97%
[18]	7 basic	SVM	CK+	91.8%

2.2 Deep-learning approaches

Deep-learning approaches to FER often use CNN to either perform classification directly or to extract latent features. In order to capture temporal aspect of expressions Recurrent Neural Networks (RNN) are sometimes used as well.

In their submission to the 2015 Emotion Recognition in the Wild contest, Winkler et. al. [19] examined effectiveness of transfer-learned CNN on FER problem with small available dataset. They used pre-trained CNN model of VGG-CNN-M-2048 [20] architecture wchich was trained on generic image recognition task using images from ImageNet. This base model was then transfer-learned in two fine-tuning phases using EmotiW and FER-2013 dataset. Resulting model achieved 55.6% accuracy on the test set.

In [21] Jung et. al. present joint model of deep temporal appearance convolutional network (DTAN) and deep temporal geometry network (DTGN). Softmax outputs of these two networks is connected by element-wise addition with softmax applied to produce the final output. DTA is a 3D convolutional network where convolutional filters are shared along the time axis. This network captures temporal difference in appearance of the input images. Sequence of facial landmarks is used as input for the DTGN. Each landmark point is centered around a nose point and normalized using division by standard deviation of according dimension. Horizontal flipping and rotation were applied to input image sequences in order to increase the amount of data available for training. Model was trained using MMI dataset. Accuracy of 97.25% on CK+ dataset is reported.

Breuer and Kimmel employed deep CNN visualization methods to examine the relation between CNN-learned features and AUs in [22]. They used architecture of three convolutional blocks (consisting of a convolutional layer of 5x5 filters, activation by ReLu and max pooling layer with 2x2 window) and two fully-connected layers to perform emotion classification. This architecture achieved 98.5% accuracy on CK+ dataset measured by 10 fold cross validation. After examining the neuron activation in individual layers they found high correlation between learned features and FACS AUs. They then performed transfer learning on the same architecture to detect individual AUs and found high accuracy of 97.5% in AU presence detection and 96.1% in AU intensity prediction. This work demonstrates viability of deep CNN networks in FER related tasks.

Submission to the 2015 Emotion recognition in the Wild challenge (EmotiW) by Kahou et al. [23] proposes using hybrid CNN-RNN network for video classification. CNN network is used to extract high-level representation of input frames. Multiple architectures of CNN network with various depths were tried. Since the data provided as part of the challenge contained only videos labelled with single emotion per video, other static datasets were used for training of the CNN network. It was observed that deeper architectures tend to overfit on the static datasets and therfore a 3 convolutional block (consisting of convolutional layer of 9x9 filters, ReLu activation and max-pooling) was chosen as the best contender. The features extracted by CNN were used as input for IRNN network (RNN of ReLu using initialization trick as described in [24]). In addition to appearance features extracted by CNN authors also used geometrical landmark features and audio features to enhance the performance of the final model. To combat different lightning conditions between datasets histogram equalization was applied to the images. Best reported accuracy on the test dataset provided as part of the challenge was 52.875% and showed an improvement over pure-CNN approaches.

Table 2.2: Deep learning FER approaches

Reference	Emotions classified	Classification method	Validation dataset	Reported accuracy
[19]	7 basic	VGG CNN	FER2013	55.6%
[21]	7 basic	DTAN & DTGN	CK+	97.25%
[22]	7 basic	CNN	CK+	98.5%
[23]	7 basic	RNN-CNN	EmotiW	52.875%

Chapter 3

Core concepts used

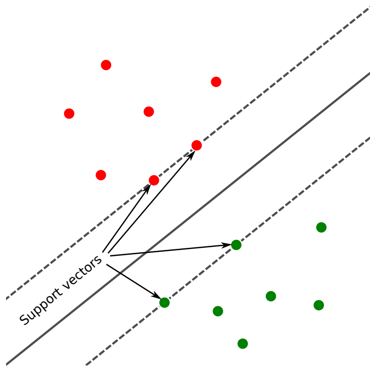
A brief introduction and overview of core concepts used in this thesis are provided in this chapter.

3.1 Support Vector Machines

SVMs are a class of supervised learning models used for classification and regression analysis originally developed by V. N. Vapnik and A. Y. Chervonenkis in 1963. In its original form, SVM is a binary linear maximal-margin classifier. Given a set of p-dimensional linearly separable binary class points as training data an infinite amount of hyperplanes separating the data exist. In order to minimalize generalization error the algorithm constructs a maximal-margin hyper-plane separating training dataset. Such hyperplane has the maximal possible distance to the closest datapoints. Training points closest to the separating hyperplane are called support vectors.
The hyperplane can be described as:

$$x^T w + b = 0;\ w \epsilon R^p, b \epsilon R$$

Figure 3.1: Support vector machine



Let n be the number of data points in the training dataset. Under the constraint of

$$y_i(x_i^T w + b) \geq 1,\ i \,\epsilon\, 1, ..., n$$

support vectors are data points that satisfy

$$y_i(x_i^T w + b) = 1,\ i \,\epsilon\, 1, ..., n$$

and their distance to the decision hyperplane can be computed as $\frac{1}{||w||}$. Therefore in order to maximize the decision margin we want to minimize $||w||$ and the optimalization problem can be defined as:

$$\underset{w}{min}\ ||w||^2;\ y_i(x_i^T w + b) \geq 1,\ i \,\epsilon\, 1, ..., n$$

Classification of a new data point is then calculated as $f(x) = sgn(x^T w + b)$. Because data is often not fully linearly separable a soft-margin variant of the algorithm was proposed by Cortes and Vapnik in 1995. The maximal-margin hyperplane constraint is relaxed to

$$y_i(x_i^T w + b) \geq 1 - \xi_i;\ \xi_i \geq 0,\ i \,\epsilon\, 1, ..., n$$

and the optimization problem becomes

$$\underset{w}{min}\ ||w||^2 + C\sum_{i=1}^{n} \xi_i^2;\ y_i(x_i^T w + b) \geq 1 - \xi_i,\ \xi_i \geq 0,\ i \,\epsilon\, 1, ..., n$$

where $C \epsilon R$ is a constant and defines the importance of all training datapoints being classified correctly. For data that is not linearly separable in the space of p dimensions U, it can be transformed into a feature space of higher dimension V where points can be linearly separated. Because feature vectors x_i only appear in inner product in both the constraint and decision function, the mapping function $\phi(x) : U \to V$ does not need to be explicitly specified but instead a kernel function is introduced. Kernel function is defined as

$$K(x_1, x_2) = \phi(x_1)^T \phi(x_2)$$

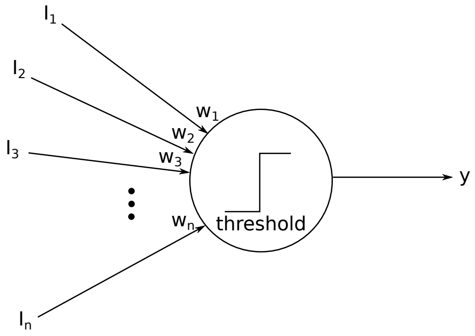
This is often referred to as the kernel trick. Popular non-linear kernel functions are the polynomial kernel:

$$K(x_1, x_2) = (x_1^T x_2 + c)^d$$

and radial basis function (RBF) kernel:

$$K(x_1, x_2) = exp(-\gamma ||x_1 - x_2||^2);\ \gamma = \frac{1}{2\sigma^2}$$

Figure 3.2: Perceptron



3.2 Artificial Neural Networks

Artificial neural networks (ANNs, further referred to simply as neural networks) are computing systems inspired by biological neural networks. The initial groundwork was laid by McCulloch and Pitts when they introduced the concept of perceptron in 1943. Perceptron is a binary classifier with n inputs and their corresponding weights, a threshold gate and one output. Decision function of perceptron is described with formula:

$$y = f(x^T w, h)$$

where $x \,\epsilon\, R^n$ is the input vector, $w \,\epsilon\, R^n$ is the vector of weights, $h \,\epsilon\, R$ is the threshold and $f : R \times R \to \{0, 1\}$ is the step function.

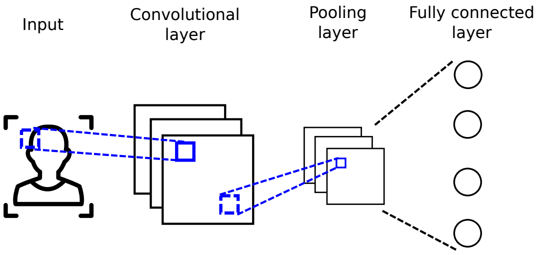
The concept of perceptron is used as a foundation for neural networks. They connect multiple perceptron-like units in layers. Neural network consists of an input layer, 0 to m hidden layers and an output layer. Neural networks are usually fully connected meaning that each neuron uses outputs of all neurons in the previous layer as its inputs. The output of single neuron is computed as

$$y = f(x^T w + b)$$

where w are the weights, x are the inputs, b is bias and f is the activation function. Multiple activation functions are used with the sigmoid function $f(x) = \frac{1}{1+\epsilon^{-x}}$ and the rectified linear unit (ReLu) function $f(x) = max(0, x)$ being the most common. The most popular method for training NNs is backpropagation of errors, a method used to calculate weight updates at each layer by calculating gradient of loss function E. The weight update is calculated as

$$\Delta w_{i,j} = \frac{\partial E}{\partial w_{i,j}}$$

Figure 3.3: Convolutional Neural Network



3.2.1 Convolutional Neural Networks

Convolutional neural networks (CNNs) are deep neural networks typically consisting of convolutional layers, pooling layers and fully connected layers. They have been proven to be very effective in various computer vision tasks such as image classification or face recognition.

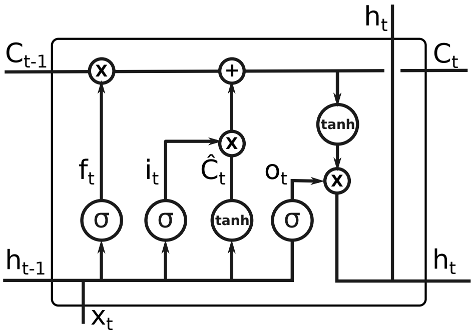
Neurons in convolutional layers are not fully connected to previous layers, every neuron is only connected to a spatial area (receptive field) of the previous layer instead. It is however fully connected along the depth axis. Convolutional layers have four hyperparameters – filter size F, stride S, zero padding P and depth D (also referred to as the number of filters). F defines the size of the receptive field, S is the offset of coinciding receptive fields, P is the amount of zero padding at the edges of previous layers and D defines the number of stacked layers. Weights are not assigned to every single connection but are rather shared among the same stacked layer. Therefore i-th convolutional layer has $F_i^2 D_{i-1} D_i$ weights. Width W_i and height H_i of the i-th layer are calculated as $W_i = \frac{W_{i-1} - F_i + 2P_i}{S_i} + 1$ and $H_i = \frac{H_{i-1} - F_i + 2P_i}{S_i} + 1$.

