A deep learning approach to face image quality assessment

Master's Thesis
MSc Business Analytics
Vrije Universiteit Amsterdam

Author:
J. Touati

Supervisors:
M.Sc. A. Gudi
Prof. Dr. S. Bhulai
Dr. E.R. Belitser

June 10, 2020

Abstract

This research presents two approaches to assess the quality of a face image. These approaches can be used to reject low-quality face images and this allows facial analysis systems to only do an analysis on face images of high quality. First, a new face quality measure is proposed, referred to as the empirical quality score, that can be used to increase the performance of a baseline facial analysis system with roughly 11%, 2%, and 4% on emotion-, gender-, and age classification, respectively. Next, a deep learning approach to face quality assessment is presented, which can be used to increase the performance of the baseline system with roughly 3% and 7% for gender- and age classification, respectively. Here, the results show that the deep learning approach does not accurately assess the quality of emotive faces. Finally, the influence of face normalization methods on the performance of the baseline facial analysis system is presented. The results show that face normalization does have a significant influence on this performance. More specifically, forcing the face to fully fit the normalized image increases the F1-Score on emotion-, gender-, and age classification with 1.35%, 0.15%, and 1.73%, respectively.

key words: Computer Vision, Deep Learning, Face Quality Assessment, Face Normalization, Face Recognition, Image Classification

Acknowledgments

This thesis is written to fulfil the final requirement for the Master Business Analytics at the Vrije Universiteit Amsterdam. Here, the student is expected to conduct a six-month research at a company that solves a business problem or gives valuable insights regarding related questions. This thesis was conducted at the company VicarVision from October 2019 till April 2020.

First, I would like to thank Tim den Uyl for giving me the opportunity to conduct this research at VicarVision. Furthermore, I would like to thank my supervisor within VicarVision, Amogh Gudi, for his valuable input and creative ideas. In addition, I would like to show my gratitude to my supervisor at the VU, Sandjai Bhulai, for his guidance and fun meetings. Finally, I want to thank Eduard Belitser for being the second reader of this thesis.

Contents

List of Figures

List of Tables

1 Introduction

Humans have always had the innate ability to recognize and distinguish between faces. Also, taking in non-verbal cues from facial expressions is an easy task for us humans. Learning this ability to extract features from faces by a computer would open up many applications, however, it is a much harder task for a computer. The first experiments related to facial analysis systems have been conducted in the 1960s, where scientists worked on using the computer to recognize human faces [5]. It was not until 1997 when scientists made the system robust enough to make identifications from less-than-perfect quality face images [6]. Later advances in deep learning algorithms would further improve the performance of these facial analysis systems [7]. In fact, deep learning approaches are until this day the state-of-the-art approach for automatic face analysis. Although face analysis has been a challenging topic in the field of computer vision, great results have been achieved on various tasks. However, most of the attention was focused on analyses based on perfect quality face images, while the most interesting applications require face analysis on lower quality face images [8]. The performance of most of the current facial analysis systems drops significantly when given stronger variations in quality of the face image [8].

VicarVision [9] is a company that develops facial analysis software that automatically analyzes facial expressions and other characteristics, such as age, gender, presence of glasses/beard, etc. The software uses a deep learning algorithm that requires image input such as a frame of a camera or webcam and analyzes the facial characteristics. There is a wide range of applications for this software which includes the usage by more than 900 institutes around the world as well as citations in 1000+ scientific publications. One of the applications is to use the facial analysis system to track, monitor and monetize the behavior of customers in a retail store.

Fig. 1: Example application: facial analytics in retail.

Multiple helpful statistics can be measured, such as demographics, emotions, and popular areas within the store. This gives a retailer insights into whether he is targeting the right audience or whether the layout of the store needs to be changed. Another application is to measure appreciation or acceptance, which can be incorporated in market research. Think of analyzing people while they are watching a new commercial and observing whether the desired response is encountered.

A difference between the applications of the software is the setting in which a person is observed. When a customer enters a retail store, he or she is not aware of the camera, and therefore is not focused on the camera as can be seen in Figure 1. In the other setting, people are sitting in front of a screen and are aware of the camera. Therefore, the latter is a much more constrained setting. Capturing face images naturally without controlling the environment or so-called in-the-wild, will result in a variety of image quality conditions, such as illumination, focus, or sharpness. Other factors are more related to the properties of the face itself like pose, presence of occlusions (e.g., glasses), and different expressions. All these factors influence the quality of the face image, where quality is to be understood as an estimator of the performance of a facial analysis system. In other words, the quality of a face image is correlated with the expected performance on face classification or recognition tasks. Especially, in-the-wild face images are known to affect this performance [8].

A face quality assessment can be used as a preprocessing step, where the system may decide to reject low-quality images. Only allowing an analysis on face images of good quality will result in a higher performance, since the system is more prone to making errors on low-quality images [10]. In addition, the computation time of a facial analysis system can be greatly reduced by only analyzing the face image of the best quality. If we assume that there are 5 seconds of footage of a person, which was taken at 25 frames per second, then there are 125 face images. Analyzing only the best quality face image can reduce the computation time by a factor 125 in this example. Another application where a face quality assessment could be used, is for Negative Identification Systems, such as security checks at banks or airports where suspects try to provide a low-quality face image to evade recognition [11]. In such cases, the system should flag such users and access should be provided only after providing a perfectly aligned face image.

In order to analyze a face image, it is desired to only analyze the face features and exclude any background information. Such preprocessing steps are referred to as Face Normalization and aim to transform an input face image to a more general form, which is done before the face analysis. Face normalization closely relates to face quality assessment as it tries to reduce variations in face images, which contributes to the quality of the face. Also, facial analysis systems perform their analysis on the normalized version of an input face image.

Both face quality assessment and face normalization have been investigated in several researches, however, predominantly for face recognition purposes. The impact on other facial features, such as the analysis of emotion, gender, or age has been given far less attention. Furthermore, multiple methods for both techniques are proposed in the literature, but a general method is still missing. From this, the importance rises to investigate methods for both face quality assessment and face normalization in order to see how they impact the performance of a facial analysis system and to compare them.

Therefore, the research question reads:

How can one assess face image quality in order to predict the suitability of a face image for face analysis purposes?

Furthermore, how does face normalization influence the performance of facial analysis systems?

In order to answer the above research questions, this thesis follows a common approach. The first step is to better understand the problem at hand, which is done by reviewing related problems and relevant literature. Section 2 provides an overview of relevant literature in the field of face normalization and face quality assessment. The second step is to collect data that suits the needs of both problems. Section 3 discusses the data that is used for both problems. Next, the methods that are used for face normalization and for face quality assessment are described. Section 4 provides an overview of all methods that are used. The experiments to test the methods are described in Section 5. Also, the evaluation procedures will be discussed here. The results of the conducted experiments will be shown and discussed in Section 6. Finally, conclusions are drawn and future work is discussed in Section 7.

2 Literature

This section discusses some related problems and relevant literature that is available in the fields of face normalization and face quality assessment (Section 2.1). This is done to get an understanding of where previous research has brought both fields and how this research can add new knowledge to both fields. After this, geometric transformations for images are discussed as these transformations are often used for image preprocessing (Section 2.2). Then, some preliminary information is provided about Neural Networks, Convolutional Neural Networks, and how these networks can be trained (Section 2.3). Finally, some evaluation measures for classification will be discussed (Section 2.4).

2.1 Related Work

2.1.1 Face Normalization

Many face normalization techniques have been proposed in the literature for face recognition tasks and even some for facial expression analysis [12]. Some studies also propose face frontalization in their normalization technique for variations in pose of the face [13], where frontalization refers to the process of synthesizing frontal facing views. Yin et al. (2017) propose a 3D Morphable Model (3DMM) conditioned Face Frontalization Generative Adversarial Network (GAN), termed as FF-GAN, to generate neutral head pose face images. FF-GAN shows promising recognition results for large pose variations of the face, however, the expression information is lost in the generated face. Zhu et al. (2014) propose a deep learning framework that can recover the frontal view of face images. They focus on pose variations, since these variations are a key challenge in many face-related applications [14]. Haghighat et al. (2016) also stress that pose variations are one of the biggest factors that cause inaccurate predictions of facial analysis systems. Although multiple studies claim these variations in pose have a big impact, none of them quantify this impact to see its extent. Hence, quantifying this will give some relevant insights.

The main problem, however, with most proposed frameworks for face normalization is that they are computationally expensive as a preprocessing step. For many real-time applications, such as facial analytics for retailers, this is not possible, because the face analysis itself is already computationally expensive. Therefore, the focus will lie on computationally cheaper face normalization methods that are applicable to real-time applications.

An efficient face location normalization algorithm based on eye locations is proposed by Li et al. (2006). This algorithm uses eye locations to adjust for in-plane rotation as can be seen in Figure 2. The experiments showed that the proposed normalization algorithm is efficient for real-time applications [15]. Figure 3 shows the types of rotation in the pose of a face, where the roll angle corresponds to in-plane rotation. The pitch and yaw angle are both referred to as the out-of-plane rotation and can not be adjusted by the proposed algorithm.

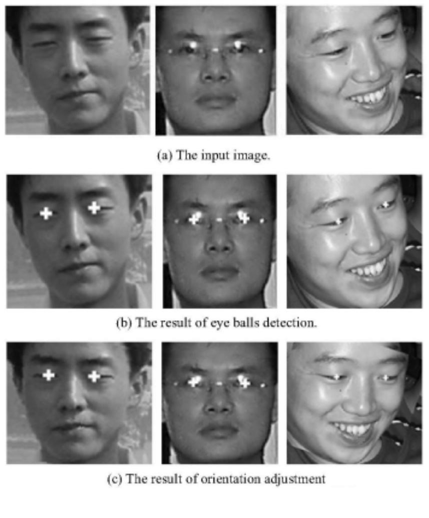

(a) The input image.
(b) The result of eye balls detection.
(c) The result of orientation adjustment

Fig. 2: The face location normalization algorithm proposed by Li et al. (2006).

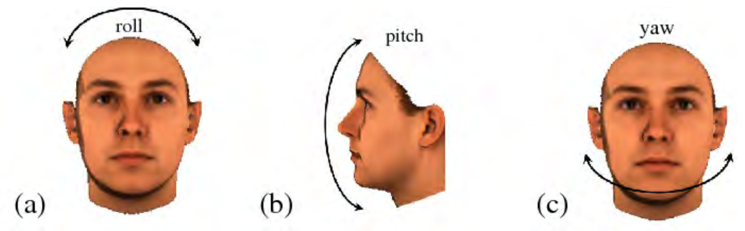


Fig. 3: Rotation types: Roll (a), pitch (b), and yaw (c) angles.

A similar algorithm is proposed by Gudi et al. (2014) where the eyes are warped onto fixed points in a new image [16][17][18]. The other parts of the face are included in the transformation. This removes the in-plane rotation and resizes the face simultaneously. Both methods use the eyes as reference points for the transformed image and enforce a zero slope on the line connecting the two eyes in the face. These types of image transformations will be discussed in more detail in Section 2.2. Other facial landmarks could also be used as reference points, however both Li et al. (2006) and Gudi et al. (2014) do not consider this. Using more or different facial landmarks in this process could make the method more robust, since face sizes can differ a lot for distinct persons. Figure 4 shows how a long shape of the face can cause this algorithm to not fully capture the jaw and the chin, while the face features of the shorter face are fully captured:

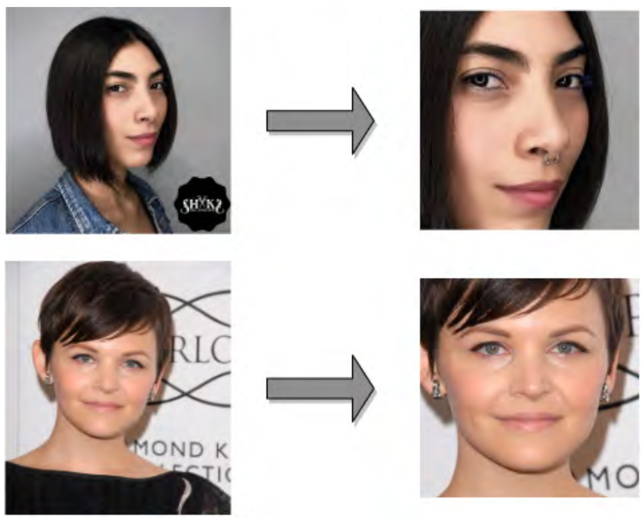


Fig. 4: Two examples of the algorithm proposed by Gudi et al. (2014)

Also using other landmarks as reference points, such as the chin or the nose, could solve this by additionally scaling the face over the y-axis to fully fit the new image. However, the original shape of the face will be lost, which is especially important for face recognition purposes [19]. For other objectives, such as analyzing facial expressions, gender, or age, this might not be important, since we are not trying to identify a person. Hence, experimenting with these types of transformations and the facial landmarks will give answers to these matters.

2.1.2 Face Quality Assessment

In general, it is difficult to explicitly define and quantify face image quality (i.e., an estimator of the performance of a facial analysis system). There are no explicit labels of face image quality, unlike labels of facial identity or facial expression. Previous research shows that there are mainly three approaches for generating a quality measure given a face image. The first approach is to empirically use face image properties, such as pose, illumination, or sharpness, in order to quantify the quality of the face image [20][21]. The second approach is to measure the similarity to a reference, or 'ideal' face image (typically frontal pose, uniform illumination, neutral expression) and use this similarity as a quality measure [10][22]. Finally, the most simple approach is to use human annotations of the perceived image quality of a face image.

The first approach can be applied by computing a combination of empirical features on the face image. The issue with this approach is that it is an ad-hoc approach, and therefore has not achieved much success [11][23]. The second approach requires a reference image of a person and this reference image is mostly not available in real-time applications. As a result, it is also necessary to train a deep learning algorithm after generating the ground truth quality labels of the face images. This way, a predictive model is created that can be used in applications where a reference image of a person is not available, such as facial analytics in retail. The third approach suffers from the fact that the human perception of face quality can be subjective, and therefore not be indicative of the performance of a facial analysis system [24][25]. In other words, the system might consider a face image to be of good quality, while a human may think it is not. In addition, human annotations are time-consuming and some rules should be made in order to give the face images an actual quality score.

In this work, the first approach will be considered, further referred to as the **empirical quality score**. A successful empirical quality score is lacking, therefore investigating new methods is relevant. Also, the second approach will be considered, further referred to as the **reference quality score**. A reference quality score has been proposed several times [10][22][11], however, to the best of my knowledge, they all focus on face recognition. None of them evaluate face image quality with the aim of predicting facial expressions, age, gender, or other characteristics. Therefore, it is very relevant to investigate whether the proposed methods are also effective for different purposes. This also holds for the empirical quality score. Finally, the third approach will not be considered, since it seems the least promising. Also, due to the six months time horizon of this research and the third approach being the most time-consuming, it will not be considered.

Multiple empirical face quality factors are discussed by Gao et al. (2007) where they state that a face image obtained by a camera is usually imperfect, since it can contain defects caused by poor illumination, improper face positioning, and imperfection of the camera [20]. The factors mentioned can be categorized into four aspects:

- The environment in which the camera is used.
- The type of camera.
- Facial conditions of the person.
- Camera positioning with respect to the person.

This research focuses on the unconstrained setting with facial analytics in retail as an example. Therefore, these categories have to be considered for that setting. Gao et al. (2007) propose several measures that focus on face symmetry, pose, illumination, and sharpness, as these are the main factors that influence face quality. These factors are also recurring in papers that focus on face quality in the unconstrained setting [8][10][22], hence they will be used to design the empirical quality score.

The deep learning framework proposed Luo et al. (2014) recovers a frontal view of a face image [14]. In this framework they select a frontal face image as a representative for each identity and then learn a mapping that transforms the face image in an arbitrary view to the frontal view. They select the representative frontal face image by combining the symmetry and rank of the face image. A very symmetrical face indicates a frontal pose of the face and can be an indicator of the absence of occlusions in the face, such as hair or hands. In addition, high rank indicates a high resolution of the image, which is qualitatively shown in their research. Although symmetry seems like a good option for measuring frontal pose, a head pose estimation network may be a better option, since an estimate of both yaw and pitch can be predicted. Furthermore, Bansal et al. (2016) show that the Laplacian operator can be used for detecting the blurriness or sharpness of an image. This is very useful for applications like facial analytics in retail, since people are walking through a store and often cause motion blur as a result. Within VicarVision, this is a common cause for wrong classifications of the facial analysis system. Next to a sharpness measure, it is also shown that the Laplacian operator can be used to detect over/under-exposed images, which is an indicator of illumination quality [26]. Experiments with the mentioned measures and combinations of them will show if they are relevant estimators of the performance of a facial analysis system.

Agarwal (2018) proposes an end-to-end system for automatic facial quality assessment [10]. The work is done in two stages:

1. Generation of the ground truth quality scores
2. Training a convolutional neural network in a supervised manner to predict the quality score.

In the first stage, the ground truth quality scores are generated for each subject in the dataset by comparing a 'gallery' face image of best quality with 'probe' face images of lower quality. Agarwal (2018) uses the Labeled Faces in the Wild dataset [27], since most of the identities in this dataset have multiple images, which is necessary for this approach. Next, Google's FaceNet [28] is used to compare the gallery images with the probe images by creating embeddings of the faces and computing the Euclidean distance between them. FaceNet learns a mapping from a face image into an Euclidean space. The idea behind FaceNet is that very similar faces should lie closer and dissimilar faces should be far away.

Agarwal (2018) is not clear about how he picks the best quality faces or gallery images, while he seems to use the same approach as Best-Rowden et al. (2017), where the gallery image is manually selected [11]. As discussed before, human perception of quality may not be indicative of the performance of a facial analysis system [24][25]. Hence, another approach for picking the gallery images is preferred. A similar end-to-end system for automatic facial quality assessment is proposed by Hernandez-Ortega et al. (2019). Here, the gallery images are selected by using a third party software to obtain automatic ICAO compliance scores [22]. This is a commercial system that is mainly used for checking the compliance of passport face images and they do not provide information on how they determine this ICAO score. Using an empirical face quality score for selecting the gallery faces seems like a better approach, since empirical factors, such as pose, sharpness, or illumination are indicative of the performance of a facial analysis system. Also, generating an empirical face quality score is already considered in this research, so also using it for this purpose makes sense.

Once the ground truth quality labels are generated, they can be used in a supervised manner to train a deep learning algorithm. Agarwal (2018) does this by using a customized FaceNet model based on Inception v3 architecture with the quality label as target. Hernandez-Ortega et al. (2019) use the VGGFace2 model [29] based on the ResNet-50 architecture in this same setting. It is not clear which network learns the target better, so experimenting with both networks will give more insights into this.

2.2 Geometric Transformations for Images

Geometric transformations are essential for image processing and are also widely used for this purpose. Everyone who has ever made a presentation in Microsoft PowerPoint will most likely have used these types of transformations by simply rescaling an image to fit the slide. Another interesting application is in training deep neural networks, where these networks require vast amounts of data. In almost all cases, these networks benefit from better generalization ability as the amount of training data increases. Randomly applying geometric transformations on existing image data can be used to generate more and new data (data augmentation). Figure 5 shows the basic set of 2-dimensional transformations.

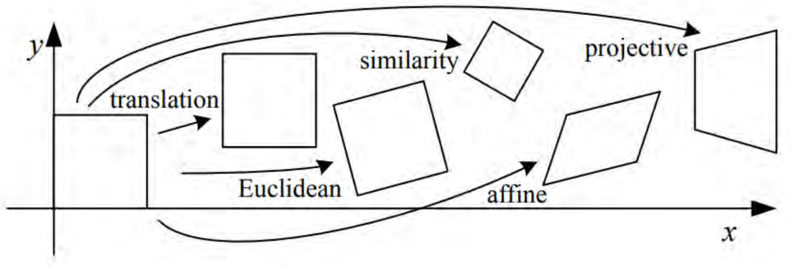


Fig. 5: Basic set of 2D geometric transformations. Source: [1]

A geometric transformation is basically a vector function T that maps the pixel (x, y) to a new position (x', y'):

$$x' = T_x(x, y), \qquad y' = T_y(x, y) \tag{1}$$

The simplest vector function achieves a 2-dimensional **translation** of the image and can be written as:

$$x' = x + t_x, \qquad y' = y + t_y \tag{2}$$

where the pair $(t_x,\, t_y)$ is called the translation vector. Using homogeneous coordinates, this can be written in matrix form:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{3}$$

When you want to rotate the image, a clockwise **rotation** transformation can be used:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{4}$$

where θ is the desired rotation angle. The **Euclidean** transformation uses translation and rotation. A **similarity** transformation also includes scaling:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} s & 0 & 0 \\ 0 & s & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{5}$$

Here, s is an arbitrary scale factor. A similarity transformation preserves the angles in the image, and therefore the original shape, as can be seen in Figure 5. The algorithm proposed by Gudi et al. (2014) uses this transformation, since it keeps the original aspect ratio. As mentioned in Section 2.1.1, an additional scaling parameter is needed to fully capture the face features in Figure 4. An additional scaling parameter can be defined for an **affine** transformation, which also allows for shearing the image. The scale and shear transformations can be expressed as follows.

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} sc_x & 0 & 0 \\ 0 & sc_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}, \quad \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & sh_x & 0 \\ sh_y & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{6}$$

Here, the pairs (sc_x, sc_y) and (sh_x, sh_y) are the scaling and the shearing parameters, respectively. To summarize, the affine transformation is a combination of **translation**, **rotation**, **scaling** (with or without preserving the original aspect ratio), and **shearing**. As stated before, these extra degrees of freedom in the affine transformation might solve the problem depicted in Figure 4.

2.3 Deep Learning

Deep learning is part of a family of machine learning methods based on artificial neural networks. To get a better understanding of the core concepts of deep learning, the basics of neural networks will be discussed (Section 2.3.1). Next, convolutional neural networks will be discussed, as these are extensively used throughout this research and generally speaking in computer vision (Section 2.3.2). Finally, the training process of these algorithms is described (Section 2.3.3).

2.3.1 Neural Networks

Neural networks are built around the ideas of Rosenblatt (1958), who designed the Perceptron algorithm [30]. A Perceptron can be seen as the simplest form of the neural network and is used to linearly separate data points in order to classify them. The Perceptron is depicted in Figure 6.

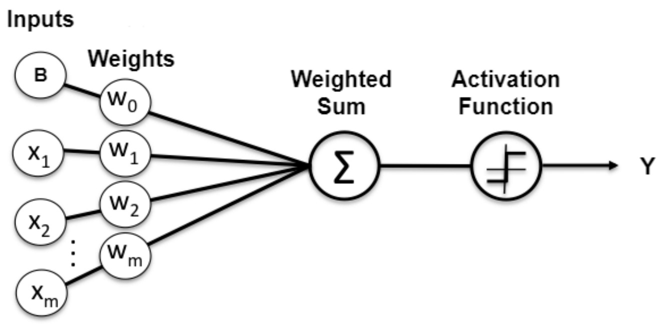


Fig. 6: Schematic representation of Rosenblatt's Perceptron.

Figure 6 shows that the Perceptron consists of a bias B, input signals X_i with $i \in \{1, 2, .., m\}$, weights W_j with $j \in \{0, 1, .., m\}$ and an activation function. The target variable y is computed by the following equations:

$$z = \sum_{i=1}^{m} x_i w_i + b w_0 \tag{7a}$$

$$y = H(z) \tag{7b}$$

$$H = \begin{cases} 1 & \text{if } z \geq 0 \\ 0 & \text{if } z < 0 \end{cases} \tag{7c}$$

First, the weighted sum of the input signals and the bias is computed (Equation 7a). Next, the weighted sum is activated by the Heaviside step function (Equation 7b), which is defined in Equation 7c. This process is referred to as Forward Propagation and is basically the computation of the output node(s). The Heaviside step function ensures that the output becomes binary, which makes the Perceptron able to classify data points. The problem, however, with the Perceptron algorithm is that it is not able to model non-linear data. Simple problems like the XOR-problem could not be solved. Consequently, a slightly more complex version of the Perceptron was designed in order to model non-linear data. Figure 7 shows this version.

