Czech Technical University in Prague
Faculty of Nuclear Sciences and Physical Engineering

Comparison of methods for unconstrained face detection

Porovnání metod pro detekci obličejů bez omezujících podmínek

Research project

Author:	**Bc. Soňa Drocárová**
Supervisor:	**Ing. Adam Novozámský, Ph.D.**
Consultant:	**Ing. Jitka Kostková, Ph.D.**

Academic year: 2021/2022

České vysoké učení technické v Praze, Fakulta jaderná a fyzikálně inženýrská

Katedra: matematiky Akademický rok: 2021/2022

ZADÁNÍ VÝZKUMNÉHO ÚKOLU

Student:	Bc. Soňa Drocárová
Studijní program:	Aplikované matematicko-stochastické metody
Název práce (česky):	Porovnání metod pro detekci obličejů bez omezujících podmínek
Název práce (anglicky):	Comparison of methods for unconstrained face detection

Pokyny pro vypracování:

1) Seznamte se s problematikou detekce obličeje pomocí moderních technik rozpoznávání obrazu. Proveďte rešerši jednotlivých přístupů a vyberte několik metod, které budete dále v práci porovnávat.
2) Vyhledejte volně dostupné datasety tváří, které se v literatuře používají při porovnávání jednotlivých algoritmů a vytvořte svůj vlastní dataset z přidělených dat.
3) U metod vybraných v bodě 1. prostudujte jejich chování na jednotlivých datasetech při různém nastavení parametrů. Promyslete také možnosti filtrování jejich výstupů za účelem snížení falešných detekcí.
4) Proveďte rešerši metodik vyhodnocování kvality detekce obličeje a na jejím základě proveďte porovnání jednotlivých vybraných metod.

Doporučená literatura:

1) R. C. Gonzalez, R. E. Woods, Digital Image Processing (4th ed.). Pearson, 2018.

2) I. Goodfellow, Y. Bengio, A. Courville, Deep learning. MIT press, 2016.

3) L. Li, X. Mu, S. Li and H. Peng, "A Review of Face Recognition Technology," in IEEE Access, vol. 8, pp. 139110-139120, 2020, doi: 10.1109/ACCESS.2020.3011028.

4) P. Borisagar, S. Jani, Y. Agrawal and R. Parekh, "An Efficient and Compact Review of Face Recognition Techniques," 2020 IEEE International Students' Conference on Electrical,Electronics and Computer Science (SCEECS), 2020, pp. 1-5, doi: 10.1109/SCEECS48394.2020.143.

5) A. Kumar, A. Kaur, M. Kumar, Face detection techniques: a review. Artif Intell Rev 52, 927–948 (2019). https://doi.org/10.1007/s10462-018-9650-2

6) M.O. Oloyede, G.P. Hancke, H.C. Myburgh, A review on face recognition systems: recent approaches and challenges. Multimed Tools Appl 79, 27891–27922 (2020). https://doi.org/10.1007/s11042-020-09261-2

Jméno a pracoviště vedoucího výzkumného úkolu:

Ing. Adam Novozámský, Ph.D.
Ústav teorie informace a automatizace, Pod Vodárenskou věží 4, 182 00, Praha 8

Jméno a pracoviště konzultanta:

Ing. Jitka Kostková, Ph.D.
Ústav teorie informace a automatizace, Pod Vodárenskou věží 4, 182 00, Praha 8

Datum zadání výzkumného úkolu: 31.10.2021

Datum odevzdání výzkumného úkolu: 15.5.2022

Doba platnosti zadání je dva roky od data zadání.

V Praze dne 25. října 2021

[illegible]

vedoucí katedry

Acknowledgment:
I would like to thank my supervisor Ing. Adam Novozámský, Ph.D. for his guidance and helpful advice as well as the consultant Ing. Jitka Kostková, Ph.D. for her notes and suggestions in writting this thesis. I am also thankful to my boyfriend for his language correction assistance.