Pooling layers are usually inserted in-between successive convolutional layers. Similiarly to the convolutional layers pooling layers also employ the idea of receptive fields however neurons are only connected along the spatial axes and connections have no weights. Instead of weighted sum a pooling operation (such as *max* or *avg*) is applied to the inputs of each neuron. Size of receptive field F and stride S are used to define a pooling layer.

3.2.2 Recurrent Neural Networks

Some tasks require the ability to recognize patterns in sequences of data, such as text, speech or numerical series. Regular NNs are not equipped with this ability since they treat each input individually. Recurrent Neural Networks (RNNs) are designed to produce output based not only on the current input

Figure 3.4: LSTM cell



but also taking previous inputs into consideration. Therefore such NN posesses a form of memory. RNNs enjoy a great ammount of interest in recent decades and have presented state-of-the-art performance in many fields.

3.2.3 Long Short-Term Memory networks

In order to capture patterns over long temporal distances a concept of Long Short-Term Memory units (LSTMs) was proposed by Hochreiter and Schmidhuber [25]. LSTM is an attempt to solve the vanishing gradient problem which prevents simpler RNNs from learning over many time steps. At the core of LSTM network is the LSTM cell (see fig. 3.4). An LSTM cell consists of multiple gates that modify the cell memory state C_t based on the hidden state at previous time step h_{t-1} and the input x_t. The first gate on the path of the information flow is the forget gate. This part of the cell is responsible for deciding what information to forget from the cell memory state and its output is defined as:

$$f_t = \sigma(w_f^T[x_t, h_{t-1}] + b_f)$$

Next the cell decides how to update its memory state utilising an intention gate which decides what is important to save to the memory state.

$$i_t = \sigma(w_i^T[x_t, h_{t-1}] + b_i)$$

$$\hat{C}_t = tanh(w_C^T[x_t, h_{t-1}] + b_C)$$

Now the cell memory state can be updated and the information flows to the last gate which is the output (or selection) gate. This gate learns to decide what information to propagate to the hidden state at the current time step h_t, which is also the output of the cell.

$$C_t = f_t C_{t-1} + i_t \hat{C}_t$$

$$o_t = \sigma(w_o^T[x_t, h_{t-1}] + b_o)$$

$$h_t = o_t tanh(C_t)$$

Chapter **4**

Available datasets

This chapter contains brief descriptions of available datasets suitable for FER task. Multiple datasets of facial images (static or sequences) are available with varying image resolutions, subject groups and types of labels. An important distinction is whether the expressions are staged or spontaneous as spontaneous expressions tend to be more subtle in intensity and shorter-lived. Only some datasets provide labels in terms of basic emotions. Because 2D based analysis has difficulty handling head pose variations, datasets of 3D images and videos are gaining popularity in the context of FER.
This thesis focuses on 2D based analysis and thus only 2D dataset are listed in this section. Also note that this section only lists datasets that were considered most suitable for purposes of this thesis and is only a subset of available 2D datasets.

The Extended Cohn-Kanade Dataset (CK+) [26] published in 2010 builds upon the original Cohn Kanade dataset [27]. It contains 593 sequences from 123 subjects of mostly posed expressions (122 sequences of spontaneous smile from 66 subjects are also available). Full FACS coding is available for the peak frames of all 593 sequences. All sequences are also labelled with basic emotions: anger, contempt, disgust, fear, happiness, sadness and surprise. All sequences begin with neutral expression and contain the onset and peak of the emotion, some sequences end with neutral emotion and some end after the peak. Participants were 18 to 50 years of age, 69% female, 81%, Euro-American, 13% Afro-American, and 6% other ethnic groups. Sequences vary in length from 10 frames to 60 frames, are of 640x480 format and are in grayscale.

Japanese Female Facial Expressions (JAFFE) database [28] is a dataset containing 213 static images of posed expressions performed by 10 japanese female models. Each image is labelled with one of 7 face expressions (anger, disgust, fear, happiness, neutral, sadness and surprise) rated by 60 japanese subjects based on six emotional adjectives. Images in the dataset have resolution of 256x256 pixels and are in grayscale.

MMI Facial Expression Database (MMI) [29] contains over 2900 video sequences and high-resolution static images of 75 subjects. Every video is fully annotated for the presence of AUs (event coding) and partailly coded on frame-level indicating AU neutral, onset, apex or offset phase. A portion of the dataset is also labelled with expressed emotion. There are a total of 238 video sequences on 28 male and female subjects. Images are colored and have resolution of 720x576 pixels.

Multimedia Understanding Group Facial Expression Database (MUG) [30] consists of 1462 sequences of posed and induced emotions performed by 35 women and 51 men of caucasian origin aged between 20 and 35 years. In the first part of the dataset (posed emotions) participants were asked to express each of the 6 basic emotions (anger, disgust, fear, happiness, sadness and surprise), the neutral expressions were also recorded. Expressions were captured at 896x896 pixels resolution with the frame rate of 19 fps. Each sequence starts and ends with neutral expression and follows the onset, apex, offset temporal pattern. The length of the sequence ranges from 50 to 160 frames. Emotion annotations are available for all sequences and a portion of the dataset is labelled with 80 facial landmark points tracked at each frame. In the second part of the dataset subjects were asked to watch an emotion-inducing video while being recorded.

The Belfast Induced Natural Emotion Database (Belfast) [31] captures spontaneous expressions as responses to emotion-inducing tasks. The database is split into three sets each collected at different time periods. Set 1 consists of 570 clips 5 to 30 seconds long capturing 70 male and 44 female subjects performing tasks designed to induce frustration, disgust, surprise, fear and amusement. 650 clips were collected for the Set 2 with lengths varying between 5 and 60 seconds. 37 male and 45 female subjects were recorded as part of Set 2 and performed tasks designed to induce disgust, surprise, fear, amusement, anger and sadness. Tasks for Set 3 were designed to explore cross-cultural differences in emotion expression and were expected to induce disgust, fear and amusement. There are 180 clips of 30 to 180 seconds duration capturing 30 male and 30 female participants from Northern Ireland and Peru as part of Set 3. Sequences are labelled with self-reported emotions.

Chapter **5**

Design and analysis

This thesis aims to create multiple models for FER in sequences. Both static (frame-by-frame) and sequence approaches and both conventional and deep-learning based meethods are utilised. To create these models following steps are performed: **data acquisition, validation and transformation**, **data preprocessing**, **feature extraction** and **model creation and training**.

5.1 Data acquisition, validation and transformation

In order to collect sufficient ammount of data for deep-learning based approaches, four datasets described in the previous chapter – CK+, JAFFE, MMI and MUG – will be used in this thesis. Because the static and sequence variants of FER require different data, these four datasets will be combined into two datasets (one static and one sequential) as a result of this step.

5.1.1 Data acquisition and validation

Each dataset has to be validated for presence of required labels, image and video orientation and face detectability. Images / sequences with missing emotion label or where face cannot be automatically detected will be discarded from the final dataset.
Out of the 593 sequences contained in the CK+ dataset only 327 are labeled with basic emotion. The other sequences do not exhibit expression that would fit the definition of a prototypic emotion. These sequences are discarded. Furthermore expression of contempt is recognized in the CK+ dataset and because it is not directly mappable to any emotion used in this thesis sequences with this expression are discarded. All frames contain a detectable face and are upright.
Only 137 video sequences are labelled with prototypic emotion in the MMI dataset. Furthermore, some of the emotion code labels do not correspond to

any of the basic emotions. Such sequences are discarded. Some sequences are recorded sideways and need to be corrected.
All sequences in MUG dataset have proper label assigned, are in correct orientation and faces are automatically detectable. There are no validation issues with images in the JAFFE dataset either.

5.1.2 Data transformation

In order to combine datasets it is necessary to unify the format, label vocabulary and temporal scale.
By examining sequence lengths and expected temporal durations of exhibited expressions it was estimated that sequences in the CK+ dataset were captured at the rate of roughly 8 frames per second. Since this dataset has the lowest capture rate the 8 fps estimate is used as the target capture rate for other datasets as well.
Video sequences in the MMI dataset were captured at 25 fps therefore, in order to equalize temporal scale, videos are subsampled keeping every third frame. Sequences in the MUG database were captured at 19 frames per second. Keeping every second frame results in sequneces with capture rate of roughly 9 fps which is an insignificant deviation from the 8 fps target.
Sequence lengths are normalized to 40 frames by duplicating first and last frames in the sequences shorter than 40 frames and truncating the beginning and end in sequences longer than 40 frames. Decision on the sequence length was done based on observation of source sequences where the cycle neutral, onset, apex always happened within 40 frames.

Common temporal regions of neutral, onset and apex phases are deduced by sequence observation in each dataset.

Table 5.1: Expression phase temporal regions

Dataset	neutral frames	onset frames	apex frames
CK+	1 to 7	8 to 20	20 to 27
MMI	1 to 6	7 to 12	13 to 20
MUG	1 to 7	8 to 16	17 to 23

Dynamic dataset is constructed such that a 10-frame long subsequence is formed for neutral, onset, apex and offset temporal regions. Because not all of the sequences contain offset phase at the end of the sequence, reversed onset sequence is used instead. Subsequences created from neutral and reversed onset regions are labelled with neutral emotion, subsequences created from apex and onset regions are labelled with the sequence label. Subsequences are normalised to the 10-frame length using algorithm described in Listing 5.1.
Because every sequence contributes two subsequences labelled with neutral

emotion there is roughly 5 times more neutral subsequences than of the rest of the expressions. In order to avoid classification bias towards dominant class, neutral subsequences are randomly culled with keep-probability of $\frac{1}{5}$. Resulting dataset contains 4982 subsequences.
Two frames are selected at random from neutral and apex subsequences and combined with the JAFFE dataset form the static dataset of 5408 images.