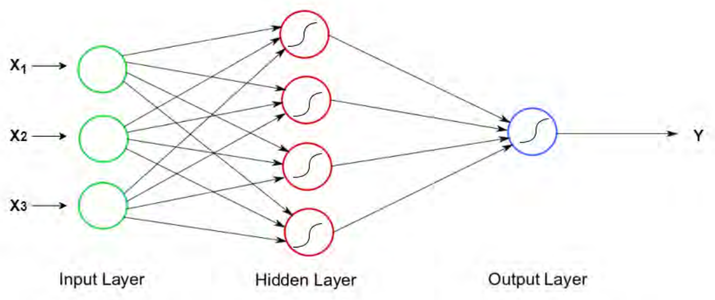


Fig. 7: The basic neural network structure (weights and biases are left out)

The addition of an extra layer between the input signals and the output layer of the network results in the basic neural network structure that is now widely used. This extra layer is the so-called hidden layer and allows the network to learn non-linear decision boundaries. Each neuron in the hidden layer acts as an independent Perceptron, i.e., each hidden neuron can make linear separations. The output layer can aggregate these independent linear separations into non-linear separations. Hence, the neural network can learn non-linear decision boundaries, such as the boundary in Figure 8.

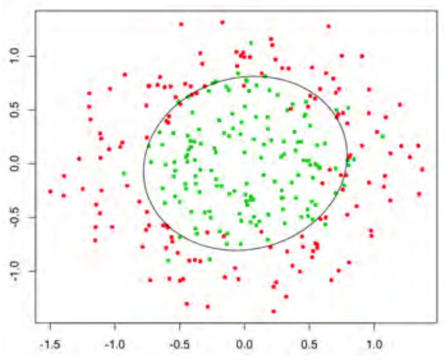


Fig. 8: A non-linear decision boundary.

Deep Neural Networks

A neural network is considered to be 'deep' when it consists of multiple hidden layers between the input and output layers [31]. The structure of a deep neural network is shown in Figure 9

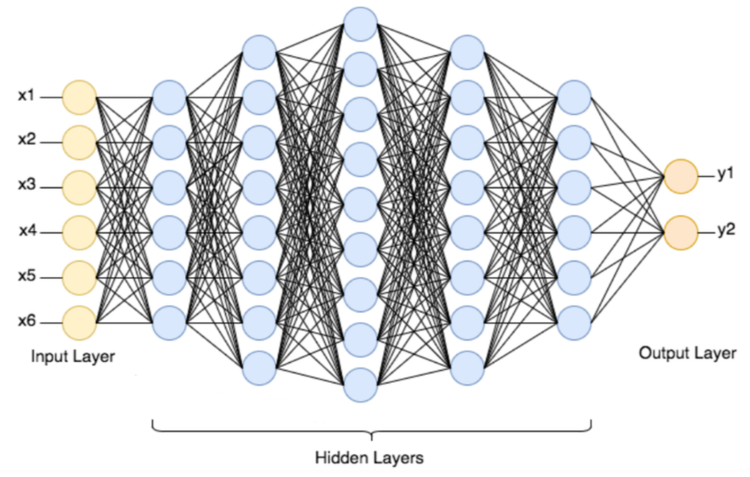


Fig. 9: The structure of a deep neural network (weights and biases are left out).

Increasing the complexity by adding more hidden layers allows deep neural networks to learn more complex non-linear relationships. A common thought is to keep adding hidden layers in order to solve more and more complex problems. The problem, however, is that this requires increasingly more computation power, which is limited to the resources at your disposal. Despite the restriction on the resources, deep neural networks have proven to be of great significance for various applications.

Activation Functions

Rosenblatt (1958) used the Heaviside step function as activation function, since it ensures a binary output. In a deep neural network, other activation functions are used. In the hidden layers, they are mainly used to transform a linear input into a non-linear output. This allows the network to model non-linear problems, as these types of problems are occurring far more in the real world than linear problems.

The choice of the activation function depends on the desired type of output. When you are considering a regression problem, you want the output layer to have a linear activation. While a multi-class classification problem requires a Softmax activation over the output layer, since this gives you probabilities of the classes. The Softmax function is defined as:

$$S(y_i) = \frac{e^{Y_i}}{\sum_j^C e^{y_j}} \tag{8}$$

where y_i is the output of the network or the so-called logit for node i and C corresponds to the number of classes. The network in Figure 9, for example, could use a Softmax activation over the output layer when y_1 and y_2 are classes. Some other common activation functions are shown in Figure 10.

The Rectified Linear Unit function or ReLU function is a commonly used activation function over the hidden nodes for one main reason. In contrast to the Sigmoid and Hyperbolic Tangent (TanH) function, it does not suffer from the vanishing gradient problem which slows down the training of the network drastically [32]. The vanishing gradient refers to the gradient of the activation function that is approaching 0 when the inputs are growing large. The ReLU function does not suffer from this as for every input larger than 0, the gradient is 1 as can be seen in Figure 10.

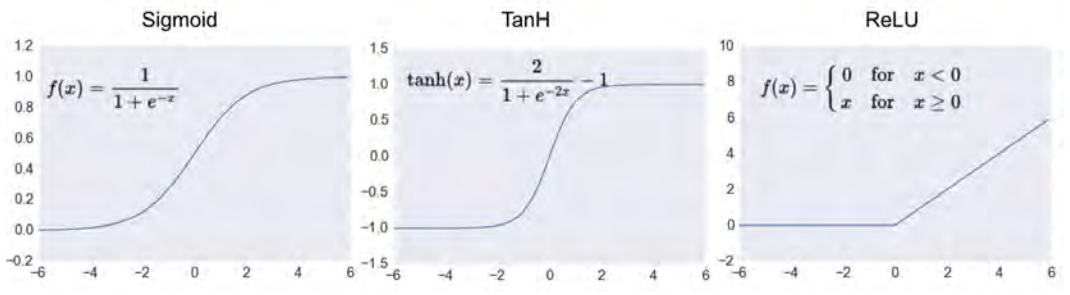


Fig. 10: Three commonly used activation functions.

Loss Functions

In order for the network to learn, a loss function has to be defined. It is a way to express how well a specific algorithm models the given data. The loss function is often defined as the difference between the target value (y) and the predicted value ($\hat{y}$). Therefore, the goal is to minimize the loss function. There are many different loss functions and similar to activation functions, the choice depends on the desired type of output.

For a regression problem, the Mean Squared Error (MSE) or the Mean Absolute Error (MAE) are often used. Both errors are defined in the following equation:

$$MSE = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}, \qquad MAE = \frac{\sum_{i=1}^{n}|y_i - \hat{y}_i|}{n} \tag{9}$$

The MSE is measured as the average of squared difference between predictions and actual observations where n is the number of observations. Due to the squaring, predictions that are far away from the actual values are penalized heavily in comparison to less deviated predictions. The MAE on the other hand, penalizes predictions that are far away equally to less deviated predictions as it measures the average of absolute differences. This makes the MAE more robust to outliers.

For a classification problem, the Cross-Entropy loss is commonly used. This loss compares the predicted probability of a class with the groundtruth label (0 or 1) for all classes. The Cross-Entropy loss is defined as:

$$CE = -\sum_{i}^{C} t_i \log(s_i) \tag{10}$$

where t_i and s_i are the ground truth label and the predicted probability for class i in C. The number of output nodes in a neural network should be equal to the number of classes C in the classification problem.

2.3.2 Convolutional Neural Networks

Convolutional neural networks (CNNs) were first introduced by LeCun et al. (1989) where they were used for recognizing handwritten zip code digits [33] provided by the U.S. Postal Service. These types of networks have proven to be superior over the standard neural network when presented image data. Tasks like image classification become extremely computationally expensive, as the image dimensions increase. Let us take an image with shape 50x50x3 and a network with one hidden layer of 1000 nodes. When you flatten the image in order to use it as input, this already corresponds to 7500 input nodes and 7.5 million(!) weight parameters, while we are only considering one hidden layer. The main reason, however, why CNNs are used is that they maintain the spatial information of the input image. In standard neural networks, you flatten the image into a vector and thereby lose information about the surrounding pixels of a certain pixel. While CNNs keep the original structure of the input by applying filters that try to capture information of the surroundings of each pixel. This is what allows CNNs to identify spatial patterns, such as edges, shapes, objects, etc. Figure 11 shows the components out of which a CNN consists.

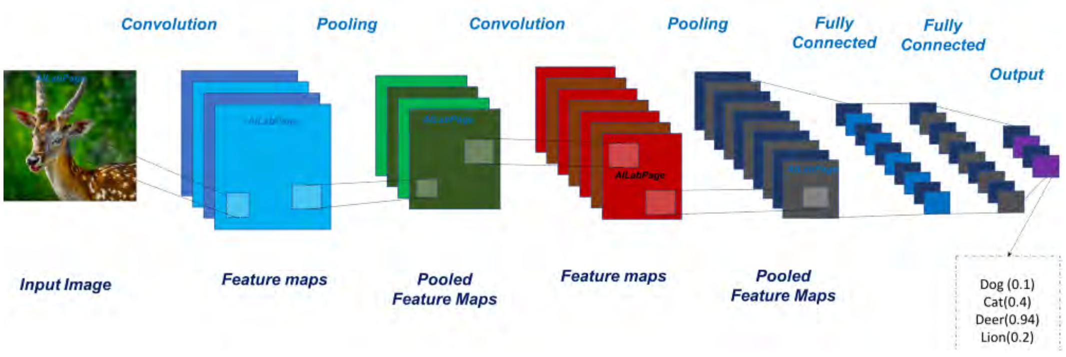


Fig. 11: The structure of a convolutional neural network.

The input layer of the CNN is an image with shape; width x height x channels, containing the pixel intensities (0-255). The channel dimension is either 1 (grayscale) or 3 (color). The main difference between a neural network and a CNN is the convolution and pooling layers. These layers act as feature extractors of the image and try to gradually extract higher level features. First, mainly edges and corners are extracted, whereafter somewhat higher level features like shapes or textures. Next, high level features are extracted that can be used to determine to which class the image belongs. These features are forwarded to the fully connected layers which are similar to a standard hidden layer where all nodes are connected (hence 'fully' connected). Finally, the output layer is used to classify the image, where the number of nodes corresponds to the number of classes.

Convolution

The convolutional layer distinguishes the CNN from the standard neural network. The idea behind the convolution is that each pixel shares information with its surrounding pixels. A filter is used to capture this information by convolving across the width and the height of the input image. The filter has usually a small size of 3x3 or 5x5 and is applied to multiple areas of the image, which significantly reduces the number of parameters. In this process, the dot products between the filter and the input positions are computed. While the filter slides as a window over the input image, a feature map is created. This process is illustrated in Figure 12. The following hyperparameters are involved in a convolution:

- Filter Size (f): The width and height of the filter expressed in the number of pixels. Common filters have size 3x3 or 5x5.

- Stride (s): When the filter is moved to a new area, the stride determines how many pixels the filter is moved. The filter normally moves from left to right, top to bottom.
- Padding (p): Padding adds extra pixels around the border of the input image. A padding of 1 adds one extra border of pixels around the image. This helps the filter to also capture information that is on the border. Furthermore, it can be used to preserve the input size in the feature map.
- Number of Filters (D): The number of filters that are applied to the input image. This number also represents the number of feature maps that are created and is often referred to as the depth dimension.

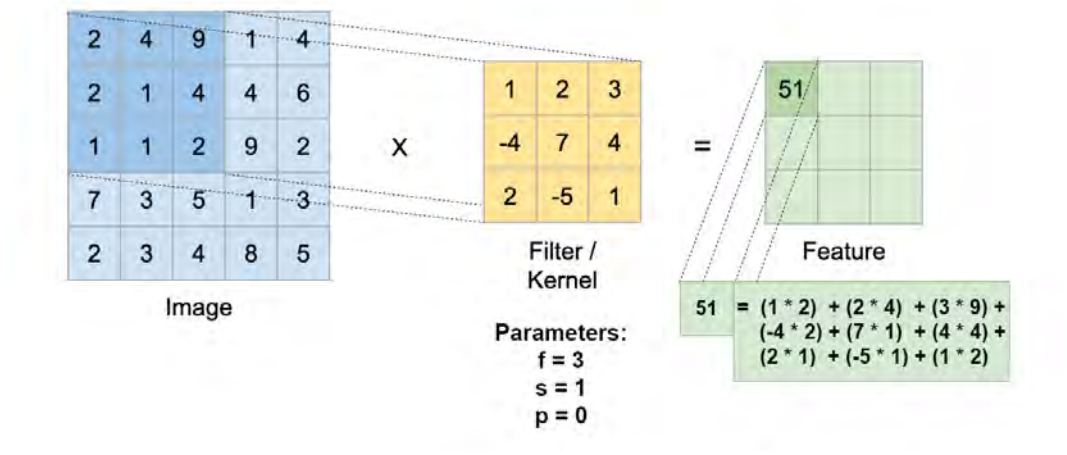


Fig. 12: Convolution by a 3x3 filter on a 5x5 image.

The output dimensions of the feature map can be computed by the following expressions:

$$W_{l+1} = \frac{W_l - f + 2p}{s}, \qquad H_{l+1} = \frac{H_l - f + 2p}{s} \tag{11}$$

where W_{l+1}, H_{l+1} and W_l, H_l are the dimensions of the feature map and the input map, respectively.

Pooling

A limitation of the feature map output of a convolutional layer is that they are sensitive to the location of the features that are present in the input image. In other words, the feature map extracted important features on precise locations of the map, so a small movement in the position of the feature in an input image will result in a different feature map. An approach to address this, is to downsample the feature maps. This makes the resulting down-sampled feature maps more robust to changes in the position of the feature in the image. This robustness against small variations in input is often referred to as **local translation invariance** [34].

Pooling layers provide such an approach to downsample feature maps by summarizing the presence of features in patches of the feature map. Two common pooling methods are max pooling and average pooling and these operations are shown in Figure 13.

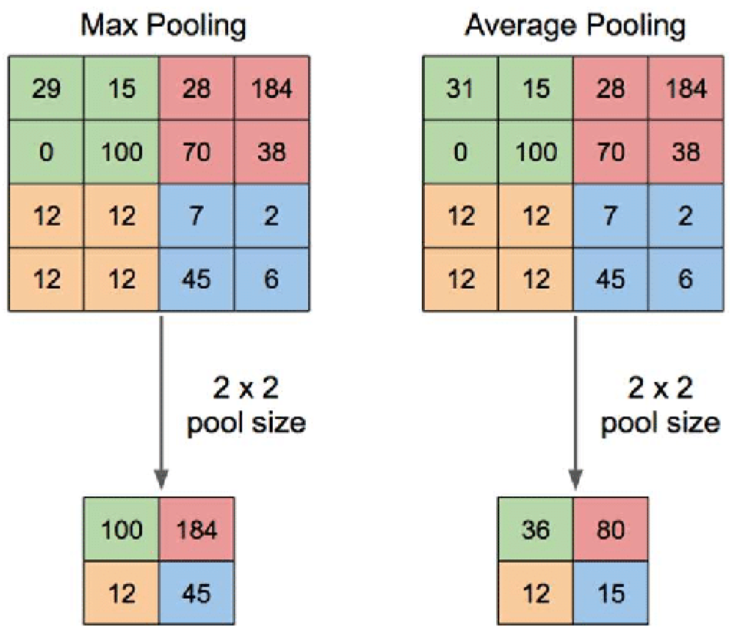


Fig. 13: Max pooling and average pooling (f = 2x2, s = 2).

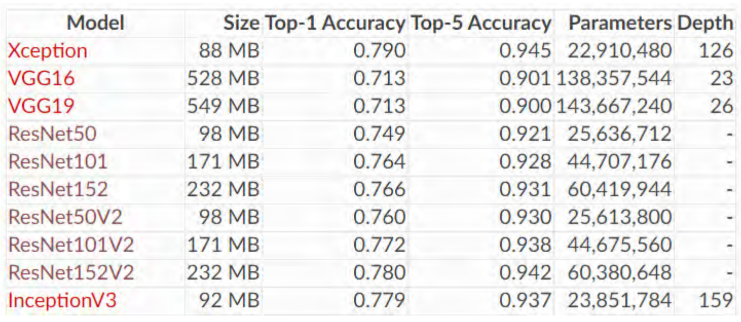
Max pooling and average pooling aim to summarize the most activated presence of a feature and the average presence of a feature, respectively. The output dimensions of the feature map is defined as:

$$W_{l+1} = \frac{W_l - f}{s} + 1, \qquad H_{l+1} = \frac{H_l - f}{s} + 1 \tag{12}$$

where $l + 1$ and l refer to the output- and input layer, respectively.

Architectures

CNNs consist of several kinds of layers that basically make up two parts of the structure. The convolution and pooling layers extract features from the input image and the fully connected layers transform these features into a prediction. The way that these layers are organized and the number of layers that are present make up the architecture of the network. The architecture determines for a great deal the performance of the network on certain classification or regression tasks. Figure 14 shows the performance of CNNs using certain architectures on the ImageNet Large Scale Visual Recognition Challenge (ILSVRC).



Model	Size	Top-1 Accuracy	Top-5 Accuracy	Parameters	Depth
Xception	88 MB	0.790	0.945	22,910,480	126
VGG16	528 MB	0.713	0.901	138,357,544	23
VGG19	549 MB	0.713	0.900	143,667,240	26
ResNet50	98 MB	0.749	0.921	25,636,712	-
ResNet101	171 MB	0.764	0.928	44,707,176	-
ResNet152	232 MB	0.766	0.931	60,419,944	-
ResNet50V2	98 MB	0.760	0.930	25,613,800	-
ResNet101V2	171 MB	0.772	0.938	44,675,560	-
ResNet152V2	232 MB	0.780	0.942	60,380,648	-
InceptionV3	92 MB	0.779	0.937	23,851,784	159

Fig. 14: Performance of some CNN architectures on the ImageNet validation dataset.

The challenge is to correctly classify 50,000 images in 1,000 classes [35]. Top-5 accuracy refers to whether the ground truth is among the top-5 predicted classes. Furthermore, depth refers to the number of layers in the network (convolution layers, activation layers, pooling layers, etc.). Clearly, the sizes of the architectures differ a lot and the impact on the performance is also quite significant.

2.3.3 Training Process

Training a network requires some knowledge about how the network learns and how the various hyperparameters and regularization techniques influence the performance of the network. The goal of training the network is to find the weights that minimize the loss function. This is an iterative process in which a dataset of N input images is propagated (Forward Propagation) through the network in batches of 1 or more. Each image in the batch is propagated through the network and the output is stored. Then, the weights have to be updated in order to minimize the loss function for that batch. There are many optimization algorithms that can be used to reach this goal. Most of them are based on the classical Stochastic Gradient Descent (SGD). SGD updates the weights every iteration according to the following update rule:

$$w_i := w_i - \alpha \frac{\partial L}{\partial w_i} \tag{13}$$

where α is the learning rate that determines how much to learn from the current batch. Furthermore, $\frac{\partial L}{\partial w_i}$ is the partial derivative of the loss function with respect to the weight w_i. This is a partial derivative, because the weights are normally not explicitly defined in the loss function. The Backward Propagation algorithm is used to compute this partial derivative and was introduced by Rumelhart et al. (1986) [36]. Using this algorithm, the weights throughout the whole network can be updated. When a batch is propagated forwards and backwards, an iteration is done. Furthermore, when all the images in the dataset have been passed forward and backwards through the network, this marks one epoch. Finally, after enough iterations, SGD is able to reach a local minimum of the loss function.

The choice of the optimization algorithm for a deep learning model can mean the difference between training time in minutes, hours, and days. Another example of an optimizer is the Adam optimizer, which has proven to minimize the loss function quite faster during training than other variants as can be seen in Figure 15.

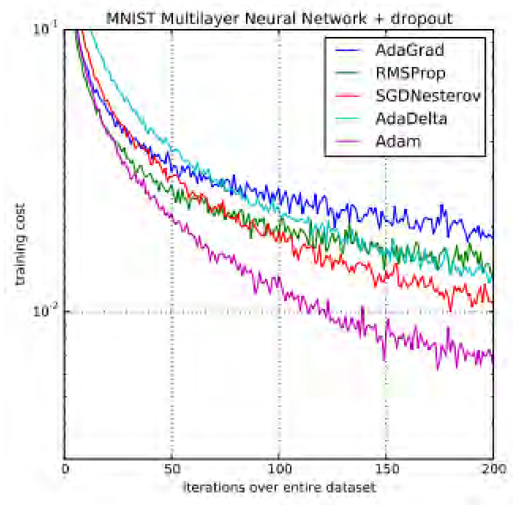


Fig. 15: The training loss for a network with various optimizers (Source: [2]).

Transfer Learning

Throughout the years, people have been creating deep learning models for all kinds of tasks. Some tasks are quite similar, such as image classification tasks where an image has to be classified in the correct class. Transfer Learning builds on the idea to take a pre-trained model and use it for a similar task. As stated before, the layers in a CNN gradually extract higher level features. The central idea behind transfer learning is that low level features are relatively similar across networks that are trained for similar tasks. This approach can help when the data is scarce for one task, as the pre-trained model is already able to extract lower level features. Hence, the pre-model only has to be fine-tuned on relatively less data to learn the high level features of the new task. Besides, it also saves a significant amount of time when the network is already designed. Figure 16 shows how transfer learning can be applied.

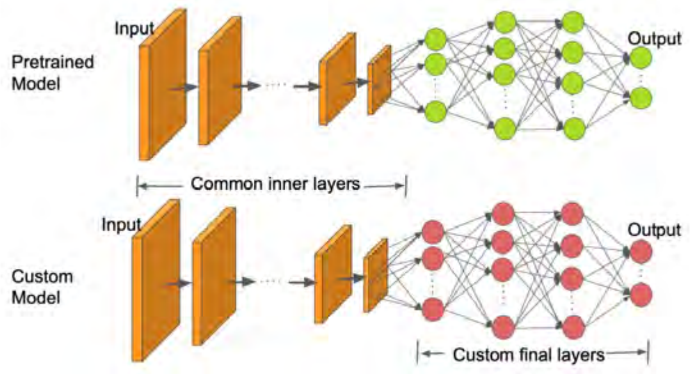


Fig. 16: The custom model reuses the convolutional layers of the pre-trained model and the final layers are customized.

This is a common transfer learning approach in which the layers are kept that contribute to feature extraction from the inputs. The fully connected layers are often replaced in order to fit the new task at hand. Especially the number of output nodes has to be adjusted to the number of classes in the new task. When the custom model is set up, it can start training in order to minimize the new loss function.

Regularization

Increasing the complexity of deep learning models by adding more layers or more nodes allows the network to learn more complex structures. At the same time, the model starts learning the training data in more detail. Figure 17 shows that the network follows the training data in more detail when you increase the number of nodes M.

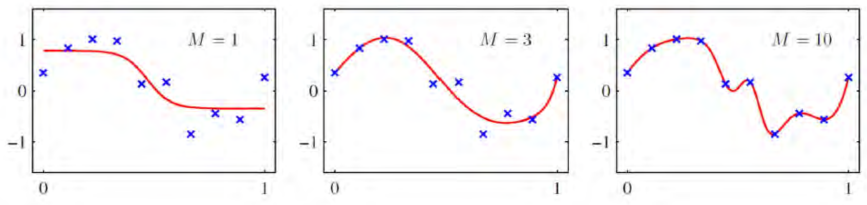


Fig. 17: The effect of increasing the number of nodes of a neural network (Source: [3]).

The problem with modeling the training data too well, as in the right graph of Figure 17, is that the model is not able to generalize to unseen examples. This phenomenon is referred to as overfitting and is a very common problem in deep learning. Figure 18 shows the four possible scenarios for a trained model.

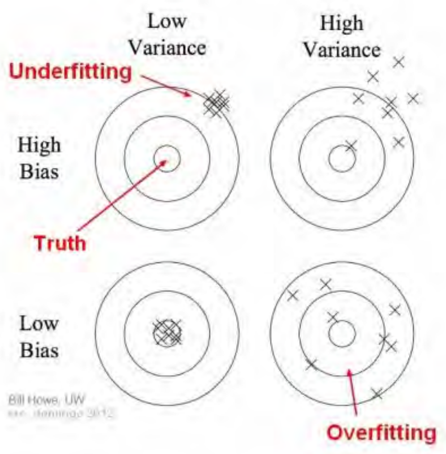


Fig. 18: Bias variance decomposition.

