Emotion Predictor: Machine Learning Based Prediction of Emotions using Facial Features

Andreas Emch
08-631-384

Master Thesis
Mai 2019

Prof. Dr. Markus Gross

Supervisors:
Rafael Wampfler
Dr. Severin Klingler

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Abstract

Software is mostly limited to the input of a keyboard and a mouse, ignoring the video which is available for most hardware devices. Therefore, a learning program or a game cannot adapt to the user affective state, ignoring the fact that a bad mood can affect the learning rate negatively.

In this thesis, we investigated how to predict the affective state of a person base on facial expression and movements. With an experiment, we conducted the affective state ratings of several people looking at pictures and solving math tasks. With our model, we performed with an accuracy of around 0.8 for predicting the valence, whereas current solutions achieved only slightly above the random level. This states the importance of considering the individual and subtle reactions of each person in this context. An important finding was that the success message from the math tasks had significant influence about in the affective state, which is essential for learning software to adapt their level of difficulty if needed.

Solving exercises at home on a tablet limits the software to rely on the built-in camera, which is most likely filming the ceiling for most of the time. With a small attached mirror on the tablet and our video restoration pipeline to reconstruct the face, we improved the facial recognition to be valid almost always. Without these adjustments, only a few sequences would have been useful from the front camera.

Zusammenfassung

Heutige Software beschränkt sich meistens nur auf die Tastatur und Maus als Eingabegeräte und vernachlässigt die Kamera, welche viele Geräte integriert haben. Somit kann eine Lernsoftware oder ein Spiel nicht auf affektiven Zustand des Benutzers eingehen und ignoriert die Tatsache, dass eine schlechte Stimmung einen negativen Einfluss auf die Lernfähigkeit hat.

In dieser Arbeit haben wir untersucht, wie der affektive Zustand einer Person anhand von dem Gesichtsausdruck und Bewegungen vorhergesagt werden kann. Mit einem Experiment haben wir von mehreren Personen die Bewertung ihres affektiven Zustandes ermittelt, während diese Bilder angesehen und Mathematik Aufgaben gelöst haben. Mit unserem Model konnten wir die Valenz mit einer Genauigkeit von etwa 0.8 voraussagen, während aktuelle Lösungen nur geringfügig über dem Zufallsniveau lagen. Dies zeigt die Wichtigkeit, die individuellen und subtilen Reaktionen jedes Menschen in diesem Zusammenhang zu berücksichtigen. Ein wichtiges Ergebnis war, dass die Erfolgsmeldung von einer mathematischen Aufgabe einen signifikanten Einfluss auf den affektiven Zustand hatte. Hier kann eine Lernsoftware ansetzten und falls nötig den Schwierigkeitsgrad anpassen.

Das Lösen von Übungen zu Hause auf einem Tablet schränkt die Software auf die eingebaute Kamera ein, die wahrscheinlich die meiste Zeit die Decke filmt. Mit Hilfe eines kleinen Spiegels auf dem Tablet und unserer Video-Pipeline zur Rekonstruktion des Gesichts verbesserten wir die Gesichtserkennung erheblich. Ohne diese Anpassungen konnten wir von der Frontkamera beinahe nichts verwenden.

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

computer graphics laboratory

Master Thesis

Emotion Predictor: Machine Learning Based Prediction of Emotions using Facial Features

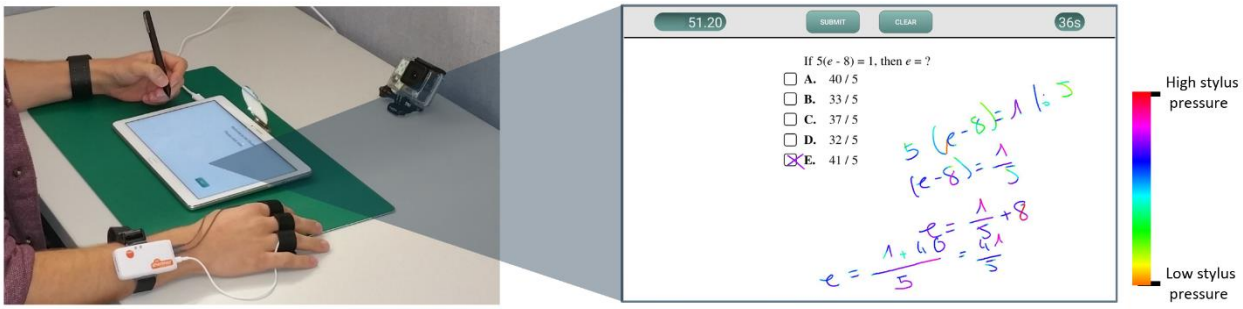


Figure. A participant during an experimental session. The task interface allows users to write solution paths directly onto the screen (the stylus pressure is color-coded for visualization purposes only).

Project Description

Being able to predict and understand human emotions is important in many areas, such as education, games and movies. Especially in education it is crucial to detect when a learner is in a bad mood which can negatively influence the learning gain. Recently we have conducted a large user study where we have collected emotions as well as video recordings of users while they have been solving math tasks (active part) and looking at pictures (passive part). In this thesis we want to analyze this dataset to build predictive models of the user emotions based on video recordings (facial expression, eye gaze, head movement) and investigate if such a model can be generalized over different domains (pictures and math).

Tasks

The main task of the thesis is the development of a data-driven model for the prediction of the emotional state of a person based on the facial expression, eye gaze, head movement and distance to the screen. Moreover, it should be investigated if the built model generalizes over domains (from pictures to math and vice versa). Last but not least, an analysis of the best performing models should allow for insights into how to improve the prediction further. The four main tasks of this project are

- Validation and statistical analysis of the interaction log files and video data (e.g., correlation of self-reports with emotions retrieved from facial expressions)
- Timestamp synchronization, preprocessing and feature extraction from the video data.
- Creating predictive models for the user emotion while the user was solving math tasks and looking at pictures based on features extracted from the video data. Investigating the generalizability by applying the model built on math tasks to the task consisting of watching at pictures and vice versa.
- Preprocessing the tablet front cam data by stitching the two parts together (separated by the mirror) and incorporating the data into the model.

Remarks

A written report and an oral presentation conclude the thesis. The thesis will be overseen by Prof. Dr. Markus Gross and supervised by Rafael Wampfler, Dr. Severin Klingler and Dr. Barbara Solenthaler. The start date of the thesis is November 19, 2018. The end date of the thesis will be Mai 20, 2019.

Acknowledgments

First of all, I would like to express my sincere gratitude to my two supervisors, Rafael Wampfler and Dr. Severin Klingler of the Computer Graphics Lab of ETH Zurich for the support, expertise and time they dedicated to me.

I would like to thank Prof. Dr. Markus Gross of ETH Zurich for giving me the opportunity to write this thesis at the Computer Graphics Lab.

A special thanks to all participants of the experiment who allowed me to use the video footage for this thesis.

I would also like to thank Simon Tännler for proofreading this thesis.

Last but not least, I would like to thank Nora Finklenburg, my close friends and family whose great discussions, challenging questions and new ideas supported me and my project.

Contents

List of Figures

List of Tables

1

Introduction

Our current affective state is controlling our behavior, the way we talk and how we interact with other people and our computer or mobile device. Even if the affective state of a person is not directly visible, we can predict it by observing the behavior and movements, and listening carefully to the voice. An essential role in this process plays the facial expressions, which are directly linked with our feelings and show temporal emotions. Furthermore, based on the work of Schachter [Sch64], emotional and body state changes have a bidirectional link.

In a recent study with over 100 students, Febrilia and Warokka [Feb14] found that a negative mood has a negative influence on learning. Interestingly, a positive attitude does not affect the learning process. These findings lead to the importance of a pleasant atmosphere and that the students have positive feelings for the subject. By observing students solving, for example, in-class exercises, a teacher can recognize a student who is always frustrated about the tasks. Possible hints can be an angrily starring on the paper, the way wrong answers are crossed or the helpless look. In this case, the teacher can directly interact with the student and can provide further help (e.g., more straightforward exercises or additional hints), which increases the probability of the student being more successful. This direct interaction is jeopardized by having either too many students per class or by letting the students solve the tasks at home. If enough personal resources are available, additional assistants for exercise lessons can be involved. However, this solution only solves the first issue. A solution to address problems that students encounter when solving tasks at home is the use of tablets for solving these tasks. Recently, tablets are more and more used in schools already to organize the school-related daily business. With the integrated camera, the student could be observed, and when finding that the student is mainly in a negative mood, the teacher can be informed. This solution relies on the fact that the emotions are shown through facial expressions and can give hints about the affective state of the person.

Similar patterns can be applied for example to video games. Some players can easily beat the game on hard settings and another player might fail badly with the same settings, resulting in

a bad experience of the game. Additionally, it might be hard to find a difficulty level, which is challenging enough and still let the player reach some achievements. Furthermore, when the game contains different mechanics, for example driving and jumping around, a player might be driving around too easily with the current degree of difficulty but spends hours for reaching the next target with jumping over platforms. Continually changing the degree of difficulty is not an option for most players. For both examples described above, usually a single person interacts with an electronic device, but the software is not responding to the affective state of the user. By interpreting and correctly reacting to the emotional changes of the user, the experience can be significantly improved. This includes adapting the level of difficulty based on the user's performance to guarantee a reasonable learning rate.

The first step for a machine to adapt to the affective state of a person using it is to recognize the person correctly and to read the facial expressions accurately. Several frameworks are already tackling this issue and can be used to analyze a webcam stream or a recorded video. Two important frameworks are Affectiva [McD16] and OpenFace [Bal16]. Both are extracting the facial landmarks and action units (please see Chapter 2) which can be used to interpret the facial expression. Additionally, Affectiva supports predicting some emotions out of the box. Based on their website[1], they are using datasets with millions of samples. However, there are no further statements on how these datasets are composed and what situations are addressed within these datasets. Analyzing available labeled datasets[2], many are either in a staged environment or consist only of pictures but not videos. This stands in sharp contrast to the above-described usages of facial expression analysis, where emotions can also occur subtly and sometimes not as defined as in a staged situation. Another problem is that the importance of dynamic facial expressions is ignored, which is found to be an essential aspect for recognizing emotions more correctly and better differentiate between genuine and fake expressions [Kru13].

In some circumstances, the available camera of the device is not optimally arranged to capture the person accurately. For example, when solving exercises for school on a tablet, there is a high probability that only the ceiling is recorded with the integrated front camera. If available, an external video camera can be installed to record the person. But having two video sources adds the complexity to synchronize the different timestamps and to combine different view angles. With various camera angles a higher inconsistency of emotion recognition between the sources is expected and based on the camera angle, some emotions might be over- or underestimated (e.g., filming top-down might result in overestimating anger as the eyebrows tend to look angrily). This was studied by Landowska and colleagues [Lan17].

Correctly reading the facial expression of a person is one thing, interpreting it accurately and to properly distinguish between fake expressions is another. It is shown that people are capable of pretending emotions, being it to gain a social advantage in a society [KM09] or to solicit the help of others [Ekm09]. Missing an additional context of the person (e.g., social or cultural circumstances) can complicate the correct classification of the underlying emotion shown in the facial expression. This additional information might not always be available in real-life situations where a person interacts with another person or a device. Furthermore, Matsumoto [Mat90] analyzed cultural similarities and differences of the display rules. Display rules are informal rules which describe how someone has to behave and express themselves in everyday

[1] https://blog.affectiva.com/emotion-ai-101-all-about-emotion-detection-and-affectivas-emotion-metrics
[2] https://en.wikipedia.org/wiki/Facial_expression_databases

situations (e.g., some cultures dampen their negative expressions in the presence of strangers). These effects can negatively influence the accuracy of an emotion prediction.

1.1. Focus of this Work

In this thesis, our primary focus was on investigating the performance for predicting affective states of students while looking at pictures and solving math tasks. This addresses the solution to help teaching facilities to improve the learning rate of their students. Each student could be equipped with a tablet for solving different exercises, and the teaching staff can observe the affective state and interact if needed. Furthermore, the software could autonomously adapt the difficulty level and give additional hints if needed.

Using tablets for solving exercises at home limits the application to rely on the front camera of the tablet. Based on the person's position, the camera might only record the ceiling and therefore cannot adapt appropriately. With the second focus on our work, we investigated on how to overcome this limitation with only small hardware adjustments. With this adaption, we needed a software framework to restore the face in the video to improve the detection rate with a face detection software.

1.2. Thesis Organization

First, in Chapter 2, we give an overview of previous research in the field of emotion recognition and image restoration. In Chapter 3, it is described how the dataset was collected and a first analysis of the dataset is given. The main focus is covered in Chapter 4 and Chapter 5, where we describe the model and results for the classification problem, respectively. The second focus, the framework to improve face detection with the front camera adjustments, is demonstrated in Chapter 6. We finish with reflecting on our work in Chapter 7 and provide an outlook of possible future work in Chapter 8.

2

Related Work

This chapter briefly introduces research related to the topic of this thesis.

Self Assessment Manikin. For experiments including a rating about the affective state of a participant, it is challenging to find an appropriate rating scale. Using a predefined list with a few available emotions can be too limiting, and the participant might not be able to match his emotion with the available options. Another issue can be the naming of the emotions, as each person can have a slightly different meaning for the same word. To let the participant write his emotion will only help partially during the experiment, but will lead to a more sophisticated analysis later.

Lang [Lan80] addressed these issues and introduced a picture-oriented scale, namely the Self-Assessment Manikin (SAM) (please see Figure 2.1). Based on the previous work of Osgood and colleagues [Osg57], it uses three different dimensions: valence, arousal, and dominance. A participant can choose from nine different levels for each aspect. This rating with pictures allows the participant to capture the idea quickly and to rate the affective state intuitively.

Based on the findings of Lang and colleagues [LB07], only valence and arousal have been used in this experiment. Dominance can lead to confusion (e.g., is the object in the picture dominant or feels the participant dominant) and seems to be highly correlated to the valence-dimension for symbolic sensory stimuli (e.g., images, sound, voice, etc.). Though, for other experiments containing social interactions, this dimension might play an essential role.

Facial Action Coding System (FACS). A commonly used system for describing various emotions is the Facial Action Coding System (FACS), which has been introduced by Paul Eckman and Wallace Friesen in 1978 [PE78]. The idea was to describe any facial movement in a movie or a single picture. To do so, the system relies on different anatomically based action units. Each action unit itself can be represented by a contraction of a muscle. By activating many muscles in the face, and therefore activating the corresponding action unit, a facial movement can be described [Coh07]. An emotion often triggers a facial movement. Therefore it can be described

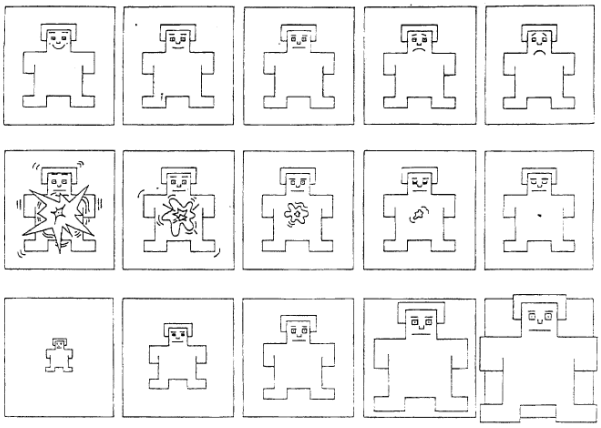


Figure 2.1.: *The Self-Assessment Manikin (SAM) [BL94]. The following affective dimensions can be rated: valence (top panel), arousal (middle panel) and dominance (bottom panel).*

as a union of several action units. For example, to express the emotion "joy", action units 6 (Check Raiser) and 12 (Lip Corner Puller) are activated. Nowadays, many facial recognition systems work with the FACS, for example, Affectiva and OpenFace. Based on the framework, different action units are supported (please see Table A.2).

Driver Drowsiness. Facial expressions are not only used for investigating the affective state but can conclude other states of a person, for example, the drowsiness. By observing the eye blinking and yawning, irregularities can be detected. These can be hints that the driver is slowly falling asleep. Both features can be directly extracted by calculating a ratio between different pairs from facial landmarks [ASP18]. Therefore, facial landmarks for the eyes are used within an equation to calculate the ratio between the height and width of the eye. With a threshold can be defined whether the eye is open or closed. Analyzing this over a more extended period, irregularities (e.g., having the eye closed for longer intervals) can be extracted, and the driver can be alerted with a sound signal to stay awake and focused. Additional features about the eyelid can give more evidence about the drowsiness state of a driver. Hu and Zheng [HZ09] analyzed the prediction based on the closing and opening speed of the eyelid and other eyelid features.

Fidgeting. The fidgeting index can be used to measure how much a person moves during a given period. It does not analyze single movement by detecting if the person is moving forward or backward, but instead uses a definition to calculate the overall energy the person currently uses to move any part of his body. The fidgeting index was introduced for predicting movie ratings from Navarathna and colleagues [Nav14]. In this paper, face and body motion features from people watching movies have been used to determine the rating of the film which was seen. One issue was that multiple people were analyzed simultaneously, and the people should not be mixed during the whole observation time. Navarathna and colleagues have compared various tracking techniques, and they found that defining an initial volume of interest for each person reduced the complexity and increased the tracking performance. A common way to extract motion is by calculating the optical flow [HS81]. Nevertheless, this method is slow for a long video with a higher resolution. The faster possibility is to obtain features from motion history images as introduced by Davis [DB97]. The correlation between features from the optical flow and motion history images is significantly high, with about 85%. Therefore, Navarathna

and colleagues [Nav14] used the motion history images, which are computational very fast to describe the overall movement energy of a person.

For this thesis, the calculation for the fidgeting index was less complex because we only track one person at a time. Additionally, we used the entire screen as the volume of interest, because we knew which person was recorded during which period.

Image Inpainting. Algorithms to inpaint images grew more interest over the last few years. It began with a basic inpainting algorithm considering the known neighborhood as used in OpenCV [Cul12]. For many applications, this can already be a computational easy and fast method. Though, a more complex picture can suffer from some artifacts because the algorithm is not aware of the object, but only of the neighborhood pixels. This can also be an issue if the missing parts of the image become more substantial and the missing pixels are not based on the remaining found neighboring pixels. The latest research has shown some notable improvement by introducing generative convolutional networks [Liu18, ea17, Uly18]. Usually, the model takes two parameters as input, the image to inpaint and the mask to specify the missing regions. With current graphics cards and the easy usage of these computation power with common frameworks (e.g., tensorflow [ea15a], keras [Cho15], etc.), the inpainting of an image can be done fast and straightforward. Some trained models can be used from the papers to start right away with inpainting some pictures. Based on the used images for training these models, the results can be not as satisfying as expected. For example, a model only trained with landscapes is great for removing people from a landscape picture, but restoring a car that is covered by a tree might not work as intended.

For this thesis, we have trained a model by ourselves with training images and masks as we expected in our restoration pipeline. For the training, we only used pictures with faces, because our inpainting is only applied to faces.

3

Dataset

In Section 3.1, we describe the setup for collecting our dataset which is used for this thesis. A first analysis of the data to find possible challenges is done in Section 3.2.

3.1. Experimental Setup

The goal of the experiment was to trigger different affective states while participants have been looking at pictures and solved math tasks. Therefore, images and math tasks have been chosen to trigger a wide range of different emotions. The experiment was conducted at ETH in 2018 with 88 participants (45 female and 43 male, all between 18 and 30 years old).

During the experiment, the participants used a Huawei MediaPad M2 tablet operated by a stylus.

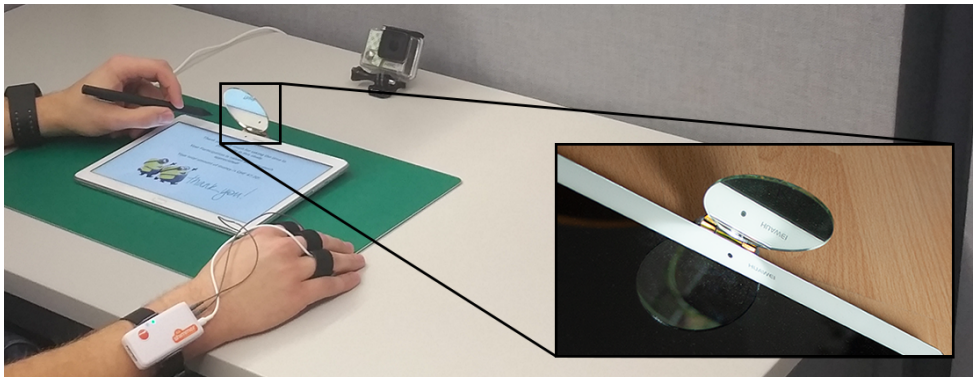


Figure 3.1.: *The experimental setup with the tablet, the GoPro Hero3 and all other sensors (please see description in the text). The circular mirror is attached with a hinge and some glue to the tablet to gain a better view on the participant with the front camera of the tablet.*

Condition	*Pay off*	*Penalty*	*ACT difficulty*	*Time limit*
Repetitive	CHF 0.20	CHF 0.20	0.76, 0.83	60 s - 75 s
Challenge	CHF 2.00	CHF 0.20	0.58 - 0.69	53 s - 93 s
Overchallenge	CHF 0.20	CHF 2.00	0.25 - 0.53	25 s - 51 s

Table 3.1.: *Pay-off and penalty for solving the math tasks correctly and wrong, respectively.*

The participant was recorded with the front camera of the tablet (frame rate per second (fps): mean = 20.022, std = 1.923) and a GoPro Hero3 (fps = 59.94), which has been installed behind the tablet on the table. Because the tablet was lying flat on the table, the view angle of the front camera was suboptimal for recording a participant sitting straight in front of it. To improve the view angle, a small circular mirror has been attached close to the front camera with a hinge and some glue on the tablet (please see Figure 3.1). To measure the skin conductance, a Shimmer GSR3 device was attached to the non-dominant hand of the participant. Skin temperature was measured with an Empatica E4 wristband. Finally, a Polar H10 chest strap has recorded the heart rate during the whole session.

This work only uses the video-recordings. Heart rate and shimmer measurements have only been used to find an optimal window for the baseline video (please see Section 4.5.4).

The experiment consisted of three essential steps, with the first as a baseline part for a person-specific normalization. Second, the participant was conducting a passive part comprised of looking at pictures and an active part consisting of solving math tasks. After each image and each math task, the participant was asked to rate the current valence and arousal.

Baseline. As biological and visual appearances tend to vary a lot between different people, this baseline was used to determine the relaxed state of the participant. To achieve this, a seven-minute long video of a waterfall with calm music is shown. During this period, the participant should be able to completely relax and get into a neutral mood without showing any emotions.

IAPS. For choosing pictures, we relied on the work from the International Affective Picture System (IAPS) dataset [PJL08]. Every image from IAPS comes with a rating for the expected valence, arousal and dominance value, which have been collected over several years and many different people during various studies. Based on these affective values, we defined four different sets (each containing ten pictures) to cover different areas in the valence-arousal space. Two sets are in the high valence region, one for high arousal and the other for low arousal. A third covered the medium valence and low arousal. For the last set which included low valence and high arousal, we additionally selected pictures that are not too cruel. Otherwise, some participants could be slightly traumatized, resulting in biasing the rating of the following images negatively. All 40 pictures have been shown to all participants, but for each, the order has been randomly determined.

In the studies carried out for IAPS, there are already several interesting findings. On the one hand, pictures rated closer to a neutral valance also tend to be rated with lower arousal. On the other hand, the more the valence rating tends to one of its extreme, the higher the arousal is rated. Therefore, a quadratic relationship between arousal and valence exists.

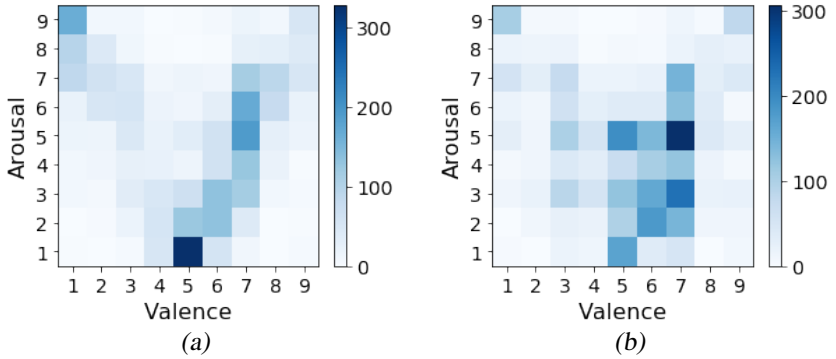


Figure 3.2.: *Distribution of ratings over all participants in the valence-arousal space for* (a) *IAPS and* (b) *math tasks.*

Math Tasks. All math tasks have been taken from the ACT and come labeled with difficulty between a range of 0 and 1 meaning demanding to effortless, respectively. Additionally, for each math task, a time limit was determined which was based on a pilot study consisting of 11 participants. A money reward is given for correctly and a monetary penalty for incorrectly solving a math task. By combining the difficulty level, the time limit and the money reward, three different exercise types have been created: repetitive, challenge and overchallenge (please see Table 3.1).

First, to form the **repetitive** block, we took two easy tasks from ACT with a difficulty of 0.76 and 0.83. To create further tasks, we only changed the variables of these basic tasks. The time limit was set so that each should have more than enough time to solve each exercise and the monetary reward and penalty were both rather low with CHF 0.20. The repetitive mode and little bonus should cover the medium valence and low arousal space. Second, the math tasks for the **challenge** block are more demanding, with a difficulty level between 0.58 and 0.69. However, the time limit is set so that most of the people should have still fairly enough time to solve them. With CHF 2.00 reward and only CHF 0.20 penalty, we expected to have high valence and high arousal for this block. Last but not least, for the **overchallenge** block, we selected math tasks with a difficulty level between 0.25 and 0.53. Additionally, the time limit was set that most participants should not be able to finish their calculations within the allowed time. Furthermore, the gain of only CHF 0.20 compared to a loss of CHF 2.00 when solving correctly and wrong, respectively, should force the participants to have low valence and high arousal while solving this block.

We expected that the stimuli and therefore the rating of one math task is based on the previously solved math tasks. Assuming that the previous math task was a frustrating one and the participant just lost CHF 2.00, correctly solving a boredom exercise is a high positively stimuli. Only after solving almost the same easy tasks over and over will result in a medium valence and low arousal rating. Addressing this effect, math tasks have been presented to participants in six blocks (2 blocks in each condition) with a different number of exercises (repetitive condition with 13 tasks, challenge condition with 5 tasks, overchallenge condition with 6 tasks). The order of the six blocks was determined randomly, with the constraint that two consecutive blocks are not the same condition. Additionally, the math tasks were shuffled within each block.