Listing 5.1: Sequence length normalisation

```
result = []
ratio = frames.length / target_length
cnt = 0
i = 1
for frame in frames:
    while cnt < 1 and i <= target_length:
        result.append(frame)
        cnt += ratio
        i++
    cnt -= 1
return result
```

5.2 Data preprocessing

Because multiple source datasets are used and conditions under which sequences were captured differ, source images vary in illumination, intensity distribution, face scale (distance from objective) and to some degree face position within the image. In order to minimize the effect of different conditions on model performance these variations have to be normalized.

5.2.1 Histogram equalization

Histogram Equalization (HE) is a popular technique to increase the global contrast of an image which effectivly spreads out most frequent intensity values across the whole intensity range. Therefore all images processed by HE have the same intensity scale. The algorithm constructs mapping from old intensity values to new ones such that the cumulative intensity function of resulting image is near-linear.
Given a greyscale image x, the probability of pixel having intensity value of i is

$$p_x(i) = \frac{n_i}{n},\ i\,\epsilon L$$

where n_i is the number of pixels at intensity level i, n is the total number of pixels and L is the intensity range. Transformation function $T(i) : L \rightarrow \{0, ..., 255\}$

mapping old intensity value to new one is defined as

$$T(i) = L_{max} \sum_{j=0}^{i} p_x(j)$$

While HE achieves improved global contrast it can in some cases (e.g. when the object of interest is significantly lighter than the rest of the image) reduce local contrast of important regions.
Adaptive Histogram Equalization (AHE) addresses this issue by transforming each pixel based on its local neighborhood. Histogram, CDF and intensity transformation function is computed for neighborhood of set size for each pixel according to the HE algorithm. When a neighborhood extending beyond the edges of source image is being processed, rows and columns are mirrored respective to the edge. While AHE improves local constrast even in cases where original HE fails, AHE can exaggerate noise in the regions of near-homogeneous intensities.
Contrast-Limited Adaptive Histogram Equalization (CLAHE) introduces clipping threshold for the local histogram in order to eliminate the noise amplification issue of AHE. Near-homogeneous regions manifest as a spike in local histogram at the according intensity bin. During local histogram computation all bins exceedig the clipping threshold are clipped at the threshold and the excess is uniformly distributed among the other bins. This modification lowers the slope of resulting CDF.

As is apparent from Figure 5.1 demonstrating the effect of discussed HE variants, CLAHE provides most stable results and achieves great local contrast. It is therefore the variant of HE used in data preprocessing in this thesis.

5.2.2 Face detection

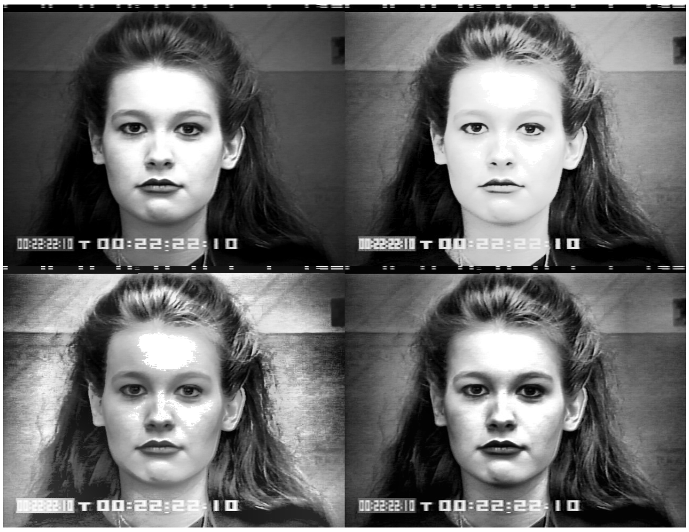
For the purposes of FER only the face region of images is required. Furthermore face alignment requires a face bounding box to prevent false-positive landmarks. In order to obtain the face boinding box a face detection is employed. There are multiple popular object-detection methods, most based either on wavelet features or Histogram of Oriented Gradients (HOG) features. HOG based detector proposed in [11] is used in this thesis.
HOG utilises gradients of intensity calculated for each pixel and consisting of magnitude g and angle ϕ. Given an image $x \epsilon \{0, ..., 255\}^n \times \{0, ..., 255\}^m$ gradient for a pixel with coordinates i, j is calculated as:

$$u_{i,j} = x_{i,j+1} - x_{i,j-1}$$

$$v_{i,j} = x_{i+1,j} - x_{i-1,j}$$

$$g_{i,j} = \sqrt{u_{i,j}^2 + v_{i,j}^2}$$

Figure 5.1: Effect of histogram equalization, from left to right: original, HE, AHE, CLAHE



$$\phi_{i,j} = arctan\frac{v_{i,j}}{u_{i,j}}$$

Image is divided into 8x8 cells. For each cell a histogram is calculated. The histogram contains bins of angles 0, 20, 40, 60, 80, 100, 120, 140 and 160. If for a pixel with coordinates i, j the closest bin angle is α and second closest is β then bin contributions are calculated as

$$\Delta H_\beta = g_{i,j}\left|\frac{\phi_{i,j} - \alpha}{\phi_{i,j} - \beta}\right|$$

$$\Delta H_\alpha = g_{i,j} - H_\beta$$

To make the descriptor indifferent to luminosity variance L2 normalization is performed on concatenated histograms of 16x16 blocks. Resulting HOG descriptor is a concatenation of all the histograms of the 16x16 blocks. In order to detect face bounding box HOG is calculated for various patches of original image and classified with learned classifier (such as linear SVM). [32]

5.2.3 Face alignment

Face alignment is a process of estimating facial landmark positions on the source image. Many methods were proposed over past decade. Method utilising cascade of tree-based regressors published in [12] is used in this thesis. The algorithm starts with initial estimate equal to learned mean shape

$$\hat{S}^{(0)} = \{a_1, ..., a_p\}$$

Figure 5.2: Histogram of Oriented Gradients

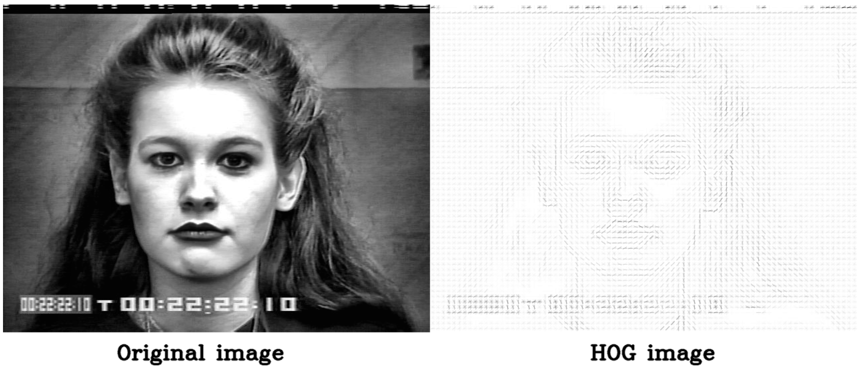
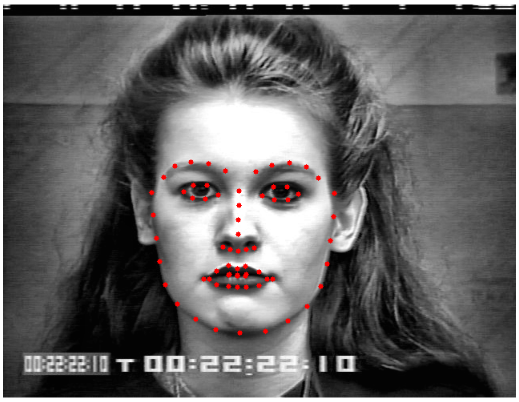


Figure 5.3: Face alignment



where $a_i, i \, \epsilon \, \{1, ..., p\}$ are the x,y coordinates of p landmarks of the mean shape. The estimate for step $t+1$ is devised as

$$\hat{S}^{(t+1)} = \hat{S}^{(t)} + r_t(x, \hat{S}^{(t)})$$

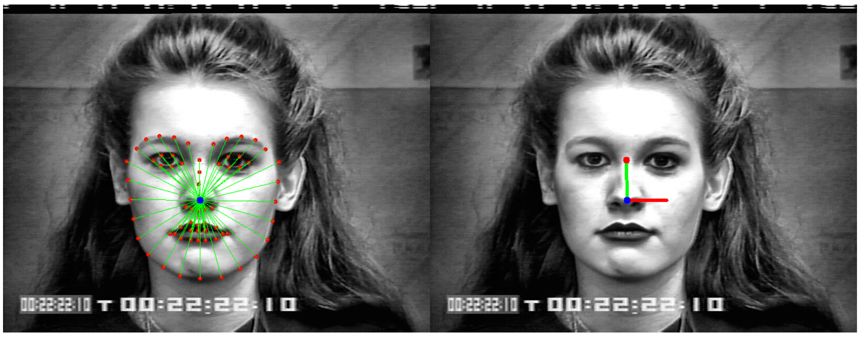
where x is the source image and r_t is the learned regressor for step t. Resulting 68 landmarks are saved along with each frame and will be used in following steps.

5.2.4 Scale, rotation and offset normalization

Because the distance between subject and camera lens can vary, faces in the dataset can be in different scales. Position of the face within the image (offset) and rotation of the face can also vary. To assist in normalization of these aspects a center of gravity (COG) of all landmarks is first computed and is used as an anchor point.

$$cog = \frac{1}{p} \sum_{i=1}^{p} a_i$$

Figure 5.4: COG, face angle



Face angle ϕ is then estimated using the COG and the root of nose, which usually lays on the y-axis of the face.

$$u = a - cog$$

$$\phi = arccos(\frac{u^T v}{||u||})$$

a is the landmark located at the root of nose and $v = (1, 0)$ is normalized vector along the x-axis. Both the image and landmarks are rotated around COG by $-\phi$ so that the y-axis of the face is vertical. Images are cropped to the face bounding box defined by points b_1, b_2 which are inferred from detected landmarks. Cropping ensures invariance to the face position within the image.

$$b_1 = (a_{xmin}, a_{ymin}),\ b_2 = (a_{xmax},\ a_{ymax})$$

$$a_{xmin} = min(\bigcup_{i=0}^{p}(1,0)^T a_i),\ a_{ymin} = min(\bigcup_{i=0}^{p}(0,1)^T a_i)$$

$$a_{xmax} = max(\bigcup_{i=0}^{p}(1,0)^T a_i),\ a_{ymax} = max(\bigcup_{i=0}^{p}(0,1)^T a_i)$$

Resulting crops are resized to 256x256 size which normalizes the scale (at the cost of potential proportion distortion). Scale normalization and position invariance in landmarks is achieved by transforming the points to COG-centric coordinates and normalizing by the largest landmark-COG distance.