The desired scenario is depicted in the bottom left, since the model's prediction is very close to the true value (low bias) and it finds this true value consistently (low variance). The model is underfitting when it is consistent, but far off the true value (top left). In this case, the complexity of the model can be increased, as the underlying pattern of the data might be more complex. On the other hand, the amount of training data might be too few and extra data should be gathered/created. When the model is relatively close to the true value and inconsistent in its predictions (bottom right), the model is overfitting. As in Figure 17, overfitting happens when the model captures noise along with the underlying pattern in the data. This can easily be solved by making the model less complex in order to model the data in less detail. However, this does not always solve the overfitting problem. Luckily, there are many other regularization techniques that can solve the problem of overfitting and some common one's will be discussed.

One of these regularization techniques is **dropout**, where each node and corresponding connections in the network are removed or 'dropped out' with probability p. By randomly removing nodes and their connections each iteration, the dependencies between the nodes are reduced. This forces the network to pay less attention to noise [37]. Figure 19 illustrates how dropout is applied.

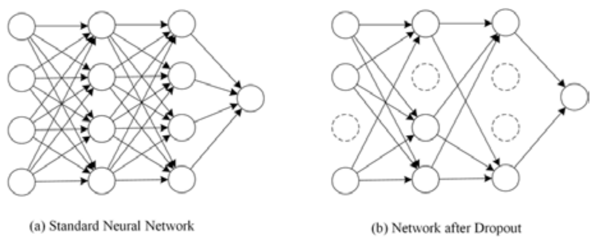


Fig. 19: Dropout.

Another approach is to stop the training process before the model starts overfitting. This approach is referred to as **early stopping** and is used to prevent the model from learning the noise in the data. Figure 20 shows how early stopping works.

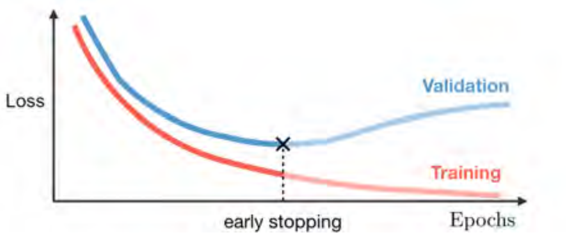


Fig. 20: Early stopping.

The training process is stopped when the validation loss does not decrease anymore, while the training loss still is decreasing. This is a clear sign that the model is overfitting, since the model is improving on the training data, but not on new data.

Two other approaches are the **L1** and **L2** regularization that both aim to penalize large weights in the network. By keeping the weights small, the model is limited in its complexity. L1 regularization (Lasso) does this by setting certain weight values to zero, while L2 regularization (Ridge) makes weights smaller. These forms of regularization are achieved by adding the weight terms to the loss function:

$$L1 = L + \lambda \sum_i |w_i|, \qquad L2 = L + \lambda \sum_i w_i^2 \tag{14}$$

where λ controls the degree of regularization.

It is common use to normalize input data in order to speed up the learning process of the model. If the model is benefiting from normalizing the input layer, why not do the same thing for the hidden layers. Normalizing all of the inputs to a standard scale allows the network to learn the optimal parameters more quickly for each node. **Batch normalization** is built around this idea and normalizes the inputs to a layer for every batch. Besides speeding up the learning process, batch normalization also has a regularization effect. Similar to dropout, it adds some noise to the inputs for each hidden layer.

A final approach to reduce overfitting or to increase the performance of a model is to use **data Augmentation**. Augmenting data is basically creating new data by applying transformations on the data that is already present. Especially in image classification tasks, data augmentation is commonly used, as large numbers of labeled images are often lacking. Some basic data augmentation techniques for images were already discussed in Section 2.2. The basic set of 2D geometric transformations (Figure 5) gives some examples of these augmentations. Some other augmentation techniques are:

- Color Augmentation: changing contrast, brightness, saturation, etc. This makes the model more robust to small changes in illumination.
- Flipping: flip the image horizontally or vertically.
- Zooming: zooming is a powerful augmentation that can make a network robust to small changes in object size. By simply taking a random crop and resizing the image, this is achieved.
- Gaussian Blur: a form of blurring that distorts the image in order to make the model more robust against quality differences in images.

2.4 Evaluation

Evaluating a classification model is commonly done with the help of the so-called confusion matrix. This matrix is used to compute some common performance measures, such as **accuracy**, **precision**, **recall**, and the **F_1-score**. The confusion matrix for a binary classification problem is shown in Figure 21.

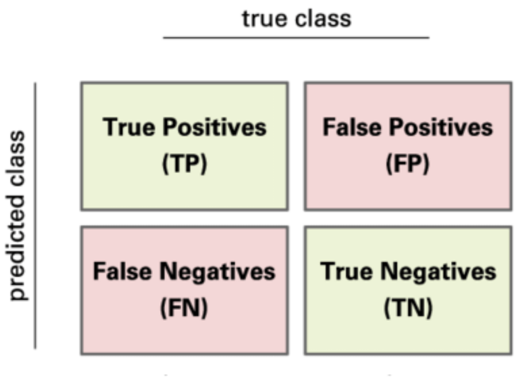


Fig. 21: The confusion matrix.

The terms in the matrix can be explained as:

- True Positives (TP): The model predicted class i and the actual class is i
- True Negatives (TN): The model did not predict class i and indeed class i was not the actual class
- False Positives (FP): The model predicted class i while the actual class is another class.
- False Negatives (FN): The actual class is i and the model did not predict class i

The accuracy is the fraction between correct predictions and all predictions that were made and can be computed by:

$$Accuracy = \frac{TP}{TP + TN + FP + FN} \tag{15}$$

The accuracy is a very useful statistic, however, it can be misleading when there is an imbalance in the dataset. In fraud detection, for example, the fraction of fraudulent instances is often lower than 1%, while the goal is to detect these exceptions. A model can easily achieve 99% accuracy by classifying all instances as non fraudulent. In the case of a class imbalance, it is good practice to use precision or recall. They are defined as:

$$Precision = \frac{TP}{TP + FP} \qquad Recall = \frac{TP}{TP + FN} \tag{16}$$

The precision of a class measures the fraction of the correct predictions of that class and all predictions of that class (including the false positives). While the recall of a class measures the fraction between the correct predictions of a class and all instances of that class (including the false negatives). The evaluation measure should be chosen based on what is relevant for the task. In the example of fraud detection, both precision and recall are important, since you want to detect all fraudulent instances and at the same time not classify all instances as fraud. In these cases, the F_1-score can be used, as it is the harmonic mean between the precision and the recall:

$$F_1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{17}$$

3 Face Image Data

This section gives an overview of the data that is used throughout this research. First, this section discusses the datasets that are used to train the deep learning algorithm based on [18][16] (Section 3.1). These datasets will also be used to apply the face normalization techniques that will be explored during this research. This makes an evaluation of the techniques possible by observing the influence of the face normalization technique on the performance of the deep learning algorithm. Next, the datasets used for assessing face image quality will be discussed (Section 3.2). These datasets are used for generating the quality scores and for training a deep learning algorithm in a supervised manner.

3.1 Face Normalization

The deep learning algorithm is trained on three distinct datasets. As stated above, the face normalization techniques that are investigated in this research will be applied to these datasets. Once the datasets are normalized, they are used for training and testing the algorithm. As face normalization is a preprocessing step on the data, these steps will not be discussed in this section, but in Section 4.1. Other preprocessing steps will be discussed here.

3.1.1 VV Dataset

The VV dataset [18][17] consists of around 25,000 face images with several annotations for the images. The dataset has the following main features:

- The dataset set contains annotations about emotions, age, gender, ethnicity, beard amount, mustache amount, and presence of glasses.
- The primary sources of the images in this dataset are specially conducted video-shoots in a studio, crowd-sourced web-cam images, as well as public access images on the internet.
- The images have an average size of 256x256 pixels in RGB and in grayscale.

The dataset contains a wide range of face image quality conditions, due to the various capturing settings of the images.

3.1.2 UTKFace Dataset

UTKFace [38] is a dataset that consists of over 20,000 face images. The main features of this dataset are described below:

- The dataset includes annotations of age, gender, and ethnicity.
- The face images in the dataset are captured in an unconstrained setting and they cover large variations in pose, illumination, resolution, facial expressions, etc.
- The dataset covers a long age span (0 to 116 years old).

Some examples of the UTKFace dataset are shown in Figure 22.

Fig. 22: Example faces from the UTKFace dataset.

3.1.3 MORPH Academic Dataset

The Academic MORPH dataset [39] consists of over 55,000 face images of around 13,500 subjects. The main features of this dataset are listed below:

- The dataset includes annotations of age, gender, ethnicity, facial hair, and presence of glasses.
- The ages range from 16 to 77 with a median age of 33.
- The average number of images per individual is 4 and the average time between photos is 164 days.
- The face images are captured in a more constrained setting, however, there are still variations in pose, illumination, resolution, facial expressions, etc.

The variations in the mentioned characteristics are visible in Figure 23.

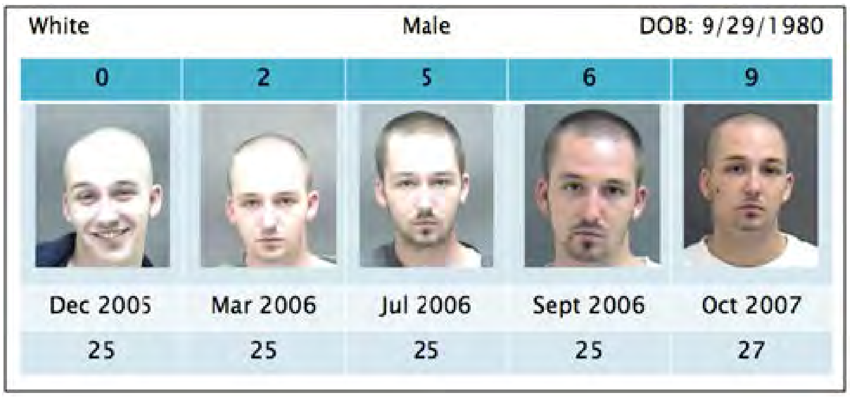


Fig. 23: Example subject from the Academic MORPH dataset.

Figure 24 gives an overview of the distributions of the classes within all three datasets:

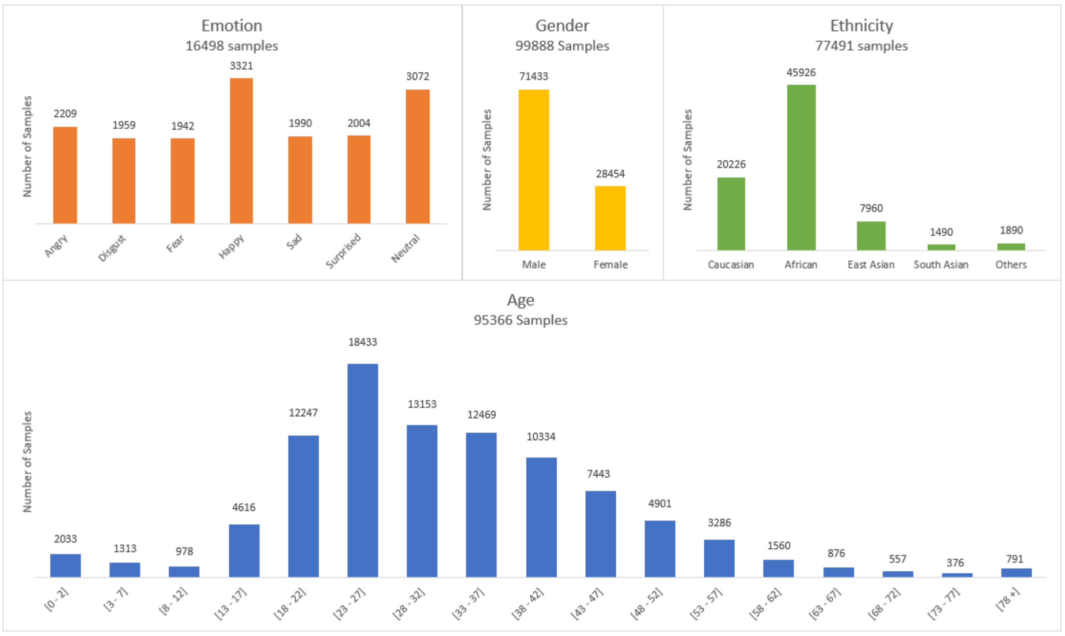


Fig. 24: Histograms of the distributions of the classes within the VV dataset, the UTKFace dataset, and the MORPH academic dataset combined.

This research will leave out any other characteristics than emotion, gender, age, and ethnicity, since experimenting with these characteristics can give enough information about the research goals that were set. Note that ethnicity will only be used to perform a bias study which is described in Section 5.2.4. This is done because I see no application where the prediction of one's race based on a face image could be of positive value.

3.1.4 Preprocessing

To create a reliable estimate of the performance of the deep learning network, it is required to evaluate the network on unseen data that was not used during training. As a result, the datasets used by the network will be divided into a train, validation, and test set. The train set will be used to iteratively update the weights of the network, while the validation set is used to get an initial indication of how the network can generalize. The validation set will also be used to perform early stopping, as this will prevent overfitting. Finally, the test set will be used to obtain a reliable estimate of the performance of the network. Table 1 shows in what partitions the data will be split.

Train	**Validation**	**Test**
80%	10%	10%

Table 1: The train, validation, and test split.

To get more robust results, the test set will be doubled in size by flipping the face images horizontally. Flipping the face images will enrich the variability in pose of the faces and specifically the yaw angle. Other augmentation techniques are not applied, since variability in illumination and sharpness are well represented in the test set.

3.2 Face Quality Assessment

Mainly the reference quality score requires a dataset, as this approach measures the similarity to a reference or 'ideal' face image and uses this similarity as the ground truth quality score. When the quality score is generated for the dataset, a network will be trained on the same dataset with the quality score as a target. This way, the network does not need a reference image to generate a quality score for new face images. A dataset for this approach has the following requirements:

- The data should contain images in which most of the complete face is visible.
- The dataset should contain multiple images of each subject.
- The face images should be annotated with the identity of the subject.
- The images should be captured in-the-wild or in an unconstrained setting in order to cover a wide variety of image quality conditions.

The empirical quality score uses empirical features of a face image to generate a quality score and consequently does not need a reference image. Therefore, the same dataset will only be used for a qualitative evaluation of the first approach.

3.2.1 Labeled Faces in the Wild

Labeled Faces in the Wild (LFW) [27] is a dataset that consists of more than 13,000 images of faces collected from the web. The main features of this dataset are described in the list below:

- Each face image has been labeled with the name of the person pictured.
- There are 5,749 people pictured in the dataset of which 1,680 have two or more distinct photos.
- The images are captured in-the-wild, and therefore cover a wide range in image quality conditions.
- The face images are detected by the Viola-Jones face detector [40].

The only constraint on this dataset is that the face images were detected by the Viola-Jones face detector, so the pose variations are limited by the pose tolerance of the detector. However, the examples of Labeled Faces in the Wild in Figure 25 show that the pose tolerance is still large enough to resemble the pose variations that could occur in the example of facial analytics in retail.

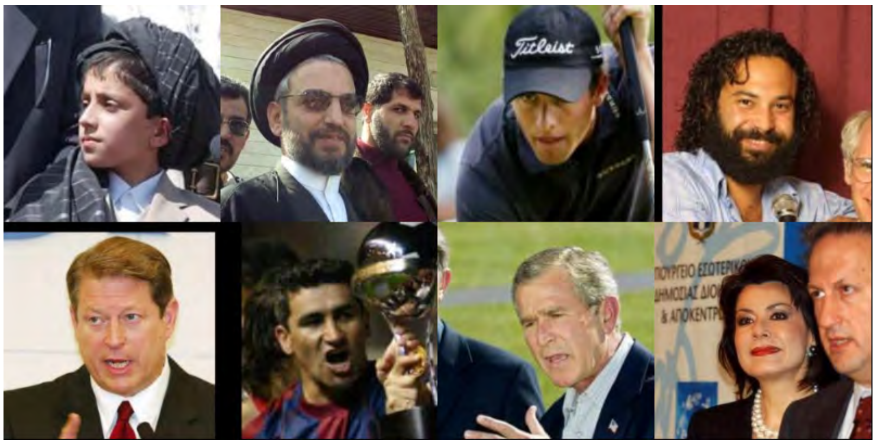


Fig. 25: Example faces from the Labeled Faces in the Wild dataset.

There are four different sets of the dataset which cover the original and three different types of 'aligned' images. The original images will be taken, since normalizing and further preprocessing these images is already a goal of this research. In addition, the face normalization technique that performs best will be applied to these images, so the original images are preferred.

3.2.2 CelebFaces

CelebFaces [41] (CelebA) is a large-scale face attributes dataset with more than 200,000 celebrity images. The main features are listed below:

- CelebFaces consists of 10,177 unique identities of which 10,113 have two or more distinct photos
- The images contain annotations about the identity, 40 binary attributes, and 5 landmark locations
- The images cover large variations in pose, background clutter, illumination, etc.

The dataset can be employed for several computer vision tasks, such as face attribute recognition, face detection, and other characteristics. For that reason, it seems like a suitable dataset for face quality assessment. Figure 26 gives an overview of some examples within the CelebFaces dataset.

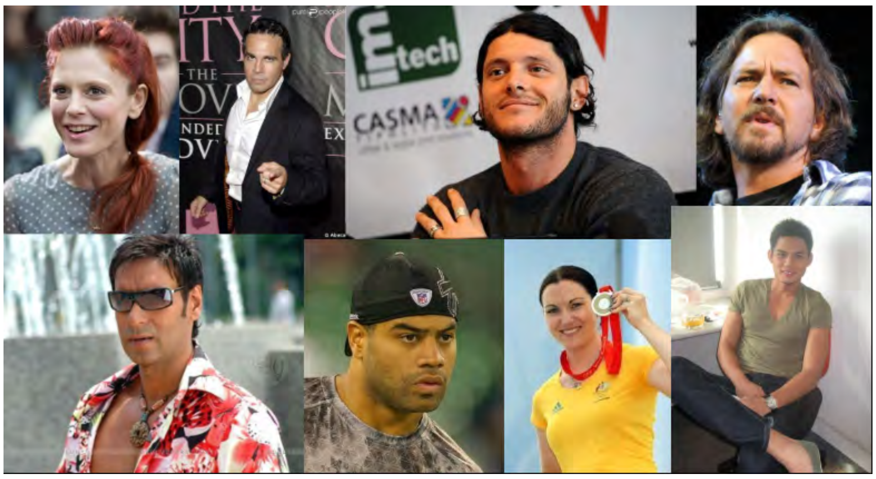


Fig. 26: Example faces from the CelebFaces dataset

The examples show clear variations in pose, illumination and sharpness, which makes it a suitable dataset.

3.2.3 Preprocessing

In order to get a reliable estimate of the performance of both the reference quality model and the empirical quality model, they will be evaluated using a different approach. Both Agarwal (2018) and Hernandez-Ortega et al. (2019) evaluate the quality model by predicting the quality of the face images in a different dataset that is used for face recognition. Then, they test whether the performance on the face recognition task increases as they remove the lowest quality images from the test set [10][22]. This research will use a similar approach that will be discussed in more detail in Section 5.2.3. As a test set of the described two datasets is not necessary, only a train and validation set is needed. Hence, the split will be as follows.

Train	**Validation**
80%	20%

Table 2: The train and validation split.

Furthermore, the following preprocessing steps will be taken for the LFW dataset:

- Subjects with fewer than two images are removed from the dataset. This ensures that a reference image is available.
- 1,680 images are assigned the gallery label
- 7,484 images are assigned the probe label

These preprocessing steps are taken in order to measure the similarity between the gallery image and the probe image(s) of a subject. The selection of the gallery and probe images will be explained in Section 4.2.2. Finally, the following preprocessing steps are taken for the CelebFaces dataset:

- Subjects with fewer than 20 images are removed from the dataset. This is done to improve the possibility of the existence of a gallery image of 'perfect' quality for each subject.
- 5,892 images are assigned the gallery label
- 140,410 images are assigned the probe label

These two datasets will be used to generate 7,484 and 140,410 ground truth quality scores for the LFW and CelebFaces datasets, respectively. CelebFaces is merely used to observe the impact of significantly more data (compared to LFW) on the performance of the quality model.

4 Methodology

This section gives a detailed description of the methods that are used for both face normalization and face quality assessment. First, the face normalization process will be discussed with all of its components. After that, two approaches to assess the quality of a face image will be discussed.

4.1 Face Normalization

Most facial analysis systems use deep learning approaches for automatically analyzing face images. It can be difficult for neural networks to handle high variations in the input data, and therefore it is necessary to reduce these variations. Face normalization techniques aim to reduce these variations by transforming a face image to a more general form. In the case of face image data, these variations are mostly caused by the pose of the face and the amount of contrast in the image [18][17]. As a result, face normalization can be divided into two parts:

- Face Location Normalization
- Color Normalization

Color normalization will not be addressed extensively in this research as it is a fairly straightforward process that has not many variations. For example, a common color normalization method is to standardize the pixel values of an image. Each pixel value x becomes:

$$x = \frac{x - \mu}{\sigma} \tag{18}$$

where μ is the image mean and σ is the standard deviation of the image. Another approach is to let μ be the global mean of the whole dataset on which the deep learning model is trained. Then, σ becomes the standard deviation of the whole dataset. This approach is used for the deep learning model in [16][18].

Face location normalization algorithms aim to reduce variations in the pose of the face. As discussed in Section 2.1.1, this research will only consider algorithms that try to remove in-plane rotation of the face, since algorithms that remove out-of-plane rotation are too computationally expensive. In-plane rotation removal corresponds to the adjustment of the roll angle of the face and was depicted in Figure 3 (a). Aside from adjusting the roll angle, face location normalization tries to remove irrelevant background information in order to only capture the facial features.

In general, the preprocessing steps for face location normalization are:

- Localize the face in the input image using a face detector and extract the crop of the face.
- Use a landmark detector to detect facial landmarks that can be used for the alignment of the face.
- Set fixed points in a new image that correspond to the facial landmarks that will be used in the transformation.
- Warp the utilized facial landmarks onto the fixed points in the new image and include the other parts of the face in this transformation.

Figure 27 shows the components of the face location normalization algorithm proposed by Gudi et al. (2014).

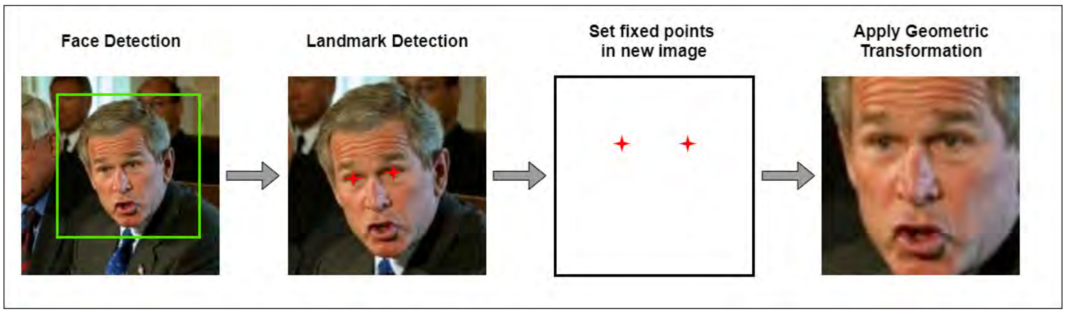


Fig. 27: The face location normalization algorithm proposed by Gudi et al. (2014).

The components of this algorithm will be discussed in more detail in this section. As the proposed face location normalization algorithm is not able to remove pose variations in the yaw and pitch angle, this section also describes an approach to measure these angles. This information will be used to quantify the impact of variations in yaw and pitch on the performance of a facial analysis system.

4.1.1 Face Detection

Face detection can be regarded as an object detection problem. Object detection aims to locate objects in an image and to classify them at the same time. A face detection algorithm tries to locate faces in an image and to ensure that it is a face. Locating a face is done by determining 4 coordinates around the face and drawing lines between them to create a bounding box.