Author's declaration:
I declare that this Bachelor's Degree Project is entirely my own work and I have listed all the used sources in the bibliography.

Prague, July 7, 2022 Bc. Soňa Drocárová

Název práce:

Porovnání metod pro detekci obličejů bez omezujících podmínek

Autor: Bc. Soňa Drocárová

Obor: Aplikované matematicko-stochastické metody

Druh práce: Výzkumný úkol

Vedoucí práce: Ing. Adam Novozámský, Ph.D., Ústav teorie informace a automatizace Pod Vodárenskou věží 4 182 00, Praha 8

Konzultant: Ing. Jitka Kostková, Ph.D., Ústav teorie informace a automatizace Pod Vodárenskou věží 4 182 00, Praha 8

Abstrakt: Táto práca sa zaoberá detekciou tváre bez obmedzijúcich podmienok. Jej úlohou je popísať metódy detekcie, zvoliť vhodné z nich ako aj datasety pre testovanie týchto algoritmov. Testovanie prebieha na dvoch voľne dostupných kolekciách obrázkov (WFLW, CelebA) a na jednom vlastnom datasete vytvoreného pre účely tejto práce označovaním prielených dát z Viedenskej knižnice. Schopnosť algoritmov fungovať v menej ideálnych podmienkach bola overená aj na podskupine súborov WFLW a CelebA, ktoré boli osadené v zťažujúcich podmienkach.

Klíčová slova: datasety pre detekciu tváre, detekcia tváre, obmedzujúce podmienky, orientačné body tváre, strojové učenie

Title:

Comparison of methods for unconstrained face detection

Author: Bc. Soňa Drocárová

Abstract: This thesis deals with face detection under unconstrained conditions. It aims to describe detection methods and select the appropriate approaches as well as datasets for testing them. The testing is done with the help of two publicly available collections (WFLW, CelebA) and one set created for the purposes of this thesis by labeling data from the Vienna City Library. The performance of these algorithms in less ideal conditions was also tested on more challenging subsets of the CelebA and WFLW datasets.

Key words: face detection, face detection datasets, facial landmarks, machine learning, unconstrained conditions

Contents

Introduction

A large part of human interaction involves face-to-face communication. Face is the first thing that people notice about one another, it expresses emotions and carries a lot of information about a person and their state of mind. We use facial features to identify each other on a daily basis.

For these and many other reasons, the detection of faces in an image is a useful tool for computers. It is a key instrument for other face-related computer tasks, such as recognition, authentication, gender, age recognition, head pose tracking, face tracking for surveillance, and many others [1]. In addition to its use in facial analysis tasks, face detectors can be found in most digital cameras providing the auto-focus feature or in social media networks to tag people in images [2]. However, in face detection, there are many challenges that the computer must be able to overcome to be successful in this task. These include variations in pose, facial expressions, lighting conditions, occlusions, etc. [1]. Therefore, many face detection approaches, as well as face detection datasets, have been developed to train and test these methods. The early algorithms, although successful under certain conditions, were unable to perform well in real world settings. Today, many approaches have been developed that are capable of performing accurately even in more complex environments [2].

This thesis aims to present the theoretical background behind some of the face detection methods used to detect faces under unconstrained conditions. It includes a chapter on the fundamentals of machine learning (Chapter 1) and a chapter dedicated to face detection, where detection methods, which are later used in experiments, are described (Chapter 3). Chapter 2 describes different challenges in detection, as well as datasets used to train and evaluate detectors under these conditions. The last Chapter 4 focuses on experiments conducted with the chosen methods on three different datasets and evaluates the results under different image conditions.

Chapter 1

Machine Learning

Face detection in digital images is a process used to identify the size and location of the human face, as well as its features [1]. The input of this process is an image or a video with the output being the information, whether it contains a face and its location as well as the extent of each of the faces present [3]. It may also be viewed as a classification problem where the objective is to locate all members of a certain class. In this case, we deal with a two-class recognition task in which one class represents all faces and the other one everything else [4].