5.3 Feature extraction

Since images are already preprocessed and CNN-based classifiers perform implicit feature extraction, this step focuses on creating multiple landmark-based feature sets.

Table 5.2: FACS AU

Emotion	AUs present	AU description
Happiness	6,12	cheek raiser, lip corner puller
Sadness	1,4,15	inner brow raiser, brow lowerer, lip corner depressor
Surprise	1,2,5,26	inner brow raiser, outer brow raiser, upper lid raiser, jaw drop
Fear	1,2,4,5,7,20,26	inner brow raiser, outer brow raiser, brow lowerer, upper lid raiser, lid tightener, lip stretcher, jaw drop
Anger	4,5,7,23	brow lowerer, upper lid raiser, lid tightener, lip tightener
Disgust	9,15,16	nose wrinkler, lip corner depressor, lower lip depressor

5.3.1 Landmark selection

There are 68 landmarks available from the face alignment part of the data preprocessing step. In order to reduce dimensionality and mitigate potential overfitting feature selection is performed. Established expression rules of FACS AUs are a great source of understanding which areas are important for each expression. AUs presence for each expression is listed in Table 5.2. For example AU1 (inner brow raiser) quite intuitively maps to landmarks located at the inner part of brows. Some landmarks, such as those located on the jaw, always move together so keeping all of them is not necessary. Examination of availabe examples of isolated AUs was used to select 28 most relevant landmarks (see Figure 5.5).

Simple random tree classifier was used to ensure no important information was lost by feature selection. Full feature set of 68 landmarks achieved 10-fold cross-validation accuracy of 78.76% and the reduced feature set achieved accuracy of 78.35%.

Figure 5.5: Selected landmarks

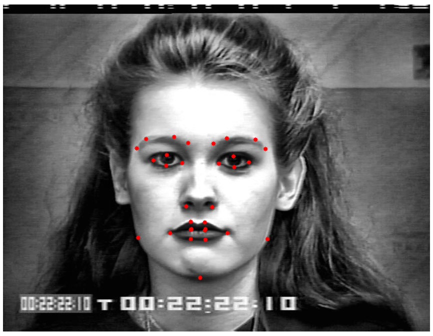


Figure 5.6: Lip corner puller AU and tracked distance

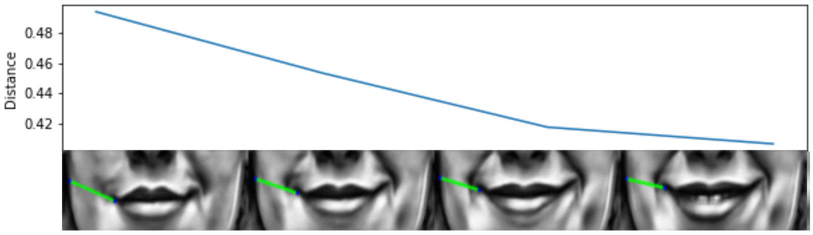
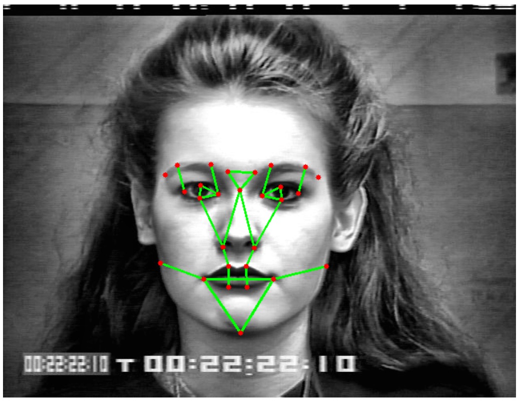


Figure 5.7: Selected landmark pairs to approximate AUs



5.3.2 Distance-based features

Because muscle activities described by AUs essentially either contract or stretch the distance between facial points, distance-based feature set aiming to approximate AUs is constructed. Similiarly to the landmark selection step, examples of AUs are used to identify possible landmark pairs that would best represent individual AUs. Some AUs are represented with multiple landmark pairs.

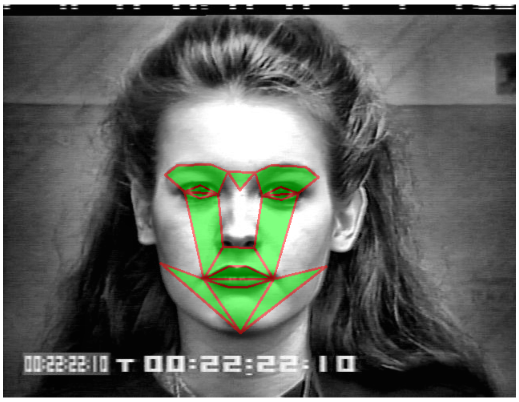
Figure 5.6 demonstrates how distance between a lip corner and a landmark located on the face outline is used to approximate AU12 — lip corner puller, which is usually present in expression of hapiness. Frames of happines onset were used to create a graph capturing the change of tracked distance during onset phase of the expression. Figure 5.7 displays selected landmark pairs.

Distance feature set achieved 76.23% cross-validation accuracy using random forest classifier. Combined with reduced landmark feature set the accuracy improved to 80.76%.

5.3.3 Area features

When multiple facial muscles are involved in an aspect of expression, compression or expansion of certain areas occurs. For example pressing lips to-

Figure 5.8: Area features



gether (AU24) greatly reduces the area between upper and lower lip. Some expressions, such as fear, are very subtle in the distance feature space. Fear manifests mainly through wide open eyes and brow raiser, but otherwise is very similiar to neutral or sad emotion. Change in the area of eye when being opened is much larger than the change in the lid-to-lid distance. Therefore a feature set of areas enclosed by feature landmarks was created to examine if areas are viable descriptors to improve model performance on subtle emotions. Area feature set achieves performance of 73.25% on its own and 78.31% when combined with landmarks.

5.3.4 Feature differentiation

As demonstrated in [15] [13], differentiated features relative to features for neutral expression lead to better performance. In [15] an assumption is made that neutral expression is always the first frame in the sequence. While that is a valid constraint when dealing with a laboratory dataset the final model of this thesis aims to be useable with real-life data. Approach used in [13], which starts with learned neutral feature vector estimate which is then updated during classification process, is better aligned with the goals of this thesis.

Neutral feature vector for static dataset is devised for each feature set as the mean of feature vectors labelled with neutral emotion in the static dataset. Differentiated feature sets are then created such that neutral feature vector is deducted from each feature vector in feature set of static dataset. In sequence dataset each sequence is differentiated by computing neutral feature vector as mean of feature vectors in the neutral subsequence, which is then deducted from all feature vectors in the sequence.

Differentiating the features improved performance on static dataset with random forests classifier from 80.76% to 82.31%.

5.3.5 Temporal differentiation

Because emotion is a dynamic psychological state temporal context is important in FER. Temporal differentiation is introduced as a method to introduce temporal context to the data. Each feature vector U_t is extended by history difference between current frame and last frame in the sequence $\Delta U_t^{(t-1)}$ as well as frame three time steps in history $\Delta U_t^{(t-3)}$

$$\Delta U_t^{(t-1)} = U_t - U_{t-1}$$

$$\Delta U_t^{(t-3)} = U_t - U_{t-3}$$

The final feature vector is then $\hat{U_t} = \{U_t, \Delta U_t^{(t-1)}, \Delta U_t^{(t-3)}\}$.

5.4 Model creation and training

This section describes architecture of constructed models and describes the classification and training process for each of them. Constructed models are categorized into conventional models and deep-learning based models.

Ten models were created as part of deep-learning based approaches. First is a CNN-RNN hybrid network utilising the Inception V3 [33] architecture and transfer learning for the CNN part, and a 4-layer RNN using CNN-processed sequence as its input. Second model is the CNN part of the hybrid network used for static classification. The rest of deep models are 5-layer RNN, one for each feature set and differentiation. For conventional approach Linear SVM, RBF SVM, k-NN and random forest methods were tried. RBF SVM showed marginally better performance on both static and sequence datasets and is thus used as the classifier for conventional models.

Following subsections describe how individual models are trained.

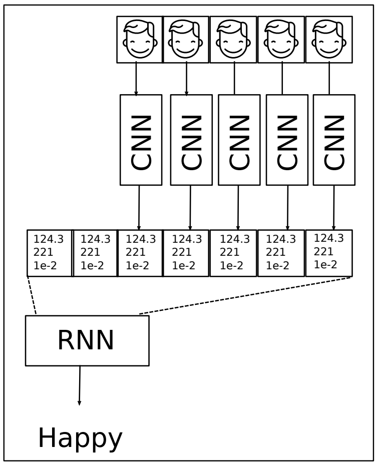
5.4.1 CNN-RNN hybrid network

CNN-RNN model performs classification based on preprocessed subsequence of images as described in section 5.2. Inception V3 network was used because of its great performance on image classification problems (achieving 93.7% accuracy on the 1000-class ImageNet test dataset).

The architecture is based on the GoogleNet architecture published in [34]. The core building block is the Inception module. Main idea behind the Inception module is that instead of selecting the convolution size on each layer of CNN, which can greatly affect the performance of resulting model, Inception modules perform multiple convolutions and let the training process decide which one is the best suited for required results.

Weights for the Inception V3 network that were pre-trained on ImageNet dataset are used for the CNN part of the hybrid network. Original fully connected layers are replaced with a 4-layer network. The output from the last

Figure 5.9: CNN-RNN hybrid network schema



convolutional block is funneled into a layer with 1024 nodes utilising ReLu activation function followed by 50% dropout, another 1024 node ReLu layer and finally a 7 node layer with softmax activation function representing the final prediction. Transfer learning is performed on resulting network.

Transfer learning is a technique where a pre-trained model is presented with new task and re-trained for it. Because many of the learned features are common for most image classification tasks, transfer learning greatly reduces the required amount of training data and training time.