OpenVino [42] offers a toolkit for applications and solutions that replicate human vision. It contains all sorts of pre-trained machine learning models that are based on convolutional neural networks. Furthermore, the models are tested on the edge and can be used for real-time applications. They offer a face detector (face-detection-retail-0004) based on SqueezeNet light [43] as a backbone for indoor or outdoor scenes and is shot by a front-facing camera. This face detector achieves an 83% Average Precision (AP) on the WIDER FACE face detection benchmark [44]. Where the AP is defined as the area under the precision/recall curve. The application setting and the performance make it a suitable face detector for this face location normalization process, and therefore it will be used.

The model outputs N bounding boxes for an image and each detection contains the following variables:

- Confidence: a confidence probability for the predicted face
- (x_{min}, y_{min}): coordinates of the top left bounding box corner
- (x_{max}, y_{max}): coordinates of the bottom right bounding box corner.

The face detector can detect multiple faces in an image. Rejecting images with multiple faces would lead to a significant loss in relevant data. Therefore, the assumption is made that the biggest bounding box belongs to the subject that corresponds to the annotations of that image. Given the LFW and CelebFaces datasets, this is a safe assumptions, as the relevant person in the images are centered. Let S be the surface of the bounding box, then:

$$S = (x_{max} - x_{min}) \cdot (y_{max} - y_{min}) \tag{19}$$

Then, the highest value for S corresponds with the biggest bounding box.

4.1.2 Landmark Detection

The next step in the face location normalization process is to detect facial landmarks that will be used in the geometric transformation. Dlib is a widely used C++ library that contains a wide range of machine learning algorithms including landmark detectors. In particular, Dlib offers two types of facial landmark detectors, namely a 5-point and a 68-point facial landmark detector. Two examples of these landmark detectors are shown in Figure 28.

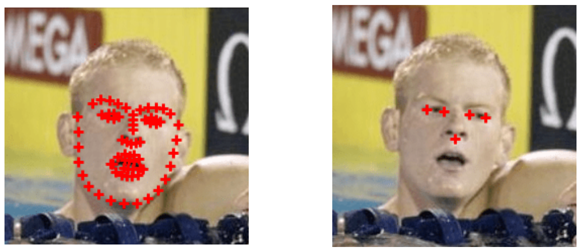


Fig. 28: Dlib's 68-point facial landmark detector (left) and the 5-point detector (right).

The algorithm proposed by Gudi et al. (2014) uses the eye centers, so they can use the 5-point facial landmark detector. The eye centers can be determined by averaging the coordinates of the eye corners. As discussed in Sections 2.1.1 and 2.2, taking more facial landmarks than the eyes could make the method more robust. Therefore, it is necessary to also consider the 68-point facial landmark detector, as it contains more landmarks. The main differences between the two landmark detectors are listed below [45]:

- The 5-point facial landmark detector is 8-10% faster than the 68-point facial landmark detector.
- The 5-point facial landmark detector (9.2MB) is over 10 times smaller than the 68-point facial landmark detector (99.7MB).
- The 68-point facial landmark detector is slightly more accurate than the 5-point facial landmark detector.

As stated, the 5-point landmark detector has an advantage over computation time and size, however, the 68-point landmark detector can still be used for real-time applications. Therefore, the choice between the landmark detectors depends on which facial landmarks will be used. The facial landmarks that are considered in the experiments will be discussed in Section 5.1.

Another facial landmark detector is provided by OpenVino. Aside from the eyes and nose landmarks, this landmark detector also detects the mouth corners, which could be useful. However, it can be very inaccurate compared to Dlib's 5-point facial landmark detector as can be seen in Figure 29.

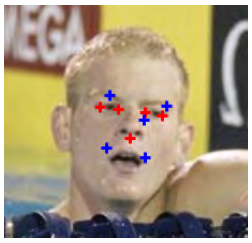


Fig. 29: The 5-point facial landmark detector from Dlib (red crosses) and the 5-point facial landmark detector from OpenVino (blue crosses). Note that Dlib's landmark detector does not detect the mouth corners.

Finally, a state-of-the-art 5-point landmark detector is proposed by Deng et al. (2019). This landmark detector is called RetinaFace and detects the same facial landmarks as the OpenVino detector [46]. It shows promising results, however, the implementation of this landmark detector is too time-consuming, as it uses a different deep learning framework (MXNet). Considering the 6-months time horizon of this research, only the facial landmark detectors of Dlib will be used.

4.1.3 Geometric Transformation

Before applying the geometric transformation on the face image, fixed points have to be determined in the new image that correspond to the facial landmarks. The facial landmarks will be warped onto these fixed points by means of a geometric transformation. The other parts of the face will be included in this transformation. The coordinates of the fixed points have to be considered for the size of the new image. This size depends on the required input size of the considered convolutional neural network. In the example below, a required input size of 96x96 is considered. Then, a trial-and-error approach to determine the coordinates of the fixed points is illustrated below:

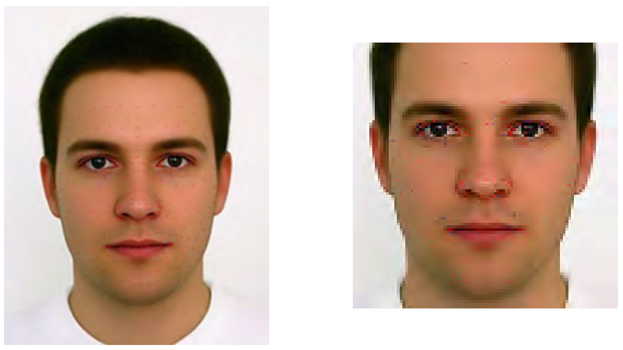


Fig. 30: The average face of 32 males (left) and the normalized version (right) (Source: [4]).

Figure 30 shows the average face of 32 males on the left and it can be used as a face that represents a 'general' face size and 'general' face proportions. This is important, because fixed points are only determined for this face, so the face should generalize to other faces. Furthermore, a male face is taken, as the female face is generally slightly smaller. This ensures that the face features are fully captured when a female face image is normalized. The fixed points can be determined by normalizing the average face image and experimenting with the coordinates of the fixed points. One should seek for a normalized face where irrelevant background information is minimized and all the face features are fully captured. Figure 30 shows the normalized version of the average face on the right, which has size 96x96. Here, the eyes are used as facial landmarks and it can be observed that the described criteria are satisfied. The coordinates of the fixed points that correspond to the eyes are set with the following formulas:

$$\begin{aligned} x_l &= \frac{1}{3} * w, \quad & y_l &= \frac{1}{3} * h \\ x_r &= \frac{2}{3} * w, \quad & y_r &= \frac{1}{3} * h \end{aligned} \tag{20}$$

The pairs (x_l, y_l) and (x_r, y_r) correspond to the coordinates of the left and right eye, respectively. Furthermore, w and h are the width and height of the new image. In this example, $w = h = 96$, so the coordinates are $(32, 32)$ and $(64, 32)$ for the left and right eye, respectively. Note that the coordinate system of an image has its origin in the left top corner.

Determining the formulas is a trial-and-error process and should result in the normalized face in Figure 30. Other fixed points, such as the chin or the nose will also be determined in this way.

Finally, the geometric transformation can be applied by using the coordinates of the fixed points. As mentioned in Section 2.2, the similarity or the affine transformation will be used depending on the facial landmarks that are used. Both the type of transformation as the facial landmarks that will be considered is discussed in more detail in Section 5.1.

4.1.4 Head Pose Detection

As mentioned in Section 2.1.1, pose variations are one of the biggest factors that cause inaccurate predictions of facial analysis systems. The proposed face location normalization process can remove variations in the roll angle, but not in the yaw or pitch angle. In order to quantify the impact of those pose variations, it is necessary to have this information about the face images in the datasets that were described in Section 3.1. As annotations of yaw and pitch are not present for those datasets, a head pose estimation network can be used to obtain this information.

OpenVino provides such a head pose detector (head-pose-estimation-adas-0001) that is based on a simple CNN architecture [42]. It was trained on the Biwi Kinect Head Pose Dataset [47] that covers about ±75° yaw and ±60° pitch. Furthermore, it achieved the following accuracy on a test partition of the dataset:

Yaw:	5.4° ± 4.4°
Pitch:	5.5° ± 5.3°

Table 3: Mean absolute error ± Standard deviation of absolute error (in angle).

This will serve as an accurate pose detector, since a 5° error in angle is quite small. Also, the variations in yaw and pitch that are covered by this pose detector are big enough, as bigger angles correspond to faces where the face features are not visible anymore. These faces are not analyzed in the first place, hence this pose detector is suitable to quantify the impact of variations in pose. Figure 31 shows an example of a pose estimation by the head pose detector.

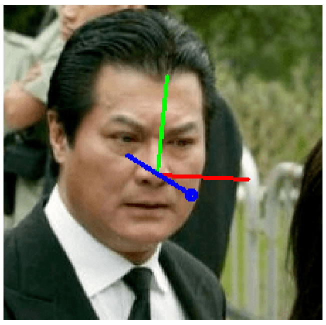


Fig. 31: An example of OpenVino's head pose detector.

A more detailed description about how the resulting pose information is used can be found in Section 5.1.2.

4.2 Face Quality Assessment

Assessing the quality of face images can be extremely useful when facial analysis systems are used to analyze faces in an unconstrained setting. Only analyzing faces of good quality can increase the performance of these systems significantly [10][22][8]. As discussed in Section 2.1.2, two approaches will be considered in this research, namely the design of an empirical quality score and a reference quality score. First, the design of the empirical quality score will be discussed, together with some experiments that demonstrate the validity of the proposed quality measures. Next, the process of generating the reference quality score for the face images in the LFW and the CelebFaces datasets will be discussed. In addition, the design of a deep learning model that is able to predict this score will be discussed. Note that the figures that are presented in this section give a visual indication of how the quality measures perform and that the actual performance will be presented in Section 6. Also, the images that are presented in the figures are normalized by the face location normalization process that is described in the previous section.

4.2.1 Empirical Quality Score

As discussed in Section 2.1.2, an empirical quality score should quantify the quality of a face image by empirically using face image properties. The main factors that affect face image quality are symmetry, sharpness, pose, and illumination, according to Gao et al. (2007). Four quality measures were mentioned that try to measure the face image properties and they will be discussed in detail below. Finally, a combination of these measures will be used to create the empirical quality score.

Symmetry

Luo et al. (2014) propose a face image measurement for frontal view face images by computing the symmetry in a face image. They define symmetry as the difference between the left and right half of the face. They formulate this by letting a matrix $\mathbf{Y} \in \mathbb{R}^{96x96}$ denote a face image of size 96x96. The symmetry measure can then be written as:

$$S(\mathbf{Y}) = \|\mathbf{YP} - \mathbf{YQ}\|_F^2 \tag{21}$$

where $S(\mathbf{Y})$ is the symmetry of image $\mathbf{Y}$. Furthermore, $\|\cdot\|_F$ is the Frobenius norm, which is the square root of the sum of the absolute squares of the elements in a $m \times n$ matrix:

$$\|\mathbf{Y}\|_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n} |a_{ij}|^2} \tag{22}$$

Furthermore, $\mathbf{P}, \mathbf{Q} \in \mathbb{R}^{96x96}$ are two constant matrices:

$$\mathbf{P} = diag([\mathbf{1}_{48}\mathbf{0}_{48}]), \qquad \mathbf{Q} = diag([\mathbf{0}_{48}\mathbf{1}_{48}]) \tag{23}$$

where $diag(\cdot)$ denotes the diagonal matrix, so $\mathbf{P}$ becomes a diagonal matrix with 48 ones and 48 zeros on the diagonal. Low values for $\mathbf{Y}$ indicate that the left and right half of the face do not differ much, hence the face is more likely to be in frontal view. Luo et al. (2014) state that low values of this symmetry measure correspond to frontal view images. They show two examples in which this is true. However, using this measure on normalized images of the LFW dataset shows that this is not true. Figure 32 illustrates this.

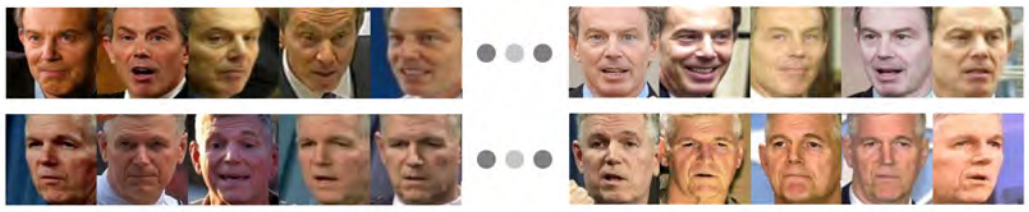


Fig. 32: The images of two persons are ranked according to the symmetry measure. In each row, the five best images are shown on the left and the worst five images are shown on the right.

In general, frontal facing faces are not preferred by this symmetry measure. The most left face image should be the face with the lowest yaw and pitch angle, however, both persons have other face images with lower angles in both yaw and pitch. It is important to rank these images accurately on their frontal pose, and therefore it is preferred to use another measure to achieve this. As mentioned before, a head pose detector can be a better option to measure frontal pose, since it can predict the yaw and pitch angle quite accurately.

The symmetry measure might still be useful to detect occlusions in front of the face, such as hands or other objects, as an occlusion would indicate a non-symmetrical face. However, experimenting with this measure did not result in any kind of occlusion detection. An example of this is shown in Figure 33.

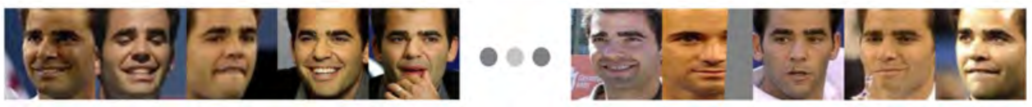


Fig. 33: The images of a persons are ranked according to the symmetry measure. In each row, the five best images are shown on the left and the worst five images are shown on the right.

It can be observed that the symmetry measure ranked an occlusion (a hand) as fifth most symmetric, namely the fifth image from the left. Hence, this measure is not an accurate indicator of occlusions and will not be used in the empirical quality score.

Rank

Luo et al. (2014) also propose a face image measurement for the sharpness of a face image. They let matrix **Y** denote a face image. Then, they define the rank of image **Y** as:

$$R(\mathbf{Y}) = \|\mathbf{Y}\|_* \tag{24}$$

where $\|\cdot\|_*$ denotes the nuclear norm, which is the sum of the singular values of a matrix. Here, the nuclear norm is used to approximate the rank of a matrix. The rank of a matrix corresponds to the maximal number of linearly independent columns. Two linearly independent columns strongly differ in pixel values, so high rank corresponds to an image with much variations in pixel values. This means that the overall image has high amounts of contrast or sharp edges. Therefore, the image is sharp, and hence high values for the rank measure indicate sharper images compared to low values that indicate less sharp images. This makes sense in theory, however, applying the rank measure on normalized images of the LFW dataset shows inaccurate behavior on measuring the sharpness of an image:

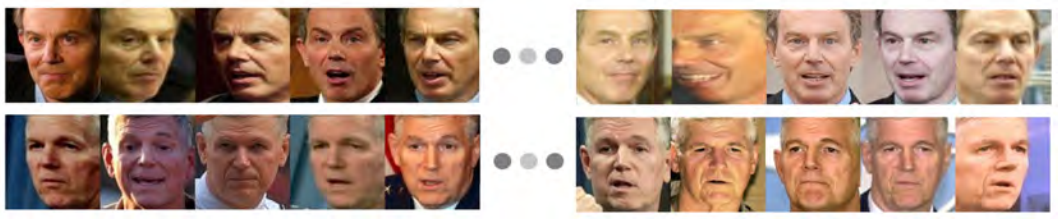


Fig. 34: The images of two persons are ranked according to the rank measure. In each row, the five best images are shown on the left and the worst five images are shown on the right.

It can be observed from Figure 34 that some of the images on the right are indeed not sharp or do not have much contrast. However, there are also sharp images among these worst five images and these should not be ranked as low as they are. In addition, there are quite blurry images among the best five images on the left, hence a better measure is preferred.

Pose

As the symmetry measure is not an accurate indicator for frontal pose faces, a head pose detector might be a better option. The head pose detector that was discussed in Section 4.1.4 will be used to investigate this. The pitch and yaw angle can be predicted accurately by this pose detector and can be used to rank the face images of a person. Again, Let matrix $\mathbf{Y}$ denote a face image. Then, a pose measure can be constructed by using the pitch and yaw angle:

$$P(\mathbf{Y}) = w_p|\text{pitch}^\circ| + w_y|\text{yaw}^\circ| \tag{25}$$

where $P(\mathbf{Y})$ is the measured pose of image $\mathbf{Y}$ and w_p and w_y are weights for the pitch and yaw angle, respectively. As it is not clear which angle influences the performance of a facial analysis system most, they are now set to $w_p = w_y$ for the validation. Quantifying this influence is a research goal on its own and Sections 5 and 6 will provide a detailed description of this influence. Furthermore, absolute values are taken to ensure that small values of this pose measure indicate a frontal facing face and bigger values indicate non-frontal faces. The same persons will be used as in Figure 32 to compare the pose measure with the symmetry measure. Figure 35 shows how the images are ranked by the pose measure.

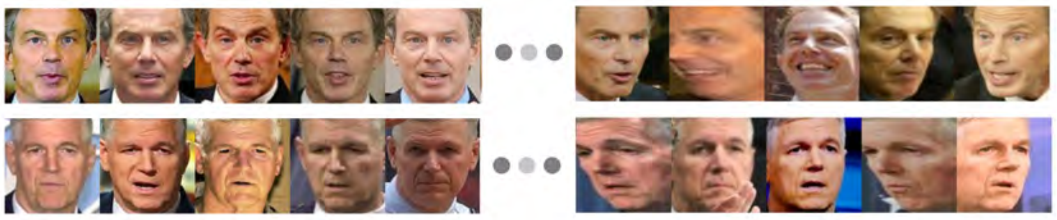


Fig. 35: The images of two persons are ranked according to the pose measure. In each row, the five best images are shown on the left and the worst five images are shown on the right.

It can be observed that frontal facing faces are clearly preferred for both persons. In addition, the biggest pose variations are clearly among the five worst images for both persons. Hence, the proposed pose measure will be used in the empirical quality score.

Variance of the Laplacian

Bansal et al. (2016) propose an effective method for blur image detection by using the Laplacian operator. They use the Laplacian operator to determine whether an image is blurred and the extent to which it is blurred. The Laplacian operator originally is a second derivative operator for a scalar function f:

$$\Delta f = \nabla^2 f = \nabla \cdot \nabla f \tag{26}$$

Besides scalar functions, the Laplacian is often used for edge detection in images, as it can indicate regions of rapid pixel intensity change. The Laplacian of an image $\mathbf{Y}$ with pixel intensity values (x, y) is given by:

$$L(\mathbf{Y}) = \frac{\partial^2 \mathbf{Y}}{\partial x^2} + \frac{\partial^2 \mathbf{Y}}{\partial y^2} \tag{27}$$

This can be computed by using a convolutional filter. The filter is used to approximate the second derivatives in Equation 27. It measures differences between the values of adjacent pixels and a big difference is an indicator of an edge. It is preferred over first derivative-based edge detectors, such as the Sobel operator, because it yields better results [48]. There are two types of Laplacian filters, namely the positive Laplacian filter and the negative Laplacian filter:

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \quad \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \tag{28}$$

Bansal et al. (2016) use the negative Laplacian filter (the right filter), as this is more common [26]. While the filter slides over the image, it produces a new image as output. Figure 36 shows a face image before and after applying the Laplacian to it.

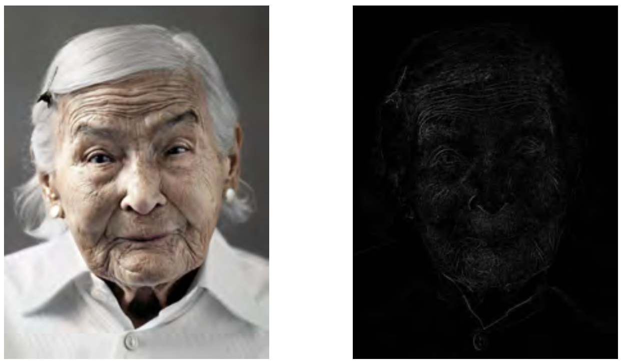


Fig. 36: A face image before (left) and after (right) applying the Laplacian to it.

It can be observed that the resulting image in Figure 36 contains a detailed overview of the edges of the face. Finally, the variance is computed over the output image. High variance indicates big differences between adjacent pixel values, which means that there are many sharp edges/corners. This is representative for a sharp image. Similarly, low variance means that there are less clear edges/corners, which indicates a blurry image. To formally define the variance of the Laplacian, let $VL(\mathbf{Y})$ denote the variance of the Laplacian of image $\mathbf{Y}$, then:

$$VL(\mathbf{Y}) = \operatorname{Var}\left(\frac{\partial^2 \mathbf{Y}}{\partial x^2} + \frac{\partial^2 \mathbf{Y}}{\partial y^2}\right) \tag{29}$$

In order to validate this measure, one can use Gaussian noise to blur an image and observe what happens to the variance of the Laplacian. This is illustrated in Figure 37 .

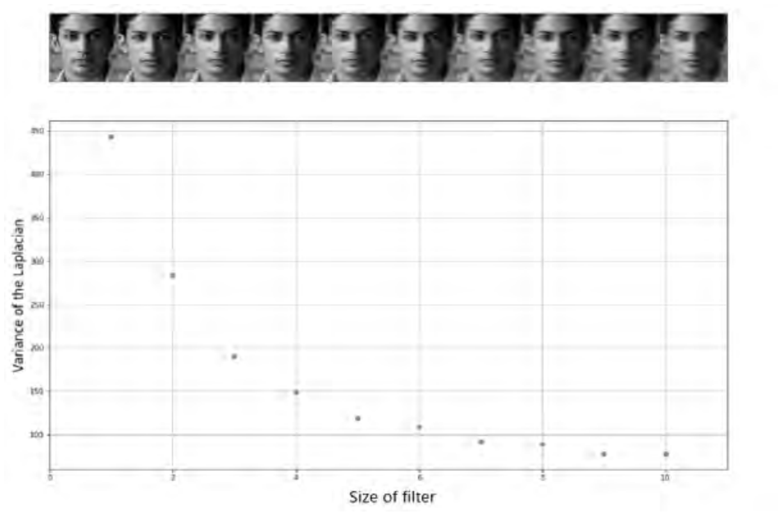


Fig. 37: Top: Face images of a person with an increasing amount of Gaussian blur from left to right. Bottom: Variance of the Laplacian measured over each face image on top. The face images correspond to the data points in the same order. Increasing the size of the filter results in adding more Gaussian blur.

Adding Gaussian noise is done by convolving a filter over the image in order to reduce image detail. The most left face image is the original image and the images to right are increasingly blurred. This is done by increasing the size of the filter. The graph in Figure 37 clearly shows that the variance of the Laplacian decreases as the filter size increases. Hence, low variance indicates a blurred image.

Using the same persons as in Figure 34 shows that the variance of the Laplacian is a better sharpness measure than the rank measure:

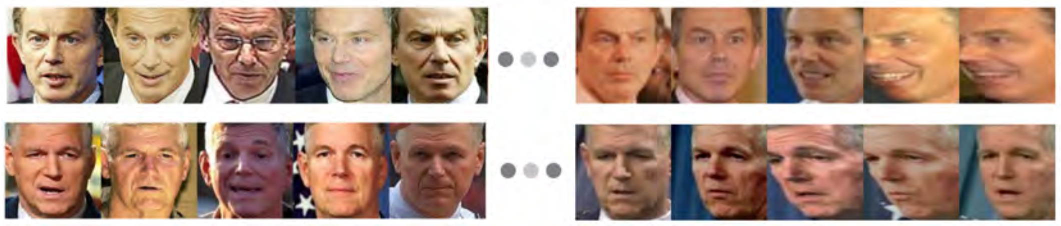


Fig. 38: The images of two persons are ranked according to the variance of the Laplacian. In each row, the five best images are shown on the left and the worst five images on the right.