Machine learning is used for many computer vision tasks, including face detection. This chapter provides a brief review of machine learning techniques and principles.

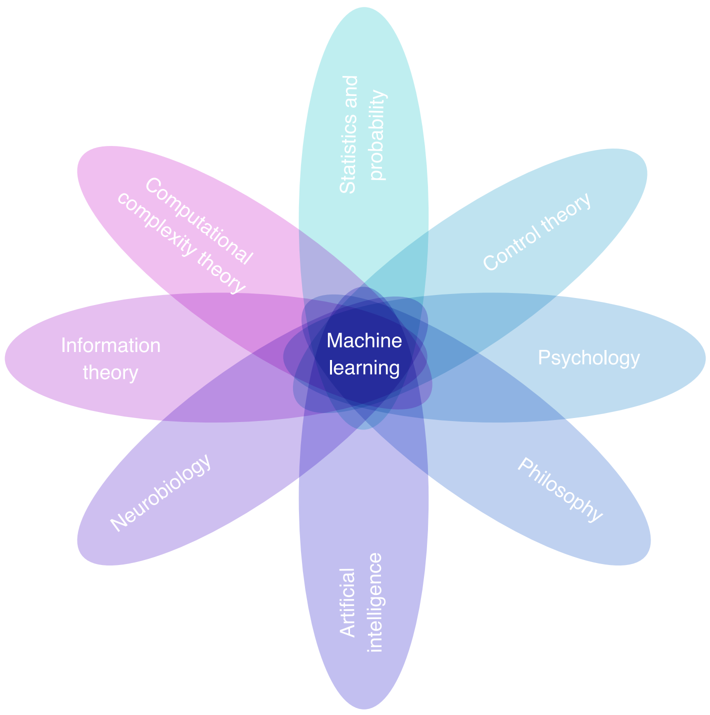


Figure 1.1: Various fields of research which are incorporated into machine learning.

As shown in Figure 1.1, machine learning is a discipline that combines various diverse fields of research. It models algorithms based on empirical data, from which it pursues to extract rules and patterns.

The main idea behind these algorithms is that they learn to perform given tasks based on acquired experience. A formal definition of learning by Tom M. Mitchell (1997) is: "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E." [5]. Some of the most common machine learning tasks are *classification*, *regression*, *feature reduction*, *anomaly detection* or *denoising*.