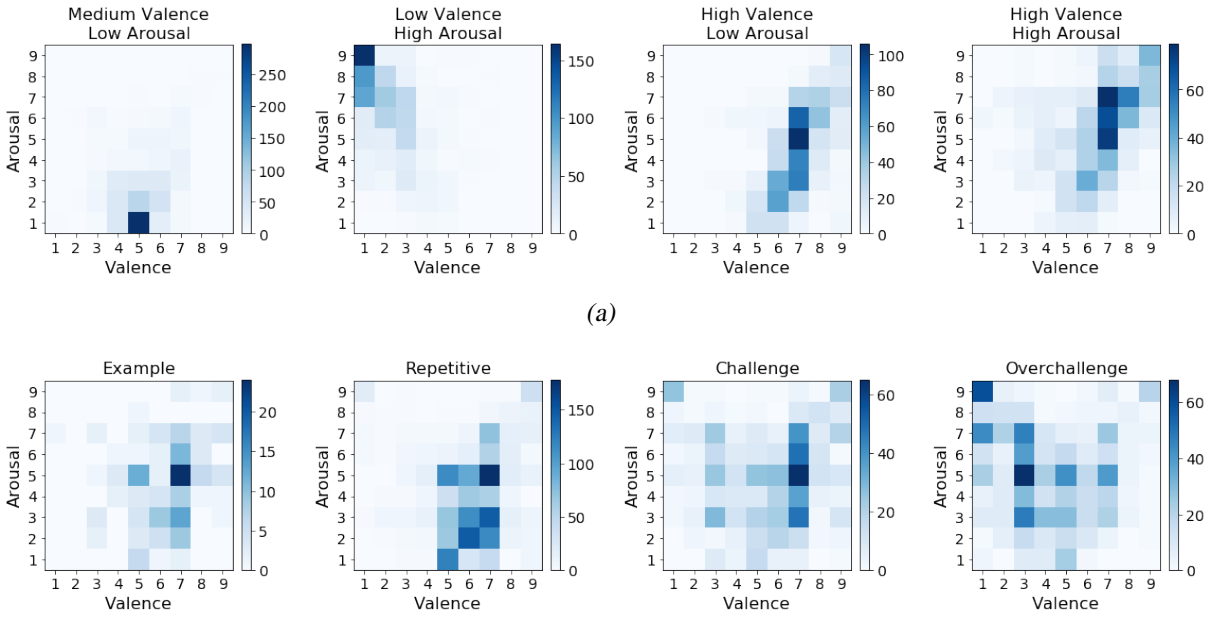


Figure 3.3.: *Distribution of ratings over all participants in the valence-arousal space.* (a) *Ratings given for each IAPS category.* (b) *Ratings given for each math task type.*

3.2. Data Analysis

3.2.1. Ratings

IAPS pictures and math tasks were chosen to cover a broad spectrum in the valence-arousal space. In Figure 3.2, we can see that a lot of the 2D space is included. For IAPS, it seems to have a little bit less variance, and the commonly known V-pattern is more prominent than for the math tasks. The lower variance for IAPS might be explained by the fact that the relative order of one picture to the others is not relevant for the rating. This can be observed in Figure 3.3(a), where ratings for each self-defined IAPS category are compared. Though, it can still be the case that the participant adjusts his rating scale slightly after seeing a highly affective picture. We assume for this experiment that this effect is relatively small. On the opposite, we assume that the higher variance for math tasks is because the emotional stimuli and therefore the rating of a single math task depends on the previously solved tasks, and not only on the correctness of the solution. For example, tasks in the boring block are reasonably easy to solve with only one substitution of a variable. Assuming that the previous math task was a frustrating one and the participant just lost CHF 2.00, correctly solving a boredom-exercise is a high positively stimuli. Only after repeating a few easy exercises will participants end up in a bored and therefore more neutral state. This reasoning has been used to group the math tasks into separate difficulty blocks (please see Section 3.1). Additionally, the emotional stimuli of solving math tasks are inextricably linked with the participant's math skills, resulting in a higher distribution of ratings between the participants. Both of these effects can be seen in Figure 3.3(b).

A fascinating observation in Figure 3.2 is the difference between the two accumulations, which

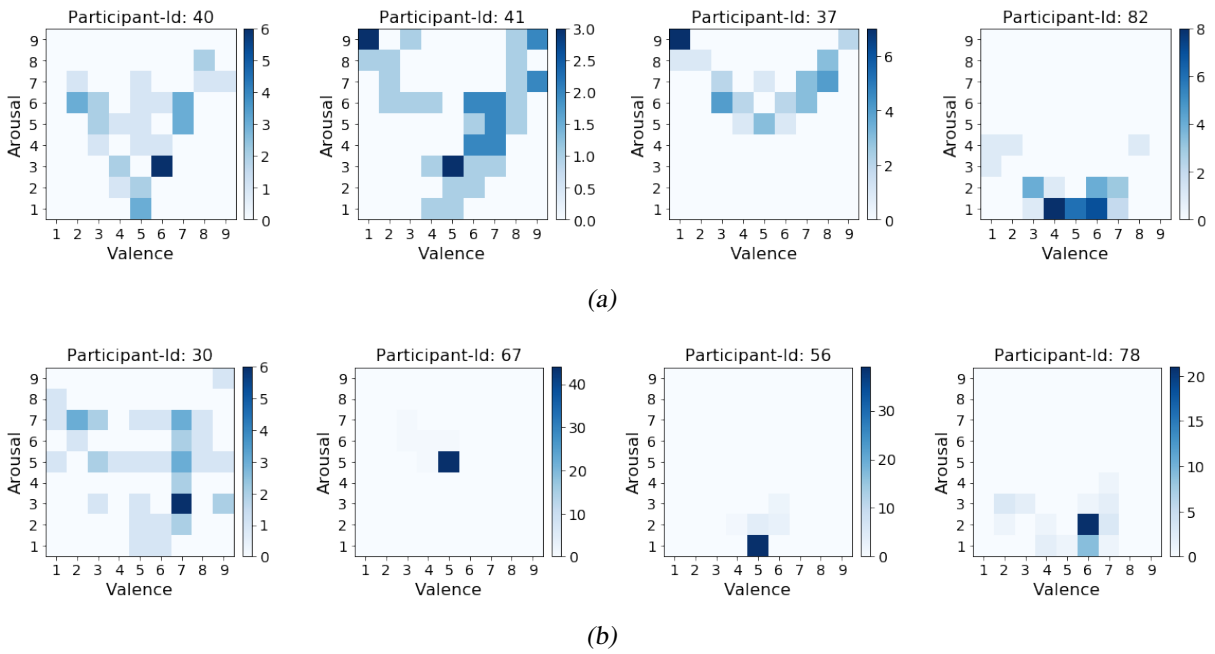


Figure 3.4.: *Distribution of ratings for specific participants in the valence-arousal space.* (a) *Most of the ratings given during the IAPS covers a huge range in the space for all participants (two left plots). Only a few participants have less variance in the arousal space (two right plots).* (b) *For math tasks, many ratings are still covering a lot of the space. Remarkably, some have only little variance through all math tasks, resulting in ratings similar to the three right plots.*

is located at medium valence and low arousal for IAPS and around high valence and medium arousal for math tasks. We expected to have the ratings for IAPS more evenly spread around the four target centroids (e.g., high valence and low arousal), because each picture set contained exactly ten pictures. For the two picture sets targeting the high valence space, the ratings look relatively similar (please see Figure 3.3(a)). We assume that arousal depends more on a participants experience and preferences for the high valence pictures, therefore having a higher variance in the arousal dimension. For math tasks, on the other hand, we believe that also a small amount of money reward increases the valence and arousal for some participants who struggle for more challenging math tasks. This could be the reason why a lot of the ratings for the repetitive block lies in the upper medium valence and higher arousal. The other blocks, challenge and overchallenge, have most ratings where we expected it, but the ratings are more spread (please see Figure 3.3(b)).

In Figure 3.2(a) it can be seen, that there is a quadratic correlation between the valence and arousal axis for IAPS ratings. A similar, but less clear behavior can be seen for the math-tasks in Figure 3.2(b). As a consequence, there are not many samples in the corners with low arousal. Additionally, few fall in the medium valence and high arousal region. This is expected, as it is harder to have more arousal in a neutral state. These conclusions results in a typically V-shape of the samples in this space.

Analyzing the ratings individually for each participant reveals some challenges. First of all, a big difference in ratings can be observed between IAPS and math tasks. The ratings for IAPS

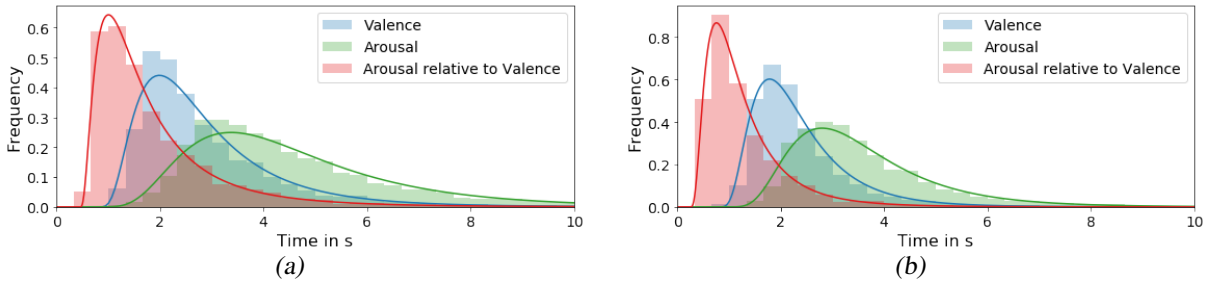


Figure 3.5.: *Seconds passed since rating screen was shown for selecting a value for valence and arousal for* (a) *IAPS and* (b) *math tasks.*

are following the desired V-shape with some variance, as can be seen in Figure 3.4(a) on the two left plots. This holds for about 95% of all participants and only a minority has a small variance in the arousal rating (please see two examples on the right plots). It is essential to state that this also might be because participants rated the pictures as society would expect it and not because of an honest rating. Participants might let their rating be influenced by the fact that everything was being recorded and thus did not want to attract attention. The ratings for math tasks reveal a different pattern. There are still about half of the participants who have a similar pattern as in the left-most plot in Figure 3.4(b). Remarkably, around one-third of all participants have a very low variance in either the valence or arousal dimension. Some participants rated only a few different ratings, covering less than 15% in the whole valence-arousal-space. Three different examples of a demanding rating are shown in Figure 3.4(b) in the three right plots. Comparing the facial expressions visually in the video for such participants brings up some doubt about an honest rating during these sessions.

3.2.2. Delay in ratings

Emotions are event-driven and tend to be only valid over a short time. Previous researches [Cab02] showed that emotion could have a short or long duration. However, for this experiment, the emotions are expected to be short-lived, as many different pictures and exercises are rated consecutively. A further time limitation for the duration of the emotion is that each image was shown for 10 seconds only. After each rating of an image, a neutral screen was shown for 5 s to set the participant in a different state, optimally neutral to be unprejudiced for the following picture. For solving the tasks, the answer had to be given within a specific time for each exercise, otherwise counting as wrong if exceeded.

One question which comes up is if there is a need to limit the time available to a participant to rate his emotion. One reason that supports a limit is the fact of the event-driven nature of emotions. As an emotion appears, it can be concise but intense. Waiting too long to rate this emotion might be hard to remember it correctly. Additionally, an emotion cannot be wrong or correct, but rather spontaneous. If somebody wants to give the most accurate rating and starts over-thinking the choice, it might not describe the emotion experienced by the participant, but instead the emotion the participant thinks was expected. One reasoning against a time-limit is the stressful nature of it. Even though most ratings are done within a quick period, it can be stressful for a participant seeing a clock counting down. As a result, the rating is endangered

to be biased into a default-rating value. Additionally, having no time limit gives the participant some time to breathe or digest some cruel picture a bit before proceeding.

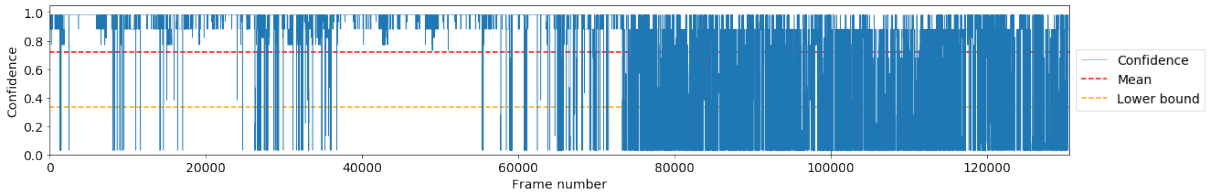
In the studies of IAPS [LB07], participants were forced to submit the rating within 15 s before showing the next picture after a 5 s neutralization screen. For our experiment, no time limit is given for the rating period, but the participants have been asked to rate each sample fast and spontaneously and proceed to the next sample quickly.

Analyzing the seconds used for selecting valence- and arousal-values relative to the display of the rating-screen reveals that most ratings are done within the suggested 10 s. Similar behavior can be seen for IAPS and math task samples, but math task samples are on average slightly faster rated than IAPS. Figure 3.5 makes it clear, that the valence is rated before arousal when looking at the blue and green histogram, respectively. These two histograms show the seconds used to rate the value relative to the start of the rating screen. Remembering the illustration of SAM (Please see Figure 2.1) which contains the valence value in the first row and the arousal in the second row, this is the expected behavior as people tend to fill out the rating from top to bottom. A second observation is that IAPS samples are slightly slower rated as math task samples for both valence and arousal. Valence is rated with 2.986 s and 2.383 s for IAPS and math tasks, respectively. Similarly, for arousal, the average of 4.671 s for IAPS is higher than 3.629 s for math tasks. The same pattern is observed for the standard deviation. Looking at valence ratings, the standard deviation for math tasks samples is with 1.095 s less than a third of the 3.592 s for IAPS. Not surprisingly, the same holds for arousal-ratings with a standard deviation of 1.556 s and 3.963 s for math tasks and IAPS, respectively. We assume that it is easier to rate the samples when the participant is playing an active role in the process and feels more immersed.

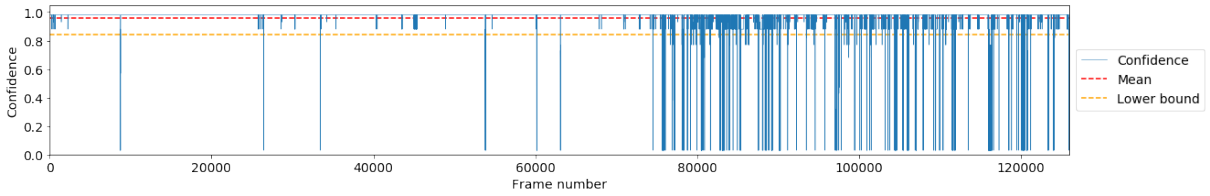
Last but not least, the red histogram shows the relative time used to rate arousal after valence has been chosen already. With an average of 1.915 s (std=1.329 s) for IAPS and 1.321 s (std=0.943 s) for math tasks, arousal is rated faster than valence. We think this is because the participant had to realize first that the rating screen is shown. As soon as this has been realized, the participant forms the rating in his mind and then start with the rating for valence. Because the decision for arousal is already made beforehand in mind and the hand and therefore the pen is now close to the choices of arousal, it is thus quicker rated.

Overall, only a few outliers can be found. The 99% percentile of ratings are within 9.195 s, 12.332 s, 6.874 s, and 9.125 s for IAPS valence, IAPS arousal, math task valence, and math task arousal, respectively. Compared to the 15 s for the IAPS studies, we think almost all of the samples are relevant and valid enough. Though, as we have no dimension dominance for our rating, a more strict time limit would be 10 s, leaving 5 s for each dimension.

Another consideration was the success message dialog, which was shown after every math task before the rating screen to inform the participant about the monetary gain or penalty. The participant had to close the dialog by herself or himself, without time limitation. However, the dialog was always open relatively short. Remarkably, the dialog remains open for a slightly more extended period for more difficult tasks, with 1.5 s (std = 0.719 s) for repetitive, 2.663 s (std = 0.803 s) for challenge and 3.853 s (std = 0.825 s) for overchallenge. This effect can be explained that for wrongly solved math tasks, the dialog was open approximately 4 s longer, and the harder the math task, the more it was answered incorrectly. For this thesis, we ignored this delay due to the success message dialog, because it is relatively short. The total time from



(a)



(b)

Figure 3.6.: *Confidence of frames during the experiment for two different participants. An example with an average confidence of* (a) *0.717 (std = 0.380) and one with* (b) *0.959 (std = 0.117).*

the delay with the dialog and the rating is still reasonably quick.

3.2.3. Change of ratings

The time used by a participant to make the rating is one aspect. The amount of how many times the rating for the same sample was changed is another. Detecting a participant switching the ratings regularly after an initial guess could hint to some uncertainty. We also considered the case that the first rating had been done intuitively, but after comparing with previously seen samples, the participant decided to adjust her or his rating accordingly to be more consistent. These might be the case when the ratings only changed by a value of one (e.g., from valence 9 to valence 8). Another case can be that a participant was shortly confused by the scale and picked the wrong side first when rating too fast. So even for substantial changes (e.g., from valence 9 to valence 2) can be considered to result in a more trustworthy rating. Looking at the data, we found that about 97% of all changes were only small. Only for a few ratings, the difference was above four (1 for the arousal of IAPS, 10 for the valence of math tasks and 10 for the arousal of math tasks). By looking at the final rating, it seems to be more reasonable for most of the few.

For this work, we assumed that the changes in ratings do not decrease the reliability and no further samples needed to be excluded.

3.2.4. Confidence Analysis

OpenFace defines a value for each frame representing the confidence of the facial detection. The range is between 0.03 and 0.98 for unsure and confident, respectively. For example, when a face is partially covered, it might be still detected and all features will be extracted, but the confidence will be rather low. Additionally, if the participant was bending over a lot and the face was only visible with a very steep angle from above, confidence went usually toward 0.03.

We analyzed the distributions for the confidence level for all participants to find possible problems and a suitable threshold for distinguishing valid from meaningless frames. For this analysis, we only analyzed frames which are relevant for our thesis. For example, we ignored frames from the introduction and at the end of the experiment as we did not use them in any other step. We did the analysis separately for GoPro and front camera videos.

GoPro. Overall, for most of the participants, the average confidence level was reasonably high with 0.954 (std = 0.121). Only eight participants had an average confidence below 0.9 and the lowest average confidence was at 0.717 (std = 0.38) for one participant (please see Figure 3.6). Additionally, we found that the average for the math task sequences was slightly lower with 0.933 (std = 0.163) compared to the IAPS part with 0.968 (std = 0.075). In contrast, the variance is more than twice as high, which can be seen as a pattern in the figure for the second half. We expected this pattern because the head position was sometimes suboptimal when participants were solving math tasks.

Front Camera. The front camera had an average of confidence for all participants around 0.63 (std = 0.436). The lower confidence and higher variance led to the second focus of this thesis, the front camera restoration, which is demonstrated in Chapter 6. Therefore, no further analysis and comparison for the front camera confidence are done here.

4

Modeling

In this chapter, the model based on the classification pipeline is described (please see Figure 4.1). First, in Section 4.1, we summarize the overall pipeline with all steps for classification. In Section 4.2, two possible frameworks for facial feature extraction are analyzed. The preprocessing of the videos is explained in Section 4.3 (step 1 in Figure 4.1) and the time synchronization between the database and video sources is shown in Section 4.4. Section 4.5 covers the used features for the machine-learning algorithms and how the data was normalized (step 3 and 4 in Figure 4.1).

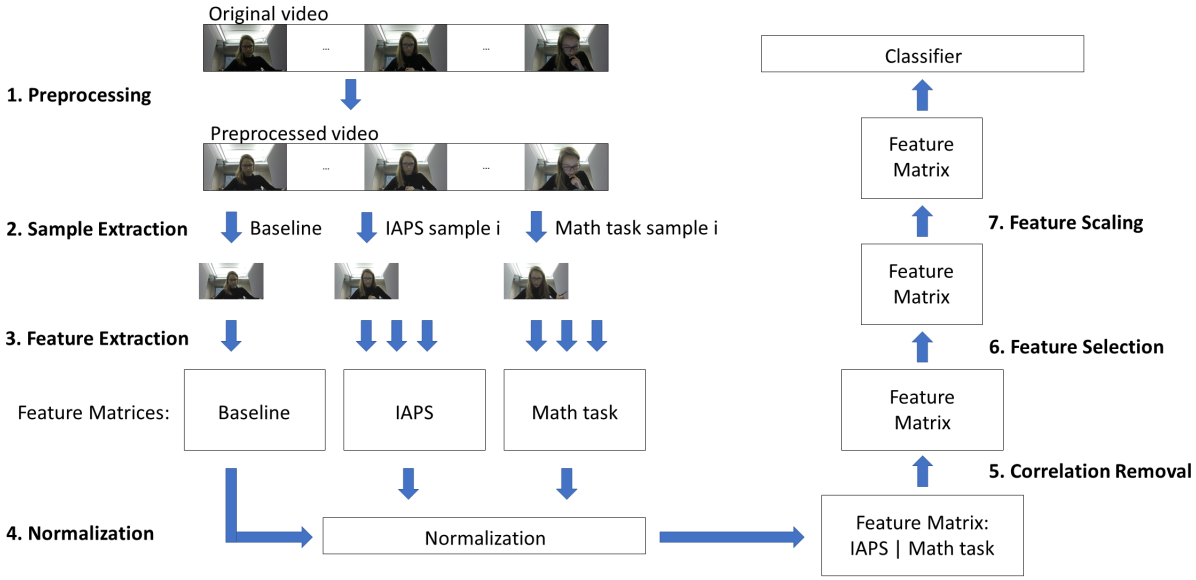


Figure 4.1.: *Classification pipeline from preparing the videos to extracting and selecting features for the classifier.*

4.1. Pipeline

In this section, we describe our pipeline for classifying valence and arousal on our dataset.

Sample Generation. We generated all samples based on two parameters. A sample is defined with the start and end time of the IAPS picture or math task from the database to extract in a next step the frames of interest for these. One parameter was the type of samples to distinguish between IAPS and math tasks samples. The other was to specify the exclusion lists to exclude some participants or only a few specific IAPS or math tasks. Additionally, we only considered samples that belong to the desired class setup. For example, when we classified the two classes "low", which contains values from 1 to 3, and "high", which includes values from 7 to 9, we ignored samples with values from 4 to 6.

Feature Extraction. In this step, we extracted all the features for the selected samples (please see Section 4.5.3). During this step, we had to fix the window size and windows shift for the samples (please see Section 4.5.2).

Baseline Normalization. After we extracted all features for the samples, we normalized each feature with the respective baseline of the participant (please see Section 4.5.4). The baseline normalization ensures to only compare state changes from one sample to the participant's neutral state and not the absolute values.

Correlation Removal. Based on the analysis of correlated features (please see Section 4.5.5), we introduced a step that removes features with a higher correlation than a defined threshold. For our default pipeline, we set this to a default value of 0.85. For this step, the correlation matrix was calculated for all features with the Equation (4.1). The indices i and k represent the index of the features for which the correlation is calculated. N stands for the number of samples and $\bar{x_j}$ is the mean of the feature j over all samples.

$$cor(x_j, x_k) = \frac{\sum_{i=1}^{N}(x_{ij} - \bar{x_j})(x_{ik} - \bar{x_k})}{\sqrt{\sum_{i=1}^{N}(x_{ij} - \bar{x_j})^2}\sqrt{\sum_{i=1}^{N}(x_{ik} - \bar{x_k})^2}} = \begin{cases} 1 & \text{if } i = k \\ r_{jk} & \text{otherwise} \end{cases} \tag{4.1}$$

In the resulting correlation matrix, the diagonal elements are always 1 by definition and it holds $r_{ik} = r_{ki}$. The range for values is between -1 and 1, with negative values representing a negative correlation. To remove correlated features, we took the absolute value of the element to find also high negative correlations with a single threshold. For each feature pair with a correlation above the threshold, we removed the feature with the higher index and kept the other.

Feature Selection. For the feature selection, we integrated the transformer SelectKBest from sklearn [Ped11] which selects the k best features based on a scoring function. If k is specified as a floating point, the respective percentage of all available features will be kept. Otherwise, it is expected to be a positive integer which stands for the total number of features to retain. For our default pipeline for the baseline performance, we used a k of 200. Different scoring functions can be passed to check for the best features. We used the f_classif, which computes the analysis of variance (ANOVA) F-value between the label and feature for classification tasks.

Feature Scaling. Many machine learning estimators require standardized data to train and predict. Features with a variance that is orders of magnitude larger than others might dominate

the objective function and estimators are not learning from other features properly. In this step, we scale each feature to have zero mean and unit variance. The formula in Equation (4.2) is applied for each feature x independently and returns the standard scaled feature z. The mean μ which is first subtracted from the feature is calculated with Equation (4.3) and uses the number of samples as N. The standard deviation is calculated with Equation (4.4) and needs the mean μ from Equation (4.3) and the number of samples as N.

$$z = \frac{x - \mu}{\sigma} \tag{4.2}$$

$$\mu = \frac{1}{N}\sum_{i=1}^{N}(x_i) \tag{4.3}$$

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2} \tag{4.4}$$

Classification. During the last step, we trained the final data on a classifier from sklearn. We decided to use four predefined classifiers, including logistic regression, support vector machine (SVM), random forest and gradient boosting. These classifiers have been selected after a first few tests on our dataset without hyperparameter optimization.

4.2. Frameworks

In this section, we analyze two well-known frameworks for recognizing and reading facial expressions, namely Affectiva in Section 4.2.1 and OpenFace in Section 4.2.2. A short comparison in Section 4.2.3 helped us to decide which framework to use for this work.

4.2.1. Affectiva

Affectiva has its origin in MIT Media Lab and was founded in 2009 by Dr. El Kaliouby and Dr. Picard. It can be used by downloading the SDK from the official website[1].

One advantage of Affectiva is the direct output of some basic emotions[2], the activation of the supported action units and additional information about the visual appearance as age, gender and if the person is wearing glasses. Additionally, an option is available to map the current emotion of the person in the video to a few predefined emoticons. A further advantage is the additional output of the valance, which reduces the step to map emotions to it. For some applications, this might be a more fundamental aspect than to analyze single emotions.

Based on some first tests, we recognized some internal adjustments for the extracted features, which are done by Affectiva internally. To test the consistency, we carried out a small analysis

[1] https://knowledge.affectiva.com/docs/using-the-emotion-sdk

[2] Based on https://imotions.com/blog/facial-action-coding-system/ in Section "Emotions and Action Units"

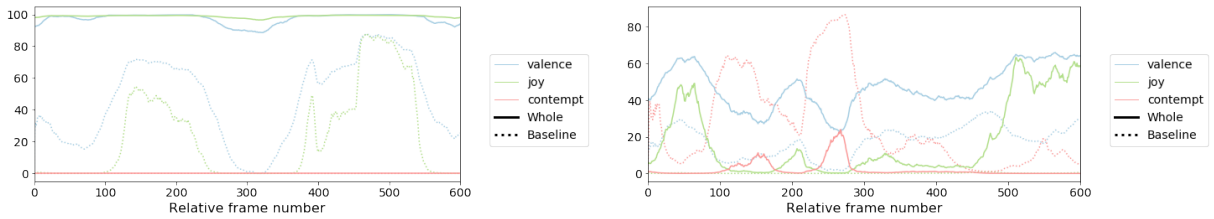


Figure 4.2.: *Comparison of extracted features by Affectiva on two identical video sequences, applied once on the whole video (solid line) and once only on the extracted video (dotted line). Only a few extracted features are shown to demonstrate the discrepancy.* (a) *Example with participant 28.* (b) *Example with participant 47.*

with the available video footage. First, we extracted all the features with Affectiva from the original videos recorded during the experiment. Second, we obtained two 10 s long sequences from the video where the person was completely visible and applied the feature extraction again on only these short sequences. With a 60 fps for the video, we have two times 600 consecutive frames. We then matched the frames of those two video sources to be sure to compare the same underlying data. Having the matching frames, we can compare the extracted features between the two video sources. For many of the participants, those extracted values differ a lot. This is not only true for the extracted emotions, but also for some action units. Still, for some participants, the two features are more consistent. In Figure 4.2(a), concerning participant 28, the extracted features project valence and joy very close to their maximum value for the full duration of the video, whereas when considering only the 10 s sequences, the values are far below the maximum and show a more apparent trend with some broad peaks. A similar difference can be seen for participant 47 in Figure 4.2(b), where the trends for contempt and valence seem to be alike, but with a different level. Only the emotion 'joy' is projected to zero for the short 10 s sequences, whereas for the whole video, it has some stronger pattern.