It can be observed that the images on the left of Figure 38 are indeed sharper than those on the right. Therefore, it seems like a good indicator of sharpness. Furthermore, Bensal et al. (2016) state that the variance of the Laplacian can detect over/under-exposed images, which is an indicator of illumination quality. This also makes sense, as very dark/light images have less clear edges, and therefore low variance in adjacent pixels, as the values are all close to each other. Hence, the variance of the Laplacian can be used as a sharpness and illumination measure in the empirical quality score.

Design of the Empirical Quality Score

In order to combine the proposed pose measure and the variance of the Laplacian, they have to be normalized to a [0,1] range, since the value range for the measures differ. Min-max normalization is used for this purpose:

$$x' = \frac{x - min(x)}{max(x) - min(x)} \tag{30}$$

where x' is the normalized value and $(min(x), max(x))$ are the minimum and maximum of the value range of x. The minimum and maximum value for the pose measure are 0 and 135, respectively. This results from the fact that the maximum tolerated yaw and pitch angles are 75 and 60 for the pose detector. The minimum value for the variance of the Laplacian is set to 0, as this occurs when all pixel values are the same. The maximum value is set to the maximum variance possible for an image with pixel values in the [0,255] range:

$$\text{Var}(X) <= \frac{c^2}{4} = \frac{255^2}{4} = 16256.25 \tag{31}$$

This holds for any set of numbers X in the range [0,c], where c is a positive real number [49]. Then, the empirical quality score of image $\mathbf{Y}$ can then be written as:

$$\text{EQS}(\mathbf{Y}) = \alpha\text{P}(\mathbf{Y}) - \beta(VL(\mathbf{Y})) \tag{32}$$

where α, β are weights for the pose measure and the variance of the Laplacian, respectively. These weights will be determined through experiments and these will be described in Section 5.2.1. Low values for the empirical quality score indicate high quality, while higher values indicate lower quality. Figure 39 shows how the empirical quality score ranks the images compared to the measures alone.

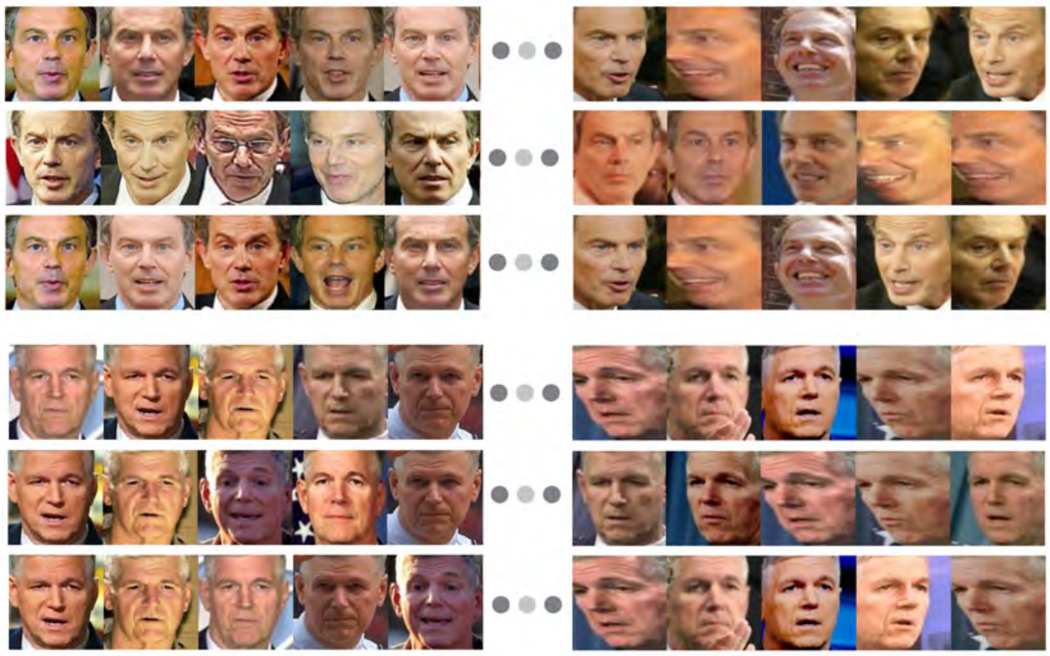


Fig. 39: The images of two persons are ranked according to three different quality measures: face pose (the first row), variance of the Laplacian (the second row), and both measures combined (the third row). In each row, the five best images are shown on the left and the worst five images are shown on the right.

It can be observed that ranking the images on the pose measure (the first row for both persons) is effective for frontal view images, but sharpness is not taken into account. Similarly, the second row shows that sharper images are preferred, however these do not necessarily have the lowest pitch and yaw angle. The combination of the two measures achieves a better result as shown in the third row. This result was achieved by setting $\alpha = 1$ and $\beta = 0.8$ in Equation 32.

4.2.2 Reference Quality Score

Instead of using hand-engineered quality measures, the reference quality score is based on the similarity between face images. As stated earlier, this approach can be divided into two stages:

1. Generation of the ground truth quality scores in order to establish a face image database with quality scores
2. Using the database to train a convolutional neural network with the quality score as target.

The resulting model can be used for automatic face image quality assessment. Both stages will be discussed in detail below.

Ground Truth Generation

Figure 40 gives an overview of how the ground truth quality scores are generated.

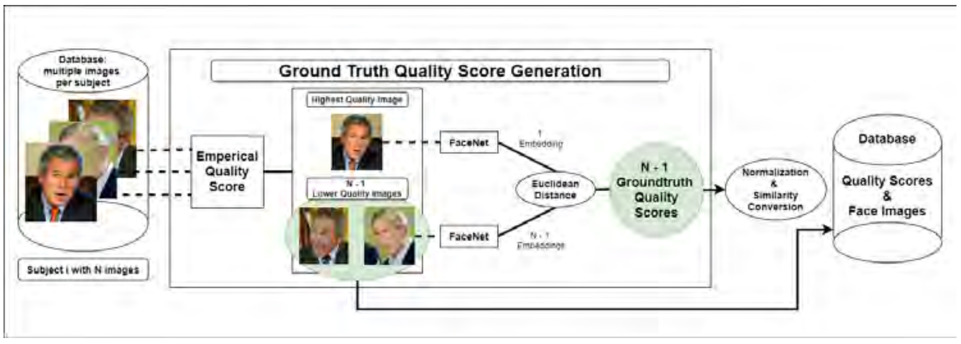

Database: multiple images per subject
Subject i with N images
Ground Truth Quality Score Generation
Emperical Quality Score
Highest Quality Image
N - 1 Lower Quality images
FaceNet
FaceNet
1 Embedding
N - 1 Embeddings
Euclidean Distance
N - 1 Groundtruth Quality Scores
Normalization & Similarity Conversion
Database
Quality Scores & Face Images

Fig. 40: The process of generating the ground truth quality scores for a dataset. Note that the $N-1$ quality scores and face images are added to the database for each subject i.

The LFW and CelebFaces datasets will be used to establish two quality-labeled face image databases. First, the empirical quality score (EQS) is used to select the gallery images, i.e., the images with the highest quality. This is done by taking the image with the lowest EQS for each subject in the dataset. Subject i has N images, so $N-1$ images are selected as probe images, i.e., the images with lower quality (higher EQS). Here, the assumption is made that an image with the lowest EQS represents good quality. Therefore, if such a gallery image is compared to a probe image of the same subject and the comparison score is high, it is safe to assume that the probe image is of good quality. On the contrary, when the comparison score is low, we can assume that the probe image is of low quality. This assumption will be further investigated through experiments that will be explained in Section 5.2.1.

The images of each subject in the dataset are assigned the gallery or probe label. The gallery image of each subject can then be compared to all of its probe images. Both Agarwal (2018) and Hernandez-Ortega et al. (2019) use FaceNet [28] to do this. FaceNet is a convolutional neural network that extracts high-level features from a face image and predicts a 128-dimensional vector representation, called a face embedding. These face embeddings can then be used to compute distances between face images as a measure of face similarity. FaceNet was trained to ensure that similar faces are closer than faces that are not alike. Hence, a large distance between a gallery image and a probe image means that the probe image is of low quality. On the other hand, a small distance means that the probe image is of good quality. The Euclidean distance is used to compare the face embeddings:

$$d(\mathbf{g},\mathbf{p}) = \sqrt{\sum_{i=1}^{n}(g_i - p_i)^2} \tag{33}$$

where $d(\mathbf{g},\mathbf{p})$ is the Euclidean distance between gallery embedding $\mathbf{g}$ and probe embedding $\mathbf{p}$. The length n corresponds to the length of the embedding, which is 128. The Euclidean distance is preferred over other distance measures, as FaceNet maps a face image to a compact Euclidean space [28]. Furthermore, the pre-trained FaceNet model based on the Inception-v3 architecture is used for generating the face embeddings. This architecture is shown in Figure 41.

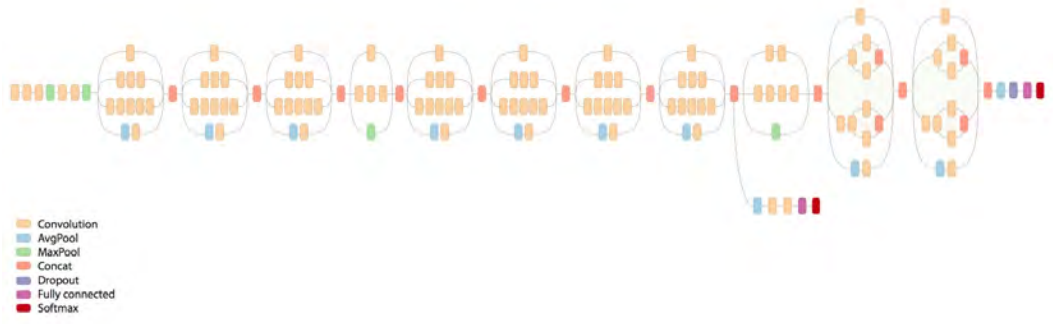


Fig. 41: The Inception-v3 architecture.

The model expects RGB color images of size 160x160x3 as input. In addition, the pixel values of the image are expected to be standardized according to Equation 18. When a face image is suitably prepared to meet the expectations of the FaceNet model, an embedding can be predicted. It is good practice to normalize the embeddings, as they are compared to each other using the Euclidean distance. Therefore, they are normalized by dividing the elements in the embedding with the Euclidean norm. The Euclidean norm of vector $\mathbf{v}$ can be written as:

$$||\mathbf{v}||_2 = \sqrt{\sum_{i=1}^{n} x_i^2} \tag{34}$$

where x_i is the i-th element in vector $\mathbf{v}$. The Euclidean distances are then computed between the normalized gallery and probe embeddings and they serve as the ground truth quality scores. Finally, when the $N-1$ ground truth quality scores of the probe images are computed for each subject i in the dataset, they are min-max normalized (Equation 30) to the [0,1] range. Note that this normalization is applied to all the ground truth quality scores of the probe images in the dataset. In addition, the normalized ground truth quality scores are converted into a similarity measure for the sake of simplicity. This done by subtracting the quality scores from one; $1 - value$. This ensures that a low quality score corresponds to a low-quality image and vice versa. In the end, two databases are built from the LFW and

CelebFaces datasets with 7487 and 140410 quality-labeled face images, respectively. Figure 42 shows how the generated ground truth quality scores rank the previously used images from the LFW dataset.

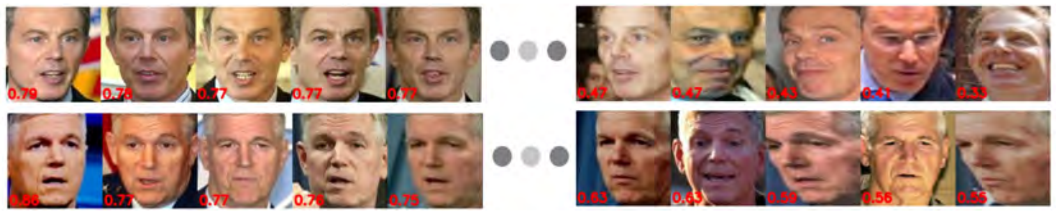


Fig. 42: The images of two persons are ranked according to the generated ground truth quality score. In each row, the five best images are shown on the left and the worst five images on the right. The score is shown in the left bottom corner.

The quality scores are added here to give an indication of how the scores relate to the images. It can be observed that the face images on the left are indeed of better quality than those on the right in terms of pose, sharpness, and illumination. Furthermore, differences in the ranking compared to the EQS are clearly visible. Experiments will show which quality measure correlates better with the performance of the facial analysis system based on [18][17].

Network Design

Now that a database is created of quality-labeled face images, a convolutional neural network can be trained to create a model that can predict the reference quality score (RQS) without the need for a reference image. Both Agarwal (2018) and Hernandez-Ortega et al. (2019) use a transfer learning approach for this, as using a pre-trained model provides a good starting point that can boost the performance by faster convergence. They customize a pre-trained model to fit the task at hand.

Agarwal (2018) uses a customized version of the pre-trained FaceNet model that is used for creating the face embeddings. A one-node fully connected layer is added at the final 128-dimensional layer of the model architecture to perform regression. A same approach will be used in this research, however, the original 128-dimensional final layer will be removed. The goal of this customized FaceNet model is not to create a face embedding, but to learn a mapping from a face image to the RQS. In transfer learning approaches it is common practice to remove the last layer for the reason mentioned. Furthermore, a sigmoid activation is used over the end node to get an output value between 0 and 1. A linear activation is used in this research, since the RQS is already in the range [0,1]. The architecture of the new FaceNet Model is shown in Figure 43.

Hernandez-Ortega et al. (2019) also use a customized pre-trained model. They selected the ResNet-50 model from [29], and removed the 8631-node classification layer of the pre-trained model. In addition, they added two fully connected layers to the new final 2048-dimensional layer. First they add a 32-dimensional fully connected layer to increase the learning ability of the network for the new task. This reduces the dimensionality from 2048 nodes to 32 nodes. Next, they add a one-node fully connected layer to perform regression. A ReLu activation is used over the first custom layer and a linear activation is used over the final layer. Finally, they only perform training on the custom layers by freezing the weights in the original layers. The idea behind freezing the weights in the original layers is that these layers are already able to extract features that are necessary for the new task, so there is no need in updating them. However, when applied in this research, freezing the weights caused learning problems, while not freezing them did not cause these problems. Therefore, this research does not freeze any weights in order to let the model adjust more to the new task at hand. In addition, the 32-dimensional fully connected layer is not used, since the network is able to learn over all layers now. The architecture of the new ResNet-50 model is shown in Figure 44.

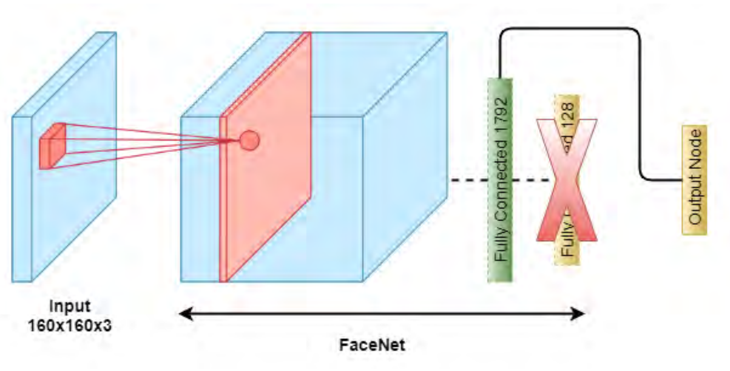


Fig. 43: The customized FaceNet model that is used in this research. The original embedding layer is removed and a one-node layer is added.

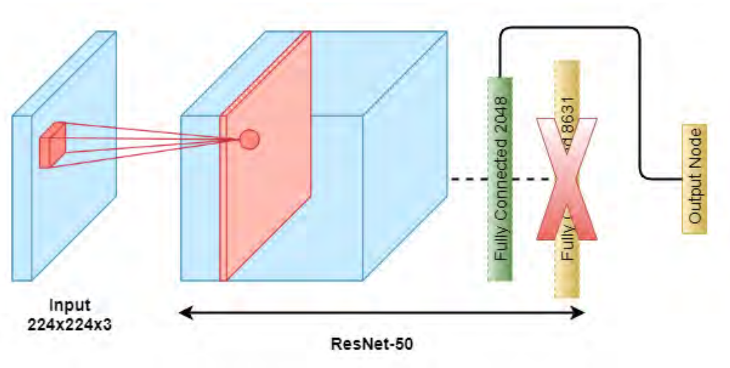


Fig. 44: The customized ResNet-50 model that is used in this research. The original classification layer is removed and a one-node layer is added.

The ResNet-50 model requires a different image size than the FaceNet model, namely a size of size 224x224x3. Both custom models will be used to predict the RQS. Section 5.2.3 gives an overview of the experiments that will be conducted with both models.

5 Experimental Setup

This section gives an overview of the experiments that are conducted for both face normalization and face quality assessment. First, experiments will be conducted with variations of the face location normalization algorithm. Evaluating these variations will give insights into the impact of face normalization on the performance of the deep learning model based on [18][16], hereby referred to as the baseline facial analysis system. Furthermore, the influence that pose variations in pitch and yaw have on this performance will be investigated. These variations are not reduced by the face location normalization algorithm, so these experiments will help to quantify this influence. Next, experiments with the empirical quality score and the reference quality score will be presented. In addition, the evaluation procedure for both quality scores will be discussed. Finally, a bias study will be performed to investigate whether the RQS models have biases towards a certain emotion, gender, age group, or ethnicity.

5.1 Face Normalization

First, a baseline performance for face normalization will be set by using the face location normalization algorithm proposed by Gudi et al. (2014). Next, multiple variations in facial landmarks, and geometric transformation will be applied to this baseline method in order to investigate the influence of these variations. An overview of the experiments that will be conducted is shown in Table 4.

Table 4: An overview of the face normalization experiments.

Experiment Nr.	**Facial Landmarks**	**Detector**	**Transformation**
1	Eye Centers	5-point detector	Similarity
2	Eye Centers, Nose	5-point detector	Affine
3	Eye Centers, Nose, Mouth Corners	68-point detector	Affine
4	Eye Centers, Chin	68-point detector	Affine
5	Eye Centers, Nose, Chin	68-point detector	Affine

The first column shows the experiment number, which will be used to refer to the variants. The facial landmarks that are used in the experiments are shown in the second column. Note that the eye centers are computed by averaging the coordinates of the eye corners. Both the 5-point and the 68-point detector contain those landmarks. Furthermore, the type of landmark detector depends on the landmarks that it can estimate. When both detectors can be used, the choice is made to use the 5-point detector, as it is a lighter and smaller model that is preferred for real-time applications. As mentioned before, the 68-point detector can also be used for real-time applications, however, it is good practice to reduce computation time when this is possible. Finally, all variants of the baseline method use an affine transformation, since the face images have to be scaled over two axes. The normalized faces will lose their original shape with an affine transformation, but as mentioned in Section 2.1.1., this might not be important when predicting facial expressions, gender, or age.

The nose, chin, and mouth corners were chosen as additional landmarks to the baseline method, because these are evident landmarks in the face. In addition, setting fixed points for these landmarks showed improvements of the problem that was depicted in Figure 4, as the face is forced to also fit the additional landmarks. This is illustrated in Figure 45 where examples of normalized images are lined up according to the variant.

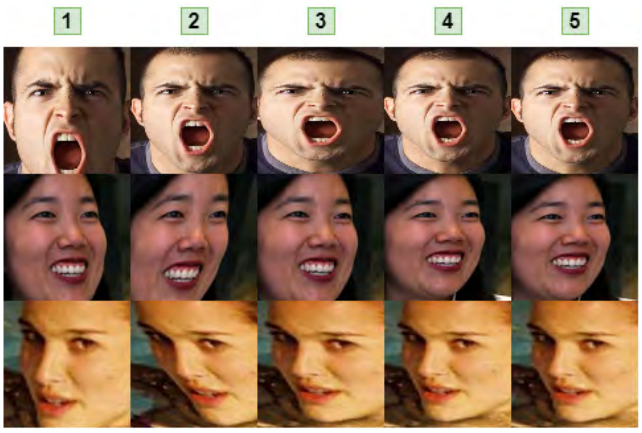


Fig. 45: The normalized face images of three persons, where each image column corresponds to a face location normalization variant. The number above links the variant to the experiment number in Table 4.

It can be observed that some faces do indeed better fit the image compared to the baseline method in column one. However, conducting the experiments will give a definite answer on whether this improves the performance of the system. The formulas that were used to set fixed points for the nose, chin and mouth corners are given below. The formulas for the eye centers were already given in Equation 20.

$$\begin{aligned} x_n &= \frac{1}{2} * w, & y_n &= \frac{15}{24} * h \\ x_c &= \frac{1}{2} * w, & y_c &= \frac{23}{24} * h \\ x_{ml} &= \frac{1}{3} * w, & y_{ml} &= \frac{17}{24} * h \\ x_{mr} &= \frac{2}{3} * w, & y_{mr} &= \frac{17}{24} * h \end{aligned} \tag{35}$$

where x_n, x_c, x_{ml}, x_{mr} are the x coordinates for the nose, chin, left mouth corner, and right mouth corner, respectively. The same holds for the y coordinates and w, h refer to the width and height of the normalized image.

5.1.1 Evaluation

The baseline facial analysis system will be used to evaluate the five experiments that are shown in Table 4. The system uses a deep learning model that will be trained on a combination of the VV dataset, the UTKFace dataset, and the MORPH academic dataset. These datasets are prepared for the model by applying the face normalization variants on them. When the model is trained on each prepared dataset, it is evaluated on a 10% test partition of this dataset. As stated in Section 3.1, this test set is doubled in size by flipping the face images horizontally, which increases the robustness of the results. Furthermore, this research considers the performance of the model on its ability to predict emotion, gender, and age. Here, both the accuracy and the F1-score will be taken as main performance measures, where the accuracy is leading. Due to class imbalances in some targets, it can be helpful to also look at the F1-score. Figure 24 in Section 3.1 gives an overview of the class distributions within the four targets. Finally, a comparison between the best performing variant and the baseline method will be made. All these results will be presented in Section 6.1.

5.1.2 Pose Error Analysis

The head pose detector that was discussed in Section 4.1.4 is used to enrich the dataset on which the baseline facial analysis system is trained with information about the pitch and yaw angle for each face image in it. The resulting information is used to perform an error analysis on the model. Here, the performance of the model will be analyzed for certain yaw and pitch angles. This makes it possible to quantify the impact of variations in these pose angles. Section 6.1.1 will give an overview of this impact.

5.2 Face Quality Assessment

This subsection will first discuss experiments that are conducted to determine the weights within the empirical quality score. This is important, because the EQS is a combination of multiple face quality measures. It is not clear how the components of the EQS correlate with the performance of the baseline facial analysis system, so evaluating the components separately will clarify this. Next, the training setups of the reference quality score models will be discussed. Then, the evaluation procedure for both the EQS and the RQS is described. Finally, the bias study will be explained, which investigates whether the EQS and the RQS have any biases.

5.2.1 Empirical Quality Score

The EQS consists of a pose measure and a sharpness/illumination measure, which are given again for readability:

$$\mathrm{EQS}(\mathbf{Y}) = \alpha \mathrm{P}(\mathbf{Y}) - \beta(VL(\mathbf{Y})) \tag{36}$$

where the pose measure is given by:

$$P(\mathbf{Y}) = w_p|\mathrm{pitch}^\circ| + w_y|\mathrm{yaw}^\circ| \tag{37}$$

Here, w_p and w_y can be determined with the help of the pose error analysis described in Section 5.1.2. This analysis will give insights into the performance of the baseline facial analysis system within certain pitch and yaw angles. Hence, the results will show which angle has a bigger impact or whether this impact is roughly the same.

The weights α and β will be determined by first evaluating the pose measure and the sharpness/illumination measure separately. This evaluation procedure is described in Section 5.2.3. The separate results will indicate which measure correlates better with the performance of the baseline facial analysis system. This information can be used to set initial values for α and β. After this, a more local search can be applied to the weight values by experimenting with multiple weight settings in the neighborhood of the initial values.

Finally, when the best weights have been found, the EQS will be evaluated with those weights. This will show whether the EQS is an estimator of the performance of the baseline facial analysis system. Also, this will show whether it is justified to use the EQS for selecting the gallery images within the RQS method.