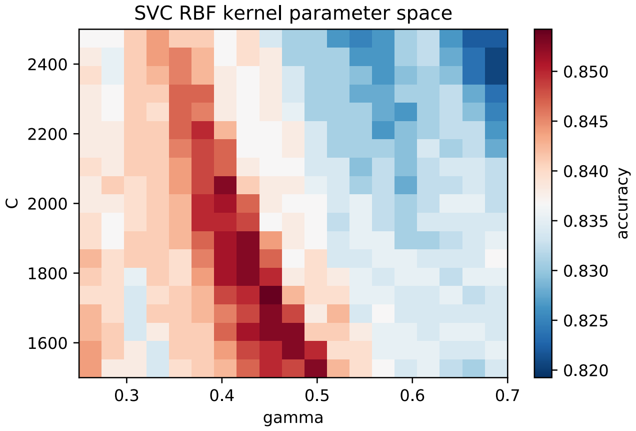
The network is trained in three phases. First all the convolutional blocks of Inception V3 are frozen and only the fully connected layers are being trained on the static dataset. Afterwards 6 convolutional blocks are unfrozen and the training is performed again. After these two steps the CNN part of the network is fully trained. The output from the last convolutional block is then used for training of the RNN network and CNN functions as an automatic feature extractor.

In order to increase the data variance data augmentation is used. All images are rotated by random angle between -20 and 20 degrees. Images are also horizontally flipped at random.

The RNN consists of an LSTM layer of 1024 nodes, dropout layer with drop probability of 50%, fully connected layer with 1024 nodes and ReLu activation function and last fully connected layer with 7 nodes and softmax activation function.

Adam optimization method with categorical crossentropy loss function achieved best results in experiments and was therefore used to train both networks.

Figure 5.10: Static SVM parameter search



5.4.2 RNN models

An RNN model is created for each feature set and differentiation. The RNN models process 10-frame long sequences of landmark features. The final architecture that was chosen by experimentation consists of 224-node LSTM layer, 30% dropout, 112-node dense layer with ReLu activation function, 32-node dense layer with ReLu activation function and finally a 7-node dense layer with softmax function. The network was trained using Adam optimization with categorical crossentropy loss function.

5.4.3 SVM models

SVM models utilise the RBF kernel function

$$K(x_1, x_2) = exp(-g||x_1 - x_2||^2);\ g = \frac{1}{2\sigma^2}$$

A model is created for each feture set described in previous section. Hyperparameter tuning of the regularization constant C and the influence area spread constant g were done by grid search of the 2-D parameter space (see Figure 5.10). This fine-tuning step is important because if g is too small the model becomes too constrained and cannot capture the complexity of data, setting g too high shrinks the radius of influence of individual support vectors and model tends to overfit. Regularization constant C counterweights the constant g by affecting the number of support vectors.

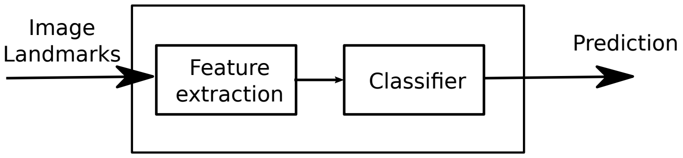
Static models perform classification on frame-by-frame basis, sequential models use concatenation of feature vectors of all frames.

5.4.4 Wrapper framework

In order to use these models with various data sources such as video streams, final models cannot depend on availability of the entire sequence at once or on a specific temporal pattern (such as each sequence must start with neutral expression). For this reason a wrapper framework, which is presented with only one image and corresponding set of 68 aligned landmarks at a time is proposed for static, static differential, sequential, sequential differential and temporal differential types of models.

Static model wrapper is the simplest case as the enclosed static model inherently operates on frame-by-frame basis. Wrapper only extracts the required features and passes them to the classifier.

Figure 5.11: Static model wrapper

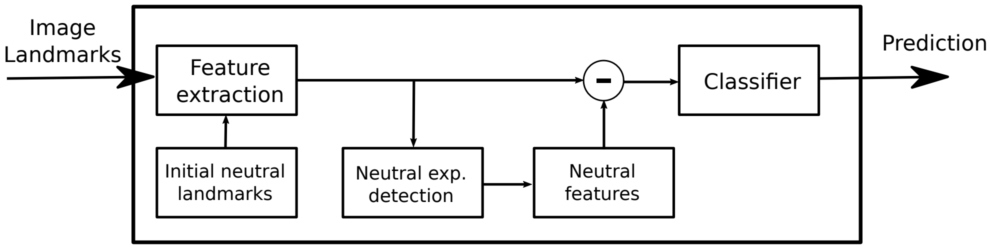


Static differential model wrapper uses neutral feature vector to perform neutral-based feature differentiation. A learned mean of face aligned landmarks for neutral expression is deployed with the model and is always used as the initial input to initialize the estimation of neutral vector. A static binary classifier is embedded in the wrapper along with the primary classifier and is used to detect neutral expression in each frame. If current frame S is classified as neutral the neutral vector estimate $S_{neutral}$ is updated as

$$S_{neutral}^{(t+1)} = \frac{1}{2}(S_{neutral}^{(t)} + S), t \geq 0$$

$$S_{neutral}^{(0)} = S$$

Figure 5.12: Static differential model wrapper



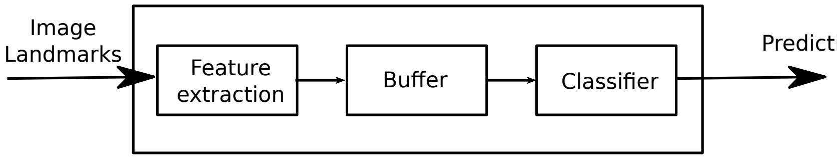
Sequential model wrapper keeps a buffer B of length N of previous frames to perform sequence-based classification. At each timestep t the buffer is updated by shifting the buffer to the left and adding current frame S to the end. Upon initialization the first frame in sequence is copied to fill the whole buffer.

$$B_i^{(t+1)} = B_{i+1}^{(t)},\ 1\ \leq\ i\ <\ N,\ t\ \geq\ 0$$

$$B_N^{(t+1)} = S,\ t\ \geq\ 0$$

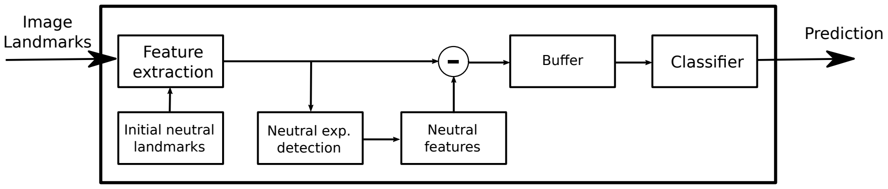
$$B_i^{(0)} = S,\ i \,\epsilon\, \{1, ..., N\}$$

Figure 5.13: Sequential model wrapper



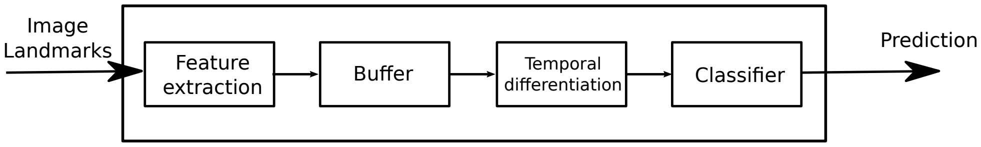
Sequential differential model wrapper is a combination of static differential model wrapper and sequential model wrapper. The neutral feature detection and differentiation process is inserted before buffer handling in the sequential model wrapper.

Figure 5.14: Sequential differential model wrapper



Temporal differential model wrapper is a modification to the sequential model. It implements identical buffer handling but instead of feeding the buffer directly into the classifier a temporal differentiation as described in 5.3.5 is performed. Buffer length N is set to 4 and the feature vector U for the classifier is constructed as $U = \{B_N, B_N - B_{N-1}, B_N - B_{N-3}\}$

Figure 5.15: Temporal differential model wrapper



Chapter 6

Implementation

This chapter briefly summarizes the implementation of data preparation and model training processes. A simple application was also developed to demonstrate the performance of implemented models and the deployability of the wrapper framework.

6.1 Language choice

Two languages were considered for the implementation — C++ and Python. C++ is a compiled, strongly typed language wchich gives it the edge in terms of computational efficiency. Many popular data science and computer vision libraries, such as the OpenCV, Dlib or TensorFlow, are either implemented in C++ directly or provide C++ API. The lack of convenience features, such as an automatic garbage collecor, can lead to slower development. Source codes in C++ have to be compiled for the target platform, which increases the deployment complexity. C++ is a popular choice in fields where computational performance of final model is of great importance, such as some computer vision tasks or AI in games.

Python is a general-purpose, dynamically typed interpreted language with tremendous popularity and support from community. Most of the data science, computer vision and deep-learning libraries provide API in Python. Its popularity can be be attributed to the ease of development, rapid prototyping capability and ease of deployment thanks to integrated tools like Pip. Python is often used in combination with Jupyter Notebook, which provides interactive, computation-cell based GUI with Python backend suited for rapid prototyping and light data exploration.

Since computational performance of resulting models is not an objective of this thesis, Python was selected as the language of implementation (mainly because of the ease of developent and intuitive APIs).

6.2 Used libraries and tools

Various machine learning, computer vision and deep-learning libraries were used for the implementation. This section lists the most import ones with brief description.

The **Open Source Computer Vision Library (OpenCV)** is a computer vision library written in C/C++ and released under BSD license. It provides API in Java, C++ and Python. It is used predominantly for image loading and manipulation in this thesis.

Dlib is a toolkit containing machine learning algorithms licensed under the Boost Software License. It is written in C++ and provides API in Python as well. Included implementation of face detection and face alignment are used in this thesis.

TensorFlow is a popular deep learning library originally developed by the Google Brain team. It was released in 2015 under the Apache 2.0 license. TensorFlow provides API to Python, C++, Java, Go and Swift. TensorFlow is used as the primary backend for deep models in this thesis.

Keras, a high level neural network API working on top of Tensorflow, CNTK or Theano backends. It offers user-friendly fluent API, modularity end extensibility. It is written in Python and is released under the MIT license. Keras is used in this thesis because of its intuitive and easy-to-use API as well as backend interchangeability.

Matplotlib [35] is used to visualize all the data in this thesis. It is an open source 2D plotting library written in Python and distributed under the PSF license.