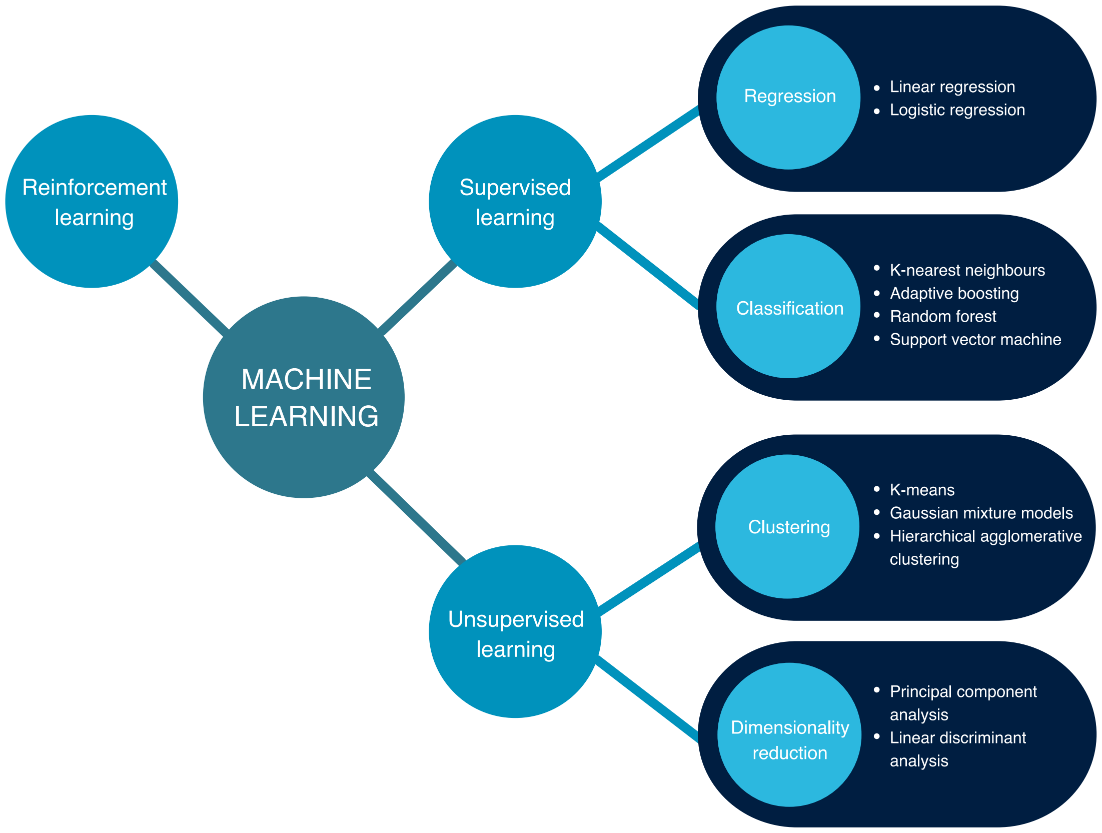


Figure 1.2: Main categories of machine learning algorithms based on experience.

The experience represents what kind of data we present the algorithm with, what it experiences [5]. Based on different learning techniques (experiences), machine learning can be divided into three main subcategories [6] (see Figure 1.2):

- **Supervised learning**: In this type of learning, examples of labels for input data are given to the computer. It is expected to associate features based on given knowledge and make decisions accordingly. Algorithms that experience this kind of data are, for example: linear or logistics regression, decision trees, naive Bayes classifier.

- **Unsupervised learning**: These techniques experience data given by features. Since no labels are provided, training is done by finding similar patterns and grouping the data according to those patterns. Examples for this category include: k-means clustering, probabilistic clustering, or hierarchical clustering.

- **Reinforcement learning**: Algorithms that use reinforcement learning maximize their performance by receiving feedback (called reinforcement signal). Therefore, they do not experience a fixed dataset as the previously mentioned approaches. This technique might be used, for example, in training neural networks.

Among the standard performance measures, the most frequently used for classification are *accuracy*, *error rate*, *precision*, *specificity*, or *sensitivity*. The choice of measures depends on the given task and the desired behavior of the algorithm used. One of the most challenging factors in machine learning is the algorithm's ability to perform well on previously unobserved input. To obtain a better insight into the performance, a testing set (which is independent of training data) is used to calculate the performance measures. A term used for the ability to give satisfactory results on previously unseen data is called *generalization*. While training a machine learning algorithm, a training error is calculated from the training data as well as a generalization (test) error - an error expected on new data. From these values, we can study the adequacy of the given algorithm by observing its ability to minimize:

- Training errors: The inability to adequately minimize the training error results in *underfitting*.
- Gap between test and training errors: If the gap becomes too large, *overfitting* occurs [5].

Examples of these unwanted behaviors of models are depicted in Figure 1.3. The likelihood of overfitting or underfitting the model can be controlled by its capacity. This property represents the model's capability of fitting different functions.