4.2.2. OpenFace

OpenFace is written by Tadas Baltrusaitis and is an open source project available on his GitHub repository[3].

Most of the action units supported by OpenFace are available as intensity and presence. The intensity of the action unit is represented as a continuous value between 0 (not available) to 5 (maximal activated), whereas the presence is a binary field (0 for inactive, 1 for active). It is mentioned that both fields are trained differently, resulting in possible inconsistencies (e.g., action unit 1 is active for presence, but the intensity value is still around 0). In addition to the action units, eye gaze features and 3D facial landmarks can be extracted with OpenFace. Please refer to the website[4] for a full list of available features.

A crucial option during the feature extraction is whether action units are static without post-processing or dynamically normalized after all frames have been processed when the range of

[3]https://github.com/TadasBaltrusaitis/OpenFace
[4]https://github.com/TadasBaltrusaitis/OpenFace/wiki/Output-Format

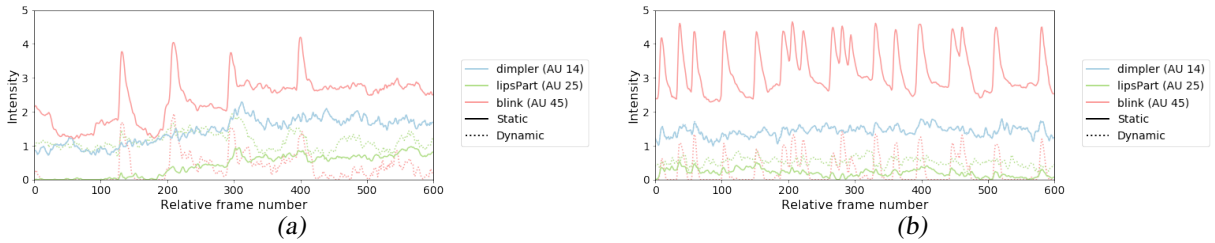


Figure 4.3.: *Comparison of static vs dynamic extracted features of OpenFace.* (a) *Example with participant 19.* (b) *Example with participant 37.*

the person is known. The general recommendation is to use static for short videos or when the person is not showing a wide range of emotions. For this work, we mainly worked with the static features, because we are normalizing each participant already with the baseline video (please see Section 4.5.4). We also did not further investigate how the dynamic post-processing is constructed and therefore trusting more in the static features. In Figure 4.4, we can see that some of the dynamic features are shifted downward and sometimes have a slightly different trend. Looking closer at the action unit 45 (blink), the static features are closer to the expectation when looking at the video itself.

To validate the consistency for OpenFace, we applied the same test as we did for Affectiva. We first run the feature extraction of OpenFace once on the original video of each participant. Second, we obtained a short video sequence of 10 s and extracted the features with OpenFace on these. Finally, we compared the extracted features for matching frames. In Figure 4.4, we can see that the difference is marginal between the extracted features for the same frames. This is only valid when extracting static features, not for the dynamic extraction, as the OpenFace internal normalization has other ranges available for each action unit. For each of the 88 participants, we only saw small deviations of the extracted features. Therefore we assume that OpenFace has a consistent feature extraction.

4.2.3. Affectiva versus OpenFace

To save time, be it in computational or organizational nature, we decided to focus ourselves in the following chapters on only one framework. In the following paragraphs, a few criteria have been compared within the two to select one.

Action Units. In total, OpenFace and Affectiva support 18 and 16 known action units, respectively. 13 are available in both. Only in OpenFace, for each action unit, there exists a presence and intensity field[5]. Furthermore, Affectiva has a continuous range between 0 and 100 for the action units and OpenFace between 0 and 5 for the intensity action units.

Emotions. Only Affectiva supports an out-of-the-box prediction of seven predefined emotions. Predictions of emoticons similar to the facial expression of the person in the video are also only available in Affectiva. However, these features are not used in this work, therefore, leaving no practical gain for it.

[5]Except for action unit 28, which is only available as presence.

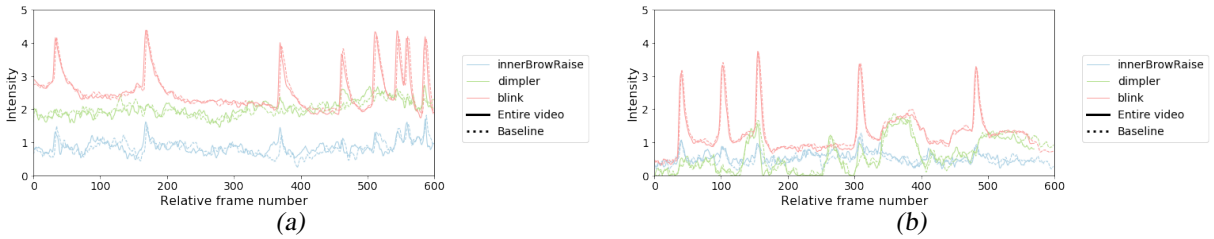


Figure 4.4.: *Comparison of extracted features by OpenFace on two identical video sequences, applied once on the whole video (solid line) and once only on the extracted video (dotted line). Only a few extracted features are shown to demonstrate the similarity.* (a) *Example with participant 37.* (b) *Example with participant 68.*

Feature Extraction. To apply the feature extraction with OpenFace, a distributed executable can be run with the needed parameters. Additionally, a model for the underlying used facial detection needs to be downloaded separately. For Affectiva, the SDK needs to be installed, and a small program needs to be written. As a starting point, examples from GitHub can be used.

Running Time. Affectiva needed around 2.5 h for an 80 minute long GoPro video (60 fps and a resolution of 1920x1080). For the same video on the same computer, OpenFace needed around 7 h but also extracted eye-gaze features and all facial landmarks.

Consistency. In the previous Section 4.2.1 and Section 4.2.2 we already covered that OpenFace has a more consistent feature extraction with the static setting.

Regarding the above criteria, we have chosen OpenFace for this work. By comparing the extracted features from OpenFace with the videos itself, it seems to work better than Affectiva. The longer runtime can be ignored, as we need to extract the features only once per video. For a real-time application, a lower fps and resolution have to be used, which both can lead to less accurate detection. Nevertheless, as we extracted all the features for Affectiva as well as to compare it with OpenFace, we are using some properties of it for the preprocessing steps (please see Section 4.3).

4.3. Video Preprocessing

In this Section, we introduce a few modifications to improve the quality of the videos and to make further process easier.

Constant fps. As GoPro recorded the videos with a constant fps of 59.94, this is not very practical for calculating time steps and relative times between frames. For the videos recorded with the front camera of the tablet, we have a variable fps (mean = 20.022, std = 1.923). This causes even more issues for calculating time steps and will result in having different numbers of frames for equally long samples (i.e., looking at a picture or solving a math task).

For the GoPro videos, we resampled all to 60 fps, whereas front camera videos have been resampled to 25 fps. For the front camera, we had the trade-off between deleting frames or interpolating many new frames. In the end, we decided to use 25 fps, as it is closer to the

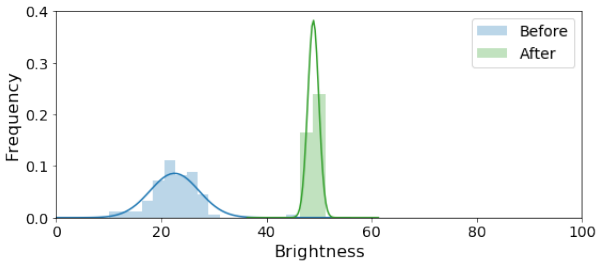


Figure 4.5.: *Average brightness of the face in GoPro videos. Without adjusting the videos (blue), most are below the recommended value of 25, whereas after processing (green) the range lies in the optimal range around 50.*

60 fps of the GoPro and therefore making it more reasonable for comparisons in a later step. To resample the videos, we used FFMPEG [Tom06].

Brightness correction. The lighting conditions for the experiment were suboptimal. The ceiling lamps from the room were just behind the participant in the video, resulting in a challenging setup for the cameras to record the face flawlessly. Affectiva also determines the brightness of the detected face. It comes within a range between 0 and 100 for dark and bright, respectively. A recommended brightness[6] for an optimal feature detection is between 25 and 75.

To validate our videos for a correct brightness, we first extracted each 100th frame with FFMPEG from the original footage. As a second step, we read the brightness of these new shortened videos with Affectiva and analyzed the average brightness factor. The average of the brightness for all videos is with 22.526 (std = 4.651) below the recommended value of 25. Adjusting the brightness is done again with FFMPEG. However, to set the correct value for the brightness parameter is not directly visible. The documentation specifies only the valid range of the parameter, which is between -1 and 1. We carried out three different brightness adjustment for two participants with slightly different values to estimate a good conversion. In the end, we found the following Equation (4.5) suitable for our application. t is our target brightness 50, and c is the average brightness level of the video.

$$b = 0.0056 * (t - c) \tag{4.5}$$

After calculating and applying the brightness parameter for each participant (please see Figure 4.6), we compared the adjusted video with the original. The brightness level over all participants increased from 22.526 (std = 4.651) to 48.911 (std = 1.045) (please see Figure 4.5).

4.4. Time Synchronization

During the experiment, each step has been documented with a start and end time in a local database. It is crucial to know when a participant has looked at which picture or what math task has been solved. To extract the correct frames matching a given sample (either one image or

[6] http://affectiva.github.io/developerportal/pages/platforms/v3_4_1/windows/classdocs/affdex-native/structaffdex_1_1_face_quality.html

(a) (b)

Figure 4.6.: *Adjustments of the brightness based on Affectiva values. For both pictures, the left half is from the original footage, whereas the right half is after adjusting to have approximately an average of 50 for the brightness. Example with* (a) *participant 68 and* (b) *participant 99.*

one math task), we need to have an accurate time synchronization to map from a date time to a frame number in the video.

One issue is the missing timestamp for each frame in a video file. However, this is quickly solved as soon as we have the correct timestamp t_0 for the first frame. Please see Equation (4.6) to calculate the timestamp t_i in milliseconds for frame i based on the first frame's timestamp t_0 and the frame rate fps of the video. For this calculation, it is important to have a constant frame rate (please see in Section 4.3).

$$t_i = t_0 + 1000 * i/fps \tag{4.6}$$

We had two options to determine the timestamp of the first frame in Equation (4.6). One is the creation date of the video file, and the other is to synchronize with a sound signal, which can be heard in the video and for which the timestamp is known. We found it more reliable to use the sound signal because GoPro stores the creation date only with a resolution of 1 s and for the front camera, it was possible that due the slow startup of the hardware, a slightly wrong timestamp has been written. The signal was played shortly (mean = 1975 ms, std = 5 ms) before the final timestamp has been written into the session table. To find the corresponding frame in the video, we used Aodbe Premiere Pro 2019 to identify the pattern in the sound layer manually. The timestamp t_p of this frame can then be calculated by subtracting the delay of 1975 ms from the timestamp t_e in the session table. Having located the correct absolute frame, we can calculate the timestamp t_0 of the first frame with Equation (4.8) or the timestamp t_i of any frame i with Equation (4.9). We only need the timestamp t_p and frame number p of the signal and the frame rate fps of the video.

$$t_p = t_e - 1975 \tag{4.7}$$

$$t_0 = t_p - 1000 * p/fps \tag{4.8}$$

$$t_i = t_p - 1000 * (p - i)/fps \tag{4.9}$$

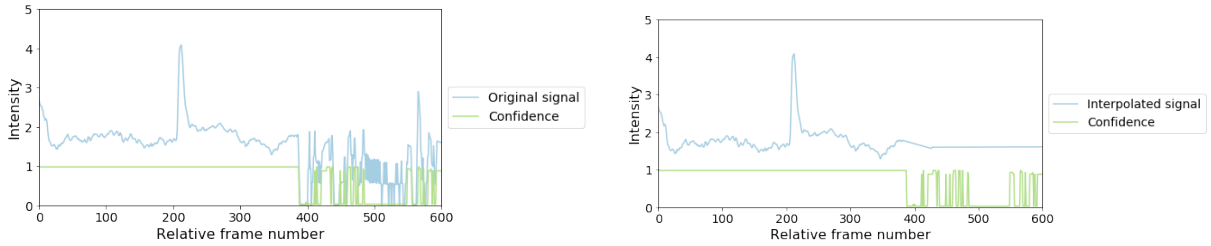


Figure 4.7.: *Features for invalid frames are interpolated if needed (e.g., for the blink detection).* (a) *The original signal has a lot of invalid frames during the last 200 frames.* (b) *The interpolated signal does not suffer from artifacts after invalidating frames manually with a window size of 11 for each frame below the threshold.*

4.5. Feature Extraction

In this section, we are describing the features of our model. First, we explain the confidence handling in Section 4.5.1 and introduce the definition of the window in Section 4.5.2. Second, in Section 4.5.3, we define all extracted features and the baseline normalization is shown in Section 4.5.4. Last, we analyze the correlation between the features in Section 4.5.5.

4.5.1. Confidence Handling

Based on the statistics from the OpenFace confidence analysis in Section 3.2.4, we fixed the threshold at the mean minus one standard deviation, which is roughly 0.85 for the GoPro videos. The confidence level from OpenFace is not continuous and the two closest possible values were 0.82 and 0.88. Therefore, we fixed it to the next lower option which is 0.82. Every frame with a lower threshold will be handled as invalid.

For most of our extracted features, we can ignore all invalid frames without introducing new artifacts (e.g., mean, min, etc.). But for obtaining the blinks which rely on peaks of a signal, an interpolation of the signal is crucial (please see Section 4.5.3). It was enough to interpolate the signal linearly without introducing artifacts that could be recognized peaks. For some consecutive frames, we observed the confidence jumping back and forth around the threshold. We decided to invalidate frames manually within a window of size 11 frames around each frame below the threshold, which results in invalidating five frames in either direction. Please see Figure 4.7 of one sample which interpolates the signal for invalid frames with a window size of eleven to invalidate consecutive frames.

4.5.2. Window Size and Shift

We introduced a window with a size and a shift to define the frames of interest for a sample (i.e., looking at one picture or solving one math task).

Window Size. The size of a window determines the duration to extract frames from the video and is specified in a float which represents the number of seconds. For example, 180 frames are considered for a window of size 3 s and a video source with 60 fps.

Window Shift. The shift determines from where the window starts for extracting frames. The shift is specified relative to the start of the display of the picture for IAPS and relative to the end of a math task when the success message appeared. We assume that the highest impact of a picture is as soon as it is shown to the participant. Therefore, the shift is relative to the start of an IAPS sample. For math tasks, on the other hand, we expected that the highest reaction is as soon as the participant sees if the given solution is correct, thus the shift is relative to the end of the math task.

With this definition, a window with size 3 s and shift 0 s will extract 3 seconds from the start of displaying the picture to the participant for an IAPS sample. For a math task sample, the beginning of the extraction is when the participant just finished the math task, and the duration is 3 seconds as well.

To have reliable samples, we analyzed the number of valid frames for a sample. A frame is valid if the confidence of the frame is above the threshold of 0.82 (please see Section 4.5.1). The window for an IAPS sample is defined to have a windows size of 10 s and a shift of 0 s, therefore covering the time when the picture was displayed. For a math task sample, we defined the window size also as 10 s and the shift as -5 s, therefore including the last 5 s of solving the math task and the period of the success message dialog. We have used the window size as 10 s to analyze the complete sequences which we assumed are essential for the classification. Calculating the percentage of valid frames for the defined window sizes gave an average of 0.990 (std = 0.068) for IAPS samples and 0.955 (std = 0.143) for math task samples. We decided to set the threshold for the required percentage of valid frames to the average minus one standard deviation. This gave a threshold of 0.922 for IAPS and 0.812 for math tasks. To have the same threshold for both sample types, we set it to be 0.8. Therefore, a sample is valid if 80% of all frames of the sequence are valid (e.g., for the GoPro with 60 fps, 480 frames have to be valid for a 10 s sequence). In total, 326 samples fell below the threshold, thereof 39 from IAPS and 287 from math tasks. This reflects the fact that the conditions for the GoPro were more challenging when a participant solved math tasks.

4.5.3. Feature Definitions

To support multiple camera sources simultaneously with different fps, all the following features are designed to be fps independently. The issue with a different fps is that there is a different amount of frames available for one source. To solve this, all features based on frame numbers are extracted as percentages. Let us assume that we have the same 10 s signal with a peak at 5 s for different sources with one 25 fps and the other 60 fps. Extracting only the relative frame number without respecting the different fps, we would obtain the location as 125 and

Id	*Name*	*Id*	*Name*
1	Inner Brow Raiser	14	Dimpler
2	Outer Brow Raiser	15	Lip Corner Depressor
4	Brow Lowerer	17	Chin Raiser
5	Upper Lid Raiser	20	Lip Stretcher
6	Check Raiser	23	Lip Tightner
7	Lid Tightener	25	Lips Part
9	Nose Wrinkler	26	Jaw Drop
10	Upper Lid Raiser	28	Lip Suck*
12	Lip Corner Puller	45	Blink

Table 4.1.: *Action-Units: The subset of action units which are recognized by OpenFace. * Only available as presence*

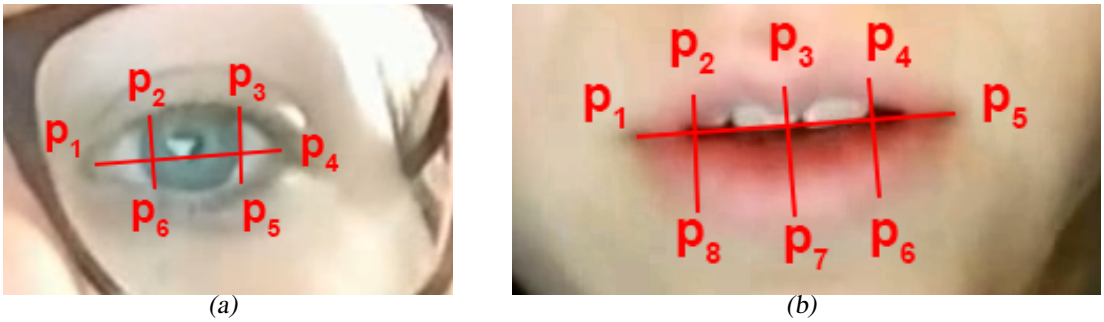
300 for the 25 fps and 60 fps source, respectively. By calculating the percentage of the feature, which is equally at 0.5 for both sources, we made sure that the model can be applied on any video without an fps constraint. It is to mention that some features might still suffer a little bit from the different sample sizes. For example, having more frames will return a more accurate variance estimate of the feature. Though, we assume that this effect is ineligible for our defined sample sizes.

Some basic statistics can be directly applied on an input signal, which can be an action unit itself or a further extracted feature. The min, max, mean and standard deviation are available for a continuous signal, whereas, for a binary signal, we calculated the percentage of the active signal state (referenced as activity-percentage). Additionally, we implemented the option to extract the location of the min and max in the signal, which can be meaningful as well for specific signals. Further, the linear regression can be applied on a signal to find a trend in the timeline. For example, when an action unit is more activated during the first few frames but not during the last frames, we captured this effect with a negative coefficient. We only extracted the linear coefficient for this feature, as the intercept was highly and significantly correlated with the mean of the signal.

Action Units. OpenFace supports 17 action units (AUs) with a continuous signal (ranges from 0 to 5) and 18 AUs for a binary signal called intensity AUs and presence AUs, respectively. Please see Table 4.1 for a list with all supported AUs.

For each intensity AU, we extracted the min, max, mean and standard deviation and furthermore, the locations of the min and max and the fit of the linear regression line. On the other hand, for each presence AU, the activity-percentage is calculated.

Combinations of Action Units to form Emotions. Based on the work from P. Eckman and colleagues [Ekm92], an emotion can be described as a set of action units. In the paper, six basic emotions (joy, sadness, surprise, fear, anger, and disgust) and their representing action units have been introduced (please see Table A.1). Based on these definitions, we extracted



(a) (b)

Figure 4.8.: *Visualization of the calculations for* (a) *EAR in Equation* (4.10) *and* (b) *MAR in Equation* (4.11).

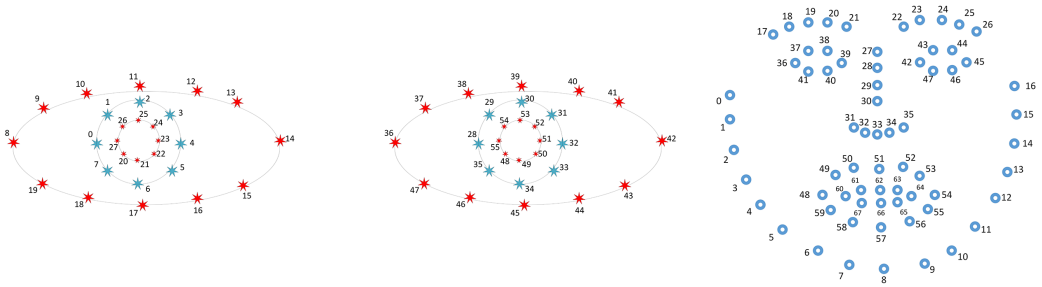
for each sample six new signals, one for each basic emotion, by combining the corresponding action units. Combining was done by adding up the action units because multiplication resulted in some combinations being zero over a complete sample as soon as one action unit was not available. For the new signals, we obtained the standard statistics (min, max, mean, standard deviation, and locations of min and max) and the fit of a linear regression line.

Eye Aspect Ratio (EAR) and Blinks. A well-known saying "The eyes are the mirror of the soul" leads to the importance of the eyes used to reflect the affective state. We assumed that blinks of the eyes play an essential role in this. A correlation between eye blink frequency and stressful situations has been found in the work from Haak and colleagues [Haa09]. We found the definition of the EAR, which is defined as aspect ratio between the height and the width of the visible eye and has been used to detect driver drowsiness by Singh and colleagues [ASP18]. Calculating EAR results in a value that is increasing for opening an eye and decreasing for closing it. It is calculated for each eye independently and averaged for the final value. To calculate the EAR as defined in (4.10), we need six 2D landmark points around the eye. Please see Figure 4.8(a) for a mapping between the formula parameters and the landmarks of the eye.

$$EAR = \frac{||p_2 - p_6|| + ||p_3 - p_5||}{2 * ||p_4 - p_1||} \tag{4.10}$$

OpenFace can return the eye landmarks from two different sets, one from the facial detector which contains 68 landmarks overall and one from the eye gaze which includes 56 landmarks dedicated only to the eye (please see Figure 4.9). We compared the results with both landmark sets to the existing action unit 45 from OpenFace, which directly reports the blink. First of all, the EAR is often very similar in the trend, independently of the used landmark set. However, when the EAR is calculated with the eye gaze landmarks, we found a higher discrepancy between the left and right eye (please see (b), (c), (e) and (f) in Figure 4.10). Second, visually there is more similarity between the action unit 45 and the negatively scaled EAR based on facial landmarks (please see (a) and (b) in Figure 4.10). The correlation factor between the action unit 45 and our calculated EAR over all samples was with -0.664 (std = 0.283)[7] and a p-value of 0.012 (std = 0.089) not as high as suspected. We visually compared the video footage

[7]The correlation is negative, because of the different definitions. For action unit 45, a higher value stands for a higher probability of a blink and therefore closed eye. The EAR decreases when the eye is closed because the detected height of the eye decreases.



(a) (b)

Figure 4.9.: *Two different landmark sets are available from OpenFace, one* (a) *from the eye gaze and the other* (b) *from the face detector. Graphics are taken from: https://github.com/TadasBaltrusaitis/OpenFace/wiki/Output-Format*

with some samples with a low correlation factor and concluded that the action unit 45 is overall more reasonable (please see Figure 4.10 in the top row a sample with high correlation and in the bottom row a sample with low correlation). As a result, we dropped the investigation of the EAR and only used the action unit 45 for further extracting the features for blinks.

Because all intensity action units of OpenFace are represented as a continuous signal, further processing needed to be done to extract the desired features (e.g., number of blinks during a sample). The default peak detection algorithm from scipy [Jon14] failed to capture different level changes between participants. Setting the same prominence[8] for all participants resulted in detecting more peaks in a noisy signal for some participants and ignoring true peaks in a clean signal for others. The issue with the prominence is that it constraints only the absolute height for a peak, which can be different for extracted action units from OpenFace (please see (a) and (d) Figure 4.10). A solution would have been to set different prominence parameters for each participant, which is not reasonable for future unknown people. Also, it does not take care of varying head poses. During looking at pictures, most participants were sitting straight in front of the tablet and the eyes are clearly visible, whereas for solving math tasks, they bent over the tablet for writing and the eyes became less visible. To address this issue, we came up with our constraint based on the predefined functions find_peaks and peak_widths from scipy. In a first step, we searched for peaks that exceed a low prominence with find_peaks to exclude peaks found in a noisy signal. In a second step, we calculated the ratio between the width and the prominence of the peak for which we defined a further constraint. The width of a peak is calculated at a relative height of 0.33 based on the peak_widths function from scipy (please see Figure 4.11(b)). Having calculated this ratio, we had to find an optimal value for detecting the peaks correctly. We used 218 samples with a total of 552 peaks and labeled each sample with the correct amount of included peaks. Applying different threshold values revealed that a value of 0.026 detects the most samples correctly (please see Figure 4.11(a)). With the new approach, we were able to identify blinks more accurately. Moreover, it allowed us to detect double blinks within a short range, which remained undetected by only considering the prominence. To detect both peaks in each double blink in Figure 4.11(b), the prominence had to be set very low, which introduced noise in other samples.

[8]The prominence is defined as the height of the peak relative to the lowest contour line which does not contain any higher peak within it.