Jupyter Notebook, part of the Project Jupyter, is a tool allowing for creation of interactive code notebooks in the browser. It supports over 40 languages, has integration with many data visualization libraries and can be easily shared with GitHub.

Scikit-learn is a Python machine learning, data mining and data analysis toolkit. It utilises the NumPy, SciPy and Matplotlib libraries. Scikit-learn is released under the BSD license. Implementations of SVM, random forrest and k-NN classifiers provided by Scikit-learn were used in this thesis.

6.3 Project structure

Large portion of work on this project was done in an unstructured manner utilising Jupyter Notebook. This allowed for easy data exploration and experimentation with different data preprocessing methods, model training, fine-tuning and evaluation. After finalization of all processes the code was divided into three modules based on functionality — a common library, main application code and accompanying scripts.

6.3.1 Common library

Common funtions were extracted into library files based on their purpose.
preprocessing.py contains functions used during preprocessing steps. An example of such function would be face angle estimation or resizing and cropping of the image to the face region.
classification.py contains implementation of all classifiers. The wrapper framework described in 5.4.4 is implemented as abstract base classes. Definition of specific classifier is done by simply extending appropriate base class and providing learned model and implementation of feature extraction function.
ui.py contains classes VideoProcessor and SequenceProcessor which are responsible for loading and handling the data source. Processor objects call a callback function (provided at initialization) frame by frame. Classes EmotionDisplay and EmotionPlot which are responsible for displaying the classification result are also defined in this file.
landmark_features.py contains functions specific to landmark features, such as computing distance features or performing COG-centric normalization.

6.3.2 Main application

The purpose of the main application is to demonstrate the performance of implemented models and deployability of the wrapper framework. Application accepts the name of the model to be used and the sequence source as command line arguments. Sequence source can be a webcam or other video stream, video file or a directory containing images.
Three windows are opened after launching the application — window displaying frames with face aligned landmarks, window displaying a bar chart of the emotion probability classification for current frame and a window containing line plot of classification development in time (see Figure 6.1).

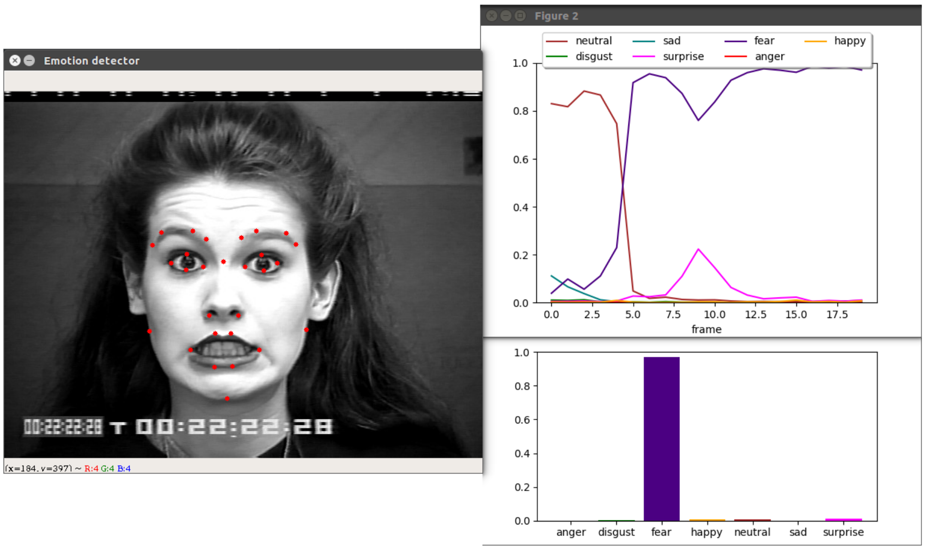
6.3.3 Accompanying scripts

Functionality that is not required by the main application was condensated into accompanying scripts. These scripts perform data transformation for each of the source datasets, combination of datasets, data preprocessing, training of the deep models and evaluation of the trained models.

6.3.4 Trained models

SVM models were trained in the Jupyter Notebook and are stored in binary form using serialization provided by the pickle library. Fine-tuning of the hyperparameters C and g was done by a linear grid search of the 2D parameter space (see Figure 5.10). An SVM classifier was trained and implemented for each combination of featureset (landmark, distance, area, combined), temporal

Figure 6.1: Main application



type (static, sequence, temporal difference) and differentiation, amounting to a total of 24 SVM classifiers.

Deep models were trained by a script. Training was set to early-stop if 10 successive epochs did not improve validation accuracy. Keras Checkpoint callback was used which stores the network weights every time validation accuracy improves. Adam optimization method was found to provide best results. Deep models are stored in h5 format. Information about the learning process was collected using TensorBoard.

Implementation of 10 deep models is provided — pure CNN static classifier, hybrid CNN-RNN sequence classifier and RNN sequence classifier for each feature set and differentiation.

Chapter **7**

Results

To asses the performance of implemented models four metrics are used — accuracy A, average precision P_{avg}, average recall R_{avg} and F-measure F. To calculate these measures a confusion matrix $M \epsilon \mathbb{N}^{C \times C}$, wehre $C \epsilon \mathbb{N}$ is the number of classes, is constructed such that c-th row represents instances where class c is the ground truth and c-th column represents instances where class c is predicted. Metrics are then calculated as:

$$A = \frac{\sum_c^C M_{c,c}}{\sum_c^C \sum_i^C M_{c,i}}$$

$$P_{avg} = \frac{1}{C} \sum_c^C \frac{M_{c,c}}{\sum_i^c M_{i,c}}$$

$$R_{avg} = \frac{1}{C} \sum_c^C \frac{M_{c,c}}{\sum_i^C M_{c,i}}$$

$$F = 2 \frac{P_{avg} * R_{avg}}{P_{avg} + R_{avg}}$$

Accuracy is the ratio of correct predictions to the total number of instances, precision for class c is the number of correct predictions of class c to the total number of predictions of class c, recall for class c is the number of correct predictions of class c to the total number of instances in class c and F-measure is the weighted average of Precision and Recall.

There are two sets of performance measurements that were collected. First was collected using 5-fold cross validation and this set does not contain measurements for deep models because retraining them is very time intensive. Because many papers publish performance of proposed models on the CK+ dataset, second set was collected by training models on the MMI and MUG datasets and splitting CK+ dataset in halves, using one half for training and one for testing.

7.1 Cross validation performance

This section lists performance metrics for SVM models measured using 5-fold cross validation. Note that these metrics show performance on datasets in the same form as was used for training (static classifiers use extracted static images, sequence classifiers use 10-frame subsequences). Therefore these values can not be used to make comparsion between different temporal types.

Table 7.1: Cross validation performance, static SVM classifiers

Feature set	Differential	A	P_{avg}	R_{avg}	F
Landmarks	No	81.2%	80.6%	80.3%	0.805
Landmarks	Yes	83.9%	83.7%	83.4%	0.836
Distance	No	79.9%	79.4%	78.0%	0.787
Distance	Yes	85.0%	85.1%	84.5%	0.848
Area	No	73.6%	73.0%	72.1%	0.725
Area	Yes	79.0%	79.1%	77.9%	0.785
Combined	No	80.5%	80.1%	78.0%	0.790
Combined	Yes	82.6%	82.4%	82.2%	0.823

Table 7.2: Cross validation performance, sequence SVM classifiers

Feature set	Differential	A	P_{avg}	R_{avg}	F
Landmarks	No	73.2%	71.4%	71.0%	0.712
Landmarks	Yes	82.7%	81.0%	79.6%	0.803
Distance	No	72.9%	70.4%	70.2%	0.703
Distance	Yes	81.7%	79.1%	77.4%	0.782
Area	No	66.7%	63.8%	63.3%	0.636
Area	Yes	79.9%	78.9%	77.2%	0.781
Combined	No	73.9%	71.4%	71.0%	0.712
Combined	Yes	83.6%	83.4%	80.9%	0.821

Table 7.3: Cross validation performance, temporal SVM classifiers

Feature set	Differential	A	P_{avg}	R_{avg}	F
Landmarks	No	69.9%	72.1%	67.2%	0.695
Landmarks	Yes	75.0%	77.1%	71.7%	0.743
Distance	No	67.6%	68.6%	64.3%	0.664
Distance	Yes	73.3%	74.7%	68.7%	0.716
Area	No	63.6%	63.7%	61.7%	0.627
Area	Yes	72.5%	73.6%	69.0%	0.712
Combined	No	69.4%	72.0%	66.2%	0.690
Combined	Yes	74.9%	78.0%	71.5%	0.746

It is apparent that area features consistently achieve worst performance in all temporal types. Differentiation of the features relative to neutral expression greatly imporves results as expected. Landmark, length and combined feature sets perform similiarly and each achieves best performance in one of the three temporal types. The F-measure is very close to the accuracy in all classifiers which suggests that the models are not biased towards single class. To better understand the relative difficulty of each class within the temporal type, aforementioned metrics were calculated on emotion basis instead of classifier basis. Note that accuracy is not very useful metric in emotion-wise performance as each emotion is considered in one-against-all manner and thus becomes a minority class.

Table 7.4: Cross validation performance emotion-wise, static SVM classifiers

Emotion	A	P_{avg}	R_{avg}	F
Anger	93.3%	76.1%	80.1%	0.781
Disgust	94.3%	81.5%	80.9%	0.812
Fear	94.2%	72.1%	64.7%	0.682
Happy	96.9%	91.0%	89.4%	0.902
Neutral	92.4%	70.5%	85.7%	0.773
Sad	94.7%	80.1%	73.2%	0.765
Surprise	95.5%	91.6%	82.8%	0.870

Table 7.5: Cross validation performance emotion-wise, sequence SVM classifiers

Emotion	A	P_{avg}	R_{avg}	F
Anger	92.3%	73.7%	77.6%	0.756
Disgust	92.9%	72.3%	76.2%	0.742
Fear	91.5%	63.8%	54.6%	0.589
Happy	96.2%	91.4%	85.3%	0.883
Neutral	92.5%	65.5%	70.1%	0.677
Sad	94.3%	77.3%	69.9%	0.734
Surprise	94.0%	80.5%	83.1%	0.818

Table 7.6: Cross validation performance emotion-wise, temporal SVM classifiers

Emotion	A	P_{avg}	R_{avg}	F
Anger	92.3%	73.7%	77.6%	0.756
Disgust	92.9%	72.3%	76.2%	0.742
Fear	91.5%	63.8%	54.6%	0.589
Happy	96.2%	91.4%	85.3%	0.883
Neutral	92.5%	65.5%	70.1%	0.677
Sad	94.3%	77.3%	69.9%	0.734
Surprise	94.0%	80.5%	83.1%	0.818

These measurements show that fear is the most difficult expression to recognize for all temporal types of classifiers while happiness is the easiest. Static classifiers appear to be best-fit for neutral classification.