5.2.2 Reference Quality Score

Both the custom FaceNet model and the custom ResNet-50 model will be trained to predict the RQS. They will be trained on the quality-labeled LFW dataset and the quality-labeled CelebFaces dataset separately. This will clarify which custom model is able to learn the RQS better and which dataset represents the RQS better. Also, it gives insights into what the impact of the size of the dataset is, as the CelebFaces dataset is significantly larger. Table 5 summarizes these experiments.

Table 5: An overview of the RQS experiments.

Experiment Nr.	Model	Dataset
6	Custom FaceNet	LFW
7	Custom FaceNet	CelebFaces
8	Custom ResNet-50	LFW
9	Custom ResNet-50	CelebFaces

The experiments in Table 5 will all use the following hyperparameter settings:

Table 6: The hyperparameter settings for the RQS experiments.

Batch Size	Learning Rate	LR Patience	LR Reduce Factor
64	0.0005	30	0.1
ES Patience	**Optimizer**	**Loss Function**	**Max Epochs**
40	Adam	Huber Loss	1000

The values for the batch size and the learning rate are set through experimentation. When the validation loss has not decreased for 30 epochs, the learning rate is decreased with a factor 0.1. The 'LR Patience' and 'LR Reduce Factor' parameter refer to this and this is used to further minimize the loss function. This concept is illustrated in Figure 46:

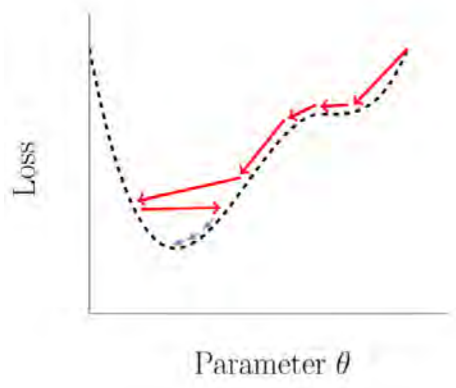


Fig. 46: The process of iteratively minimizing the loss function. The red arrows correspond to the initial learning rate and the blue arrows to a reduced learning rate.

The learning rate is reduced for a maximum of 2 times, which helps to speed up the convergence. In addition, early stopping is performed after two learning rate reductions with a patience of 40 epochs. The 'ES Patience' parameter refers to this. Furthermore, the Adam optimizer is used, as convergence is reached faster compared to other optimizers. Also, as we are considering a regression problem, the Huber loss is taken as loss function. This loss function is a combination between the Mean Squared Error and the Mean Absolute Error (Equation 9). It is basically the MAE, which becomes the MSE when the error is small. How small that error has to be depends on the hyperparameter δ, which is set to 0.2. This means that when the difference between predicted value and actual value is smaller than 0.2, the MSE is taken, while the MAE is taken for larger values. This is done to make the model robust to outliers, while keeping the advantages of the MSE for smaller errors. Finally, the training process is allowed to run for a maximum of 1000 epochs, which is a broad margin.

5.2.3 Evaluation

As a face quality score is being estimated, a different evaluation procedure has to be used, since the estimations can not be compared to ground truth values. These ground truth values are simply not available. With the RQS, ground truth quality scores are generated, however, it can not be assumed that these correlate to the performance of a facial analysis system, as this is being investigated. Hence, this research follows a similar evaluation procedure as Agarwal (2018) and Hernandez-Ortega et al. (2019). Here, the quality score is predicted for face images of a dataset that is used for face recognition purposes. Next, they test whether the performance on the face recognition task increases as they remove the lowest quality images from the test set.

This research already uses the baseline facial analysis system for the evaluation of the face normalization experiments. Here, the performance of the system is considered for emotion, gender, and age. The performance is evaluated on a 10% test partition of the combination of three datasets. In order to evaluate the EQS and the RQS, they are both used to predict the quality of this same test set. As a result, the test set will be enriched with a quality score for each face image. Next, we observe what happens with the performance of the system on predicting emotion, gender, and age, when the lowest quality images are removed from the test set. If this performance increases, it shows that the used quality score does correlate to the performance of the baseline facial analysis system. Figure 47 illustrated this procedure for clarity.

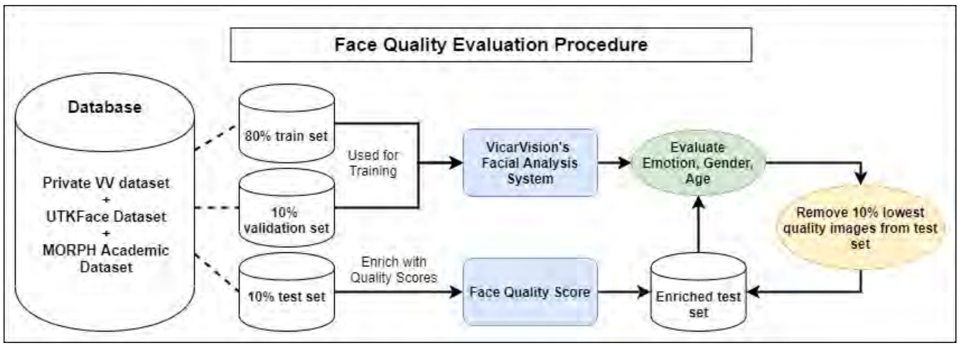

Face Quality Evaluation Procedure
Database
Private VV dataset
+
UTKFace Dataset
+
MORPH Academic Dataset
80% train set
10% validation set
10% test set
Used for Training
Enrich with Quality Scores
VicarVision's Facial Analysis System
Face Quality Score
Evaluate Emotion, Gender, Age
Enriched test set
Remove 10% lowest quality images from test set

Fig. 47: The process of evaluating the quality scores (EQS/RQS) that are proposed in this research.

This evaluation procedure will be used to evaluate the EQS and the RQS quality models. In addition, it will be used to evaluate the separate components of the EQS. Section 6.2 will present all these results.

5.2.4 Bias Study

The evaluation procedure that is described above gives insights into whether the proposed quality scores are estimators of the performance of a facial analysis system. The results might indicate that this is true or false, however, these results give no indication about potential biases that may be present in the quality scores. In general, a bias is a phenomenon that occurs when a model produces results that are systematically prejudiced due to inaccurate assumptions of the model. For example, when the EQS or RQS predicts a low quality score for a face image, because of its race, facial expression, age group, or gender. It could occur that the quality score labels East Asian people as low quality while the facial analysis system is just less accurate in predicting this ethnicity group. Hence, to be sure that the quality models score face images as low quality, because it has low quality, a bias study will be performed. The bias study will be performed by observing the class distributions of the lowest quality faces for each target.

6 Results

This section presents the results of the experiments that are explained in the previous section. First, the results of the experiments concerning face normalization will be shown. This will give insights into how face normalization influences the performance of a facial analysis system. Furthermore, the results of the pose error analysis will be shown. Next, the results of the experiments concerning the empirical quality score and the reference quality score will be presented. In addition, the results of the bias study will be provided.

6.1 Face Normalization

Figure 48 shows the performance of the baseline facial analysis system on emotion-, gender-, and age classification for the five experiments of Table 4.

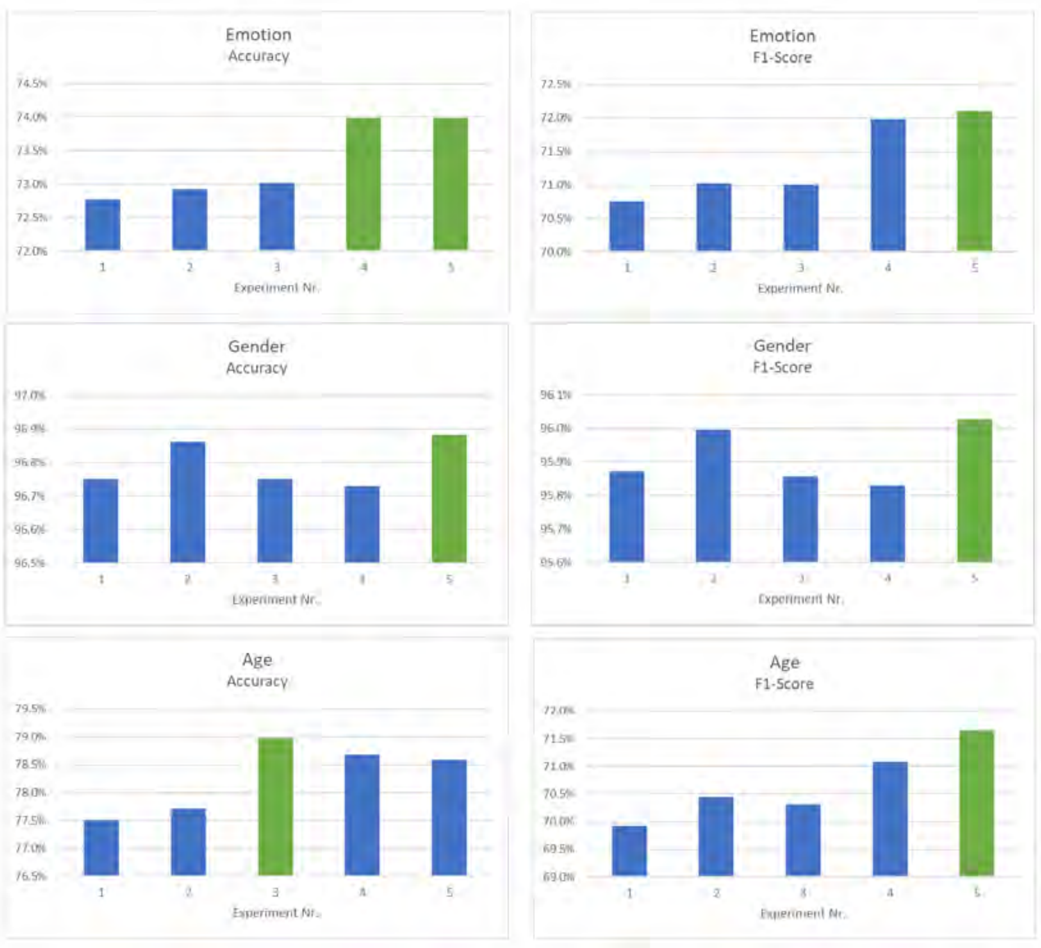


Fig. 48: The performance of the baseline facial analysis system on emotion-, gender-, and age-classification. The accuracy is shown on the left and the F1-score on the right. The best performing face normalization method is marked as a green bar for each graph.

First, it can be observed that experiment 1, which is the baseline face normalization algorithm proposed by Gudi et al. (2014), is outperformed in emotion-, gender, and age classification. Especially, face normalization method 5, where the eye centers, nose, and chin are taken, outperforms the baseline method for all three targets. In addition, method 5 outperforms all other variants, except for predicting age in terms of accuracy. However, it performs best in terms of the F1-Score, which is slightly more indicative of the performance on age, since the distribution of classes within the age data is somewhat imbalanced. This can be observed in Figure 24. As face normalization method 5 performs best for all three targets, it will be used to normalize the LFW and CelebFaces datasets. These datasets are used to train the RQS models on and using the best face normalization technique should enhance their performance. Table 7 gives an overview of the performance gains by using face normalization method 5 compared to the baseline method.

Table 7: The performance gains of using the face normalization method of experiment 5 compared to the baseline method.

	Emotion	**Gender**	**Age**
Accuracy	1.21%	0.13%	1.08%
F1-Score	1.35%	0.15%	1.73%

It can be observed that the improvement of the face normalization algorithm proposed by Gudi et al. (2014) has a significant performance gain on emotion-, gender-, and age classification. Note that this performance gain is the result from only applying a different face normalization method to the dataset on which the system is trained. The training process has been executed in the same manner for all five experiments.

6.1.1 Pose Error Analysis

First, a visual inspection is performed to investigate the impact of variations in pitch and yaw angles. Figure 49 illustrates this impact for emotion.

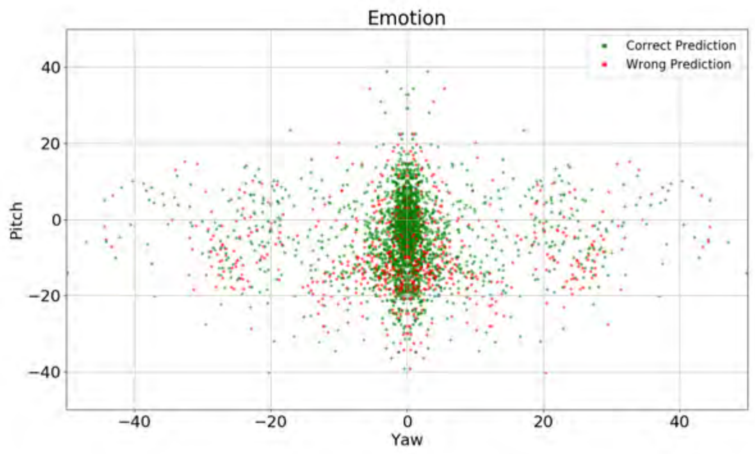
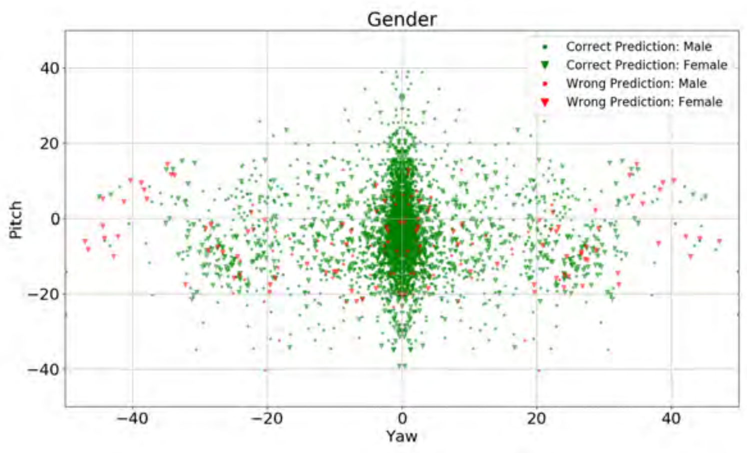


Fig. 49: Correct and wrong emotion classifications of the baseline system for variations in pitch and yaw. The x-axis represents the yaw angle, where a positive/negative yaw corresponds to a face that is looking to the right/left (POV of viewer). The y-axis represents the pitch angle, where a positive/negative pitch corresponds to a face that is looking up/down.

It can be observed that there are more green dots in the middle than there are red dots. In other words, the proportion of correct predictions to wrong predictions is in favor of the correct predictions for low pitch and yaw angles. This proportion gets more equal as the pitch and yaw angles increase, since it seems there is an equal proportion of green to red dots outside the middle. This means that the baseline system makes more mistakes for increasing pitch and yaw angles. Figure 51 shows similar graphs, but now for gender- and age classification.



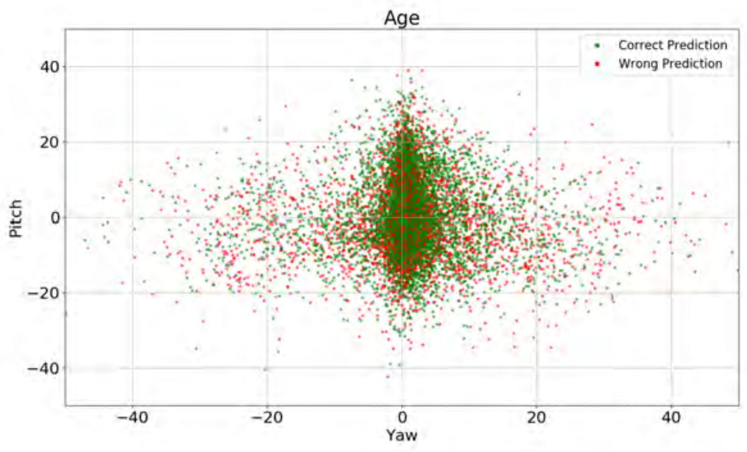
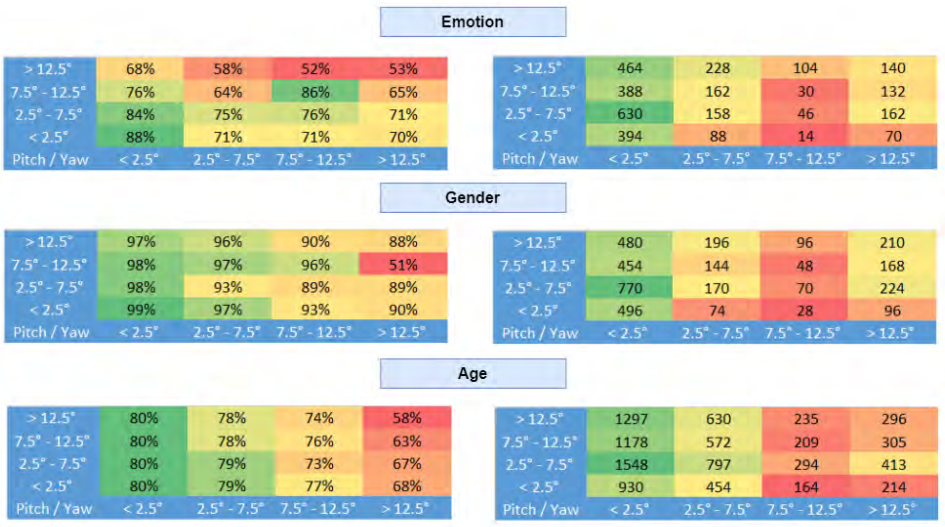


Fig. 51: Top graph: correct and wrong gender classifications of the baseline system for variations in pitch and yaw. Each prediction also shows whether the system predicted a male or female. Bottom graph: Correct and wrong age classifications of the baseline system for variations in pitch and yaw. The x-axis represents the yaw angle, where a positive/negative yaw corresponds to a face that is looking to the right/left (POV of viewer). The y-axis represents the pitch angle, where a positive/negative pitch corresponds to a face that is looking up/down.

Similar behavior can be observed in Figure 51. Again, the proportion of correct to wrong predictions equalizes more as the pitch and yaw angles increase. In addition, for gender classification, it seems that the system misclassifies males mainly as females when there is a big yaw angle in the face in. This indicates a bias where faces with large yaw angles are recognized as females, while they are males. This might be a general bias for facial analysis systems or this is something specific for the baseline system.

Finally, to confirm the observations described above, a quantitative evaluation is performed. The quantitative results of the pose error analysis for emotion, gender, and age are presented below. Note that these results are shown for absolute pitch and yaw values. This is done in order to obtain reliable values, as taking real values would result in too few samples within the angle intervals. This also seems as a safe assumption to make, because the figures above do not indicate that a positive or negative yaw have a different impact on the number of misclassifications. The same holds for the pitch angle



Emotion

Pitch / Yaw	< 2.5°	2.5° - 7.5°	7.5° - 12.5°	> 12.5°
> 12.5°	68%	58%	52%	53%
7.5° - 12.5°	76%	64%	86%	65%
2.5° - 7.5°	84%	75%	76%	71%
< 2.5°	88%	71%	71%	70%

Pitch / Yaw	< 2.5°	2.5° - 7.5°	7.5° - 12.5°	> 12.5°
> 12.5°	464	228	104	140
7.5° - 12.5°	388	162	30	132
2.5° - 7.5°	630	158	46	162
< 2.5°	394	88	14	70

Gender

Pitch / Yaw	< 2.5°	2.5° - 7.5°	7.5° - 12.5°	> 12.5°
> 12.5°	97%	96%	90%	88%
7.5° - 12.5°	98%	97%	96%	51%
2.5° - 7.5°	98%	93%	89%	89%
< 2.5°	99%	97%	93%	90%

Pitch / Yaw	< 2.5°	2.5° - 7.5°	7.5° - 12.5°	> 12.5°
> 12.5°	480	196	96	210
7.5° - 12.5°	454	144	48	168
2.5° - 7.5°	770	170	70	224
< 2.5°	496	74	28	96

Age

Pitch / Yaw	< 2.5°	2.5° - 7.5°	7.5° - 12.5°	> 12.5°
> 12.5°	80%	78%	74%	58%
7.5° - 12.5°	80%	78%	76%	63%
2.5° - 7.5°	80%	79%	73%	67%
< 2.5°	80%	79%	77%	68%

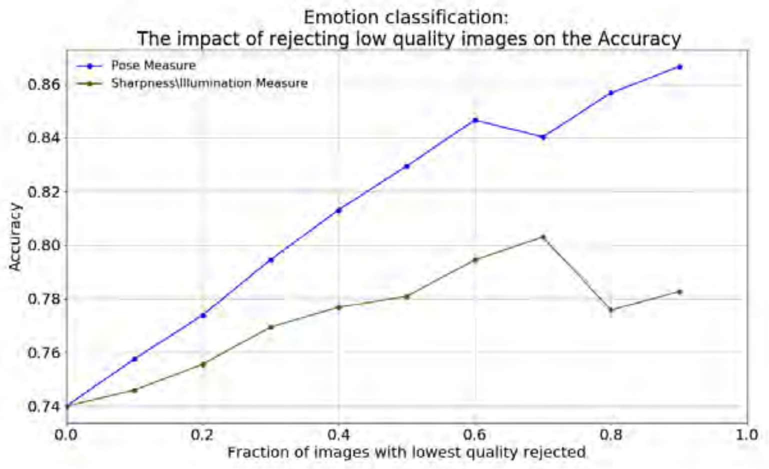
Pitch / Yaw	< 2.5°	2.5° - 7.5°	7.5° - 12.5°	> 12.5°
> 12.5°	1297	630	235	296
7.5° - 12.5°	1178	572	209	305
2.5° - 7.5°	1548	797	294	413
< 2.5°	930	454	164	214

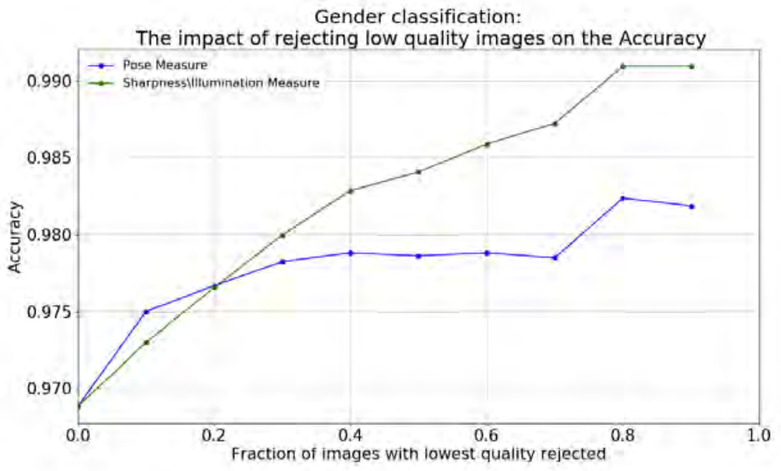
Fig. 52: On the left, the accuracy of the baseline facial analysis system within specific pitch and yaw angles is shown for emotion-, gender-, and age classification. The corresponding number of face images within those yaw and pitch angles are shown on the right. Face normalization method 5 was used to normalize the face images of these results. Note that absolute values for the pitch and yaw angles are taken.

It can be observed that the accuracy decreases as the yaw and pitch angles increase, which is what the literature suggests [14][8]. However, compared to the literature, these results show the exact impact that variations in pose have on the performance. For example, it can be observed that the accuracy for emotion decreases with 35% for pitch and yaw angles above 12.5° compared to faces with a pitch and yaw angle below 2.5°. This tells us that the impact of variations in pose on the performance of a facial analysis system is quite significant. Furthermore, it can be observed that the accuracy decreases more in the yaw direction than the pitch direction for gender and age. For emotion, this seems to be roughly the same, so it not clear which angle has a bigger impact. Also, it can be observed that there are more samples in the pitch direction than the yaw direction. Hence, the baseline facial analysis system is also trained on more samples with pitch variation than yaw variations. This might cause the model to classify face images with variations in pitch better than face images with variations in yaw, as it simply had more training data for variations in pitch. For both these reasons, it can not be concluded whether the pitch or the yaw angle have a bigger impact on the performance. Hence, the weights w_p and w_y will be kept equal for the pose measure (Equation 37) within the EQS (Equation 36). Finally, bigger pitch and yaw angles are not considered, because the number of samples within those angles were too small.