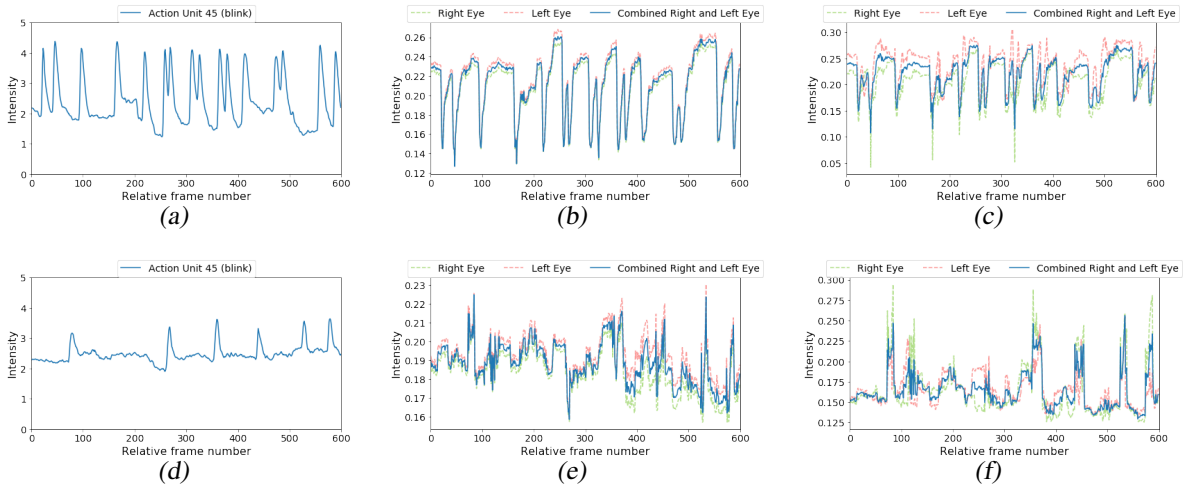


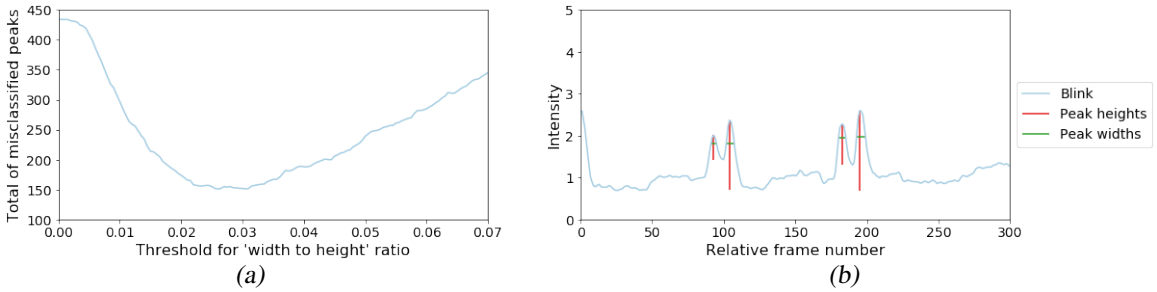
Figure 4.10.: *Comparison between the action unit 45 (blink) and EAR features of two 10 s samples. The upper sample has a high correlation between the different extracted features, whereas the bottom sample has a low correlation.* (a) *and* (d)*: Extracted action unit 45 (blink).* (b) *and* (e)*: Calculated EAR with facial landmarks.* (c) *and* (f)*: Calculated EAR with eye gaze landmarks.*

From the action unit 45, we extracted the min, max, mean, standard deviation and locations for the min and max as for all other AUs. Additionally, based on the peak detection we extracted the number of found blinks and the min, max, mean and standard deviation from the duration between the blinks. We also calculated the min, max, mean and standard deviation of the prominence and width of each blink.

One issue with these features is that the shorter the sample, the less the chance is to capture one or more blinks. To overcome this, we calculated a similar signal as it is known for the inter-beat (RR) intervals of the heart rate. The calculation is done beforehand over the entire length of the video with all available frames. First, all peaks are detected with our method and the optimal found ratio threshold. For each location of a peak, we stored the duration between the previous peak in milliseconds. To have a continuous signal, we linearly interpolated the signal with the known peaks. Finally, we can calculate the frequency for any position in the signal by taking the inverse of the value in the new signal The advantage is to have a continuous signal for the entire signal and be able to extract the statistics even if no blink is available during the sample itself. This indirectly leads to the disadvantage, that we also considered information which does not belong to the sample itself. From this new interpolated blink signal, we extracted the min, max, mean and standard deviation.

Mouth Aspect Ratio (MAR). The MAR works similarly as the EAR and was also covered in the work of Singh and colleagues [ASP18] for detecting drowsiness. It is defined as the ratio between the height and the width of the mouth and has a higher value when the mouth is open and a lower when it is closed. The MAR is calculated with the formula in (4.11) and needs eight different landmark points (please see Figure 4.8(b) for a visualization).

$$MAR = \frac{||p_2 - p_8|| + ||p_3 - p_7|| + ||p_4 - p_6||}{3 * ||p_5 - p_1||} \tag{4.11}$$



(a) (b)

Figure 4.11.: (a) *To find the optimal threshold for the ratio between the width and prominence of a peak, a range of thresholds have been tested against a labeled subset.* (b) *With the new approach we were able to detect double blinks.*

Based on the 68 point landmark model for face detection in OpenFace, we had two different pairs of landmarks for the mouth. One pair is the contour line between the mouth lips and skin (outer), whereas the other is the contour line between the mouth and the inside of the mouth (inner). We calculated the MAR based on both landmark sets and derived the average from both to extract the final MAR. OpenFace already supports a few action units which are representing some state of the mouth, namely the action units 12 (lip corner puller), 25 (lips part) and 26 (jaw drop). The correlation between action unit 12 and MAR was -0.148 (std = 0.398) with a p-value of 0.149 (std = 0.319). Between action unit 25 and MAR, it was 0.248 (std = 0.4) with a p-value of 0.127 (std = 0.296) and action unit 26 and MAR correlated with 0.045 (std = 0.343) and a p-value of 0.262 (std = 0.4). With these numbers, a correlation between those action units and the MAR is almost inexistent, and we decided to include this as a new feature signal. We expected a stronger relationship between the outline and inside of the lips, but analyzing all samples, this is rather low with an average of 0.27 (std = 0.434) and is significantly on average with a p-value of 0.047 (std = 0.157). To visually validate the accuracy of the MAR, we extracted it for a few short samples and compared the signal with the video itself (please see Figure 4.12). In many cases, the signal followed the expected trend, but sometimes it was either noisy or slightly off value. The reasons for this were merely suboptimal lighting conditions or when the head was turned away from the camera in either direction. From the MAR signal, we calculated the standard statistics (min, max, mean, standard deviation, and locations of min and max) and the fit of a linear regression line.

Eye-Gaze. Another essential hint on the current affective state of a person is the eye-gaze. In our experiment, participants were asked to watch more or less always on the screen. Though, a participant could still have a natural reflex to look away for a short moment (e.g., for thinking about a solution or to look away from a cruel picture). Eye gaze is supported by OpenFace out of the box with the direction for each eye as well as combined angles of both eyes. The default statistics are extracted directly on the continuous angles of the eye gaze. Additionally, we discretized the eye gaze by splitting the screen into nine different regions: four for the horizontal and vertical directions, one for each corner and one for the center (please see Figure 4.13(a)). For each region, we summed up the frames in which the eye gaze was pointing to this region and normalized it with the duration of the extracted sample. Therefore, we have one new feature per area, which makes a total of nine. For the GoPro, the feature was mainly highlighting that for most samples, the participants are looking downward, which was the expected behavior as

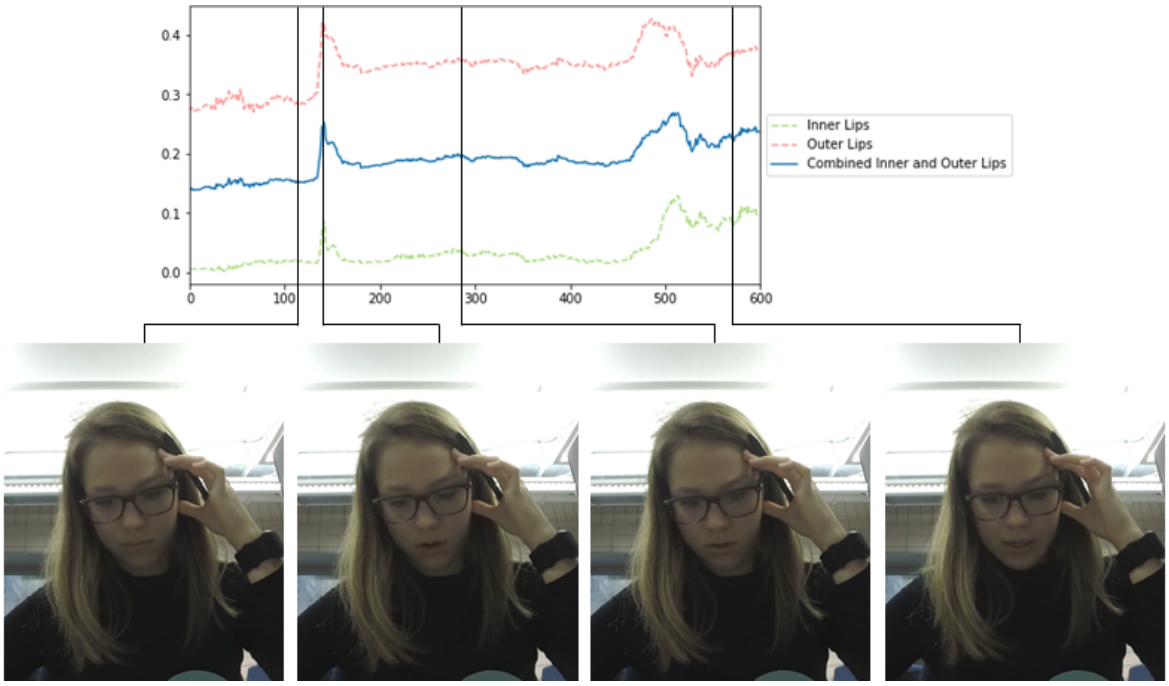


Figure 4.12.: *The extracted MAR in the plot on top captures well the changes of the mouth state over the 10 s. The green line is calculated with the inner landmarks of the mouth and the red line with the outer landmarks. The blue line is the average between these.*

the GoPro was behind the tablet. Still, with this feature, we found a few samples in which the participant is looking away to the ceiling. For the front camera of the tablet, there is slightly more variety in the signal. Though participants were focusing their eye gaze a lot in the center, again and again, they looked around to the edges or corners (please see Figure 4.13(b)).

Movement. The movement of a person is not a facial expression itself, but we assumed that it could help to understand the affective state of a participant better. We found for example that some participants moved closer to the screen when looking at a cruel picture to verify what they are seeing (please see Figure 4.14). Our first consideration was to extract the longest movement of the participant during a sample. This is done by merely parsing the signal frame by frame

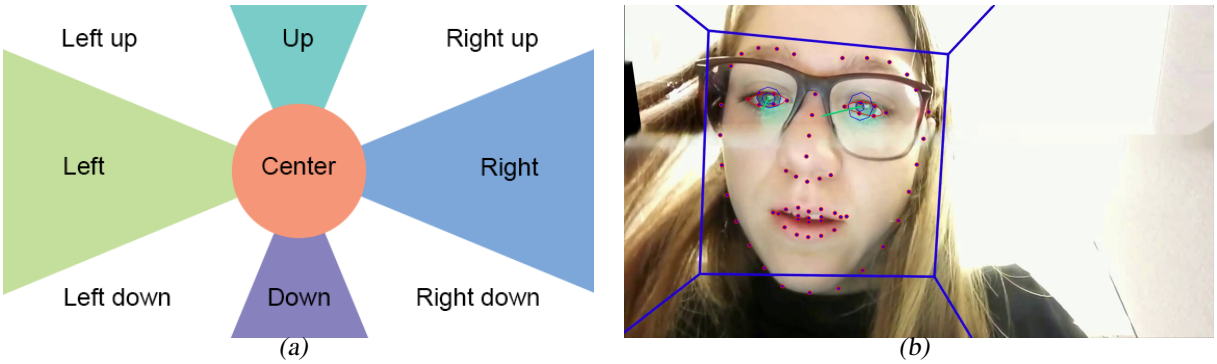


Figure 4.13.: *Eye gaze feature example.* (a) *Discretization of the camera screen into nine disjoint regions.* (b) *Example of the eye gaze from OpenFace.*

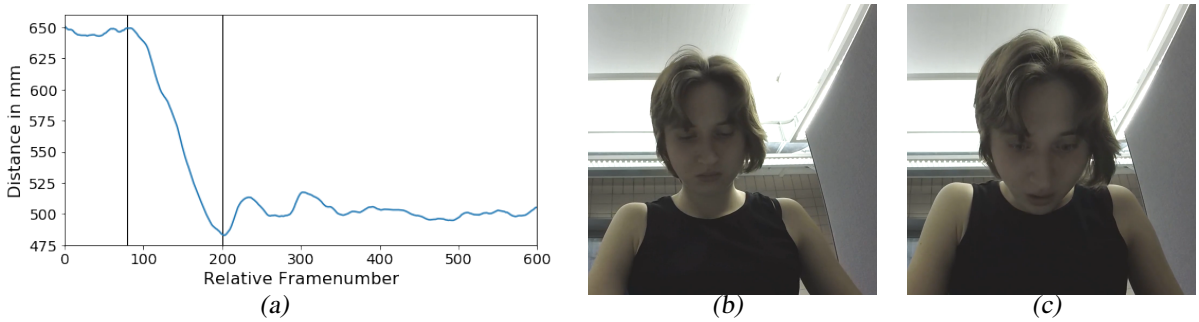


Figure 4.14.: *Example of a participant moving closer to screen.* (a) *Distance to camera during the sample, where* (b) *around frame 80 she is farthest away and* (c) *around frame 200 closest.*

and count consecutive frames in which the direction of the movement remains the same. We then extracted the start, length and total distance of this segment. Second, we summed up the total distance moved over the whole sample to capture somebody continually moving back and forth. Additionally, we calculated the velocity and acceleration over the whole sample and extracted the min, max, mean, standard deviation and location of the min and max. OpenFace calculates the position of the head in 3D with respect to the camera being the origin. Therefore, we extracted the above-mentioned features on each axis (i.e., x-axis, y-axis and z-axis) independently to capture the movements toward and backward to the camera in a separate feature. Additionally, we calculated the distance between the head position and the camera in 3D for each frame and extracted the features on this new signal.

Fidgeting. The fidgeting index has been introduced by Navarathna and colleagues [Nav14] for predicting movie ratings from audience behaviors and is based on motion history images from Davis and Bobick [DB97]. The main idea is to detect the total energy a person is using for the movement. This does not only include the movement of the head but the whole visible body. The energy is calculated for each frame based on the difference between two consecutive frames. Furthermore, it is defined as the number of pixels whose value crossed a given threshold normalized by a defined bounding volume of interest. For our experiment, we set the bounding volume to be the entire region of the video, because many participants moved within the full screen during the whole session. In the paper [Nav14], not only one frame but an average over a few previous frames is used to calculate the difference for a new frame. This resulted in a smoother outcome and decreased the artifacts. To speed up the computation, we stored an adaptive grayscale background which is updated framewise with a weighted average[9] (please see Equation (4.12)).

$$b_{gray} = (1 - w) * b_{gray} + w * f_{gray} \tag{4.12}$$

The variable b_{gray} represents the adaptive background in grayscale and f_{gray} is the currently used frame in grayscale. Both images are stored with a pixel range from 0 to 255. The parameter w represents the weight for the weighted average, and we fixed this value to 0.2. To calculate the difference and therefore the movement for a new frame, we subtracted the adaptive back-

[9]The adaptive background is updated with the new frame only after it has been processed to determine the energy.

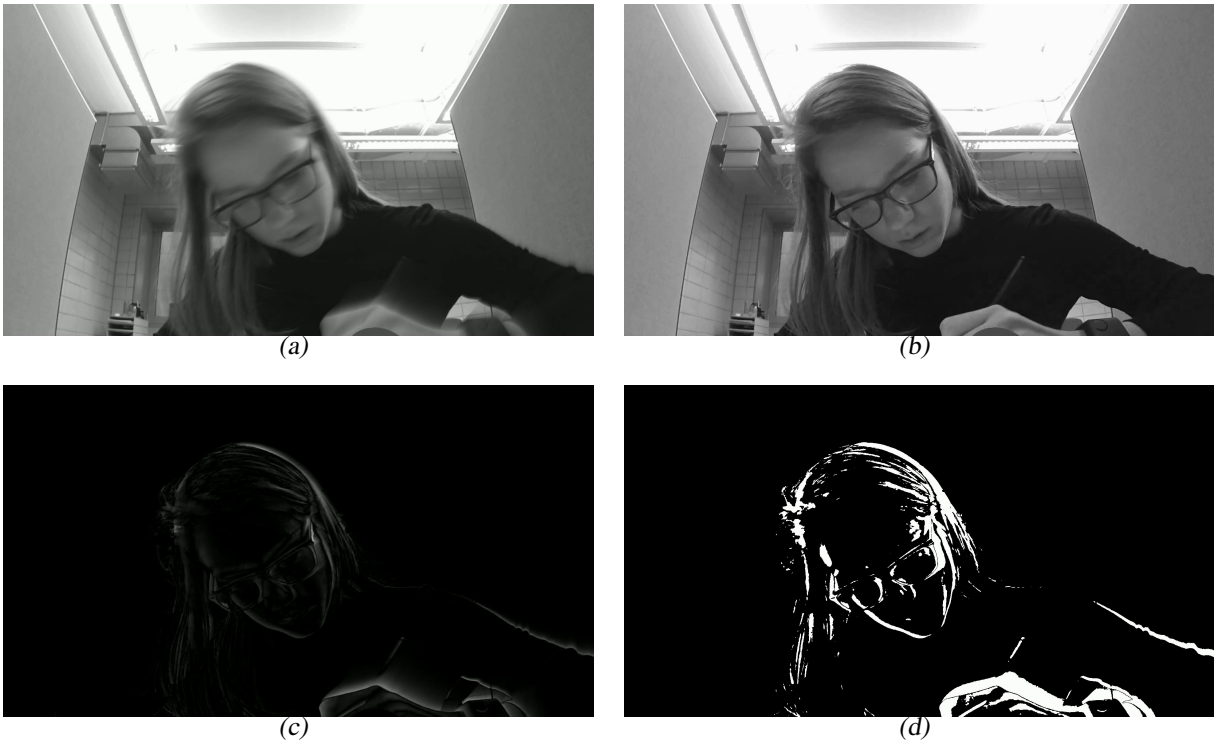


Figure 4.15.: *Illustration of the fidgeting extraction pipeline.* (a) *The adaptive background is stored in grayscale.* (b) *The new frame is converted in grayscale and* (c) *the absolute difference between* (a) *and* (b) *is calculated.* (d) *Finally, the frame is converted with a threshold into a binary format.*

ground from the new grayscale frame and took the absolute value for each pixel. Finally, we extracted the binary result with a threshold of 20, counted all remaining pixels and calculated the percentage of active pixels with respect to the camera resolution. Please see Figure 4.15 for a complete overview of the steps for calculating the fidgeting index. From the fidgeting index, we extracted the fit of the linear regression, min, max, mean and standard deviation of the energy changes for all frames. Additionally, we calculated the total energy by summing up all energy changes and took the location of the minimal and maximal energy change during the sample.

4.5.4. Baseline Normalization

Because every person has subtle deviations for facial expressions and the position relative to the camera varies for everyone, we decided to do a normalization for each participant. At the beginning of the experiment, a seven-minute nature video was presented on the tablet to let the participant relax and come in a neutral state (please see Section 3.1). We first investigated with the help of heart rate and skin conductance measurements if there exists an optimal window with the same window size as for the samples. We considered that the facial expression is relatively neutral around the time including the heart rate minimum or analog for the shimmer minimum. A first analysis of the extracted action units from OpenFace and the video sequence revealed that this approach has some drawbacks. First, only extracting a short sequence can result in a

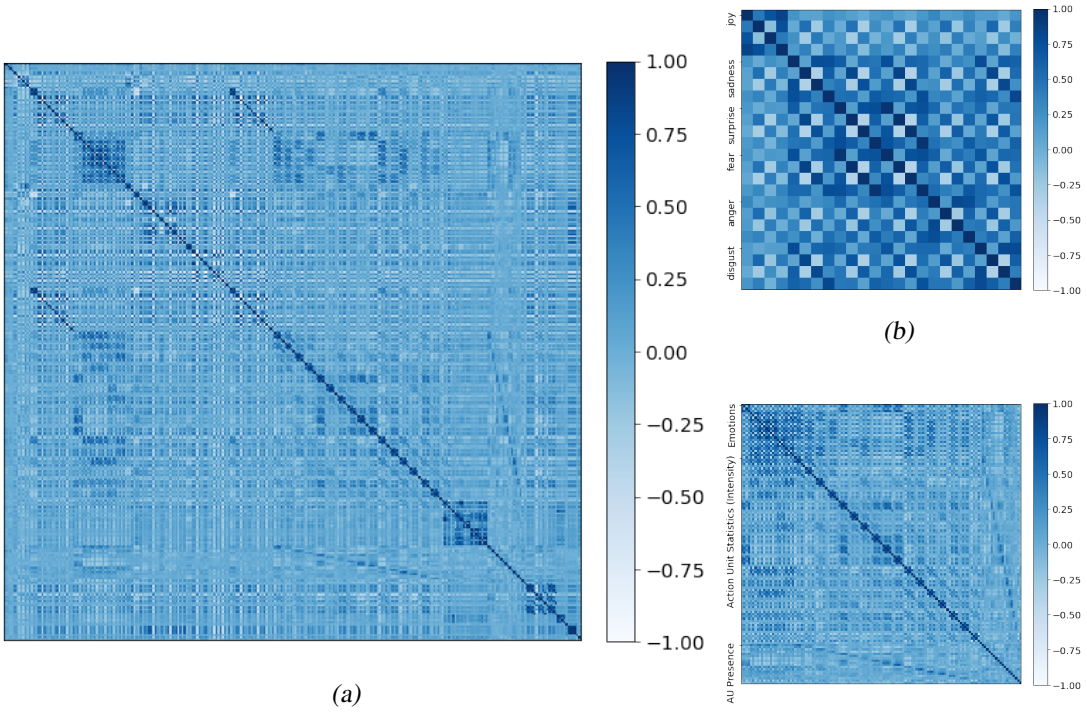


Figure 4.16.: *Correlation matrix of extracted features:* (a) *all features,* (b) *combined emotions versus intensity action units,* (c) *intensity versus presence actions and* (d) *combined emotions.*

biased estimate when the participant during this time has some emotional facial expression. We detected a few participants smiling during the extracted 10 s baseline video, but a few seconds later the expression was neutral again. There is no clear evidence about what triggered the emotion. We assumed the participants remembered something spontaneously while trying to relax, forcing a reaction in the facial expression. Second, there is still a chance that no face can be detected during the determined window. We found one participant drinking during this period, which resulted in a covered face for most of the frames. Therefore, only a few frames extracted meaningful values for the action units. In the end, we decided to use the complete baseline video. This increased the number of valid frames and the amount spontaneous stimuli should be relatively small, resulting in the less biased ground truth. For the baseline, the same features were extracted as for the rest of the samples. During the normalization step, the baseline features are subtracted from the sample features to get the relative change of the facial expression during the sample.

4.5.5. Feature Correlation

All features combined from Section 4.5.3 result in a total of 282 features. More features are not only slowing down the classification algorithms but also increase the complexity of the model itself. As a first manual attempt, we analyzed correlations between features to reduce possible highly correlated features (please see Figure 4.16(a)). First, we can see that most correlations happened to be around the diagonal axis. The most common pattern is the small square with a

size of four features in around the middle which represents statistical features (min, min, mean and std) of action units. Another block that caught our attention are the features representing the six basic emotions (please see Figure 4.16(b)). It can be seen that some emotions have a higher correlation with others, whereas joy (the first four features) is not visibly correlated with the others. This correlation was expected because some emotions share one or more action units. The same reasoning can be used for the association between emotions and the respective action units (please see Figure 4.16(c)). Activity-percentage features from presence action units are slightly correlating with their respective intensity action units. However, the correlation is not as strong as assumed beforehand. This might be a result of differently trained classifiers for both types of action units as stated from OpenFace. The statistics (max, mean, std and sum) for the fidgeting index are highly correlated with each other. We assume that one reason for this high correlation is the way a movement is detected. Looking at the videos with the extracted energy, we saw that there was always a similar pattern for a move. Most of the time, the same silhouette of the participant is detected as an energy change, but the distance slightly varied based on the movement speed. Another correlation, which was expected, occurs between the fidgeting index and movement features. The locations of the minimum and maximum and the linear regression about the fidgeting index were not interacting with any other feature (please see Figure 4.16(a), the last eight features). Another strong relationship is found between movement features in 3D and along the Z-axis[10] and a weaker relationship for the other axes. This is because we combined the single axes into one distance vector for the movement in 3D, and participants were more likely to move back and forth instead of side to side.

[10] In OpenFace, the Z-axis is defined along the camera direction

5

Results

In this chapter, the results of the classification are shown. First, in Section 5.1 the overall setup is described. Second, the pipeline and features are validated with another dataset in Section 5.2. In Section 5.3, we find the optimal window. Finally, in Section 5.4, we analyze the principal component analysis (PCA) and in Section 5.5 the feature importance. The performance is shown and analyzed in Section 5.6.

5.1. Experimental Setup

In this section, we describe the setting for the classification pipelines.

Classes. Per default, we classified each dimension of the valence and arousal space independently. For each dimension, we defined two classes. Ratings from 1 to 3 are called "low" where ratings from 7 to 9 are called "high" (e.g., low valence and high valence). For training and testing, we only included samples with a rating in one of those classes. Ratings between 4 and 6 are ignored to have a more significant margin between the classes. With this setting, we have a total of four different test environments, each with two classes: IAPS and valence, IAPS and arousal, math tasks and valence, math tasks and arousal. For all settings, we have slightly imbalanced classes (please see Table 5.1).

Cross-Validation. To validate our model, we used a nested cross-validation approach. In the outer loop, we validated the performance from the inner loop, which does the hyperparameter optimization. Because the data in the validation fold from the outer loop is never seen during the hyperparameter selection, it is a more accurate estimate of the generalization error. We decided to use the *Stratified Shuffle Fold* (*SSF*) and *Leave One Group Out* (*LOGO*) for generating the folds. *SSF* considers the class imbalance for creating the splits randomly. Therefore, each fold is expected to have the same ratio between the samples as in the complete dataset. With *LOGO* we simulated training on all participants except one, which has been used for validation. The

	GoPro			*Front Camera*		
Classification Dataset	*Low*	*High*	*Total*	*Low*	*High*	*Total*
IAPS on valence	843	1218	2061	695	975	1670
IAPS on arousal	1206	982	2188	993	757	1750
Math tasks on valence	724	1422	2146	273	486	759
Math tasks on arousal	1380	726	2106	585	212	797

Table 5.1.: *Number of samples for each classification problem. Each number represents the number of samples for the classification problem (row) and class label (column).*

main difference between these two fold mechanics is that for *SSF*, we train with samples from every participant, whereas for *LOGO*, one participant is excluded for training and only used for predicting. By comparing these two methods, we gained insight into how the performance is affected by including a participant in the learning step or not. Because *LOGO* generates 88 splits, one for each participant, it is a computationally demanding task. Therefore, we decided to use *LOGO* only for the outer loop whereas hyperparameter optimization was always used in conjunction with *SSF*.

Metrics. For reporting the performance of our model, we used two different metrics. The first we used is the *accuracy* metric, which is the ratio between the number of correct predictions and the total number of predictions. This metric is easy and fast to calculate but does not handle the class imbalance well. To compare the results with a metric that takes care of the class imbalance, we introduced the *Area Under the Receiver Operating Characteristic Curve (ROC AUC)* metric. The ROC curve is calculated by comparing the true positive rate with the false positive rate at different threshold levels. *ROC AUC* is only defined for binary classification. However, for a multiclass problem, it is calculated for each class against all other classes independently, and the average from the single scores is calculated as a final score. The average can be defined as "micro", which aggregates all predictions before calculating the *ROC AUC*, and "macro", which calculates each class independently and takes the average in the end. Furthermore, some participants rated the samples for math tasks to be only in either the *low* or *high* class. In this case, *ROC AUC* is not defined and can not be calculated for these folds. To prevent excluding participants, we aggregated the predictions for each fold in the *LOGO* cross-validation and calculated the *ROC AUC* with all predictions in the end. For this performance, there is no standard deviation available and will be marked as *ROC AUC Aggregated*. Both metrics, accuracy and *ROC AUC*, are performing better for a higher value and range between 0 and 1 for bad and perfect performance, respectively. The random level for accuracy is based on the number of classes and is defined as $1/n$, where n is the number of classes. For *ROC AUC*, the random level is at 0.5 for binary and multiclass classification problems.