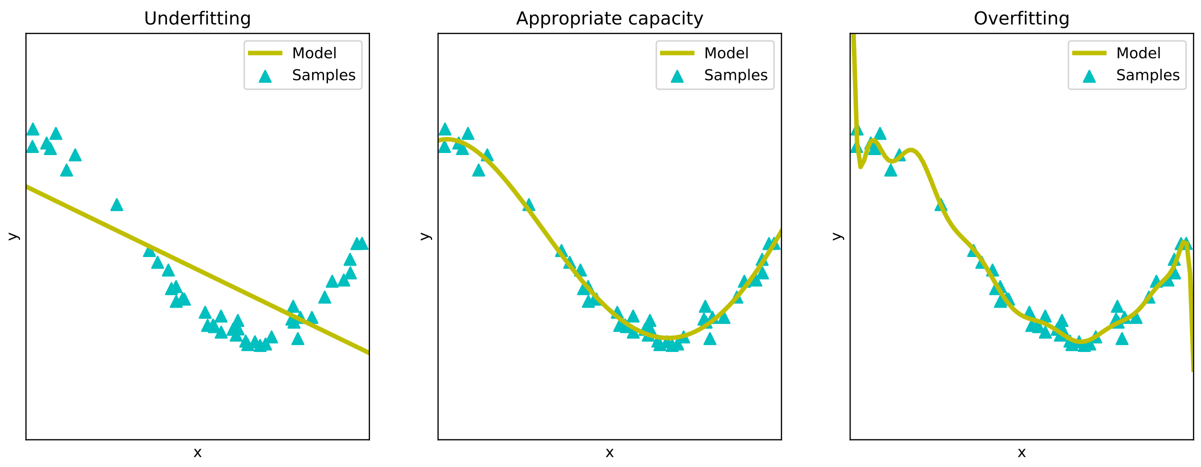
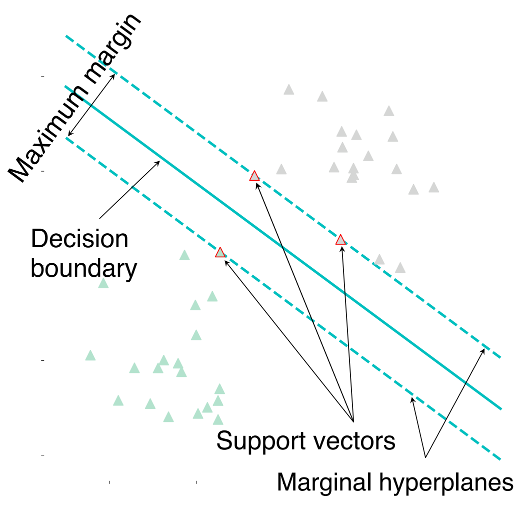


Figure 1.3: Visualisation of model behaviors if insufficient minimization of training errors (underfitting) or gap between training and test error (overfitting) occurs and a model with appropriate capacity.

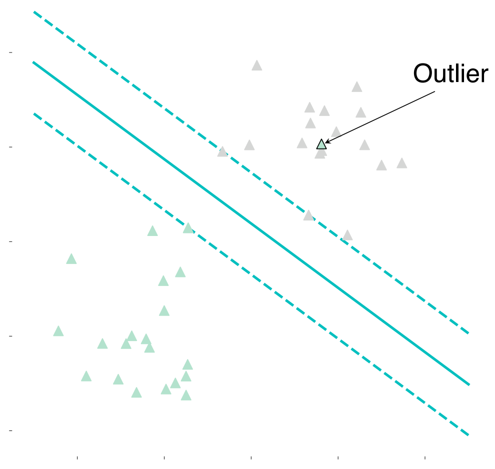
The remainder of this chapter is focused on machine learning methods that are common in face detection applications.

1.1 Support Vector Machine

Support vector machine (SVM) is a machine learning algorithm that uses supervised learning for classification tasks. It is designed to look for parallel hyperplanes that separate data belonging to different classes by the maximum possible distance. The resulting decision boundary is also a hyperplane parallel to these so-called marginal hyperplanes, which lies between them. The name support vector machine is derived from the term used for training samples that the separating hyperplanes pass through - *support*



(a)



(b)

Figure 1.4: (a) Hard margin SVM approach to finding decision boundary for two classes and two-dimensional features where hyperplanes are lines. It shows where the decision boundary support vectors and maximum margin are located. (b) Soft margin approach which allows the outlier to lie on the wrong side of the marginal hyperplanes.

vectors [7]. Figure 1.4a depicts all the above-mentioned concepts related to SVM for two classes of two-dimensional features.