6.2 Face Quality Assessment

The results for the empirical quality score and the reference quality score will be presented in this subsection. First, the results of the experiments around the EQS will be provided. In order to find the weights α and β in Equation 36, the pose measure and the sharpness/illumination measure are evaluated separately. This will show how the measures correlate to the performance of the baseline facial analysis system. Figure 54 shows the results for emotion-, gender-, and age classification





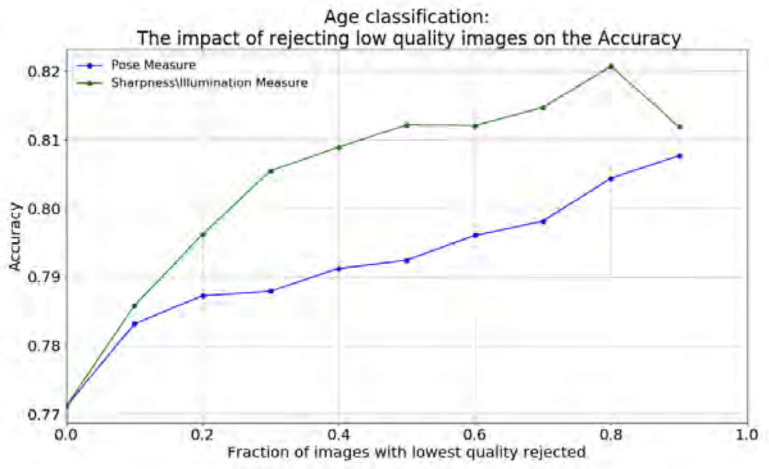


Fig. 54: The graphs show the performance of the baseline system on emotion-, gender-, and age classification for different fractions of the test set. The data is ranked according to the pose measure (blue line) and according to the sharpness/illumination measure (green line). The x-axis shows the fraction of data in the test set that is rejected based on the corresponding measure and the accuracy is shown on the y-axis. Note that the most left data point represents the accuracy on the full test set. Also, one data point to the right represents the accuracy of the test set when 10% of the lowest quality data is rejected.

Figure 54 shows that, in general, the accuracy increases as more ranked test data gets rejected based on the quality measures. This indicates that both measures are correlated to the performance of the baseline system for emotion-, gender-, and age classification. Furthermore, for emotion classification, it can be observed that rejecting 90% of the test data based on the pose measure and the sharpness/illumination measure increases the accuracy by roughly 12% and 4%, respectively. Note that rejecting 90% of the images is a realistic scenario for real-time applications as a common camera films with a rate of 25 images per second. In addition, it can be observed that the pose measure correlates considerably better with the performance for emotion than the sharpness/illumination measure. The other graphs about gender- and age classification show that the sharpness/illumination measure correlates better with that performance than the pose measure. For gender classification, the graph shows that rejecting 90% of the test data increases the accuracy by roughly 1% and 2% for the pose measure and the sharpness/illumination measure, respectively. Finally, for age classification, rejecting 90% of the test data increases the accuracy by roughly 4% for both measures.

These results confirm the visual inspections of Section 4.2.1 and show that both measures of the EQS are estimators of the performance of the baseline facial analysis system. This research aims to create a general quality measure, hence the weights of the EQS have to be chosen based on the general performance on emotion-, gender-, and age classification. The figures show that the pose measure is considerably better correlated to the performance on emotion classification. However, the sharpness/illumination measure is better correlated to the performance on gender- and age classification. As they are both important, the following weight settings for (α, β) will be considered: $(1, 0.5), (1, 1), (1, 1.5)$. The result for emotion classification is shown in Figure 55:

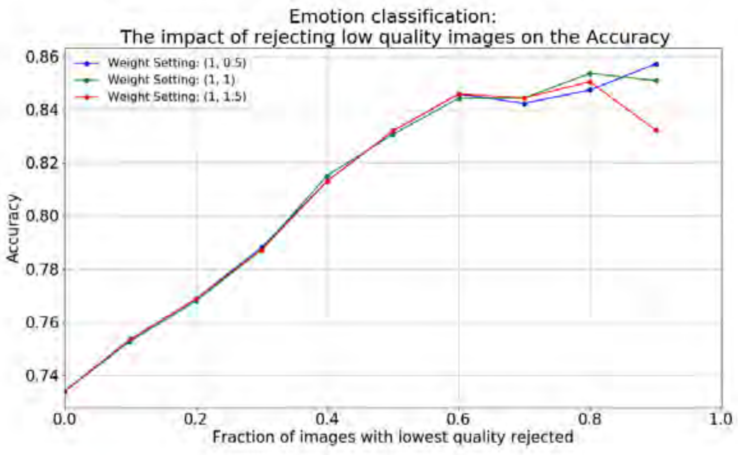
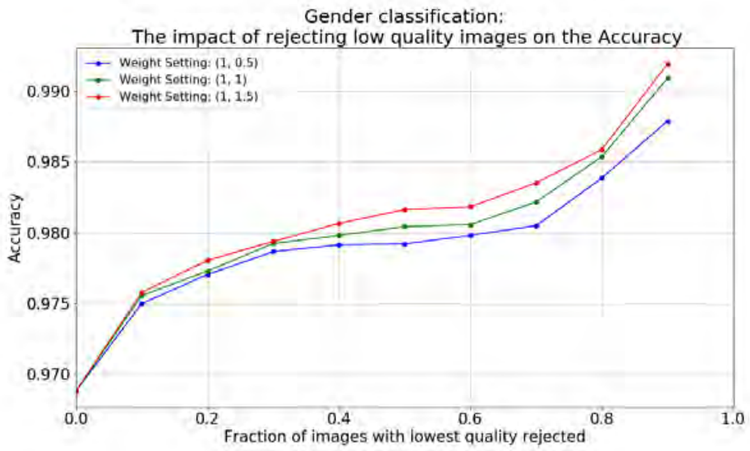


Fig. 55: This graph shows the accuracy of the baseline system on emotion classification for different fractions of the test set. The data is ranked according to three weight settings for the EQS. The x-axis shows the fraction of data in the test set that is rejected based on the EQS and the accuracy is shown on the y-axis. Note that the most left data point represents the accuracy on the full test set. Also, one data point to the right represents the accuracy of the test set when 10% of the lowest quality data is rejected.

First, it can be observed that the proposed EQS is correlated to the performance of the baseline facial analysis system for all three weight settings. Furthermore, the EQS with weights setting $(1, 0.5)$ achieves the highest accuracy when 90% of the test data is rejected. However, the differences between the three weight settings are small. Figure 57 shows the result for gender- and age classification.



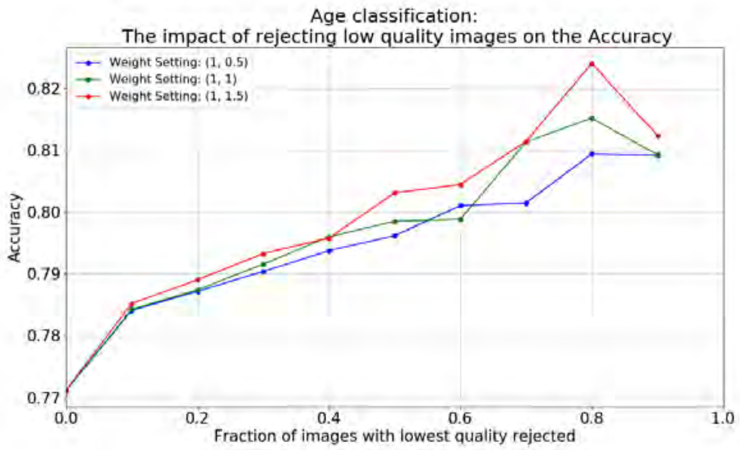


Fig. 57: The graphs show the performance of the baseline system on gender- and age classification for different fractions of the test set. The data is ranked according to three weight settings for the EQS. The x-axis shows the fraction of data in the test set that is rejected based on the EQS and the accuracy is shown on the y-axis. Note that the most left data point represents the accuracy on the full test set. Also, one data point to the right represents the accuracy of the test set when 10% of the lowest quality data is rejected.

The figure shows that the EQS also correlates to the performance on gender- and age classification. Furthermore, the differences are again quite small, but the EQS with weights $(1, 1.5)$ achieves the highest accuracy for both targets. This is also expected, as the sharpness/illumination measure has more weight in this setting.

The three figures above show a similar trade-off in performance between the pose measure and the sharpness/illumination measure as in Figure 54. Here, more weight on the pose measure results in a slightly better performance for emotion classification, while more weight on the sharpness/illumination measure results in a slightly better performance for gender and age classification. As the differences between the weight settings in all three figures are small and both components of the EQS seem to correlate equally (on average) to the performance, they will be treated with equal weight. Hence, the final EQS equation becomes:

$$\mathrm{EQS}(\mathbf{Y}) = \mathrm{P}(\mathbf{Y}) - VL(\mathbf{Y}) \tag{38}$$

The results in this section have also shown that it is justified to use the EQS for selecting the gallery images within the RQS method.

Next, the results for the experiments that were described in Table 5 about the RQS will be presented. The four RQS models that have been created will be evaluated in the same manner as the EQS. First, the results for emotion classification are shown in Figure 58.

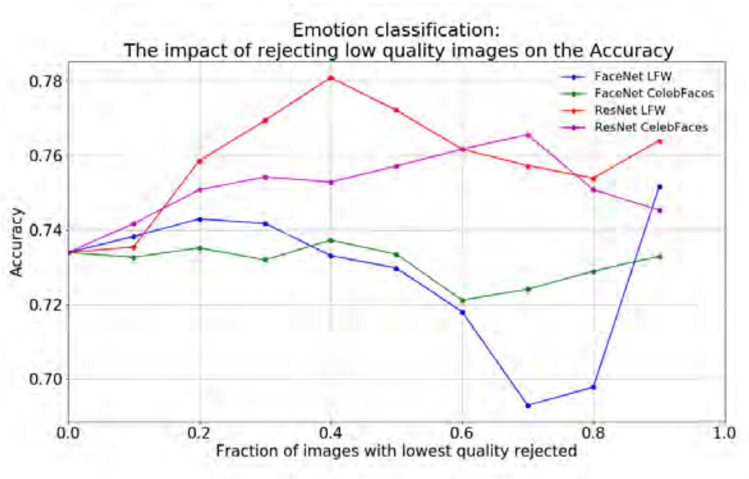


Fig. 58: This graph shows the accuracy of the baseline system on emotion classification for different fractions of the test set. The data is ranked according to four RQS models. The x-axis shows the fraction of data in the test set that is rejected based on the model predictions and the accuracy is shown on the y-axis. Note that the most left data point represents the accuracy on the full test set. Also, one data point to the right represents the accuracy of the test set when 10% of the lowest quality data is rejected.

The figures show that the accuracy does not increase as more ranked test data is rejected based on all four models. Hence, they are all not correlated to the performance of the baseline facial analysis system on emotion classification. This may be caused by the fact that the reference quality scores on which the custom FaceNet and ResNet models are trained, are generated by using the original FaceNet model. FaceNet is originally designed for face recognition purposes and therefore focuses on identities. As a consequence, a larger Euclidean distance can exists between two face embeddings of one identity where one of them contains a not neutral facial expression compared to two face embeddings containing neutral expressions. Figure 59 shows that this is indeed true.

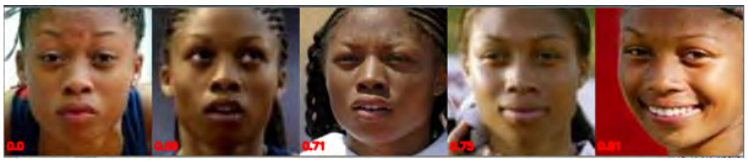
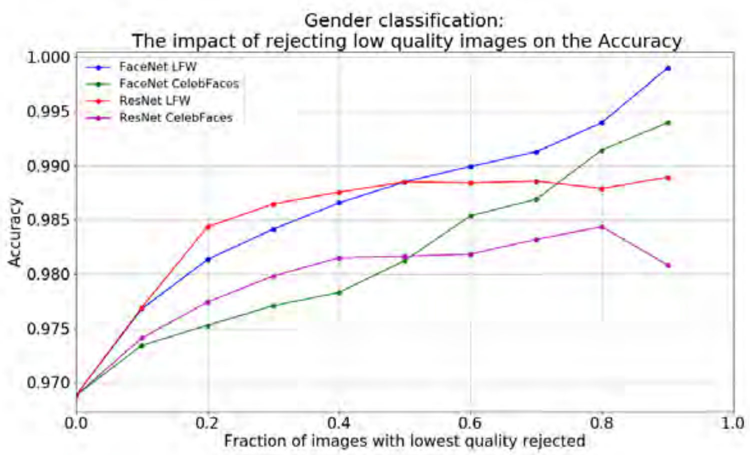


Fig. 59: Face images ranked on the Euclidean distance (left corner) to the most left face image.

The most right face image contains the only face with a not neutral facial expression and seems of good quality, however, the Euclidean distance is the largest. This may have caused emotive faces to receive a higher Euclidean distance compared to neutral faces and therefore a lower reference quality score, which indicates low quality. This seems to explain why the predicted reference quality scores do not correlate well with the performance of the baseline facial analysis system on emotion classification, however this should be examined more thoroughly. Next, the results for gender-, and age classification are shown in Figure 61



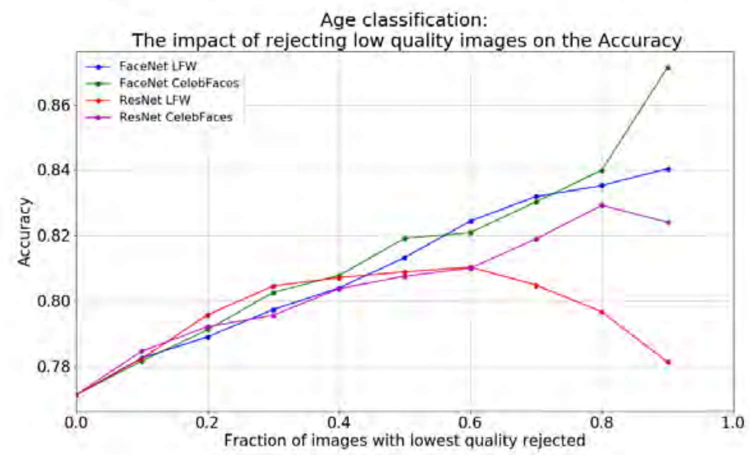


Fig. 61: The graphs show the performance of the baseline system on gender- and age classification for different fractions of the test set. The data is ranked according to four RQS models. The x-axis shows the fraction of data in the test set that is rejected based on the model predictions and the accuracy is shown on the y-axis. Note that the most left data point represents the accuracy on the full test set. Also, one data point to the right represents the accuracy of the test set when 10% of the lowest quality data is rejected.

Both FaceNet models show clear monotonically increasing lines, which indicates that both models correlate well with the performance on gender- and age classification. On the other hand, both custom ResNet models correlate less well to the performance compared to the FaceNet models, as they show drops in accuracy when rejecting more ranked test data. For gender classification, these are small drop, but for age classification these are bigger drops. Also, in general, the lines are below the lines of the FaceNet models, which indicates that they correlate worse to the performance on gender- and age classification. Furthermore, the custom FaceNet model that is trained on the LFW dataset outperforms the custom FaceNet model that is trained on the CelebFaces dataset for gender classification. It is the other way around for age classification at 90% test data rejected, however, at the other fractions, both FaceNet models show similar correlation. From these three figures, one can conclude that the custom FaceNet models outperform the custom ResNet models. In addition, on average, the FaceNet models seem to perform equally. In other words, the effect of the dataset seems to average out over gender and age classification.

Finally, a comparison between the EQS and the best RQS models is shown in Figure 63.

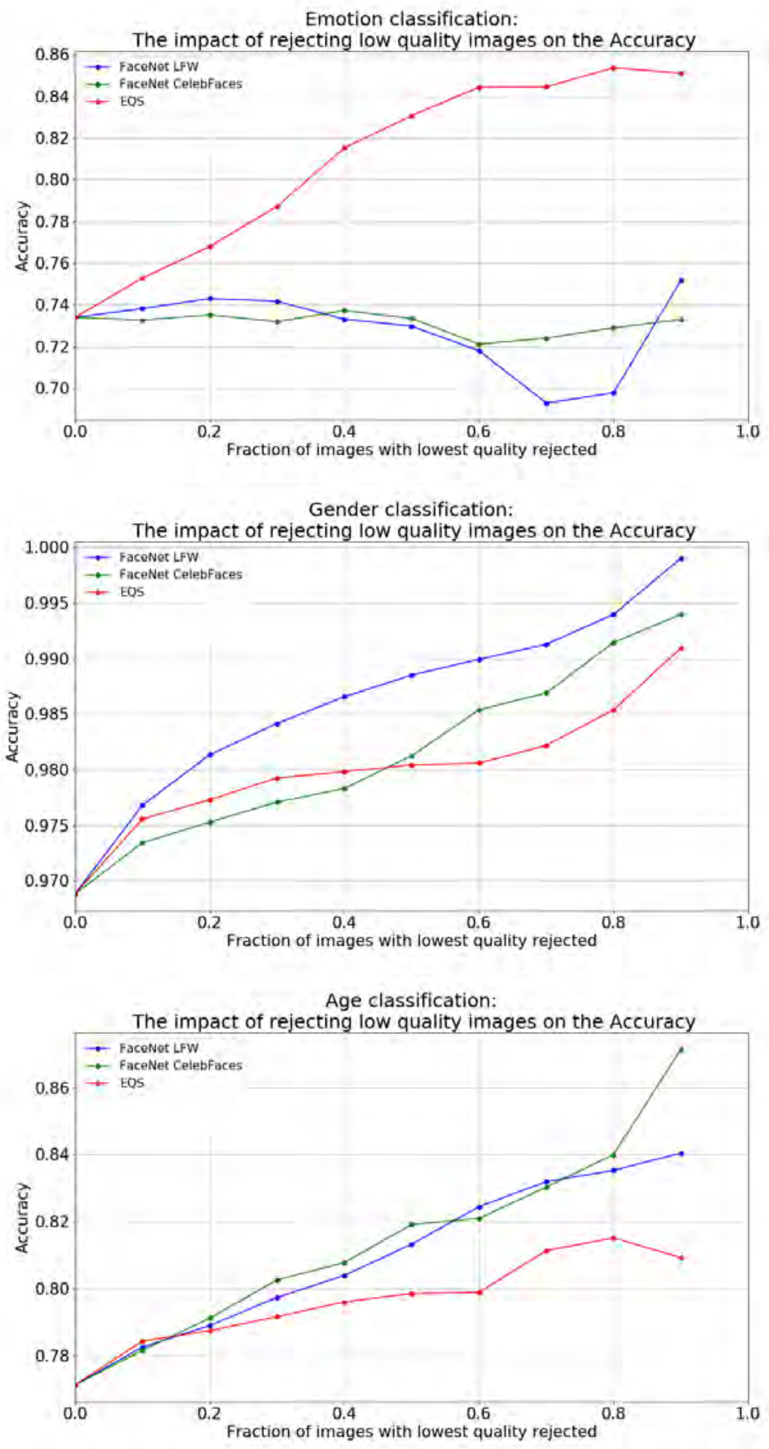


Fig. 63: The graphs show the accuracy of the baseline system on emotion-, gender, and age classification for different fractions of the test set. The data is ranked according to the custom FaceNet models and the EQS. The x-axis shows the fraction of data in the test set that is rejected based on the corresponding measure and the accuracy is shown on the y-axis. Note that the most left data point represents the accuracy on the full test set. Also, one data point to the right represents the accuracy of the test set when 10% of the lowest quality data is rejected.

Clearly, the EQS correlates well with performance on emotion classification, while both custom FaceNet models do not correlate at all with this performance. However, both RQS models correlate better with the performance of the baseline facial analysis system for gender and age classification than the EQS.

6.2.1 Bias Study

The bias study will be performed on the EQS and the RQS models, since these have shown to be estimators of the performance of the baseline system. As the predictions of both RQS quality models are not correlated to the system's performance on emotion classification, it makes no sense to check for biases within this target. However, the EQS does correlate well with emotion classification, so biases within this target will be investigated for the EQS.

In order to check for biases, the distribution of rejected images will be visualized and compared to the distribution of the full test set. This is done for emotion, gender, age, and ethnicity and shows what classes the quality model is rejecting within the target. Note that the RQS models do not consider emotion as mentioned above. This can show whether the EQS and the RQS models have biases towards certain classes. First, we will examine the distributions of the rejected data for the RQS models. Figure 64 illustrated this for the custom FaceNet model that is trained on the LFW dataset.

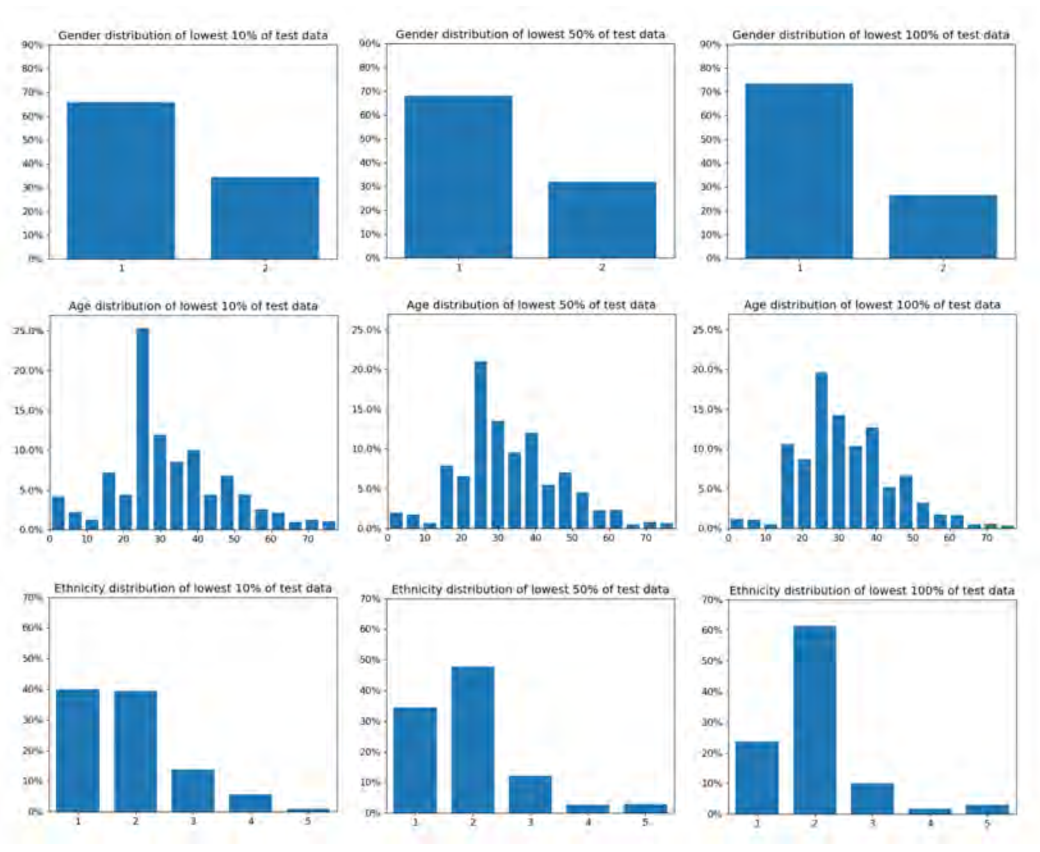


Fig. 64: The first row shows the gender distribution of classes within the 10%, 50%, and 100% lowest quality test data according to the quality scores predicted by the custom FaceNet model (LFW) (1=male, 2=female). Note that the 100% lowest quality test data is the full test set. The second row shows the age distribution for the same subsets of the test set. The third row shows the same for the ethnicity distribution (1=Caucasian, 2=African, 3=East Asian, 4=South Asian, 5=Others)

The first row of histograms shows that the gender distribution of the lowest 10% and lowest 50% test data are almost identical to the gender distribution in the full test set. This indicates that the custom FaceNet model (LFW) does not predict low quality for a specific gender, as a similar gender distribution as the full test set is rejected. In other words, these histograms indicate that this quality model does not have a bias towards a specific gender. Furthermore, the second row of histograms shows that the age distribution of the lowest 10% of test data contains more face images within the [23-27] and [0-2] age groups than the age distribution of the full test set. This is also the case for the age distribution of the lowest 50% of test data, but somewhat less. This suggests that there might be a bias towards these age groups by the quality model, as these are rejected proportionally more compared to the full distribution. Finally, the third row shows that the ethnicity distribution of the lowest 10% of test data contains considerably more Caucasian face images than the ethnicity distribution of the full test set. This suggests that this quality model has a bias towards face images with a Caucasian race. Note that these results are indications and do not provide evidence of potential biases. Figure 65 shows the same histograms for the custom FaceNet model that is trained on the CelebFaces dataset.