Hyperparameter Optimization. The hyperparameter optimization was applied for the steps in the feature selection and the classifiers. The validation was done within the nested cross-validation in the inner loop with the random search. For the *SSF*, we used 194 iterations for the random search, whereas we only used 48 repetitions for *LOGO*. A multiple of 48 for the number of folds was used because we could use 48 processes in parallel on the available cluster. We reduced the iterations for *LOGO* because we have 88 splits instead of only 10 compared to

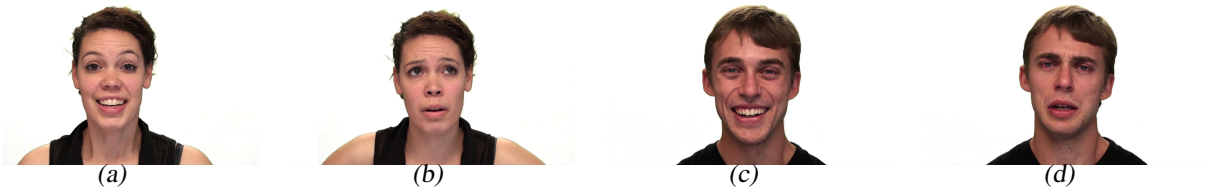


Figure 5.1.: *Two actors from the RAVDESS dataset. In* (a) *and* (c)*, actors were asked to say a phrase with the emotion happy, whereas in* (b) *and* (d) *with the emotion sad.*

the stratified shuffle, which is about nine times slower for computing.

For the hyperparameter optimization, we focused only on one window configuration. This window is based on the optimal window which is analyzed in Section 5.3. We used the same window size for IAPS and math tasks to have better comparability between the two models. Additionally, we only used two different windows shifts, one for IAPS and one for math task samples. We reason that it is more meaningful to have the same window for valence and arousal because it relates to the same emotion from the participant. Additionally, it saves computation power as we can use the same extracted features for classifying valence and arousal, which might be relevant when running the classifications on a tablet or other limited devices. We used integer uniform distributions $\mathcal{U}_I(min, max)$ and float uniform distributions $\mathcal{U}_F(min, max)$ to specify parameter ranges.

We used $\mathcal{U}_F(0.7, 1.0)$ for testing the threshold to remove correlated features and $\mathcal{U}_I(100, 240)$ to cover the range for the parameter k to select the k best features.

Each of the classifiers has at least one tuning parameter. In the following, we are showing the ranges for all tuning parameters.

Logistic Regression: solver = ["lbfgs", "sag"] and C = $\mathcal{U}_F(0.0, 10.0)$.

SVM: C = $\mathcal{U}_F(0.0, 10.0)$, kernel = ["linear", "poly", "rbf", "sigmoid"] and degree = $\mathcal{U}_I(1, 3)$.

Random Forest: n_estimators = $\mathcal{U}_I(100, 300)$, max_depth = $\mathcal{U}_I(1, 5)$ and class_weight = ["balanced", "balanced_subsample", None].

Gradient Boosting Classifier: n_estimator = $\mathcal{U}_I(100, 300)$, max_depth = $\mathcal{U}_I(1, 3)$ and min_samples_leaf = $\mathcal{U}_I(1, 3)$.

5.2. Proof of Concept

To validate our approach and our features with a synthesized dataset, we verified it on the Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS) dataset from Livingstone and Russo [LR18]. This dataset was gathered in the middle of 2018 with 24 professional actors (12 female and 12 male), each singing and saying two different phrases with eight emotions: neutral, calm, happy, sad, angry, fearful, surprise and disgust. Please see Figure 5.1 for two actors saying a sentence with two emotions.

There are a few significant differences to our dataset. First, in our dataset, the emotion was

triggered by an event (looking at a picture or solving a math task), whereas for RAVDESS the participants were ordered to show a specific emotion with singing or saying a sentence. Second, the actors for RAVDESS played the emotion actively as they knew which one to show. For our participants, it was the opposite. They had to think about how they felt after the emotion happened. Third, the lighting condition for RAVDESS was optimal and the actor was standing straight in front of the camera and looking directly into it. Because our participants had to focus on the screen of the tablet, they are not looking into the camera and the angle might vary depending on if they are looking at pictures of solving math tasks. Last, the dataset has eight different labels, not only two. The pipeline handled this implicitly because each classifier can handle multiclass predictions.

The same pipeline and features have been applied to the RAVDESS, with one exception: we did not extract the interpolated RR feature for the blinks, because each video is only about 3 s long and during many videos, there occurs no blink and the feature would be undefined. For the normalization step, we used one sample which was labeled as "neutral". In total, 2452 samples were used from the dataset. The eight classes were relatively balanced. The classes calm, happy, sad, angry and fearful contained 376 samples each, disgust and surprise comprised 192 samples each and neutral included 188 samples.

Accuracy. For this classification problem with 8 classes, the random level is 0.125. Using the *Stratified Shuffle* as a cross-validation fold, performance is not influenced by normalizing an actor with her or his normal state. The performance with baseline normalization is at 0.928 (std = 0.011) and around 0.911 (std = 0.016) without, both with SVC. The reason for this is most likely that each actor exists in the training and testing folds. Therefore we already see subtle variations from each actor during the training step. Using the *LOGO* folds, where all samples of a single actor are only in the test set, normalization increased the performance significantly from 0.715 (std = 0.108) to 0.768 (std = 0.078). This concludes that emotions are universal, but subtle differences can be hard to estimate correctly without knowing anything about a person. This effect has been covered by Elfenbein and Ambady [EA03].

ROC AUC. We made the same findings with the *ROC AUC*. Using the *Stratified Shuffle* and the micro-average for the multiclass handling, the performance with SVC is at 0.958 (std = 0.010) without baseline normalization and similarly at 0.969 (std = 0.008) with normalizing. Using the macro-average results without normalization in 0.911 (std= 0.016) and with normalization 0.928 (std = 0.011) for SCV. However, the baseline normalization increased performance when using *LOGO*. Using SVC with micro average and no baseline normalization performed with 0.850 (std = 0.061), whereas with normalization it increased to 0.894 (std = 0.047). For the macro-average, the difference was slightly higher: 0.715 (std = 0.105) without baseline normalization and 0.768 (std = 0.081) with baseline normalization.

The results with this dataset were promising. However, we strongly assumed that there is a big difference in recognizing actively played emotions by actors compared to rating felt emotions by students.

Classification Dataset	*Window size*	*Window shift*	*Accuracy*
IAPS on valence	2 s	11 s	0.780 (std = 0.020)
IAPS on arousal	2.5 s	11.5 s	0.738 (std = 0.026)
Math tasks on valence	2.5 s	1.5 s	0.814 (std = 0.022)
Math tasks on arousal	3.5 s	5 s	0.918 (std = 0.014)

Table 5.2.: *Window configuration for the best performance for each classification problem.*

5.3. Optimal Window

To determine the best window size and window shift for the samples, we first analyzed the OpenFace features during a defined period for the samples. For an IAPS sample, we looked at the 600 frames during which the picture is shown, and for a math task sample, we looked at 600 frames around the end when the participant received the success message dialog. Looking at these plots visually, we were not able to find a strong definite that leads to an optimal size and shift for the extraction window.

As a second approach, we applied different windows (i.e., distinct shifts and sizes) for the feature extraction and performed cross-validation to calculate the performance. For both sample types, IAPS and math tasks, we defined the possible window sizes to be between 0.5 s and 10 s with 0.5 s steps. However, we defined different ranges for the window shifts. For IAPS we determined the window shift to be between -5 s and 15 s with 0.5 s steps and for math tasks, we tested shifts between -10 s and 10 s with 0.5 s steps.

We tested the performance for all combinations given by these two parameter ranges and for each of the four combinations independently: IAPS and valence (IV), IAPS and arousal (IA), math task and valence (MV), math task and arousal (MA). This resulted in 3280 different cross-validations (20 window sizes, 41 window shifts, 4 test set). Please see Figure 5.2 for an overview of the performance for each classification problem with a fixed window size of 2.5 s and the GoPro videos. Out of all combinations with the different window sizes and window shifts, this revealed an optimal window configuration for each classification problem (please see Table 5.2).

For IV, we can see that prediction is around the random level before the picture is displayed to the participant, which is reasonable as there is no direct connection with the frames and the emotion (please see Figure 5.2(a)). The performance increases slightly as soon as the picture is displayed and holds the same level for the 10 s while the image is displayed. However, predicting the emotions based on the frames during which the participant did the rating has the best performance, with around 0.8. This effect vanishes quickly for a later window shift. To have the best performance during the rating screen instead of during the time the image was displayed surprised us, as we expected it to be during the frames when the picture was shown. We argue that this is because people are used to seeing many images during the daily business and became slightly numb to certain stimuli. However, to rate a picture might trigger the accurate emotion because it has to be compared to similar personal experiences. Analyzing the window size of 10 s for IV, the performance has only one small plateau during the displaying of the picture but never passes a performance above 0.75.

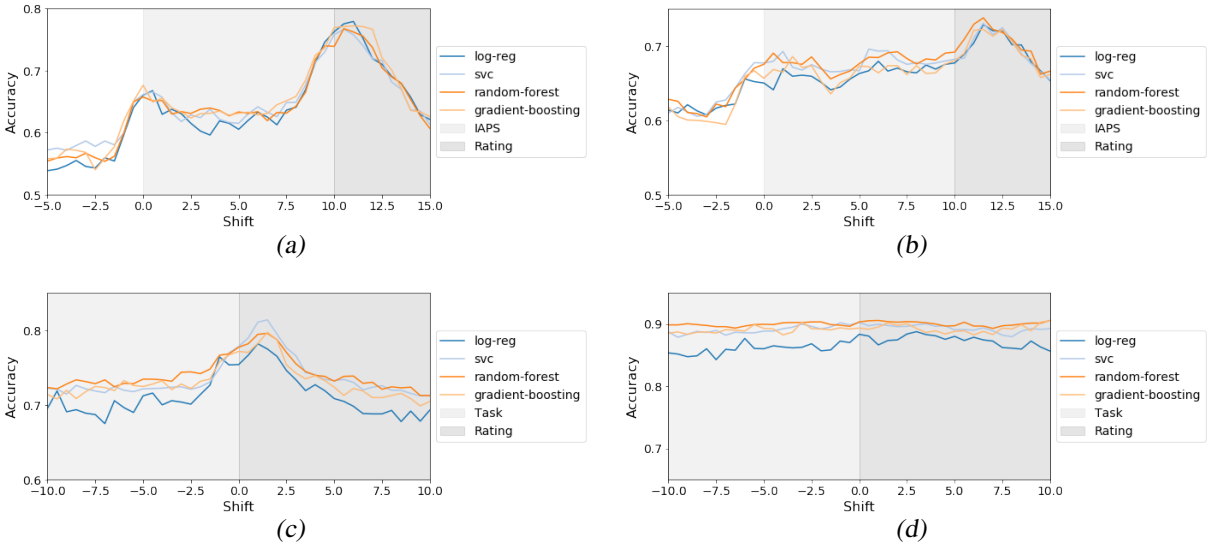


Figure 5.2.: *The optimal window size and shift for feature extraction. All plots are with respect to a 2.5 s window size and show a trend to find the optimal shift. The light gray background marks the frames which belongs to the sample, whereas frames during the dark gray background are during the SAM rating.* (a) *IAPS on valence* (b) *IAPS on arousal* (c) *Math task on valence* (d) *Math task on arousal.*

A similar effect can be seen for IA (please see Figure 5.2(b)). However, it is not as definite and the performance only reaches about 0.72. We reason that it was harder for participants to specify their arousal level compared to the valence.

For MV, it is different (please see Figure 5.2(c)). Already 10 s prior the end of the math task and the fixed window size of 2.5 s, we achieved an accuracy above 0.7. The peak of the performance is around the shift of 1 s, which is during the time the participant sees the dialog with the message if the solution was correctly solved or not. We argue that this is because the participants were actively solving math tasks and possibly knew shortly before the end, whether they could solve it or not. However, to see the monetary reward or penalty made the effect more clear and performance increases up to 0.85.

A completely different trend can be seen for the MA (please see Figure 5.2(d)). The accuracy is about 0.9, independently of the window size and window shift. We argue that the same reasoning applies as for the MV. The active solving of the task helps to show more of the emotions during the sample. However, we do not know what the final reason might be for such good accuracy for any window setting.

Based on the previous analysis, we argue that the window size of 2.5 s is reasonable for IAPS and math tasks. For IAPS, we set the window shift at 11 s, which is the start of the rating screen, whereas, for math tasks, we fixed it to 1.5 s, which is shortly after the end of the math task when the participant saw her or his monetary reward or penalty. We reason that it is more meaningful to have the same window for valence and arousal, because it relates to the same emotion of the participant. Additionally, to have the same window size for IAPS and math tasks allows for better comparability between the models.

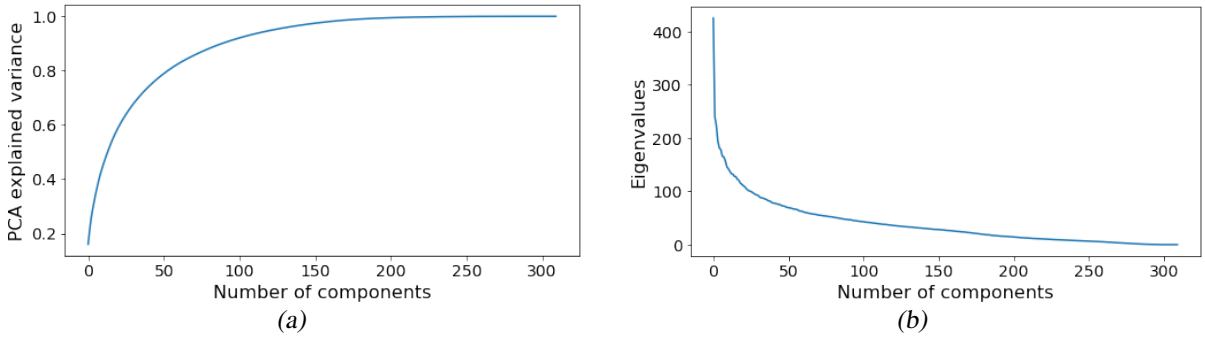


Figure 5.3.: *Principal component analysis of all math task samples.* (a) *Cumulative explained variance based on the number of components.* (b) *Eigenvalues of principal components ordered by the absolute value.*

5.4. Principal Component Analysis (PCA)

With the PCA, we wanted to find out the features which are most important for the variance. We applied it to the extracted and normalized features directly without further selection and analyzed the load of the ten features with the highest contributions for the first five principal components.

First of all, in total 282 features are extracted for the model (please see Chapter 4. To explain 95% of the variance of the data, about 122 and 126 components are needed to explain for IAPS and math tasks, respectively. For 99%, around 162 components for IAPS and 166 components for math tasks are enough (please see Figure 5.3). The load of the features for each principal component is remarkably similar between the IAPS and math task samples. Therefore, the following analysis holds for both and differences are stated. For the **1st** principal component, the most prominent features are from the movement extraction based on the head position. Besides these, only one from eye gaze is used to form this principal component. The **2nd** principal component contains most features from the emotions based on the action units and the **3rd** is a mix between emotions and action units itself. Eye gaze and again some movement features are important to form the **4th** principal component. Only the **5th** component is formed differently for IAPS and math tasks. Again, a significant portion of the movement features is contributing the most for IAPS samples, whereas for math task samples the blinks are highly relevant.

Unfortunately, testing the performance for applying PCA was not promising. Using only a few components received a quite poor performance around the random level. The more principal components we have used, the better the performance became. Nevertheless, it slowly converged to the highest performance that we achieved with all principal components.

5.5. Feature Importance

The feature importance was extracted with the random forest classifier and the Gini criterion. We analyzed the importance with all sound samples of each classification problem (i.e., IAPS and arousal, etc.). Please see Figure 5.4(a) for the ten most important features based on the

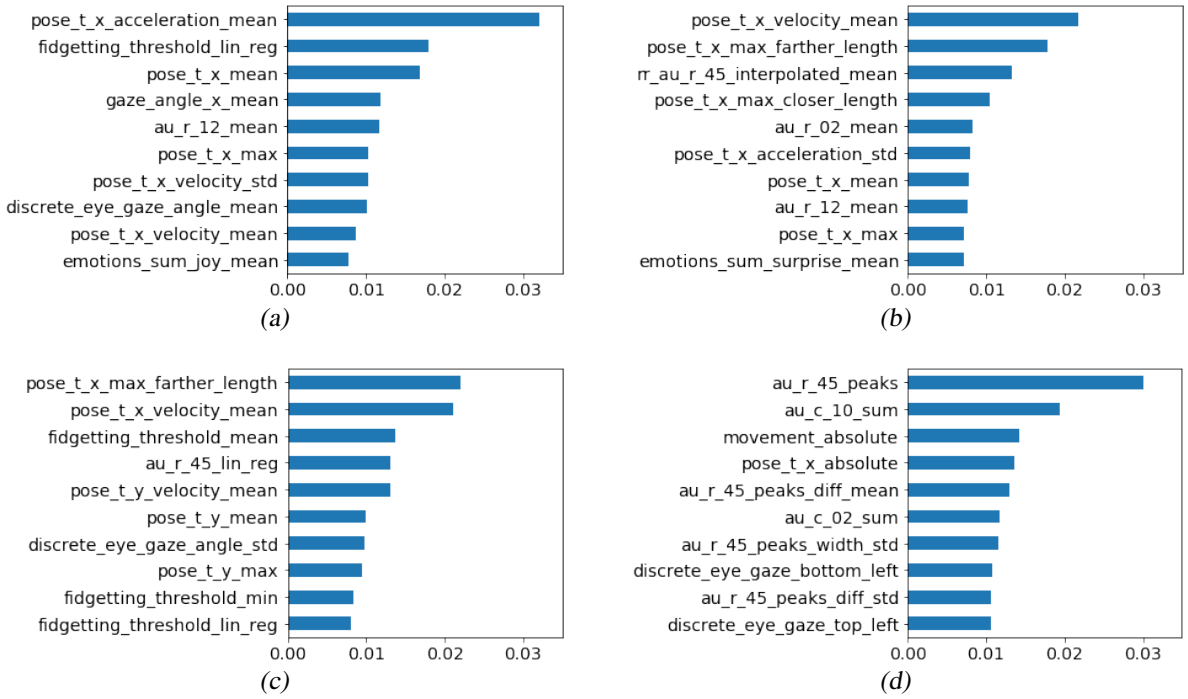


Figure 5.4.: *10 features with the highest feature importance for the random forest.* (a) *IAPS on valence* (b) *IAPS on arousal* (c) *Math task on valence* (d) *Math task on arousal.*

random forest importance for each classification problem (i.e., IAPS on valence, etc.).

Overall, features from the movement and blinks have higher importance for all classifications. This includes the velocity, the moved distance and statistics (i.e., min, max, mean, etc.) from the interpolated RR signal for blinks. However, features from almost all analyzed sections (i.e., action units, eye gaze, blinks, movement, etc.) are present in the 30 most important features. Generally, for math tasks, more features are present from the fidgeting index, whereas for IAPS, more of the movement features are essential. Against our consideration, movement along the x-axis (left and right) has higher importance than along the z-axis (forward and backward).

For IAPS, many of the most prominent features are statistics that are directly extracted from the action units (i.e., max of action unit 12, mean of action unit 6, etc.). Only two features from the emotions are in the top 30 features for valence, namely the mean and standard deviation of joy. On the opposite, for arousal the emotion surprise is essential.

For math tasks, the maximum of the combined emotions joy, sadness, and anger are prominent for valence and arousal. This was expected, as we reason these are the most triggered emotions when the success screen is shown. For valence, the mean of the MAR feature seems to be determining. We argue that this is reasonable because the participants were sometimes visibly swearing while solving math tasks and seeing their solution was wrong. Additionally, for the arousal, the number of blinks is a notable prominent feature with approximately 0.031. We reason that this can be due to the relationship between blinks and stressful situations, as shown by Haak [Haa09]. We further argue that it is more important for math tasks than for IAPS because the participant was more active for solving the math tasks, and therefore, stress is more likely to affect her or him.

	Accuracy		*ROC AUC*	
Classification Dataset	*Stratified Shuffle*	*LOGO*	*Stratified Shuffle*	*LOGO*
IAPS on valence	0.800 (0.016)	0.719 (0.133)	0.879 (0.017)	0.779*
IAPS on arousal	0.751 (0.018)	0.673 (0.127)	0.820 (0.023)	0.733*
Math tasks on valence	0.821 (0.022)	0.722 (0.171)	0.883 (0.015)	0.759*
Math tasks on arousal	0.909 (0.019)	0.619 (0.295)	0.968 (0.010)	0.577*

Table 5.3.: *Performance after hyperparameter optimization for GoPro videos. For each value in a cell, the performance is written with the standard deviation in the brackets. * For the ROC AUC Aggregated is no standard deviation available (please see Section 5.1).*

5.6. Performance

5.6.1. GoPro

Accuracy. With *Stratified Shuffle* cross-validation, the performance for valence is similar for IAPS and math tasks with 0.800 (std = 0.016) and 0.821 (std = 0.022), respectively. But for arousal, the difference is eye-catching, with 0.751 (std = 0.018) for IAPS and 0.909 (std = 0.019) for math tasks. We argue that it was easier for participants to rate their arousal for math tasks with actively solving them and harder for looking at pictures. Additionally, when visually inspecting the videos from the participants, it can be seen that the facial expressions and movements were generally more suggestive while solving tasks compared to only looking at pictures. For the *LOGO*, the performance drop is bigger within the math tasks, from 0.821 (std = 0.022) to 0.722 (std = 0.171) for valence and from 0.909 (std = 0.019) to 0.619 (std = 0.295) for arousal. We argue that this is the same effect as seen for the RAVDESS dataset (please see Section 5.2), where we found that it is easier to predict the affective state when we trained already with the person. Also, around one-third of the participants rated the arousal only in one class (i.e., all low or all high), which could also affect the generalization over participants. For IAPS, the drop from *Stratified Shuffle* to *LOGO* was only about 0.07 for valence and arousal. Again, it improves accuracy when we already know something about the participant.

ROC AUC. With the ROC AUC, the same observations can be made. With *Stratified Shuffle*, performance for valence is approximately the same with 0.879 (std = 0.017) for IAPS and 0.883 (std = 0.015) for math tasks. Arousal again performs better for math tasks with 0.968 (std = 0.010) compared to IAPS with 0.820 (std = 0.023). Using *LOGO* as cross-validation, performance suffers more for predicting math tasks than for IAPS. For math tasks, valence falls from 0.881 (std = 0.014) to 0.742 and arousal drops from 0.968 (std = 0.010) to 0.577. For LOGO and ROC AUC, we have no standard deviation because of the high class imbalance for some participant's ratings (please see Section 5.1 under *Metric*). IAPS suffers less and performs still with 0.779 for valence and 0.733 for arousal, which is both about 0.1 less than with *Stratified Shuffle*. The reasoning is the same as for *Accuracy*.

The ROC plots compare sensitivity versus specificity and are visualized with true positive rate versus false positive rate (please see Figure 5.5). In our case, the sensitivity defines the low

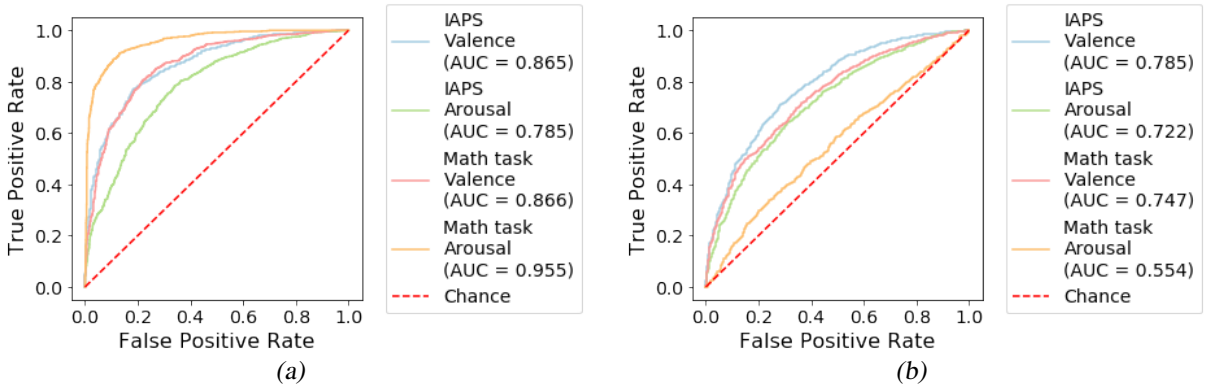


Figure 5.5.: *GoPro: ROC plots for* (a) *stratified shuffle and* (b) *LOGO cross-validation. (Remark: Plots are based from the optimal window search directly. Therefore, numbers might differ slightly from the Table 5.3)*

samples that are correctly predicted as low, whereas the specificity is the high samples that are correctly classified as high. We extracted the ROC plots based on the best classifier from the optimal window (please see Section 4.5.2). For all classification problems with stratified shuffle, we have a ROC AUC clearly above the random level. For IAPS on valence has a similar shape as math tasks on valence, which results in an almost identical performance level. Only the shape for math tasks on arousal is different and follows almost the optimal shape, which would be going through the top left corner. On the other hand, for the LOGO cross-validation, math tasks on arousal is almost along with the random level. Math tasks on valence, IAPS on valence and arousal are again clearly above the random level and show a ROC curve with a reasonable performance level.

Confusion Matrices. We calculated the row-wise normalized confusion matrix to analyze the misclassified samples. To have a more accurate analysis, we calculated the confusion matrix for each fold (i.e., ten matrices for stratified shuffle) and averaged all into one confusion matrix. For IAPS on valence, the high class is predicted more accurately with 85.7% compared to the low class with 69.5%. It is more likely that a low sample is predicted as a high sample than the other way around (please see Figure 5.6). IAPS on arousal is predicting the low class more reliable with an accuracy of 78.6%. On the other hand, the high class is correctly classified with an accuracy of 65.9%. For math tasks it is similar as for IAPS (please see Figure 5.7). Predicting on valence is correct only for about 58.8% of low samples and 41.2% are wrongly predicted as high. On the other hand, high samples are correctly classified for about 91.8%. Remarkably, predicting on arousal is correct for 96.6% for low samples and 79.2% for high samples. In this case, it is more likely to misclassify a high class as a low class.