7.2 CK+ performance

Performance on CK+ dataset was measured such that MMI, MUG and training half of CK+ dataset were used for training. The testing half of CK+ dataset was then used to measure performance. This performance was measured using the final wrapped models on the original dataset. Using the knowledge from the Data transformation step where the shortest neutral phase was 5 frames long, frame threshold is defined $T = 5$. Each sequence $S = \{S_1, ..., S_l\}$ is assigned predicted label c_S equal to the emotion with maximum cummulative probability for frames after the frame threshold.

$$c_S = \underset{c}{argmax} \sum_{i=T+1}^{l} p_c(S_i)$$

where p_c is the predicted probability of frame S_i belonging to class c.

Table 7.7: CK+ performance, static classifiers

Feature set	Classifier type	Differential	A	P_{avg}	R_{avg}	F
Image	CNN	No	88.8%	84.5%	85.4%	0.849
Landmarks	SVM	No	93.3%	90.0%	91.1%	0.906
Landmarks	SVM	Yes	86.9%	81.7%	84.9%	0.832
Distance	SVM	No	92.4%	88.8%	89.2%	0.890
Distance	SVM	Yes	85.3%	80.4%	83.4%	0.819
Area	SVM	No	86.9%	83.3%	84.5%	0.839
Area	SVM	Yes	85.3%	79.9%	83.6%	0.817
Combined	SVM	No	93.6%	91.5%	91.1%	0.913
Combined	SVM	Yes	85.0%	80.2%	83.1%	0.816

Most feature sets with SVM classifier outperformed the deep CNN classifier. An interesting observation is that contrary to the cross validation case, differentiation on original CK+ dataset leads to decrease in performance. This might be explained by the static SVM classifier, which is used in the differentiating wrapper models to detect neutral faces, making more incorrect predictions on the CK+ dataset. Area features exhibit the worst performance and landmark, distance and combined features perform similiarily, which is consistent with measurements done by cross validation.

Table 7.8: CK+ performance emotion-wise, static classifiers

Emotion	A	P_{avg}	R_{avg}	F
Anger	94.7%	89.1%	70.6%	0.788
Disgust	96.2%	92.8%	85.9%	0.892
Fear	97.3%	82.5%	86.2%	0.843
Happy	98.8%	96.7%	97.9%	0.973
Neutral	95.3%	55.1%	86.4%	0.673
Sad	96.1%	76.9%	80.2%	0.785
Surprise	98.7%	98.3%	96.7%	0.975

Emotion-wise performance supports previous hypothesis. The lowest emotion-wise performance for static classifiers on CK+ dataset is the neutral emotion, which may explain the worse performance of differentiating models.

Table 7.9: CK+ performance, sequence classifiers

Feature set	Classifier type	Differential	A	P_{avg}	R_{avg}	F
Image	CNN+RNN	No	91.9%	90.6%	89.0%	0.898
Landmarks	RNN	No	85.3%	80.2%	75.8%	0.779
Landmarks	RNN	Yes	83.2%	76.6%	77.9%	0.772
Distance	RNN	No	83.5%	76.8%	76.2%	0.765
Distance	RNN	Yes	82.9%	76.8%	74.2%	0.754
Area	RNN	No	81.0%	76.6%	76.0%	0.763
Area	RNN	Yes	71.9%	71.1%	66.3%	0.686
Combined	RNN	No	81.3%	74.1%	73.8%	0.740
Combined	RNN	Yes	75.5%	72.9%	67.6%	0.702
Landmarks	SVM	No	91.4%	90.2%	86.7%	0.884
Landmarks	SVM	Yes	88.4%	84.3%	83.3%	0.838
Distance	SVM	No	89.9%	89.1%	82.3%	0.856
Distance	SVM	Yes	84.1%	82.1%	78.0%	0.800
Area	SVM	No	87.2%	88.6%	79.1%	0.836
Area	SVM	Yes	84.1%	78.6%	80.4%	0.795
Combined	SVM	No	93.6%	93.7%	88.7%	0.912
Combined	SVM	Yes	86.9%	81.1%	82.1%	0.816

The temporal information captured by RNN network from CNN-extracted features considerably improved on accuracy of the pure-CNN model from 88.8% to 91.9%. SVM sequence classifiers show no significant improvement over their static variants, in some cases the performance is worse. SVM sequence classifiers consistently outperform pure-RNN sequence classifiers.

Table 7.10: CK+ performance emotion-wise, sequence classifiers

Emotion	A	P_{avg}	R_{avg}	F
Anger	95.2%	90.0%	72.5%	0.803
Disgust	96.8%	95.2%	86.4%	0.906
Fear	96.9%	79.4%	86.0%	0.826
Happy	98.7%	95.2%	98.9%	0.970
Neutral	96.7%	67.6%	81.2%	0.738
Sad	96.1%	75.4%	85.7%	0.802
Surprise	97.8%	95.9%	95.5%	0.957

Table 7.11: CK+ performance, temporal classifiers

Feature set	Classifier type	Differential	A	P_{avg}	R_{avg}	F
Landmarks	SVM	No	95.1%	93.3%	92.6%	0.929
Landmarks	SVM	Yes	85.9%	81.4%	83.7%	0.825
Distance	SVM	No	92.4%	90.4%	87.1%	0.887
Distance	SVM	Yes	85.6%	80.3%	83.8%	0.820
Area	SVM	No	93.3%	91.0%	92.7%	0.918
Area	SVM	Yes	81.3%	76.3%	79.9%	0.781
Combined	SVM	No	94.2%	92.1%	91.1%	0.916
Combined	SVM	Yes	84.7%	79.5%	81.9%	0.807

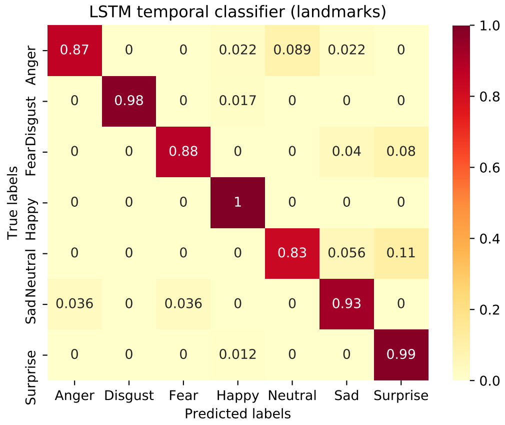
Temporal differentiation outperforms either of other temporal types examined in this thesis achieving accuracy of 95.1% with landmark, non-differential feature set. It scores higher accuracy with all other feature sets as well.

Table 7.12: CK+ performance emotion-wise, sequence classifiers

Emotion	A	P_{avg}	R_{avg}	F
Anger	94.3%	86.7%	69.7%	0.773
Disgust	94.7%	87.9%	81.5%	0.846
Fear	95.2%	73.1%	67.8%	0.703
Happy	97.8%	92.0%	98.6%	0.952
Neutral	95.0%	65.1%	60.1%	0.625
Sad	95.0%	70.6%	75.4%	0.730
Surprise	97.7%	94.0%	97.7%	0.958

Figure 7.1 shows the confusion matrix for the best model.

Figure 7.1: Confusion matrix of the best model



7.3 Disadvantages of used methods

Because this thesis focused on analysis of 2D images and all of the datasets used capture subjects looking directly into the camera in laboratory conditions, there is very little head pose variance in the dataset. As a result the trained models are sensitive to the pose and work well only when subject is looking directly into the camera. Using dataset with 3D landmark annotation would alleviate this problem but obtaining a 3D landmark annotation in real time requires special equipment. One of the goals for this thesis was to create a model that would perform recognition based on 2D streams with sources such as web camera. Figure 7.2 demonstrates how prediction changes for constant smile expression when the subject is looking to the sides.

One of qualitative aspect is stability of prediction. Because static models do not use any temporal information about the development of expression their predictions can be erratic, especially if there is a large amount of noise in the source images. Figure 7.3 demonstrates the stability of static and sequence classifier respectively when transitioning from smile to surprise face.

Figure 7.2: Prediction development based on head pose

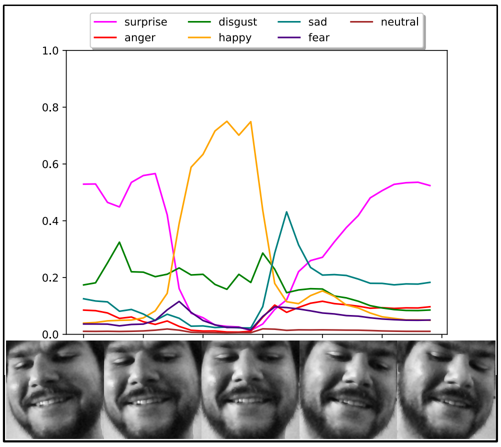
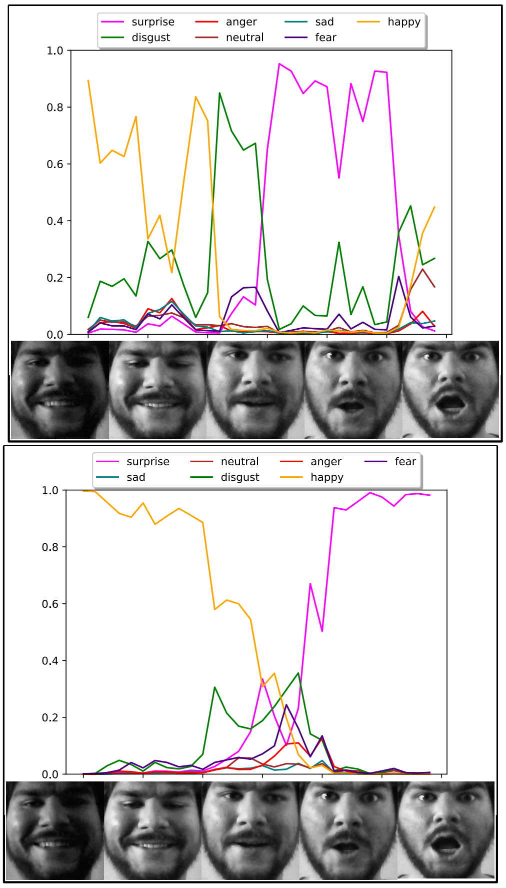


Figure 7.3: Prediction stability for static classifier (top) and sequence classifier (bottom)



Conclusion

The task of this thesis was to propose and implement a model for human emotion state recognition based on a sequence of frontal face images captured from video stream with the output of probabilities for each emotion.