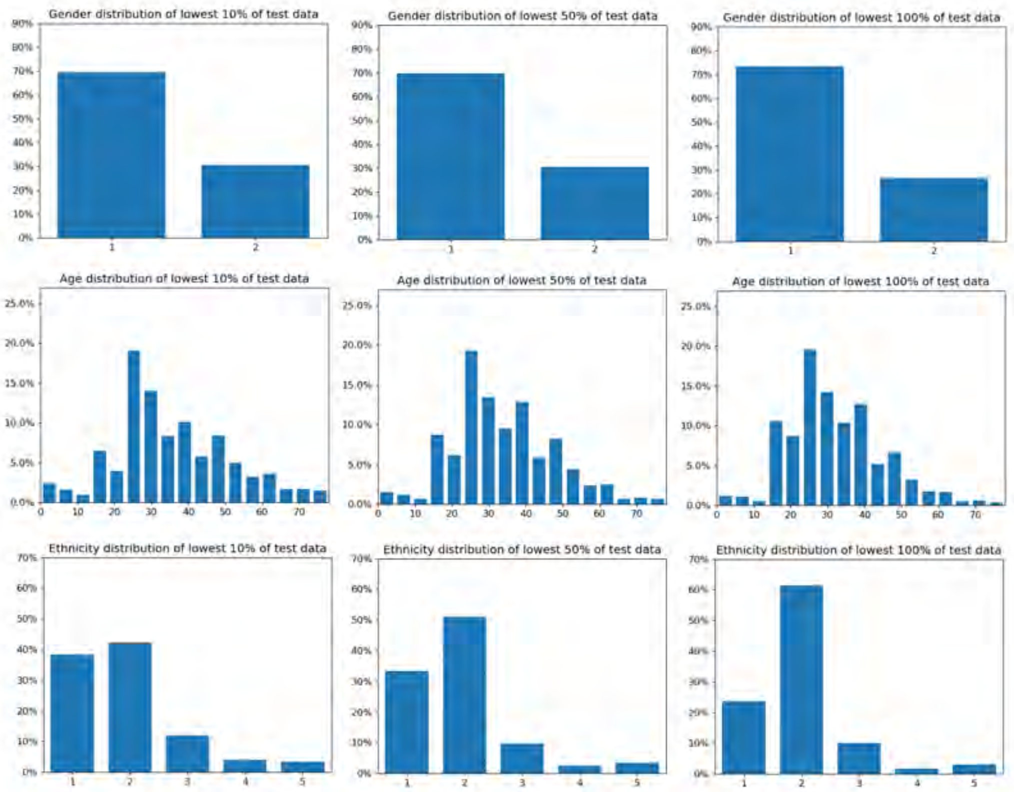


Fig. 65: The first row shows the gender distribution of classes within the 10%, 50%, and 100% lowest quality test data according to the quality scores predicted by the custom FaceNet model (CelebFaces) (1=male, 2=female). Note that the 100% lowest quality test data is the full test set. The second row shows the age distribution for the same subsets of the test set. The third row shows the same for the ethnicity distribution (1=Caucasian, 2=African, 3=East Asian, 4=South Asian, 5=Others)

It can be observed that the gender distribution of the lowest 10% and 50% of test data are again almost identical to the gender distribution in the full test set. The same holds for the age distributions of those subsets, however, there are some slight differences in the older age groups. Furthermore, a similar bias can be observed for this custom FaceNet model, namely a bias towards face images within the Caucasian class. Again, these results are indications and they have to be further investigated to provide evidence. Finally, the same investigation is performed for the EQS, but now with the emotion target. Figure 66 illustrates this.

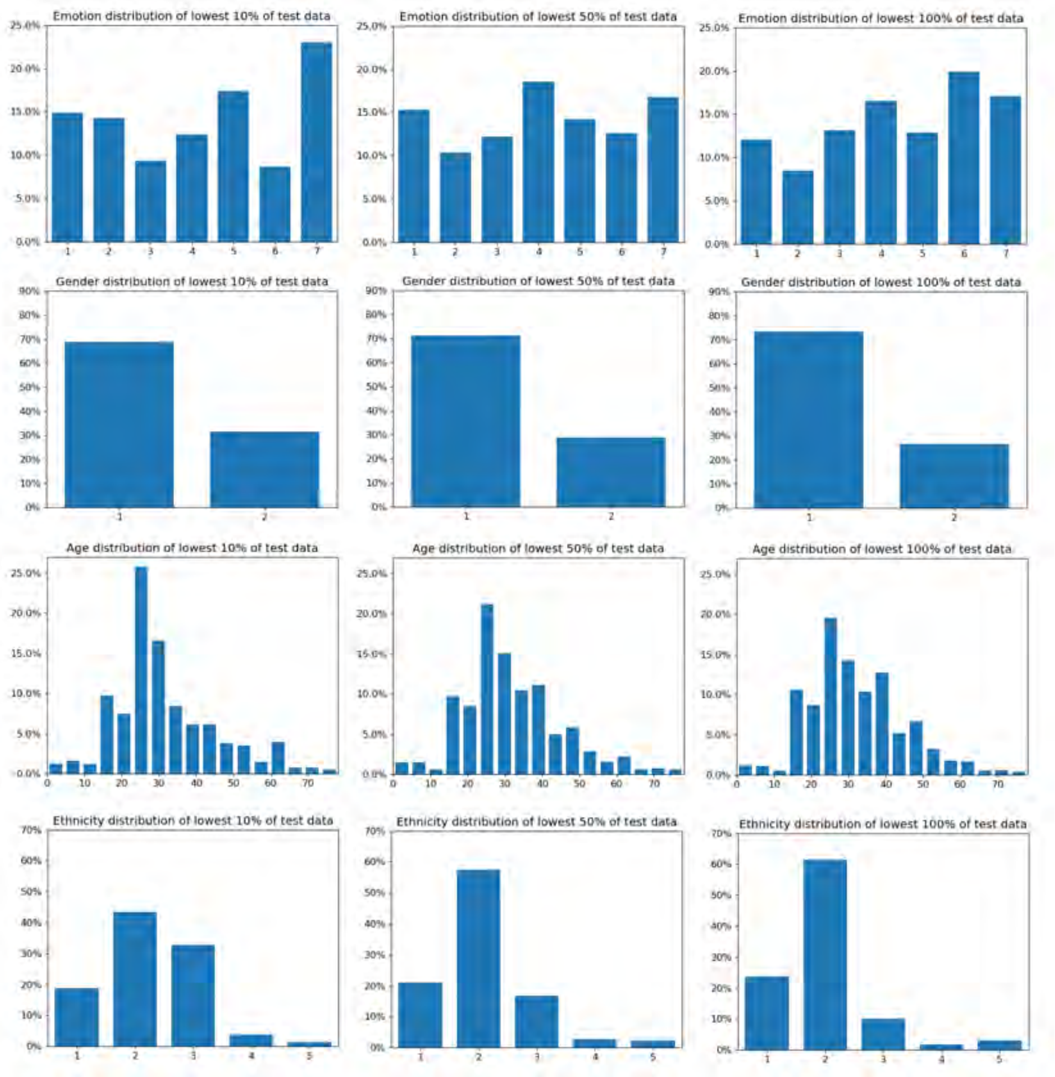


Fig. 66: The first row shows the emotion distribution of classes within the 10%, 50%, and 100% lowest quality test data according to the quality scores estimated by the EQS (1=angry, 2=disgust, 3=fear, 4=happy, 5=sad, 6=surprised, 7=neutral). Note that the 100% lowest quality test data is the full test set. The second row shows the gender distribution of classes for the same subsets (1=male, 2=female). The third row shows the age distribution for the same subsets of the test set. Finally, the fourth row shows the same for the ethnicity distribution (1=Caucasian, 2=African, 3=East Asian, 4=South Asian, 5=Others)

It can be observed in the first row of histograms that the emotion distribution of the lowest 10% of test data contains more face images that express the emotions disgust (2), sad (5), and neutral (7) than the emotion distribution in the full test set. The same can be observed for the emotion distribution of the lowest 50% of test data, but to a lesser extent. This indicates a potential bias to the mentioned emotions. The second row shows that the gender distribution of the lowest 10% and 50% are very similar to the gender distribution in the full test set. Hence, a bias towards a certain gender is not present here. Furthermore, in the third row of histograms, an indication of a bias can be observed within the [23-27] age group, as face images within these groups are rejected proportionally more in the lowest 10% of test data compared to the full test set. Finally, the fourth row of histograms shows that East Asian face images are present in greater numbers within the lowest 10 % of test data than in the full test set. This indicates a bias towards face images with an East Asian race.

7 Conclusion

The main goal of this research was to investigate methods to assess the quality of a face image. This research proposed an empirical quality measure and a deep learning approach to perform face quality assessment. Aside from that, face normalization methods were investigated in order to observe its influence on the performance of a facial analysis system. These research goals were summarized into the following research questions:

How can one assess face image quality in order to predict the suitability of a face image for face analysis purposes?

Furthermore, how does face normalization influence the performance of facial analysis systems?

These research questions will be answered in the same order as was done in all other section, so a conclusion will first be drawn from all results around face normalization. The results showed that improving the face location normalization algorithm proposed by Gudi et al. (2014) positively influenced the performance of the baseline facial analysis system on emotion-, gender-, and age classification. More specifically, the F1-Score on emotion-, gender-, and age classification increased with 1.35%, 0.15%, and 1.73%, respectively. It is important to note that this performance gain was achieved by only applying a different face location normalization algorithm to the dataset on which the baseline system is trained and tested. Hence, this research shows how face normalization can have a significant influence on the performance of a facial analysis system.

As the face location normalization algorithm does not remove pose variations in pitch and yaw, it was found interesting to investigate the impact of these variations on the performance of a facial analysis system. Also, multiple studies claim that these variations have a significant impact on this performance, however, they do not quantify this impact. Therefore, this research showed exact performance metrics within specific pitch and yaw angles for emotion-, gender-, and age classification. These results show that the performance decreases substantially when the pitch and yaw angles increase. Pitch and yaw angles above a 12.5° angle could decrease the accuracy with 35%, 11%, and 22% for emotion-, gender-, and age classification, respectively. This tells us that it is important to include the removal of pitch and yaw angles in the face normalization technique. Recommendations about this are given in Section 8.

Next, a conclusion will be drawn upon the results around face quality assessment. Two approaches were considered to assess the quality of a face image. First, this research proposes a new face quality measure, which empirically uses a pose measure and a sharpness and illumination measure. This face quality measure is referred to as the empirical quality score and the results show that this measure correlates well with the performance of the baseline system on emotion-, gender-, and age classification. In other words, it was shown that the empirical quality score is able to predict the suitability of a face image for face analysis purposes. Furthermore, the results showed that rejecting 90% of the lowest quality face images according to the empirical quality score could increase the performance of the baseline system with roughly 11%, 2%, and, 4% for emotion-, gender-, and age classification, respectively. Aside from that, this is a lightweight quality measure that can be easily implemented in real-time applications. The computation time is roughly equal to the the computation time of the head pose estimation network that is used, because the computation time of the Laplacian operation is negligible next to the pose estimation. An average of 15 milliseconds was measured on the machine used for this research, which has an CPU with the following properties: Intel Core i7-2600 CPU @ 3.40GHz, 3401 Mhz, 4 Core(s), 8 Logical Processor(s).

Finally, this research investigated whether the deep learning approach proposed in [10], [22], and [11] could also be used for emotion-, gender-, and age classification. The results show that both custom FaceNet models which were trained on the LFW dataset and the CelebFaces dataset, respectively, correlate well with the performance of the baseline system on gender- and age classification. However, they do not correlate with the performance of the baseline system on emotion classification. The reason for this might lie in the fact that the reference quality scores are generated by using FaceNet, but this should be examined more thoroughly. Furthermore, the results show that rejecting 90% of the lowest quality face images according to the predictions of both models could increase the performance of the baseline system with roughly 3% and 7% for gender- and age classification, respectively. Hence, the results show that the predictions of the custom FaceNet models correlate better for gender- and age classification than the empirical quality score.

To conclude, this research proposes an empirical quality score that can predict the suitability of a face image for emotion-, gender-, and age classification. Furthermore, the deep learning approach proposed in this research shows superior results for predicting the suitability of face image for gender- and age classification. For the implementation of one of the quality prediction methods, one should make a trade-off between computation time and performance, as the deep learning approach is more computationally expensive. Section 8 gives some recommendations about how to decrease that computation time.

8 Discussion

Although this research shows that the face quality assessment methods and the face normalization methods are effective for their purposes, there are still some improvements possible. This section discusses the limitations of the proposed methods. In addition, further improvements that could be made are recommended for future research.

Computation time

The decision was made to only consider computationally cheap face normalization methods in this research, because the actual face analysis is already quite computationally expensive. Furthermore, this research showed that both face quality assessment methods can be used to reject low-quality face images. This way, facial analysis systems can only do an analysis on face images of good quality. As a result, the computation time for face analysis decreases as more low-quality face images are rejected. For example, when a system only analyzes 10% of the input images of a camera, the computation time decreases with 90%. This reduction in computation time can be used to apply more computationally expensive face normalization methods, such as those discussed in Section 2.1.1. These normalization methods also include face frontalization in their methods, which removes variations in pitch and yaw of the face. This can further improve the performance of facial analysis systems, as was shown in the pose error analysis in Section 6.1.1. Note that some of these frontalization methods will also remove the expression information, as they synthesize a neutral frontal face. Hence, it should be considered for which target they are used.

Additional reductions in computation time can be achieved when using a deep learning approach to face quality assessment. The custom FaceNet models that were constructed are still quite computationally expensive for predicting face quality. The computation time can be reduced by using lighter architectures and it should be investigated whether these architectures are also able to learn the RQS. A lighter network can be build from scratch or an existing lighter network can be used to do this. Finally, knowledge distillation can be applied to a custom FaceNet model to create a lighter version of it. Knowledge distillation is a process where a smaller model learns from the predictions of a bigger model. An example of this is given by Sanh et al. (2019), where a lighter version of the language model BERT is created [50].

A general or specific method

Another point of interest is whether to use one face normalization method when predicting all targets (emotion, gender, and age) or to use a face normalization method for each target separately. In other words, it might be better to use a different face normalization method when the facial analysis system is predicting one's emotion than when it is predicting one's gender. After all, determining one's gender usually requires to see one's hair, while determining one's emotion does not require this. This research considered one face normalization method for all targets as network architecture of the baseline system allows it to predict all targets at once. As a consequence, the system uses one dataset for training, so only one face normalization method can be used. Furthermore, the results showed that one face normalization method was best for all targets, so this suggests that it was indeed better to use one face normalization method when predicting all targets. However, when using other face normalization methods or when predicting other targets than considered in this research, this might be different. Hence, future research should consider whether to use a general face normalization method for all targets or one that might be more tailored to each specific target that is predicted.

The same question should be asked for the empirical quality score, because this research aimed to create a general quality measure. It was already shown that some weight settings were better for emotion classification and some weight setting were better for gender- and age classification. Therefore, it should be considered how to choose the weights of the empirical quality score and whether to use one weight setting for all targets.

Deep learning approach with emotion prediction

The results showed that the predictions of the custom FaceNet models did not correlate with the performance of the baseline system on emotion classification. As mentioned before, this might lie in the fact that the reference quality scores are generated with FaceNet. However, this is an indication, and therefore it should be examined more thoroughly. In addition, it is important to examine what the cause of this is, as the results show that the deep learning approach is superior to the empirical quality score on gender- and age classification. Hence, this deep learning approach to face quality assessment might be superior for all targets.

Pitch and yaw variations

Future research should also better investigate the impact of pose variations in pitch an yaw on the performance of facial analysis systems. This research quantified this impact and gave relevant insights into the impact. However, it did not become clear whether the pitch angle or the yaw angle has a bigger influence on the system's performance. Also, bigger pitch and yaw angles than 12.5° were not considered, as the number of samples within those angles were too small. Hence, more data is needed to better investigate the influence of variations in pitch and yaw. A dataset that can be used for this purpose, is the the M2FPA pose dataset [51]. This dataset contains 397,544 images of 229 subjects with 62 poses, which includes 13 yaw angles, 5 pitch angles and 44 yaw-pitch angles, so enough data is present. Another advantage of this dataset is that all images already contain information about the pitch and yaw angle, so a head pose estimation network is not needed to generate the ground truth values.

Bias study

A final point of interest is to better investigate the bias indications that were shown in Section 6.2.1. It should be examined whether the same biases occur when the empirical quality score and the custom FaceNet models are applied to another system's test set. This will give answers to whether the biases were specific to the test partition of the VV dataset or whether they occur again. When the biases occur towards the same classes within a target, this indicates that there are indeed biases in the quality measures. Then, future research should further examine what causes these biases to better understand them.

References

1. R. Szeliski. *Computer Vision*. Springer, London, 2011.
2. Lei Ba J. Kingma, D.P. Adam: A method for stochastic optimization. 15, 2015.
3. C.M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.
4. `https://www.uni-regensburg.de/Fakultaeten/phil_Fak_II/Psychologie/Psy_II/beautycheck/english/durchschnittsgesichter/durchschnittsgesichter.htm`.
5. Bergstra J. de Leeuw, K. *The History of Information Security: A Comprehensive Handbook*. pp. 264–265. Elsevier, Amsterdam, 2007.
6. University Of Southern California. Mugspot can find a face in the crowd – face-recognition software prepares to go to work in the streets. *ScienceDaily*, 7, 1997.
7. M. W. Pontin. Better face-recognition software. *MIT Technology Review*, 2007.
8. Abdel-Mottaleb M. Alhalabi W. Haghighat, M. Fully automatic face normalization and single sample face recognition in unconstrained environments. *Elsevier*, 12, 2016.
9. `http://www.vicarvision.com/`.
10. V. Agarwal. Deep face quality assessment. *CoRR*, 6, 2018.
11. Jain A. K. Best-Rowden, L. Automatic face image quality prediction. *CoRR*, 2017.
12. Cohn J. F. Kanade T. Saragih J. Ambadar Z. Matthews I. Lucey, P. The extended cohn-kanade dataset (ck+): A complete dataset for action unit and emotion-specified expression. *CVPR*, 8, 2010.
13. Yu X. Sohn K. Liu X. Chandraker M. Yin, X. Towards large-pose face frontalization in the wild. *CoRR*, abs/1704.06244, 2017.
14. Luo P. Wang X. Tang X. Zhu, Z. Recover canonical-view faces in the wild with deep neural networks. *CoRR*, abs/1404.3543, 2014.
15. Cai X. Li X. Liu Y. Li, G. An efficient face normalization algorithm based on eyes detection. *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 6, 2006.
16. Welling M. Tasli H.E. Gudi, A. Recognizing semantic features in faces using deep learning. 68, 2014.
17. Emrah Tasli H. den Uyl T. M. Maroulis A. Gudi, A. Deep learning based facs action unit occurrence and intensity estimation. 5, 2015.
18. A. Gudi. Recognizing semantic features in faces using deep learning. 9, 2016.
19. Ma L. Fan H. Li, Y. and K. Mitchell. Feature-preserving detailed 3d face reconstruction from a single image. *CVMP '18*, 10, 2018.
20. Li S.Z. Liu R. Zhang P. Gao, X. Standardization of face image sample quality. *in Proc. Int. Conf. Biometrics*, 11, 2007.
21. Zhiguang Yang, Haizhou Ai, Bo Wu, Shihong Lao, and Lianhong Cai. Face pose estimation and its application in video shot selection. 1, 2004.
22. Galbally J. Fierrez J. Haraksim R. Beslay L. Hernandez-Ortega, J. Faceqnet: Quality assessment for face recognition based on deep learning. *CoRR*, 8, 2019.
23. Beveridge J. R. Bolme D. S. Draper B. A. Givens G. H. Lui Y. M. Cheng S. Teli M. N. Phillips, P. J. and Zhang H. On the existence of face quality measures. *IEEE Conf. on Biometrics: Theory, Applications and Systems*, pages 1–8, 2013.
24. Tabassi E. Grother, P. Performance of biometric quality measures. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 29(4):531–543, 2007.
25. Vatsa M. Singh R. Bharadwaj, S. Can holistic representations be used for face biometric quality assessment. *IEEE International Conf. on Image Processing*, 2013.
26. G. Choudhury T. Bansal R., Raj. Blur image detection using Laplacian operator and open-cv. pages 63–67, 01 2016.
27. Ramesh M. Berg T. Learned-Miller E. Huang, G.B. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical Report 07-49, University of Massachusetts, Amherst, October 2007.
28. Kalenichenko D. Philbin J. Schroff, F. Facenet: A unified embedding for face recognition and clustering. *CoRR*, abs/1503.03832, 2015.
29. Zhang X. Ren S. Sun-J. He, K. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.
30. F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 1958.
31. Y. Bengio. Learning deep architectures for ai. *Foundations*, 2:1–55, 01 2009.

32. . Bordes A. Bengio Y. Glorot, X. Deep sparse rectifier neural networks. In Geoffrey Gordon, David Dunson, and Miroslav Dudík, editors, *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pages 315–323, Fort Lauderdale, FL, USA, 11–13 Apr 2011. PMLR.
33. Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Comput.*, 1(4):541–551, December 1989.
34. Müller A. Behnke S. Scherer, D. Evaluation of pooling operations in convolutional architectures for object recognition. *20th International Conference on Artificial Neural Networks (ICANN)*, 10, 2010.
35. Deng J. Su H. Krause-J. Satheesh S. Ma S. Huang Z. Karpathy A. Khosla A. Bernstein M. Berg A.C. Fei-Fei L. Russakovsky, O. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.
36. Hinton G.E. Williams R.J. Rumelhart, D.E. The back-propagation algorithm. 1986.
37. Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014.
38. Song Yang Zhang, Zhifei and Hairong Qi. Age progression/regression by conditional adversarial autoencoder. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2017.
39. Karl Ricanek and T. Tesafaye. Morph: A longitudinal image database of normal adult age-progression. volume 2006, pages 341 – 345, 05 2006.
40. Jones M.J. Viola, P. Robust real-time face detection. *IJCV*, 19, 2004.
41. Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.
42. `https://docs.openvinotoolkit.org/latest/index.html`.
43. Moskewicz M. Ashraf K. Han-S. Dally W. Keutzer K. Iandola, F.N. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and textless1mb model size. 02 2016.
44. Luo P. Loy C.C. Tang-X. Yang, S. Wider face: A face detection benchmark. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
45. A. Rosebrock. (faster) facial landmark detector with dlib. *https://www.pyimagesearch.com/2018/04/02/faster-facial-landmark-detector-with-dlib/*, 2018.
46. Guo J. Yuxiang Z. Yu-J. Kotsia I. Zafeiriou S. Deng, J. Retinaface: Single-stage dense face localisation in the wild. In *arxiv*, 2019.
47. Dantone M. Gall J. Fossati-A. Van Gool L. Fanelli, G. Random forests for real time 3d face analysis. *Int. J. Comput. Vision*, 101(3):437–458, February 2013.
48. Burt P. Hingorani R. Bergen-J. Peleg, S. A three-frame algorithm for estimating two-component image motion. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 29(09):886–896, sep 1992.
49. `https://math.stackexchange.com/questions/83046/maximum-of-the-variance-function-for-given-set-of-bounded-numbers/83144#83144`.
50. Debut L. Chaumond J. Wolf-T. Sanh, V. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *ArXiv*, abs/1910.01108, 2019.
51. Wu X. Hu Y. He-R. Sun Z. Li, P. M2fpa: A multi-yaw multi-pitch high-quality dataset and benchmark for facial pose analysis. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.