Comparing the *Stratified Shuffle* with the *LOGO*, IAPS has more or less similar confusion matrices. The true predictions are slightly lower, which can be seen on the diagonal of the confusion matrix. As a result, the wrongly classified samples are slightly higher. Still, a higher proportion gets predicted correctly. Different confusion matrices are generated by the *LOGO* and math tasks. For valence and stratified shuffle folds, the class high was predicted well with around 91.8% high samples classified as high. The low class, on the other hand, was closer to the random level with about 58.8%. Testing with LOGO, this became a significant problem, where the

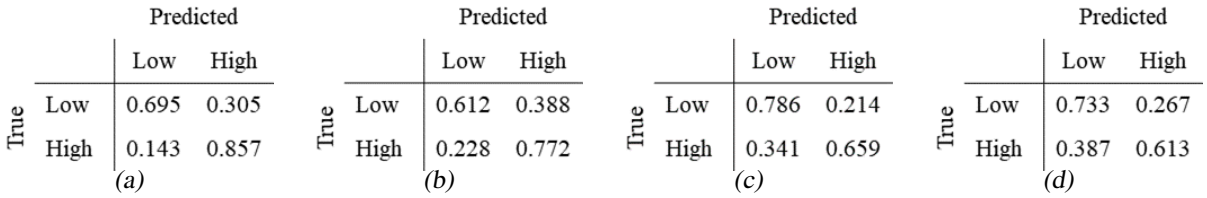


Figure 5.6.: *Confusion matrices for IAPS.* (a) *On valence with Stratified shuffle.* (b) *On valence with LOGO.* (c) *On arousal with Stratified shuffle.* (d) *On arousal with LOGO.*

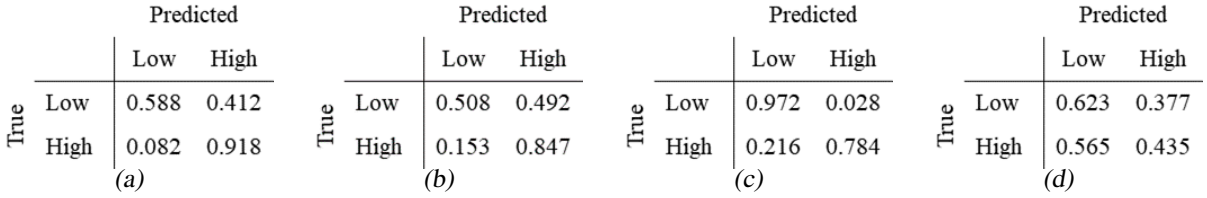


Figure 5.7.: *Confusion matrices for math tasks.* (a) *On valence with Stratified shuffle.* (b) *On valence with LOGO.* (c) *On arousal with Stratified shuffle.* (d) *On arousal with LOGO.*

low samples are classified as low only for 50.8%, which is the random level. High samples are still relatively well recognized as high with about 84.7%. For math tasks on arousal with LOGO, the high classes are classified correctly for only 43.5% of all high samples. Also, the correctly classified low samples dropped from 97.2% with stratified shuffle to a 62.3% with LOGO. With this sharp contrast, we reason that there are more subtle differences between participants for actively solving math tasks. We assume that there is also an influence from existing knowledge about mathematics and if the participant likes solving math tasks. For example, we found some participants still smiling, even when they rated the math task as low valence. Visually analyzing these samples, it seems that some have an insincere and disbelieving smile. This can sometimes not only be seen through the face but also with some hand signals and body movement (e.g., raising the hands close to the face, opening the hand, etc., please see Figure 5.8). We reason that this behavior is because the participant cannot believe that she or he solved the math task wrong, but that the money penalty lowers the rating about the affective state.

5.6.2. Front Camera

Because only a few samples were available for the front camera (please see Table 5.1), we first focused mainly on the GoPro for our classifications. Only after the restoration, we were able to use almost the identical number of samples (please see Section 6.3). Already in this section, we will briefly discuss all the results based on the final restored front camera videos. The final analysis of the improvement through the restored videos is discussed in Section 6.5.

For the front camera videos, we used the same pipeline as for the GoPro samples. To determine the optimal window for the front camera, we run the same search with the new samples (please see Section 5.3). The differences in the found parameters were only small, up to 0.5 s. Therefore, we decided to stick with the same window for the front camera as we used for the GoPro. With this setup, we were learning with the corresponding sequences from both video sources.

(a) (b)

Figure 5.8.: *For math tasks, sometimes the hand gestures could be used to correctly identify the affective state.* (a) *The hands are risen close to the face and form a disbelief signal.* (b) *The hand with the pen is open fast and "aggressively" because she can not believe that the solution was wrong.*

	Accuracy		*ROC AUC*	
Classification Dataset	*Stratified Shuffle*	*LOGO*	*Stratified Shuffle*	*LOGO*
IAPS on valence	0.789 (0.026)	0.732 (0.165)	0.870 (0.026)	0.803*
IAPS on arousal	0.721 (0.050)	0.656 (0.143)	0.809 (0.028)	0.703*
Math tasks on valence	0.778 (0.018)	0.683 (0.172)	0.836 (0.025)	0.726*
Math tasks on arousal	0.893 (0.023)	0.571 (0.352)	0.958 (0.016)	0.538*

Table 5.4.: *Performance after hyperparameter optimization for front camera videos. For each value in a cell, the performance is written with the standard deviation in the brackets. * For the ROC AUC Aggregated is no standard deviation available (please see Section 5.1).*

Comparison with GoPro. The achieved performance with the front camera samples are incredibly close to the performance with the GoPro and differ only about ± 0.03 for IAPS samples (please see Table 5.4). For IAPS on valence, front camera samples tend to perform around 0.03 better with LOGO and both metrics, whereas for the stratified shuffle, it is almost identical. On the other hand, for arousal, LOGO performs 0.03 worse for both metrics and stratified shuffle is again similar to the GoPro samples. For math tasks, the performance is always lower with front camera samples compared to GoPro samples. With front camera samples and LOGO, classifying valence has an accuracy of 0.683 (std = 0.172) and the ROC AUC Aggregated is 0.726. For math tasks on arousal, LOGO performs only slightly above the random level (accuracy: 0.571, std = 0.352, ROC AUC Aggregated: 0.538). However, stratified shuffle cross-validation has an accuracy of 0.893 (std = 0.023) and a ROC AUC of 0.958 (std = 0.016). The performance with the front camera and GoPro samples is overall remarkably similar, which supports the conclusions from the GoPro analysis (please see Section 5.6.1).

ROC AUC. Analyzing the ROC plots for the front camera, we can see the curves for the stratified cross-validation is almost identical with the ROC plots for the GoPro (please see Section 5.6.1). Additionally, the shape of the curve does not differ a lot for most classification datasets. Only for math tasks on arousal with LOGO, the ROC curve is below the random level

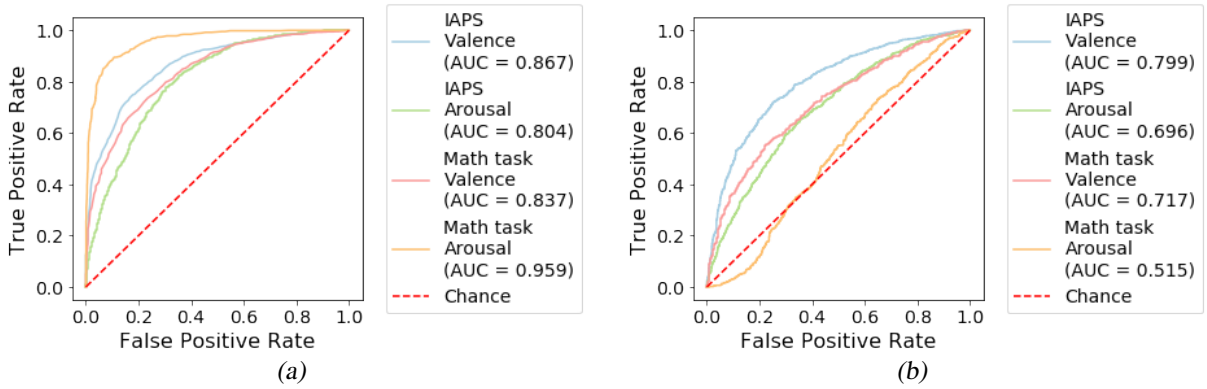


Figure 5.9.: *GoPro: ROC plots for* (a) *stratified shuffle and* (b) *LOGO cross-validation.*

for the first part and does not improve significantly in the upper part. Based on the feature importance (please see Section 5.5), the blinks have a high influence for this classification dataset. We reason that due to the fact that many samples only became available after restoration, there might be the issue that the eyes are inpainted for many frames. We demonstrate in Section 6.5, that inpainting the eyes is not reliable, because there is no guarantee to restore the eyes with the correct eye gaze and eyelid positions.

Confusion Matrices. The confusion matrices are again very similar to the already discussed confusion matrices based on GoPro samples. But for math tasks and LOGO, low valence is only predicted correctly for about 46.8%. Also, high arousal is only classified correctly for 25.7%. As already stated for the ROC plots, we strongly assume this arises from the fact that the eyes are inpainted for many samples, which is a known problem.

Feature Importance. We calculated the feature importance for the front camera samples in the same manner as we have done for GoPro (please see Section 5.5). Remarkably, more or less, the same features are in the 30 most important features. Though the load is slightly different and the order of the features also differs a little. Only for IAPS on valence, the fidgeting index became essential for the front camera compared to the GoPro, with the mean, min and linear regression coefficient of the fidgeting signal as the three most prominent features (please see Figure 5.10).

5.6.3. Comparison with Affectiva

Affectiva calculates the valence already for each frame with the feature extraction (please see Section 4.2). To compare our model, we calculated the accuracy and ROC AUC for the estimated valence from Affectiva.

Experimental Setup. To predict a sample, we collected all values of the valence feature from Affectiva for the respective frames and calculated the average for further labeling. We also analyzed other statistics (i.e., min and max), but because the results were similar, we only considered the mean. The mapping from the range from Affectiva (i.e., -100 to 100) to our classes (i.e., low and high), we tried two different conversions. In a first attempt, we scaled the complete valence signal for each participant to our scale, which is between 1 and 9. With this

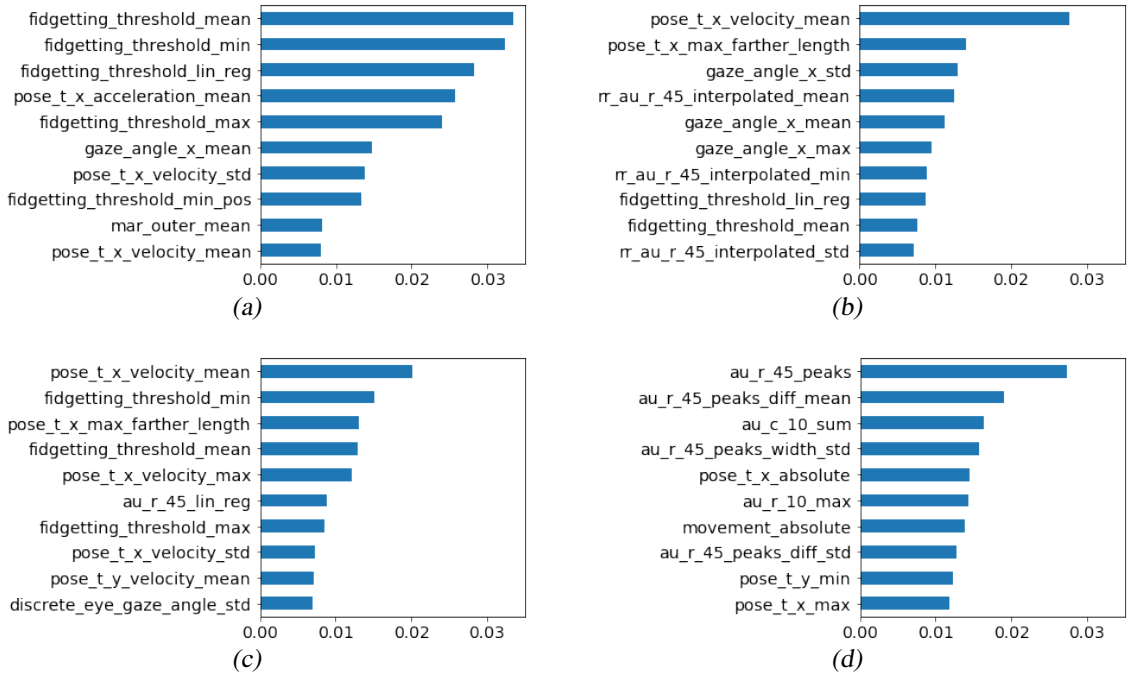


Figure 5.10.: *10 features with the highest feature importance for the random forest (front camera).* (a) *IAPS on valence* (b) *IAPS on arousal* (c) *Math task on valence* (d) *Math task on arousal.*

rescaled signal, we labeled an average of the valence below 5 as low and the remaining as high. The second variation was to calculate the average directly on the original signal and label all negative values as low and the remaining as high. With the first approach to rescale the signal before labeling, we achieved slightly higher performance. Here, we used the same samples as in our analysis in the previous sections. This includes samples that are valid and which have a valence between 1 and 3 or 7 and 9.

Performance. We calculated the accuracy of each participant independently. Here, we report the average of all participants, which is similar to the LOGO cross-validation. The prediction is based on the average of the valence from all respective frames, and the labels are derived from the rescaled signal. For IAPS, the accuracy is at 0.541 (std = 0.145), and for math tasks, it is 0.567 (std = 0.215). Also, the ROC AUC has an insignificant performance, with 0.511 for IAPS and 0.498 for math tasks (this is the aggregated ROC AUC without standard deviation, please see Section 5.1 under *Metric*).

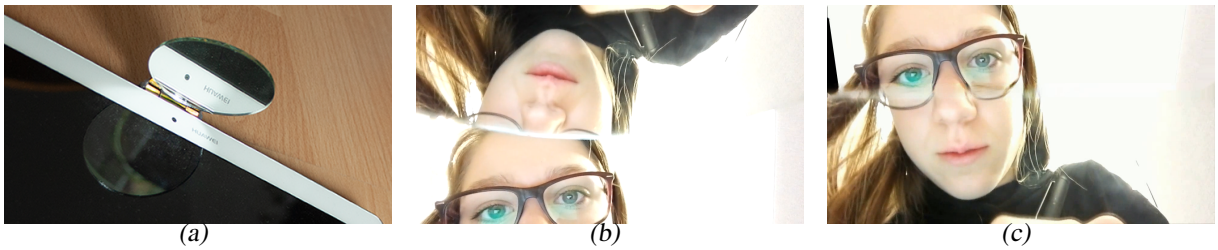
Conclusion. With Affectiva, we were able to achieve only a random level. This performance was expected, as we reasoned already that the underlying training data of Affectiva might not match emotion prediction in our situation. Besides, we assume that Affectiva makes a frame prediction independently of dynamic facial expressions, which can result in misinterpreting some temporal expression. Many participants also had rather subtle changes in their mimic, but some more evident pattern in the body movement. Affectiva only considers the facial expression and ignores other features which might be as essential for understanding the affective state of a participant under these conditions. Based on these findings, we were able to improve the performance significantly by introducing features that consider dynamic facial expressions and include movement with the entire body (please see Section 4.5.3).

6

Face Restoration in Videos

We demonstrated in Chapter 3 our modification of the tablet to have a higher chance of recording the participant (please see Figure 6.1(a)). As a result, we recorded the participant within two different regions on the camera screen (please see Figure 6.1(b)). The focus of this chapter is on the pipeline to convert this input to a usable output for facial recognition software (please see Figure 6.1(c)).

First, in Section 6.1, we describe the first step of the pipeline to rearrange the two separate parts. Second, in Section 6.2 we present how the face is detected with missing facial information to finally restore the face in Section 6.3. In Section 6.4, we briefly explain our framework for the video editing pipeline. The first results are demonstrated in Section 6.5.



(a) (b) (c)

Figure 6.1.: *To record the participant sitting in front of the tablet more reliable with the front cam,* (a) *a small circular mirror was installed.* (b) *This recorded a video which is split into two parts.* (c) *The target image of the pipeline to make the video usable for facial recognition software.*

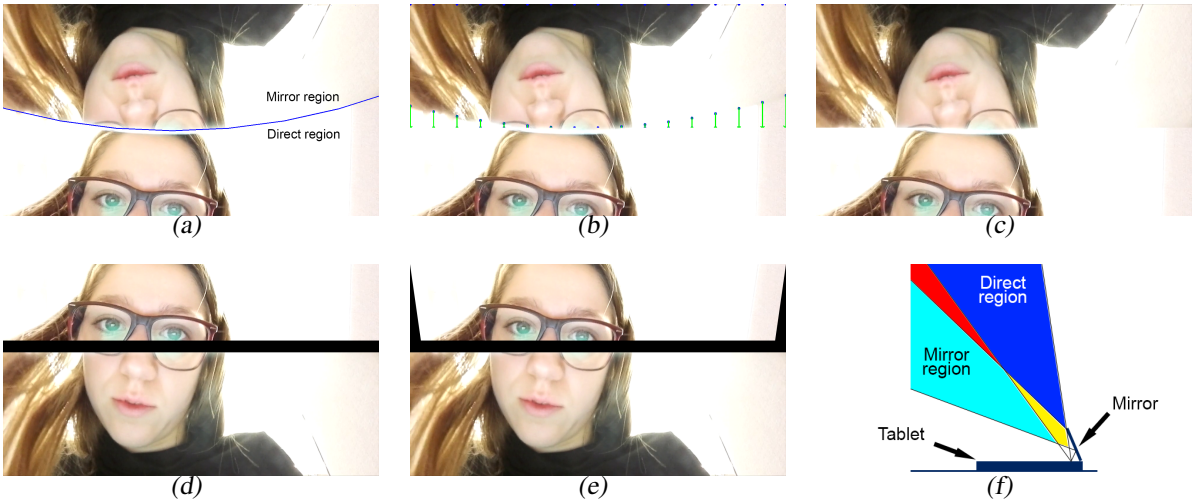


Figure 6.2.: *Steps to rearrange the mirror and direct regions.* (a) *First, we defined participant specific parameters.* (b) *By splitting the mirror into multiple pieces,* (c) *we warped the image to a straight line.* (d) *Then we flipped the mirror region along the horizontal axis and swapped it with the direct region by added a padding.* (e) *After warping the lower corners to fit the edges better, we arrived at the desired output.* (f) *Illustration of the regions as the front camera is recording it.*

6.1. Mirror Handling

For the following algorithm, we define the mirror region as the part of the frame above the circular line in Figure 6.2(a), and the direct area as the part of the frame below the line. Because the mirror has been glued to the hinge to not move during the experiment (please see Figure 3.1), the regions remain the same for the entire time. As it can be seen in Figure 3.1, the mirror has been glued to the hinge to not move during the experiment, which reduced the complexity for transforming the input frame into the desired output frame. We have tried to find the edge of the mirror with the Hough Circle Transform in OpenCV. Unfortunately, the lighting conditions were too bad, and for most of the participants, there was no edge visible. For each participant, we have defined an optimal parameter set, containing the center and radius to cover the mirror. This was adjusted visually and manually for each participant. Because the edge of the mirror was a little bit blurry in the video, we defined the mirror region a little bit smaller to only have the clear image included (please see Figure 6.2(a)).

In addition to the mirror parameters, we have defined a padding that we introduced between the direct and the horizontally flipped mirror image. The padding was needed because in many cases there is a discrepancy between the mirror and the direct view. When the participant was too close to the tablet, the view angle from the direct and the mirror regions were overlapping (please see Figure 6.2(f) the yellow area). On the other hand, when the participant was too far away, neither of the view angles recorded some parts of the face (please see Figure 6.2(f) the red area). Even though the padding is based on the distance of the participant, we decided to choose a static padding to begin with. This reduces the complexity of calculating the optimal padding for each frame. The last parameters covered the fix of the perspective wrap for the direct region. Because the participant was very close and mostly with a non-orthogonal head

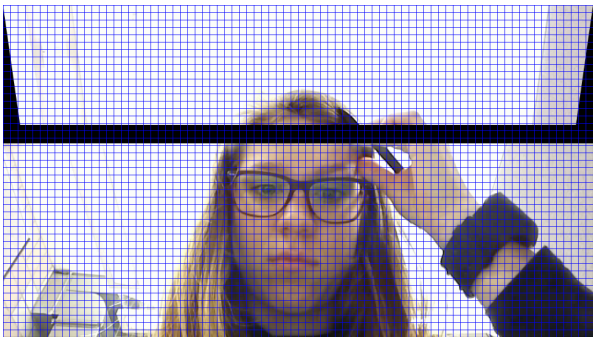


Figure 6.3.: *Splitting image into multiple cells for Histogram of Gradients (HoG).*

position, the top of the head was narrower than the bottom of the face and therefore looked a bit skewed. Furthermore, it did not fit the size of the respective cut to the mirror region.

As a first step, we transformed the mirror region to have a straight edge instead of a curved edge created by the circular mirror. We reason this straight edge improved the overall look of the face and felt more natural, which we verified visually. To transform the edge, we split the mirror region into 16 trapezoids with the same width and the height designed to go along the curved mirror edge (please see Figure 6.2(b)). To calculate the points on the mirror edge, we first calculated the width of each piece based on the width of the video w_{video} and the desired number of pieces p with Equation (6.1). Then, we calculated the position of the i$^{\text{th}}$ point on the mirror edge with Equation (6.2) ($i = 0$ is the leftmost point and $i = p$ is the rightmost point). For this, we required the radius r and the position on the x-axis m_x from the mirror center and the width w of the piece from Equation (6.1).

$$w = \frac{w_{video}}{p} \tag{6.1}$$

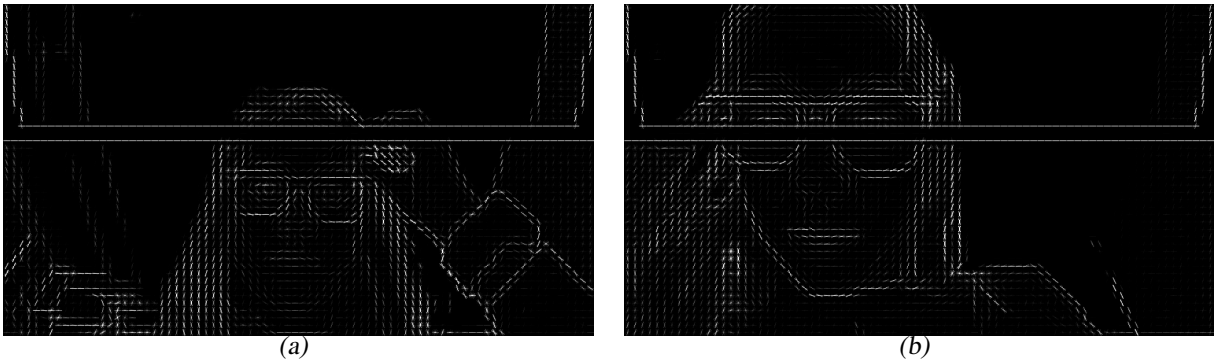
$$y_i = -r + \sqrt{r^2 + (m_x - i * w)^2} \tag{6.2}$$

As a second step, we rearranged the regions correctly (please see Figure 6.2(d)). We extracted the mirror region with the new straight edge, flipped it along the horizontal axis and moved it to the bottom, whereas the direct region with the predefined padding was moved to the top. We decided to rather cut the top of the head as this region contains less important action units compared to the bottom of the face.

As the last step, we corrected the perspective warping of the direct region by pushing the bottom corners closer to the middle. With this, the width of the face matches better between the two edges. Finally, we arrived at the desired output frame as in Figure 6.2(e).

6.2. Face Detection

As soon as some parts of the face are missing, standard face detectors fail to recognize it as such. As a consequence, OpenFace was not able to detect the face in many frames using the front camera video. Also, the face detection with dlib [Kin09], which is integrated into OpenCV, had trouble recognizing the face properly for some frames. We argue that the issue with our set



(a) (b)

Figure 6.4.: *Face detection issues for HoG.* (a) *The HoG visualization for a frame with a positive face detection.* (b) *For this frame, the face was not detected.*

up was the hard transition from the direct into the mirror region with the padding. This affects the histogram of gradient (HoG) extraction negatively which is used in some face detection algorithms.

The HoG is a feature description which is typically used in image recognition where classifiers are trained to detect a defined pattern. To calculate the HoG, the image is first divided into small cells (please see Figure 6.3), each having the same size. For each pixel in the cell, the gradient is calculated and with those, a histogram with n bins is generated for each cell. For example, with using a histogram with nine bins, each bin is covering a range of about 20 degrees. The histograms per cell are further normalized block-wise with the neighbor cells. Each cell can then be visually displayed with the n representative angles, each having the length representing the size of the bin in the histogram. Please see Figure 6.4 to compare two HoG visualizations of one participant, where for the frame in (a), face detection succeeded, wherein (b) it did not. For our examples to calculate the HoG, we used 16 pixels squares as cell size and nine orientations for the histogram. Other facial classifiers might be trained on different parameters and thus might suffer less or more of the found effect.

This situation motivated us to restore the face in the video to improve the detection rate. To do so, we first tried to detect the face on the output after the mirror handling. If the face was detected, we stored the position of the found facial landmarks and continued to the inpainting step (please see Section 6.3).

In cases where no face was detected, we restored the image based on the last known position of the face (please see Section 6.3) and applied the face detector a second time on the reconstructed image. When the face was detected successfully with the temporary restored image, we saved the position and rotation of the detected face for the inpainting step. If the face was not detected, we did not reconstruct this frame and used the original frame for the output video. This should reduce the introduced artifacts based on wrong facial information.

With this two-step face detection, we improved the detection rate significantly. For our restored videos, we found that around 74.5% of all processed frames were restored in total. For about 41.9% of all total frames, the face detection was successful on the first attempt with the original frame. Approximately 32.6% of all total frames were reconstructed based on the indirect detection step, where the face was found after a first inpainting attempt. For videos, which had a

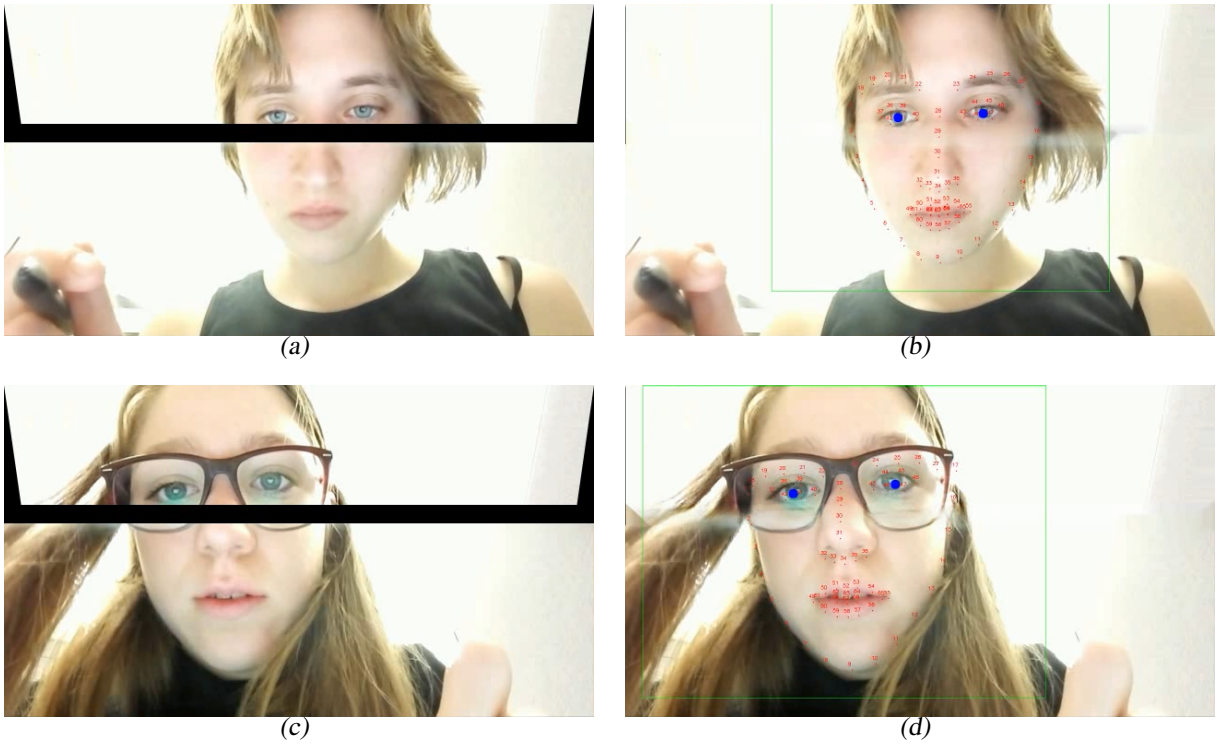


Figure 6.5.: Face detection after inpainting with the last known position of the face. (a) *and* (c) *Even though a lot of the face is visible, no face was detected with OpenFace.* (b) *and* (d) *show the debug frame from the face detection step, where the face was detected after inpainting the frame with the last known location of the face. Additionally, the detected facial landmarks and the centroid for the eyes are shown for debug reasons.*

lower average confidence, proportionally more frames are reconstructed with the two-step face detection. Because these statistics cover the whole video of each participant, also irrelevant frames were included into this statistics (e.g., the introduction of the experiment).