After researching the nature of emotion expression and the psychological background work done primarily by Paul Ekman, 7 basic emotional states were selected for recognition — anger, disgust, fear, happiness, neutral, sadness and surprise. The research of previous related work led to the decision of using facial landmarks and CNN-extracted embeddings as descriptors of the facial expression for emotion recognition.

Based on a survey of available viable datasets four of them were selected — MMI [29], CK+ [26], MUG [30] and JAFFE [28]. An increased attention was paid to the vaidation, transformation, preprocessing and normalization of the data. Multiple preprocessing techniques like the histogram equalization or face angle estimation and correction were used.

Four feature sets based on facial landmarks are proposed. The main source of information when designing these sets was the observation of prototypic Action Units (AUs, movements of muscles or groups of muscles changing the appearence of the face) in the Facial Action Coding System (FACS) [9]. A subset of originally detected landmarks with equal information value was selected. Using this subset distance-based and area-based features best describing the AUs were proposed. Last feature set is a concatenation of the three sets.

Because of the temporal properties of human emotion three types of temporal context were proposed. One set of models performs recognition on frame-by-frame basis having no temporal context at all. Second set of models utilises temporal differentiation — each frame is extended by the changes respective to previous frames in the sequence. Third set of models focuses on 10-frames long subsequences, initially trained with subsequences of expression onset, apex and offset.

Inspired by the success of Convolutional Neural Networks (CNN) in various copmuter vision tasks in recent years and by the work of Winkler et al. [19]

a deep CNN model based on the InceptionV3 architecture [33] was proposed. Because training deep networks require a lot of data transfer-learning method was deployed. The learned CNN model was used to create a hybrid CNN RNN network which utilises the CNN network as feature extractor.

The landmark features are processed by RNN networks and Support Vector Machine (SVM) classifiers. Relatively shallow RNN of 5 layers utilising LSTM cells was used. This architecture was selected after experimentation with different architectures. Out of conventional classification methods the Radial Basis Function (RBF) SVMs produced best results.

In order to ensure usability of produced models on real-time streams such as web cameras a wrapper framework encapsulating the classifiers was proposed. Frames are presented one at the time to the wrapper and it handles all necessary feature extraction and classification internally.

Proposed solution was implemented in Python programming language. Deep networks use TensorFlow backend with Keras high-level library for model definition and training. An application consuming a video stream, video file or a directory with sequence of images and displaying the emotion prediction in real time was implemented.

A total of 10 deep models and 24 SVM classifiers were implemented and their performance examined and compared. Models were tested on the CK+ dataset for comparability of achieved results with other related work. SVM classifier with temporal differentiation context using selected landmarks as features achieved the best performance — accuracy of 95.1%.

Bibliography

[1] EKMAN, P. Introduction. *Annals of the New York Academy of Sciences*, volume 1000, no. 1, 2003: pp. 1–6, ISSN 1749-6632, doi: 10.1196/annals.1280.002. Available from: `http://dx.doi.org/10.1196/annals.1280.002`

[2] Ekman, P. Happy, sad, angry, disgusted. volume 184, 10 2004: pp. 4–5.

[3] Abhang, P.; Rao, S. N.; et al. Emotion Recognition Using Speech and EEG Signal –A Review. 2011.

[4] Ekman, P. Expression and the nature of emotion. *Approaches to emotion*, volume 3, 1984: pp. 19–344.

[5] Ekman, P. Are there basic emotions? 1992.

[6] Darwin, C. *The Expression of the Emotions in Man and Animals*. Cambridge Library Collection - Darwin, Evolution and Genetics, Cambridge University Press, second edition, 2009, doi:10.1017/CBO9780511694110.

[7] Ekman, P.; ; et al. Facial Expressions of Emotion. *Annual Review of Psychology*, volume 30, no. 1, 1979: pp. 527–554, doi:10.1146/annurev.ps.30.020179.002523, `https://doi.org/10.1146/annurev.ps.30.020179.002523`. Available from: `https://doi.org/10.1146/annurev.ps.30.020179.002523`

[8] Gendron, M.; Roberson, D.; et al. Perceptions of emotion from facial expressions are not culturally universal: evidence from a remote culture. *Emotion*, volume 14, no. 2, Apr 2014: pp. 251–262.

[9] Ekman, P.; Friesen, W. V. Measuring facial movement. *Environmental psychology and nonverbal behavior*, volume 1, no. 1, 1976: pp. 56–75.

[10] Viola, P.; Jones, M. J. Robust real-time face detection. *International journal of computer vision*, volume 57, no. 2, 2004: pp. 137–154.

[11] Zhu, Q.; Yeh, M.-C.; et al. Fast human detection using a cascade of histograms of oriented gradients. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, IEEE, 2006, pp. 1491–1498.

[12] Kazemi, V.; Josephine, S. One millisecond face alignment with an ensemble of regression trees. In *27th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, United States, 23 June 2014 through 28 June 2014*, IEEE Computer Society, 2014, pp. 1867–1874.

[13] Suk, M.; Prabhakaran, B. Real-time mobile facial expression recognition system-a case study. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2014, pp. 132–137.

[14] Cootes, T. F.; Taylor, C. J.; et al. Active shape models-their training and application. *Computer vision and image understanding*, volume 61, no. 1, 1995: pp. 38–59.

[15] Ghimire, D.; Lee, J. Geometric feature-based facial expression recognition in image sequences using multi-class adaboost and support vector machines. *Sensors*, volume 13, no. 6, 2013: pp. 7714–7734.

[16] Wiskott, L.; Krüger, N.; et al. Face recognition by elastic bunch graph matching. *IEEE Transactions on pattern analysis and machine intelligence*, volume 19, no. 7, 1997: pp. 775–779.

[17] Happy, S.; George, A.; et al. A real time facial expression classification system using local binary patterns. In *Intelligent Human Computer Interaction (IHCI), 2012 4th International Conference on*, IEEE, 2012, pp. 1–5.

[18] Ghimire, D.; Jeong, S.; et al. Facial expression recognition based on local region specific features and support vector machines. *Multimedia Tools and Applications*, volume 76, no. 6, 2017: pp. 7803–7821.

[19] Ng, H.-W.; Nguyen, V. D.; et al. Deep learning for emotion recognition on small datasets using transfer learning. In *Proceedings of the 2015 ACM on international conference on multimodal interaction*, ACM, 2015, pp. 443–449.

[20] Chatfield, K.; Simonyan, K.; et al. Return of the devil in the details: Delving deep into convolutional nets. *arXiv preprint arXiv:1405.3531*, 2014.

[21] Jung, H.; Lee, S.; et al. Joint fine-tuning in deep neural networks for facial expression recognition. In *Computer Vision (ICCV), 2015 IEEE International Conference on*, IEEE, 2015, pp. 2983–2991.

[22] Breuer, R.; Kimmel, R. A Deep Learning Perspective on the Origin of Facial Expressions. *arXiv preprint arXiv:1705.01842*, 2017.

[23] Ebrahimi Kahou, S.; Michalski, V.; et al. Recurrent neural networks for emotion recognition in video. In *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*, ACM, 2015, pp. 467–474.

[24] Le, Q. V.; Jaitly, N.; et al. A simple way to initialize recurrent networks of rectified linear units. *arXiv preprint arXiv:1504.00941*, 2015.

[25] Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural computation*, volume 9, no. 8, 1997: pp. 1735–1780.

[26] Lucey, P.; Cohn, J. F.; et al. The extended cohn-kanade dataset (ck+): A complete dataset for action unit and emotion-specified expression. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on*, IEEE, 2010, pp. 94–101.

[27] Kanade, T.; Cohn, J. F.; et al. Comprehensive database for facial expression analysis. In *Automatic Face and Gesture Recognition, 2000. Proceedings. Fourth IEEE International Conference on*, IEEE, 2000, pp. 46–53.

[28] Lyons, M. J.; Akamatsu, S.; et al. The Japanese female facial expression (JAFFE) database. In *Proceedings of third international conference on automatic face and gesture recognition*, 1998, pp. 14–16.

[29] MMI Facial Expression Database. Accessed: 2018-03-01. Available from: `https://www.mmifacedb.eu/`

[30] Aifanti, N.; Papachristou, C.; et al. The MUG facial expression database. In *Image analysis for multimedia interactive services (WIAMIS), 2010 11th international workshop on*, IEEE, 2010, pp. 1–4.

[31] Sneddon, I.; McRorie, M.; et al. The belfast induced natural emotion database. *IEEE Transactions on Affective Computing*, volume 3, no. 1, 2012: pp. 32–41.

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

[33] Szegedy, C.; Vanhoucke, V.; et al. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2818–2826.

[34] Szegedy, C.; Liu, W.; et al. Going deeper with convolutions. Cvpr, 2015.

[35] Hunter, J. D. Matplotlib: A 2D graphics environment. *Computing In Science & Engineering*, volume 9, no. 3, 2007: pp. 90–95, doi:10.1109/MCSE.2007.55.

Appendix A

Acronyms

FACS Facial action coding system

AU Action unit

FER Facial expression recognition

SVM Support vector machine

ANN Artificial neural network

CNN Convolutional neural network

RNN Recurrent neural network

RBF Radial basis function

ASM Active shape model

EBG Elastic bunch graph

DTW Dynamic time warping

LBP Local binary pattern

PCA Principal component analysis

LSTM Long-short term memory (cell)

HE Histogram Equalization

AHE Adaptive Histogram Equalization

CLAHE Contrast-Limited Adaptive Histogram Equalization

CDF Cummulative Distribution Function

HOG Histogram of Oriented Gradients

COG Center of gravity

Appendix B

Contents of enclosed CD

- `readme.txt` the file with CD contents description
- `src` the directory of source codes
  - `emotion_detect` implementation sources
  - `thesis` the directory of LaTeX source codes of the thesis
- `text` the thesis text directory
  - `thesis.pdf` the thesis text in PDF format