A hyperplane for separating two classes with n-dimensional feature vectors $\boldsymbol{x}$ is defined as:

$$\boldsymbol{wx} + b = 0 \tag{1.1}$$

and $\boldsymbol{w}$ is the vector of weights. Two marginal hyperplanes can be represented as:

$$\boldsymbol{wx} + b = 1, \tag{1.2}$$

$$\boldsymbol{wx} + b = -1, \tag{1.3}$$

satisfying the condition that the hyperplanes separate data: $\omega_i(\boldsymbol{wx}_i + b) \geq 1$ where ω_i is class identifier ($\omega_i \in \{1, -1\}$). Let us have two feature vectors: $\boldsymbol{x}_+$ lying on the positive plane and $\boldsymbol{x}_-$ on the negative plane closest to $\boldsymbol{x}_+$. Vector $\boldsymbol{x}_+ - \boldsymbol{x}_-$ is normal to both hyperplanes, hence for any λ:

$$\lambda \boldsymbol{w} = \boldsymbol{x}_+ - \boldsymbol{x}_-. \tag{1.4}$$

Multiplying Equation (1.4) by vector $\boldsymbol{w}$ yields:

$$\lambda ||\boldsymbol{w}||^2 = \boldsymbol{x}_+\boldsymbol{w} - \boldsymbol{x}_-\boldsymbol{w}, \tag{1.5}$$

$$\lambda ||\boldsymbol{w}||^2 = 2, \tag{1.6}$$

Substituting $\lambda = 2/||\boldsymbol{w}||^2$ (derived from (1.6)) in (1.4) leads to:

$$\boldsymbol{x}_+ - \boldsymbol{x}_- = \frac{2}{||\boldsymbol{w}||}, \tag{1.7}$$

which is the equation for the margin, we are looking to maximize. This can be achieved by minimizing $||\boldsymbol{w}||$ with regard to the constraint $\omega_i(\boldsymbol{wx}_i + b) \geq 1$. Lagrangian function can be used to accomplish this task [8]. This kind of approach to SVM uses *hard margin* constraint and is suited for linearly separable classes without outliers. To avoid this drawback, the *soft margin* constraint is used. A penalty term is added to the objective function so that the algorithm allows the training samples to lie on the other side of the hyperplane for a given class (see Figure 1.4b). If classes are not linearly separable classes, a mapping of the feature space is carried out. This approach to SVM is known as the *kernel trick* [7].

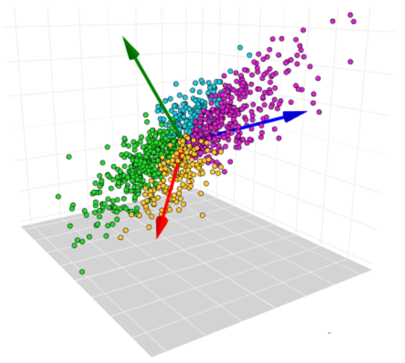
1.2 Principal Component Analysis

Principal component analysis (PCA) is an unsupervised learning algorithm. It is designed to reduce the dimensionality of given data while removing their linear correlation. Covariance for $m \times n$ matrix representation of data $\mathbf{X}$ with zero mean (which can be achieved by subtracting the mean from all samples) is defined as:

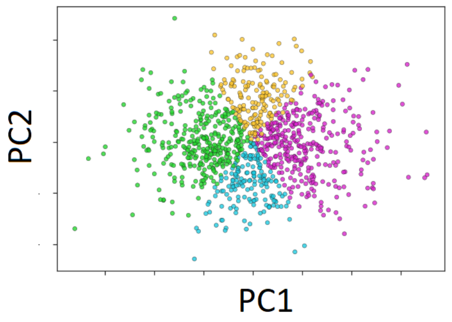
$$\text{Cov}[\mathbf{x}] = \frac{1}{m-1}\mathbf{X}^T\mathbf{X}. \tag{1.8}$$

PCA finds a linear orthogonal transform, in this case rotation $\mathbf{z} = \mathbf{x}^T\mathbf{W}$, such that matrix Cov[$\mathbf{z}$] is diagonal [5].