6.3. Face Restoration

In this section, we describe the steps for our face restoration. We used an existing implementation for an inpainting model[1], which is based on the latest research from Liu and colleagues [Liu18].

In Section 6.3.1, the dataset to train the model is shown. The required transformation of the video is explained in Section 6.3.2 and the step to apply the model on the video is demonstrated in Section 6.3.3.

[1] Keras implementation by Mathias Gruber. Code: https://github.com/MathiasGruber/PConv-Keras

(a) (b)

Figure 6.6.: *Examples from the inpainting model with pictures from CelebA-HQ.* (a) *The right eye and both eye brows are completely covered and inpainted reasonably well, as both eyes seem to point in the same direction.* (b) *A position, which could occur when the tabled is turned for better writing. The inpainting places the missing nose and completes the missing eye and mouth.*

6.3.1. Dataset and Training

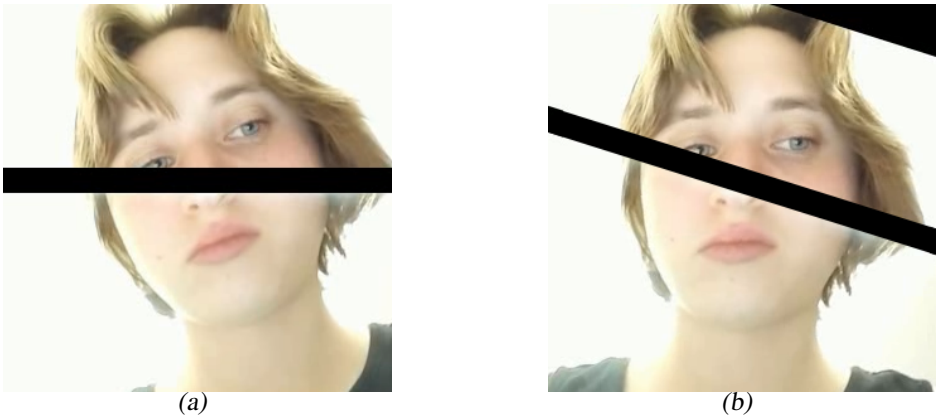
Our model is trained with the dataset CelebA-HQ [ea15b] which contains 30'000 face aligned images from celebrities. For this dataset, face alignment was done to have the eyes and mouth in the same position. To train our model, we split the dataset into a train set of 25'000 images, a test set of 2'500 images and a validation set of 2'500 images. We trained on a single NVIDIA GTX 1070 (8GB) with a batch size of 2. During the first 50 epochs, we used batch normalization in all layers and set a learning rate of 0.0001. For the last 20 epochs, batch normalization was disabled and the learning rate set to 0.00005. Masks have been generated randomly for each batch and were defined similarly to the expected mask which covered the face for our front-cam video (please see Figure 6.6). Even though the mask in Figure 6.7(b) is rather steep, we found training our model with these was more reliable. For the first attempt, we were only training with fewer step angles, and for some participants, who rotated the head a bit more for writing, the inpainting did fill in an arbitrary pattern. Training finished within nine days. The training loss was 170338.48 and the validation loss 181162.97.

6.3.2. Input Transformation

The input for the inpainting model has to be as close as possible to the trained data. The resolution of the input layer is a 512x512 color image with a range from 0 to 1 for each color channel. Based on the CelebA-HQ dataset, the eyes need to be horizontally aligned and the position is expected to be relative to the top left as origin at 190 pixels | 245 pixels and 390 pixels | 245 pixels for the left and right eye, respectively. The mouth is always at the height of around 375 pixels.

The following algorithm was inherited from PyImageSearch[2] and adjusted to our problem. To horizontally align the image, we used the landmarks of the eyes found in Section 6.2 to compute the centroid of each eye by averaging the respective landmark points. Based on the differences on each axis between the two eye centers, we calculated the rotation angle with the arc-tangent.

[2] "https://www.pyimagesearch.com/2017/05/22/face-alignment-with-opencv-and-python/



(a) (b)

Figure 6.7.: *Face alignment.* (a) *Input frame.* (b) *Rotated frame to have the eyes horizontally aligned.*

The final rotation matrix was generated with the midpoint between both eyes and the previously found rotation angle. This rotation matrix is applied to the image, and the inverse transformation matrix is stored for the later inpainting process (please see Figure 6.7). Based on the midpoint and distance between the eyes, we extracted the bounding box which outlines the face respecting the eye position constraints.

With the rotation matrix, we also created the binary mask which is needed as a second parameter for the inpainting model. Based on the previous step, where we adjusted the mirrors (please see Section 6.1), we had the information about the missing part in the output frame. By applying the rotation matrix on this binary mask, we generated the matching mask for the model.

6.3.3. Apply Inpainting

Having the input image and mask, we could call the model with these parameters and read the restored image directly, but we only replaced the region of the mask in the original image. During testing different inpainting models, some introduced artifacts on areas which the face was visible. This step ensures the correct replacement of only missing parts, independently of the underlying inpainting model. Even though our final model does only restore pixels as defined in the input mask, we left our replacement constraint active as a sanity check.

As the last step, we inpaint the remaining empty region from the mask which was not affected by the face restoration. For this, we use the inpainting method from OpenCV with an inpaint radius of 3.

6.4. Video Pipeline

To speed up our implementation for applying different steps on various videos, we created a framework to simplify this process. We implemented a video-pipeline class, which has a similar idea as the pipeline class from sklearn. The steps which need to be applied to the video processing must be passed in the correct order. The pipeline itself ensures that each step is

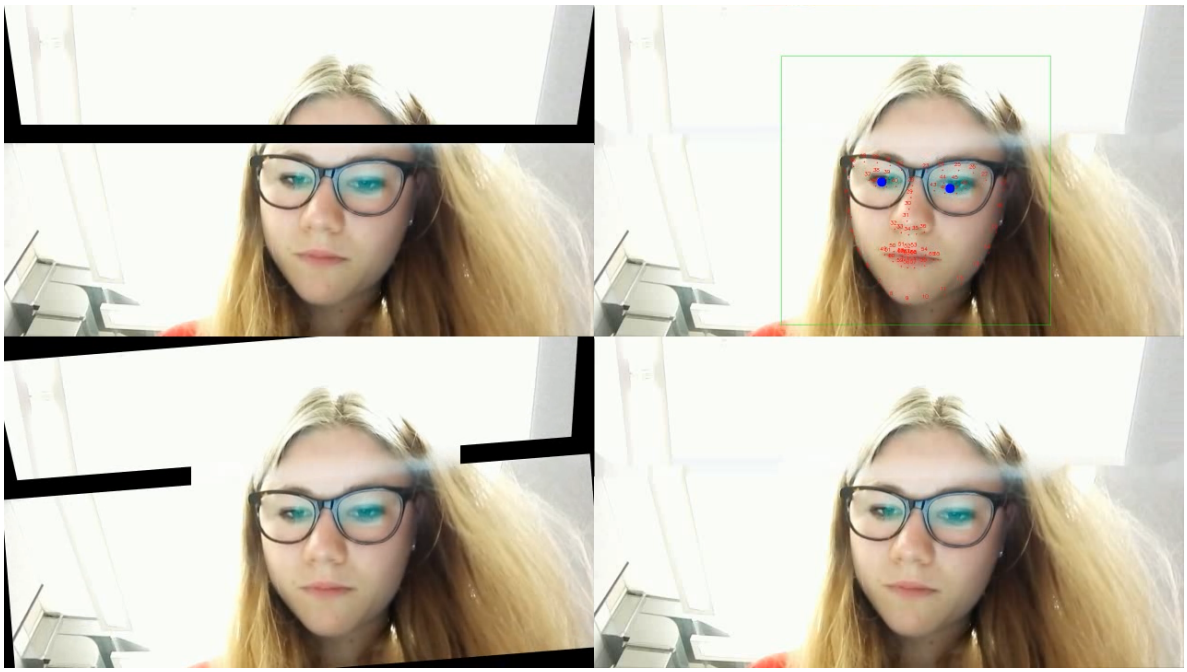


Figure 6.8.: *Composition of the video pipeline to validate the correctness of the steps. Top left: Input frame. Top right: Output frame with additional debug information. Bottom right: Face aligned input frame. Bottom left: Final output frame.*

initialized with the correct video parameters, which might have been changed from previous steps (i.e., the first step is a scaling step and the second relies on the height of the image). The same pipeline will work when one step is removed or a video with another resolution is applied. This interchangeability was needed for this thesis, because the front camera videos needed different preprocessing steps as the GoPro videos. To be able to watch the process of the pipeline of the video and to verify the correctness of the steps, a composition can be applied to display different steps. Please see Figure 6.8 as an example with a 2 by 2 composition, showing different steps and debug screens during the inpainting of a video.

Pipeline. The pipeline is the main class connecting everything together. It is initialized with the steps, controllers and composition. Additionally, it initializes the steps correctly for a given video file and applies the steps on each frame and saves the final output into a video file.

Steps. A step is a small, encapsulated transformation of an input frame to a desired output frame. The input frame can be from the video file itself or from a previous step. The output frame can be written to a new video file and is passed to the next step if there is any. Each step should be independent of other steps, but also not contain to many different computations as it will be harder to maintain. Currently, steps cannot pass information from one to another, except during the initialization to specify if the step is changing the height or width of the frame. Each step can have different initialization parameters and implements a function which is used by the composition to be displayed during the processing. Nevertheless, additional functions can be defined and included. For our work, we implemented the following steps:

Saver: Saves the output from the previous step into a specified video file. The input frame is given without any transformation to the next step.

	Front Camera (original)			*Front Camera (restored)*		
Classification Dataset	*Low*	*High*	*Total*	*Low*	*High*	*Total*
IAPS on valence	695	975	1670	826	1192	2018
IAPS on arousal	993	757	1750	1169	964	2133
Math tasks on valence	273	486	759	682	1310	1992
Math tasks on arousal	585	212	797	1324	652	1976

Table 6.1.: *Number of samples for each classification problem with restoration. Each number represents the number of samples for the classification problem (row) and class label (column).*

Viewer: Displays the output from the previous step on screen. The input frame is given without any transformation to the next step.

Scaling: Scales the frame by the given factor.

Switch Mirror: Applies the transformation of the image and normal region based on Section 6.1.

Movement Detector: Implements the fidgetting index and has a binary image as output (please see Section 4.5.3).

Face Inpainting: This is the implementation for the face detector and face restoration from Section 6.2 and Section 6.3, respectively. We combined those two steps into one class to have the information about the detected face directly available for inpainiting. Additionally, as the face detection is based on the capabilities of the inpainting, it simplified the implementation and dependencies between separate classes.

Composition. A composition is basically a combination of different images generated by the steps. This can be a final output of one step or an intermediate picture generated by a step. The composition takes the reference of the function generating the image to display and can point to any function of any step. Currently defined compositions are the 2x2, which shows 4 images in parallel, a 2x1 and 1x2 which shows two images horizontally or vertically aligned, respectively, and a 1x1 which shows simply on image. Additionally, when specified, the composition can be saved in a video file to analyze the steps in a later state.

Controller. A controller is handling the keyboard input during the video processing. It can be used do either manipulate the position of which frame has to be processed next, changing parameters live for some steps (if implemented), taking snapshots from the current frame or quit the processing.

6.5. Results

We have already seen in Section 6.2, that around 74.5% of all processed frames from the front camera videos have been restored, with either directly or indirectly detecting the face. Not all of these frames are relevant, as the introduction and frames which do not belong to any sample are included in this statistic.

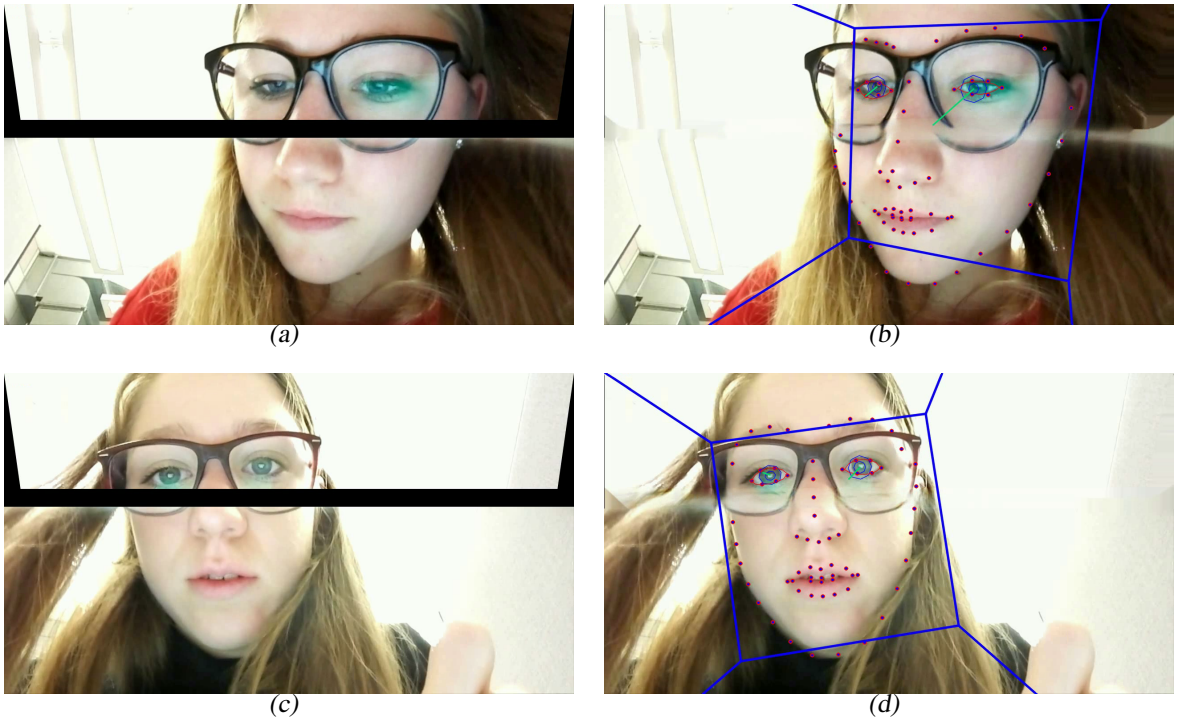


Figure 6.9.: *Improvement of face detection by detecting the face for new frames.* (a) *and* (c)*: Face was not detected with only adjusting the mirror and direct region.* (b) *and* (d)*: After the face restoration, the face was detected.*

Confidence Analysis. The confidence with OpenFace increased significantly from the original videos with 0.630 (std = 0.436) to the restored videos with 0.927 (std = 0.179). The achieved confidence level was similar to the average the GoPro with 0.954 (std = 0.121) (please see Section 3.2). Analyzing only frames that belong to IAPS sequences, the average confidence is 0.945 (std = 0.138), whereas, for frames from math task sequences, the average is 0.899 (std = 0.223). We assume this is because participants bent over the tablet for solving math tasks, and some came too close to the camera for an accurate face detection because the face was cut off.

Additionally, we were able to use more valid samples for training (please see Table 6.1). In total, for IAPS on valence we were able to use 348 additional samples and for arousal 383. For math tasks, the restoration was even able to validate additional 1233 samples for valence and additional 1179 samples for arousal. For checking if a sample is valid, we have used the same thresholds as defined for the GoPro, because the confidence levels were similar after restoration. The frame is valid for a confidence level over 0.82, and about 80% of all frames for a sample must be valid for the sample to become valid (please see Section 4.5.1).

Performance Analysis. In Section 5.6.2, we already analyzed the performance with the reconstructed videos. In this section, we only show the differences between the original and inpainted videos. Interestingly, the performance for stratified shuffle cross-validation is about the same, independently of the reconstruction. Because we only used valid samples, we did not extract features from noise, but with the original videos, fewer samples could be generated. Therefore, already with fewer samples, we were able to achieve approximately the same performance as

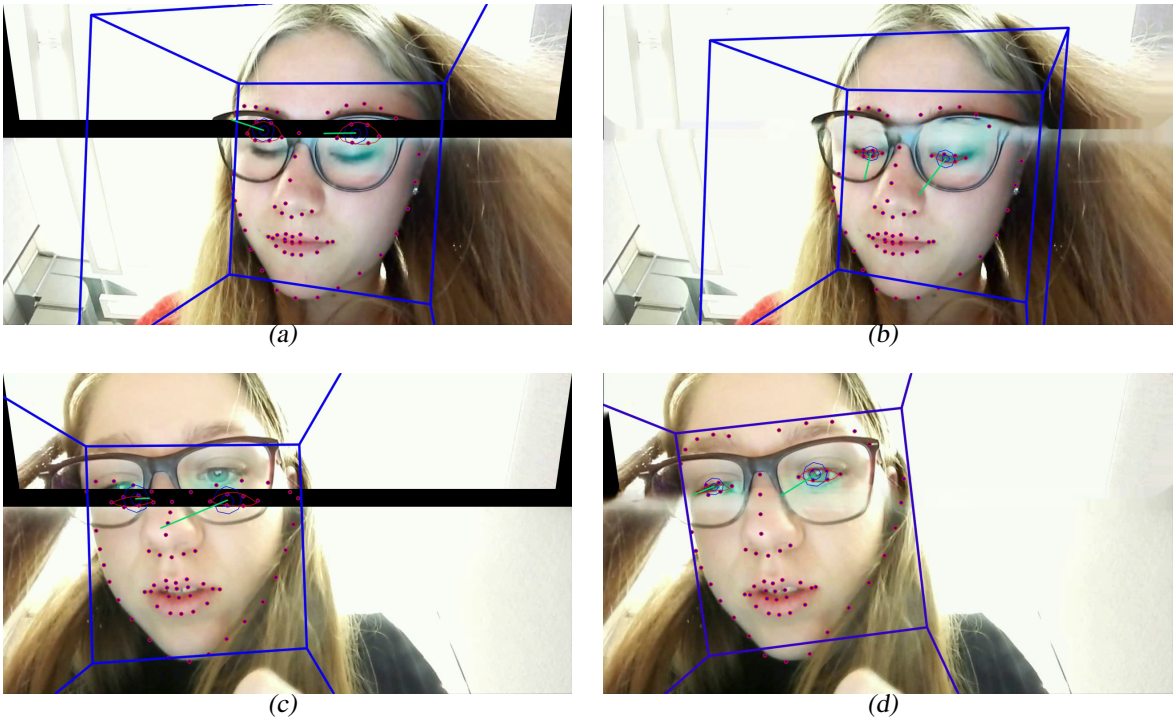


Figure 6.10.: *Face is detected more accurate.* (a) *and* (c)*: Face was detected with a poor accuracy.* (b) *and* (d)*: Accuracy of landmark assignments improved.*

with all samples from the GoPro. For math tasks, this is even more interesting because the dataset based on front camera samples was only about one-third of the dataset based on GoPro video samples. On the other hand, with using the LOGO cross-validation, the reconstructing of the videos improved the accuracy of math tasks on valence from 0.598 (std = 0.261) to 0.704 (std = 0.162), which is closer to the performance of the GoPro. The same holds for ROC AUC that increased from 0.606 to 0.745. For other performance levels, the restoration improved mostly by a small amount, up to 0.05. Only in the case for math tasks on arousal, the performance is slightly worse for the accuracy and ROC AUC. We assume that more training samples are improving the performance to a certain extent. On the other hand, for arousal, it might be harder because the variance within each participant is known to be little (please see Section 3.2).

Visual Comparison. Comparing the face detection before and after the face restoration revealed that in many cases, the results improved. First, the face was detected in many frames where it was not successful before (please see Figure 6.9). With this, more frames have a confidence level over the threshold and can be used for the pipeline. To analyze the reliability of the reconstructed videos, we first wanted to calculate the correlation between features that are obtained from GoPro videos and features that are extracted from front camera videos. But even for the participant with the highest average confidence in the front camera video (mean = 0.944, std = 0.160), we found only a weak correlation for many features from OpenFace. We argue that this is due to the different head pose for the two independent video sources. Another can be, that only small changes in the time synchronization or slightly different trends will result in a weak correlation, even if visually there seems to be a stronger relationship. Therefore, we compared a few action units visually for the same sequences. For most, the trend overall trend

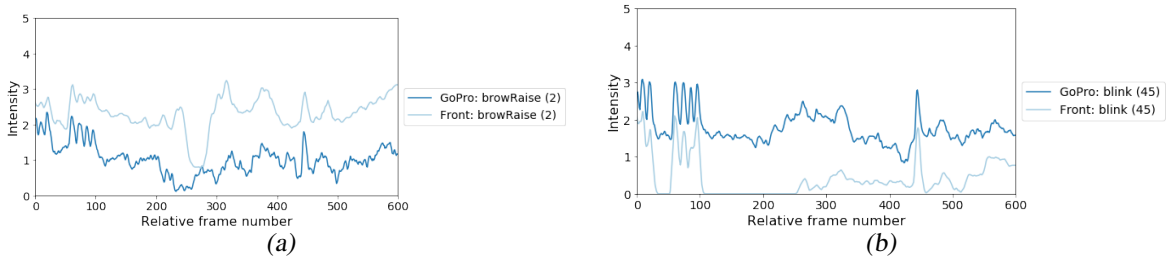


Figure 6.11.: *Comparison of extracted action units from reconstructed and original videos. Both action units show a similar trend for one example.* (a) *Action unit 12 (brow raise) and* (b) *action unit 45 (blink).*

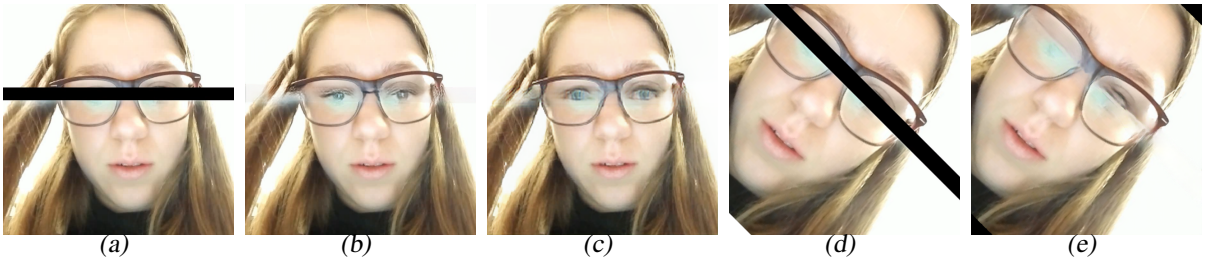


Figure 6.12.: *Face reconstruction NVIDIA online demonstration.* (a) *For the aligned input,* (b) *the eyes could be restored.* (c) *Same input reconstructed with our trained model.* (d) *As soon as the face was rotated,* (e) *The face was not reconstructed properly.*

seems to be similar for both (please see Figure 6.11).

Comparison to NVIDIA-demo and Simple Inpainting. NVIDIA has an online demo[3] with their inpainting model, which implements the same paper [Liu18] as for our self-trained model. With the demo, it was possible to restore some faces with a mask from our video pipeline. After a few tests, we found that it has the same limitations about the input. The eyes are expected to be on a certain height, and the face needs to be aligned to have the eyes together on the equal height. Otherwise, it does not reconstruct the face correctly (please see Figure 6.12). As we have the same limitation with our model, we have the advantage to align the face as desired before we reconstruct it. Therefore, no direct comparison is possible, and for a video restoration, each frame would have been manually restored via the demo.

For a quick comparison, we applied the inpainting algorithm from OpenCV directly on the videos. This is a somewhat naive but simple restoration, as it is not aware of the object, but only about the neighboring pixels. Remarkably, for some head positions and missing parts, this was already improving the face detection with OpenFace. We inpainted the videos of nine participants, and the confidence was around 0.906 (std = 226). Mainly, when no critical areas of the face were covered (e.g., the nose), the face could be recognized with this simple approach. This could be already a good starting point for a fast reconstruction when it is known to have all the essential parts available most of the time. Also, this could be considered as a starting point for our algorithm when no face can be found or when only a short sequence needs to be reconstructed. Nevertheless, when the eyes or the mouth were missing, it did not reconstruct

[3]https://www.nvidia.com/research/inpainting/selection, 01.03.2019

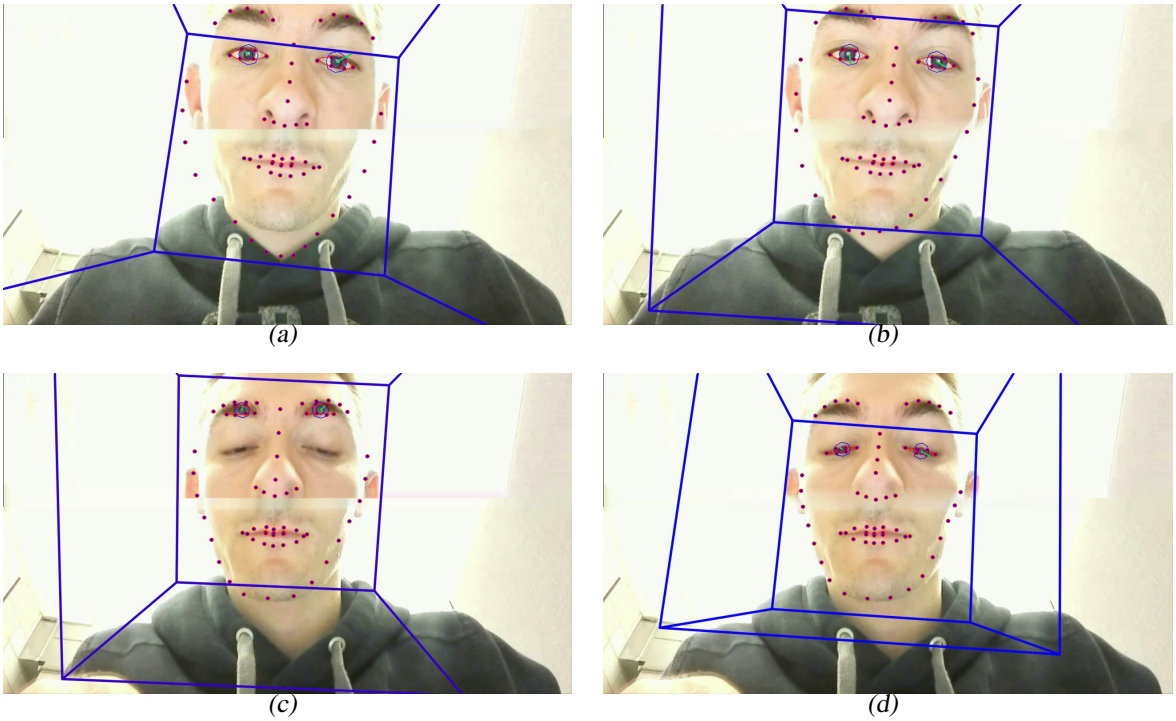


Figure 6.13.: *Comparison with our face reconstruction and the simple inpainting.* (a) *and* (c)*: With the simple inpainting algorithm, the landmarks are sometimes not accurately detected.* (b) *and* (c)*: The landmarks are more accurately aligned with our face reconstruction.*

the face, but only stretched the pixels along with the padding (please see Figure 6.13(a)).

Analyzing a few frames, we found that in many cases, with our approach, the facial landmark mapping of the face detector was more accurate (please see Figure 6.13). For both methods, the face was found, but in some cases, the face seems to be detected wider as it is after only inpainting with OpenCV. We reason that this is due to the contour of the face, which is not as continuous around the inpainted part as in our approach. In Figure 6.13(c)), the eyes are detected incorrectly with only the simple approach, whereas with our approach, the landmarks are appropriately assigned.

Known Issues Currently, our restoration has to be applied on the complete video because the two-step face detection relies on a successful face detection from a previous frame. The issue for only restoring the samples was that the chance to detect the face was smaller. If the participant had the eyes in the top region and the mouth in the bottom area, the face was never discovered during the sample. When frames beforehand are analyzed and restored, there was a higher chance that the face was once identified and can be tracked with the indirect face detection during the sample. This could probably be solved with face detection, which is trained for our specific occlusions. After visually inspecting the restored videos, we found some issues, which need further investigations for improving the restoration process.

Face Lost because of False Detection. A few times, it happened that the face was immediately found, but it turned out to be a false detection. In the next frames, where the face was not found anymore, even the restoration with the last known position did not solve the problem, because

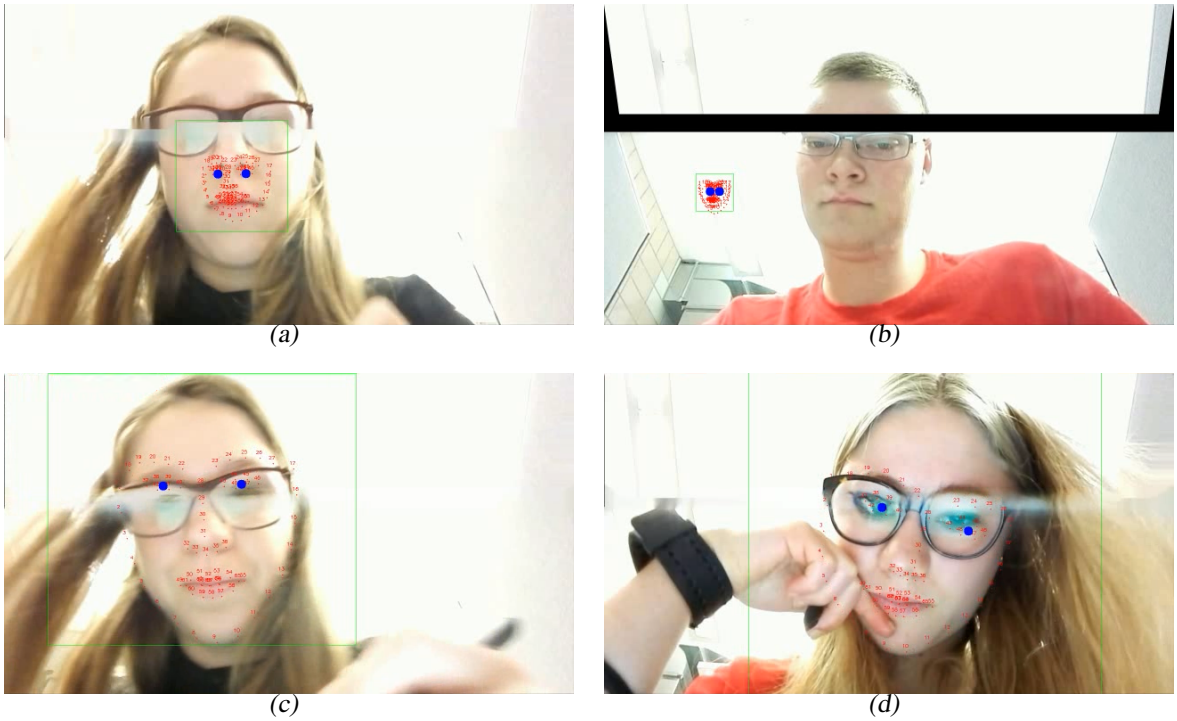


Figure 6.14.: *Known issues for our face restoration.* (a) *and* (b)*: The face was detected once wrong to be only around the nose and mouth.* (c) *and* (d)*: The eyes are missing partially or completely, and the face is not aligned as accurate, which leads to a wrong inpainting model input.*

the restored region is not relevant (please see Figure 6.14(a)). To reduce this issue, a sanity check needs to be implemented, which verifies the changes in the detected faces between two consecutive frames is reasonable.

Eyes. The eyes were the most challenging parts. As soon as one or both eyes were missing in the original frame, it became more demanding to restore the frame correctly. One issue was the possible wrong alignment by an inaccurate landmark detection (please see Figure 6.14(c)). We argue that this can be improved by a slight random rotation and random shift in the training data to have a model that is less limited with the eye positions. Another issue is the static padding, which might not always fit the actual missing part. This can also lead to some incorrectly restored images as the face will be more variable in height for different frames. We reason this should be a lesser problem for detecting the action units, as they are not based on the face height itself.

Illumination. For some participants, the lighting condition was better than for others. This was also dependent on the position of the participant and therefore, was dynamically changed during the entire video. For some frames, the border of the face has only little contrast, and the face detection does not align properly on the correct contour. As a result, the found landmarks could represent a too broad face. In most cases, this was not a problem for the action units, as long as other landmarks for the mouth and eyes were correctly aligned. Nevertheless, sometimes this misalignment gave wrong calculations for calculations based on the head position. One approach could be to normalize each frame based on the found face, similar to the brightness factor of Affectiva.

7

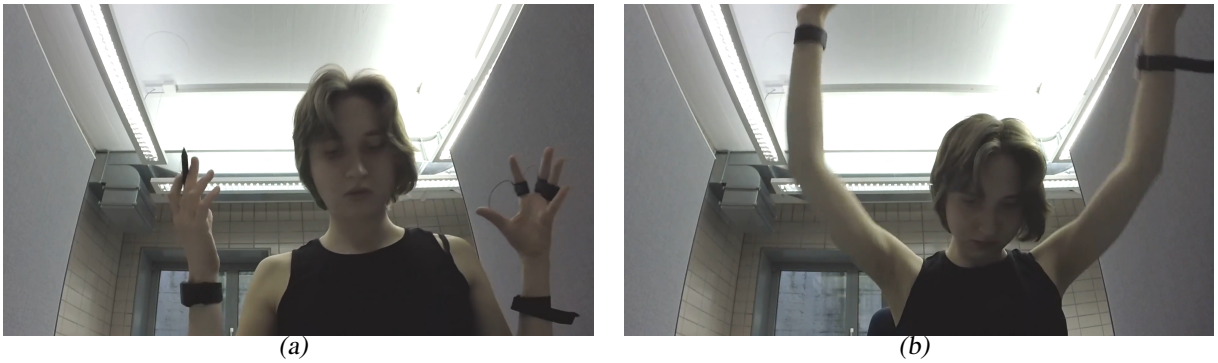
Conclusion

One goal of this thesis was to classify the affective state of participants during an experiment, which was subdivided into looking at IAPS pictures and solving math tasks. The setup of the experiment was similar to students solving their exercises in school or at home on a tablet. Learning software could be improved with personally adjusting the level of difficulty based on the student solving the tasks.

The affective state can be subdivided into different affective dimensions. Two of these are the valence and arousal, which have been used to let participants rate their actual state. Based on the ratings given by the participants, we tried to classify the affective state by analyzing different features extracted only from the videos.

We achieved a reasonable performance above the random level with the stratified shuffle cross-validation. For IAPS we classified valence with an accuracy of 0.792 (std = 0.030) and arousal with 0.702 (std = 0.028). However, with an accuracy of 0.814 (std = 0.016) for valence and 0.908 (std = 0.018) for arousal, it was easier to classify math tasks. We argue that this is because the participant was more involved in the experiment for solving math tasks and was, therefore, able to rate it more accurately. Another fact could be that participants were already used to the rating system in the second half of the experiment, which could influence the rating behavior slightly. Furthermore, because participants were in an active state for solving math tasks, they tend to show more variations and stronger facial expressions and movement. In the passive state when looking at pictures, less change was visually seen.

Better results were achieved when data from participants is used in training and predicting step. This can be observed by using LOGO cross-validation, where we did not train anything from the participant we tried to predict. The accuracy dropped for IAPS to 0.727 (std = 0.134) on valence and 0.643 (std = 0.643) on arousal. However, math tasks suffered more and are about 0.687 (std = 0.196) on valence and 0.596 (std = 0.315) on arousal. We argue this is because emotions are universal but still have individual variety, which might be hard to estimate when we know nothing about a person. This effect was investigated by Elfenbein and Ambady [EA03].



(a) (b)

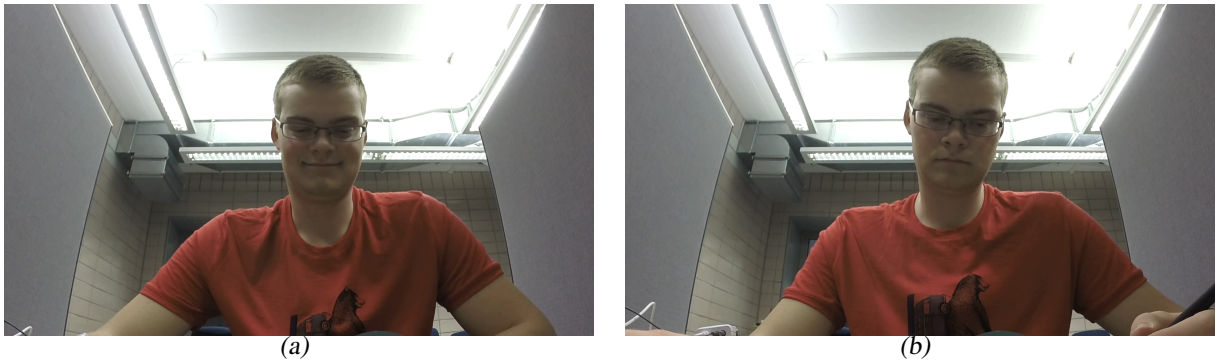
Figure 7.1.: *Little facial expression visible.* (a) *The facial expression remained unchanged during the video sequence and her body language communicates astonishment (rating: valence = 1, arousal = 5)* (b) *The body shows a winner pose by putting the hands up, whereas the facial expressions did not change (rating: valence = 6, arousal = 7).*

The higher variance in the performance suggests that some participants were similar to others whereas some participants seem to be more individual in showing their facial expressions.

Additionally, performance is based on the window for feature extraction. A longer duration above 5 s did only decrease performance. We claim this is because the emotions triggered during this experiment are expected to be short. Furthermore, we argue this is similar to solving exercises for school at home. There still might be a chance that some students are frustrated for a longer period while solving tasks and continuously failing. However, it should always be valid to predict the negative affective state by only considering short sequences in this case as well. Not only the duration of the window has an impact on the performance, but also the shift from where to extract features. Fascinatingly, for IAPS and math tasks, we found the best shift to be about 1 s after the picture was shown and 1 s after the math task was finished. We reason that for IAPS the participants had to compare the seen image with personal experiences to rate it, which resulted in facial expression changes. On the other hand, for math tasks, we argue that the success message was the event to trigger the affective state changes, which can be seen as a reaction with facial expression changes.

While finding the optimal window for feature extraction, we tried to find some reasons for wrongly classified samples. During the data analysis, we already discovered that there are some differences between the participants and that there might be some unexplainable variance. Even though the rating with the SAM can be intuitively used, there can still be personal differences in ratings. By visually analyzing wrongly classified samples, we found different problem cases.

Little Facial Expression. A few participants did not show a lot of variety in the facial expressions. For some, it was even hard to distinguish if the video was playing while looking at a picture. Having almost no visual differences in the facial expression between a highly aroused negative and highly aroused positive sample makes it questionable if the participants rated honestly. Some reported they were tired, which can also influence the facial expressions. Introducing medium valence and medium arousal as new classes might at least be able to capture these as neutral. However, this would still be wrong concerning the rating of the participant. Math task samples did suffer less of this phenomenon as the participants were actively writing for solving math tasks, but we still found some participants who had minimal facial expression. We



(a) (b)

Figure 7.2.: *Misinterpretations in facial expressions.* (a) *Even though the participant was smiling, he rated the seen picture as low valence (2) and high arousal (9).* (b) *He rated the seen picture as high valence (7) and medium arousal (6), although he does not look as happy.*

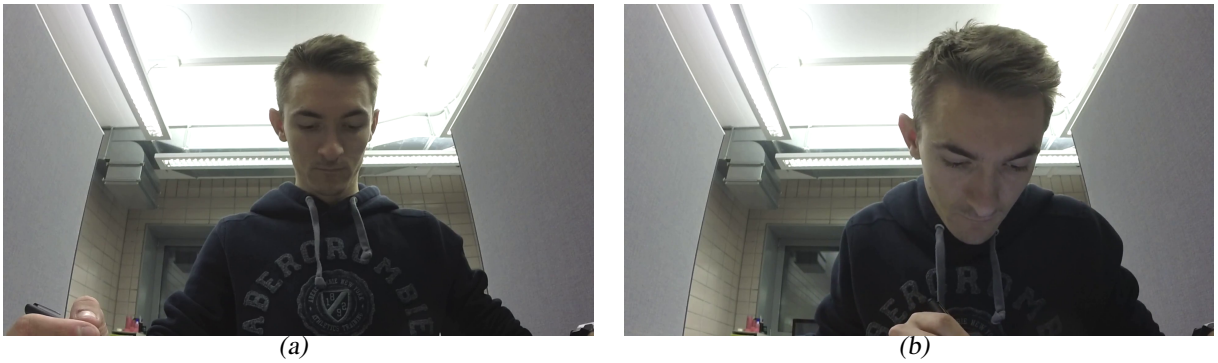
even found one example where no change in the face was shown, but she did a winner pose by raising their arms as the solution was correct for an overchallenge task (please see Figure 7.1).

Inconsistent Rating. Comparing samples with the same rating of the participant showed sometimes extreme visual differences. For example, we found a participant who rated some pictures as high valence and high arousal, but for some, there was almost no change in the facial expression, and for others, a happy face is visible with a huge smile. We argue that to some extent such inconsistency might be that the participants had first to get used to the rating scales. However, the discrepancy was not expected to be this high. Another small factor might be the individual order for the participant, that one picture still affects the rating for the following images. Similarly, this can be applied to the math tasks where a repetitive math task can be a welcoming change from the overchallenge math tasks.

Misinterpretations. Being filmed by cameras or observed by other people can influence our behavior. The cameras, the GoPro as well the mirror for the front camera, were clearly visible and might have influenced the participants. We found one participant who was happily smiling while looking at a picture but rated it as low valence and high arousal, which is the opposite. Because the picture was rated as high arousal and high arousal from the studies of IAPS, we argue that for this sample the participant just rated the image how it would be expected by society (please see Figure 7.2).

The same effects can be seen during math tasks. A few participants rated the math task as low valence but were smiling. However, the smile itself did not look as a happy smile but rather as a contempt smile (i.e., they should have known the answer but gave the wrong solution). Visually, there was not a lot of difference, but it was more the whole appearance of the participant leading to this conclusion. Therefore, as soon as an algorithm associates the smile with a high valence, it might be hard with only subtle variations to find the correct meaning of the smile. That it is hard to detect feigned from genuine smiles was already investigated by Krumhuber and Manstead [KM09].

Noise in Extracted Features. Not only the lighting conditions were challenging and resulted in sometimes noisy signals in the extracted features, but also the head position of the participant (please see Figure 7.3). Even though that OpenFace calculates the head position and rotation,



(a) (b)

Figure 7.3.: *Typical pose during the experiment.* (a) *For watching pictures, most stayed calm and upright, whereas* (b) *for solving math tasks they bent over the tablet.*

the action units seem to be not normalized against the rotation.

During the experiment, participants were filmed with a GoPro and the front camera of the tablet. Unfortunately, not many frames could be used from the front camera because the face was not detected correctly, even though most of the face was visible. We investigated several options to create a video pipeline to restore these videos with the main focus to improve the accuracy of face detection software. In the end, we had a pipeline which corrected the mirror and restored the face with an inpainting network.

After applying our pipeline to the front camera videos of each participant, we improved the confidence from 0.630 (std = 0.436) to 0.927 (std = 0.179). We further demonstrated, that in many situations, the face detection worked more accurate after the restoration. However, there are a few limitations for inpainting the face. The most challenging situation is when one or both eyes are covered. One reason for this is that each picture in the training data for the inpainting model was precisely aligned the same way (i.e., the eyes and mouth are always at the same position). If in our frame, one or both eyes are missing, we might not be able to align the picture as precisely as needed to restore the eyes correctly. The padding was chosen to be static to reduce the complexity of the video processing pipeline. Because the padding depends on the distance of the person in the video, a reliable measure is needed. Most measurements for the distance are using the gap between the two eyes. Because we had some noise in the eyes, we did not want to rely on this to reduce the artifacts.

After we have reconstructed all front camera videos of each participant, we applied the machine learning model on features extracted from these videos. Remarkably, the performance is extremely close to the performance with GoPro videos. Though it is good to have the same accuracy with only the built-in camera and our adjustments, it brings up some doubt about the honest ratings of the participants. But we also assume that some do not show a lot of changes in the facial expressions for similar stimuli from the experiment, and they more likely rated logical instead of emotional. Another reason could be that the cameras influenced the behavior in the visual field of the participants.

8

Future Work

The accuracy for extracting action units highly depends on the head position of the participant. As soon as the head is tilt, the values for the action units are different, even for the same facial expression. Future work could investigate the possibilities to normalize these action units to have a more concise result for the same facial expressions filmed from different angles.

For extracting the fidgeting index, a new video was created with the binary changes of two consecutive frames. A further idea can be to use this basis to create motion history images and find discrete patterns of movements. This could detect when somebody is shaking the head or open the hand (i.e., for being surprised). This leads to another possible investigation, where more features from the body language are considered.

Lighting conditions can be problematic for the front camera. In many cases, some lamps can be on the ceiling and behind the person, which makes it more challenging for extracting the facial features without noise. Further studies can investigate on adjusting the frames to have similar conditions independently of the environment.

The second goal was to improve the face detection for the front camera with a small mirror attached to the tablet and a framework to restore the video properly. While we have demonstrated a first working version, which already improved the face detection rate significantly, there are some improvements possible.

For further use, there is the need to apply the video pipeline directly on the webcam stream. To do so, the pipeline needs to improve in computational performance, as the first-time restoration a 70 minute video (25 fps, resolution 1920x1080) took about 20 hours. One bottleneck is that face detection is applied twice on most frames. Another performance improvement could be to write the framework in C++.

Face detection works reliably under ideal circumstances. However, as soon as the face is partially covered, current classifiers are failing to detect the face accurately. Hard transitions in the pictures are making this more challenging. Another issue is that the entire face is always

expected and all the landmarks need to be fitted on the picture. The research could try to find a method to accurately detect the visible part of the face and ignoring other landmarks.

The restoration worked reliable, except when one or both eyes were covered. As a result, the face alignment is not as accurate for the trained inpainting model, which expects the eyes and mouth in the same position. To investigate on a model which is more flexible and still reasonable for training could be explored. We defined the padding between the two regions to be static. This can lead to restoration issues when the mirror is not fixed and changes the angle. We have seen that, based on the distance and the head pose of the person, the padding should also be adjusted. However, this is only reliable when the eyes are both visible. It could be investigated, how to calculate the distance trustworthy when one eye is missing.

A

Appendix

A.1. Action Units used to show an emotion

Emotion	*Imotions*	*P. Ekman*
Joy	6, 12	1, 6, 12, 14
Sadness	1, 4, 15	1, 4, 15, 23
Anger	4, 5, 7, 23	2, 4, 7, 9, 10, 20, 26
Fear	1, 2, 4, 5, 7, 20, 26	1, 2, 4, 5, 15, 20, 26
Disgust	9, 15, 16	2, 4, 9, 15, 17
Surprise	1, 2, 5, 26	1, 2, 5, 15, 16, 20, 26

Table A.1.: *Emotions described with the FACS.*

A.2. Action Units supported by OpenFace and Affectiva

Id	*Name*	*OpenFace*	*Affectiva*
1	Inner Brow Raiser	X	X
2	Outer Brow Raiser	X	X
4	Brow Lowerer	X	
5	Upper Lid Raiser	X	
6	Check Raiser	X	X
7	Lid Tightener	X	X
9	Nose Wrinkler	X	X
10	Upper Lid Raiser	X	X
12	Lip Corner Puller	X	X
14	Dimpler	X	X
15	Lip Corner Depressor	X	X
17	Chin Raiser	X	X
18	Lip Puckerer		X
20	Lip Stretcher	X	X
23	Lip Tightner	X	
24	Lip Pressor		X
25	Lips Part	X	
26	Jaw Drop	X	X
28	Lip Suck	X*	X
43	Eyes Closed		X
45	Blink	X	

Table A.2.: *Action-Units: The subset of action units which are supported by OpenFace and Affectiva. * Only available as presence*

Bibliography

[ASP18] Chandrakant Chandewar Ashlesha Singh and Pranav Pattarkine. Driver drowsiness alert system with effective feature extraction. *International Journal for Research in Emerging Science and Technology*, 5(4):26–31, 2018.

[Bal16] Tadas et al. Baltrušaitis. Openface: an open source facial behavior analysis toolkit. In *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1–10. IEEE, 2016.

[BL94] Margaret M Bradley and Peter J Lang. Measuring emotion: the self-assessment manikin and the semantic differential. *Journal of behavior therapy and experimental psychiatry*, 25(1):49–59, 1994.

[Cab02] Michel Cabanac. What is emotion? *Behavioural processes*, 60(2):69–83, 2002.

[Cho15] François et al. Chollet. Keras. https://github.com/fchollet/keras, 2015.

[Coh07] Jeffrey F et al. Cohn. Observer-based measurement of facial expression with the facial action coding system. *The handbook of emotion elicitation and assessment*, pages 203–221, 2007.

[Cul12] Ivan et al. Culjak. A brief introduction to opencv. In *2012 proceedings of the 35th international convention MIPRO*, pages 1725–1730. IEEE, 2012.

[DB97] James W Davis and Aaron F Bobick. The representation and recognition of action using temporal templates. In *IEEE conference on computer vision and pattern recognition*, pages 928–934, 1997.

[EA03] Hillary Anger Elfenbein and Nalini Ambady. Universals and cultural differences in recognizing emotions. *Current directions in psychological science*, 12(5):159–164, 2003.

[ea15a] Martín Abadi et al. TensorFlow: Large-scale machine learning on heterogeneous

systems, 2015. Software available from tensorflow.org.

[ea15b] Ziwei Liu et al. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[ea17] Ching-Wei Tseng et al. General deep image completion with lightweight conditional generative adversarial networks. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2017.

[Ekm92] Paul Ekman. Are there basic emotions? *Psychological Review, Vol 99(3)*, 1992.

[Ekm09] Paul Ekman. *Telling lies: Clues to deceit in the marketplace, politics, and marriage (revised edition)*. WW Norton & Company, 2009.

[Feb14] Ika et al. Febrilia. The effects of positive and negative mood on university students' learning and academic performance: Evidence from indonesia. *Factors Affecting English Language Teaching and Learning*, 2014.

[Haa09] Martijn et al. Haak. Detecting stress using eye blinks and brain activity from eeg signals. *Proceeding of the 1st driver car interaction and interface (DCII 2008)*, pages 35–60, 2009.

[HS81] Berthold KP Horn and Brian G Schunck. Determining optical flow. *Artificial intelligence*, 17(1-3):185–203, 1981.

[HZ09] Shuyan Hu and Gangtie Zheng. Driver drowsiness detection with eyelid related parameters by support vector machine. *Expert Systems with Applications*, 36(4):7651–7658, 2009.

[Jon14] Eric et al. Jones. SciPy: Open source scientific tools for Python, 2014.

[Kin09] Davis E. King. Dlib-ml: A machine learning toolkit. *Journal of Machine Learning Research*, 10:1755–1758, 2009.

[KM09] Eva G Krumhuber and Antony SR Manstead. Can duchenne smiles be feigned? new evidence on felt and false smiles. *Emotion*, 9(6):807, 2009.

[Kru13] Eva G et al. Krumhuber. Effects of dynamic aspects of facial expressions: A review. *Emotion Review*, 5(1):41–46, 2013.

[Lan80] PJ Lang. Self-assessment manikin. *Gainesville, FL: The Center for Research in Psychophysiology, University of Florida*, 1980.

[Lan17] Agnieszka et al. Landowska. Limitations of emotion recognition from facial expressions in e-learning context. In *CSEDU (2)*, pages 383–389, 2017.

[LB07] P Lang and Margaret M Bradley. The international affective picture system (iaps) in the study of emotion and attention. *Handbook of emotion elicitation and assessment*, 29, 2007.

[Liu18] Guilin et al. Liu. Image inpainting for irregular holes using partial convolutions. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 85–100, 2018.

[LR18] Steven R Livingstone and Frank A Russo. The ryerson audio-visual database of

emotional speech and song (ravdess): A dynamic, multimodal set of facial and vocal expressions in north american english. *PloS one*, 13(5):e0196391, 2018.

[Mat90] David Matsumoto. Cultural similarities and differences in display rules. *Motivation and emotion*, 14(3):195–214, 1990.

[McD16] Daniel et al. McDuff. Affdex sdk: a cross-platform real-time multi-face expression recognition toolkit. In *Proceedings of the 2016 CHI conference extended abstracts on human factors in computing systems*, pages 3723–3726. ACM, 2016.

[Nav14] Rajitha et al. Navarathna. Predicting movie ratings from audience behaviors. In *IEEE Winter Conference on Applications of Computer Vision*, pages 1058–1065. IEEE, 2014.

[Osg57] Charles Egerton et al. Osgood. *The measurement of meaning*. Number 47 in 9. University of Illinois press, 1957.

[PE78] Wallace V. Friesen Paul Ekman. Facial action coding system. *Facial Action Coding System*, 24(4):56–75, 1978.

[Ped11] F. et al. Pedregosa. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[PJL08] Margaret M. Bradley & Bruce N. Cuthbert NIMH Center for the Study of Emotion & Attention University of Florida Peter J. Lang. International affective picture system, 2008. https://csea.phhp.ufl.edu/Media.html#topmedia.

[Sch64] Stanley Schachter. The interaction of cognitive and physiological determinants of emotional state. In *Advances in experimental social psychology*, volume 1, pages 49–80. Elsevier, 1964.

[Tom06] Suramya Tomar. Converting video formats with ffmpeg. *Linux Journal*, 2006(146):10, 2006.

[Uly18] Dmitry et al. Ulyanov. Deep image prior. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9446–9454, 2018.

ETH

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Declaration of originality

The signed declaration of originality is a component of every semester paper, Bachelor's thesis, Master's thesis and any other degree paper undertaken during the course of studies, including the respective electronic versions.

Lecturers may also require a declaration of originality for other written papers compiled for their courses.

I hereby confirm that I am the sole author of the written work here enclosed and that I have compiled it in my own words. Parts excepted are corrections of form and content by the supervisor.

Title of work (in block letters):

Emotion Predictor: Machine Learning Based Prediction of Emotions using Facial Features

Authored by (in block letters):

For papers written by groups the names of all authors are required.

Name(s):	**First name(s):**
Emch	Andreas

With my signature I confirm that

- I have committed none of the forms of plagiarism described in the 'Citation etiquette' information sheet.
- I have documented all methods, data and processes truthfully.
- I have not manipulated any data.
- I have mentioned all persons who were significant facilitators of the work.

I am aware that the work may be screened electronically for plagiarism.

Place, date	**Signature(s)**
Burgdorf, 17.05.2019	A. Emch

For papers written by groups the names of all authors are required. Their signatures collectively guarantee the entire content of the written paper.