The basic algorithm for principal component analysis consists of finding the mean vector $\boldsymbol{\mu}$ and the covariance matrix Cov[$\mathbf{x}$] from the original data. Subsequently, the eigenvectors with their corresponding eigenvalues of the covariance matrix are calculated and sorted in decreasing order according to the eigenvalues. Afterwards, k eigenvectors with the highest eigenvalues are used as new dimensions that the original m-dimensional space will be transformed into [9]. The number of eigenvectors k can be determined by the number of dimensions needed for given problem or with the help of a measure called proportion of explained variance which is calculated from the eigenvalues [10]. Figure 1.5 shows an example of PCA for 3-dimensional data.



(a) Original data in 3-dimensions.



(b) Scatterplot after using PCA to reduce the dimensionality to 2-dimensions.

Figure 1.5: Visualization of PCA on 3-dimensional dataset [11].

1.3 Adaptive Boosting

Adaptive Boosting or AdaBoost is a supervised learning algorithm used for classification. The idea behind this approach is to create a classifier by combining several imprecise or weak ones.

Let us have N training samples $(\mathbf{x}_i, y_i)_{i=1}^N$ where $\mathbf{x}_i \in \mathbb{R}^k$, $k \in \mathbb{N}$ and $y_i \in \{-1, +1\}$. The definition of $0-1$ loss function I for weak classifiers $f_m(\mathbf{x}) \in \{-1, +1\}$ is given by the equation:

$$I(f_m(\mathbf{x}), y) = \begin{cases} 0 & \text{if } f_m(\mathbf{x}_i) = y_i \\ 1 & \text{if } f_m(\mathbf{x}_i) \neq y_i. \end{cases} \tag{1.9}$$

The algorithm for adaptive boosting consists of the following steps:

1. Initialize the weights for each weak classifier: $w_i^{(1)} = 1/N$ for all i.
2. For all classifiers ($m = 1, \dots, M$) do:
   (a) Train weak classifier m with given weights, minimizing weighted error:

   $$\epsilon_m = \frac{\sum_i w_i^{(m)} I(f_m(\mathbf{x}_i) \neq y_i)}{\sum_i w_i^{(m)}}, \tag{1.10}$$

   and count the weight of the m-th classifier $\alpha_m = \ln \frac{1-\epsilon_m}{\epsilon_m}$.

   (b) Update the weights for all i according to the following formula:

   $$w_i^{(m+1)} = w_i^{(m)} \exp(\alpha_m I(f_m(\mathbf{x}_i) \neq y_i)). \tag{1.11}$$

This learning process produces the final classifier in the form of a linear combination given by equation [12]:

$$g(\mathbf{x}) = \operatorname{sign}\left(\sum_{m=1}^{M} \alpha_m f_m(\mathbf{x}) \right). \tag{1.12}$$

1.4 Artificial Neural Networks

These machine learning algorithms are inspired by human biology. The main idea behind this approach is to construct a network of so called neurons, which operate in parallel [7]. Unlike other machine learning approaches, artificial neural networks (ANNs) learn the representation of given data, which becomes increasingly abstract in each layer. However, human input in form of specifying the parameters is still needed even for these learning algorithms.

One of the first artificial neuron models was the perceptron (see Figure 1.6). It works for a two-class classification problem where the data are linearly separable. The equation for the separating hyperplane in this case yields:

$$f(\mathbf{w}) = \mathbf{w}^T \mathbf{x} = 0, \tag{1.13}$$

where $\mathbf{x} = (x_1, x_2 \dots, x_n, 1)^T$ is a vector of inputs, $\mathbf{w} = (w_1, w_2, \dots, w_{n+1})^T$ is a vector of the corresponding weights and f is called *input function*. When performing classification on linearly separable data with two classes C_1 and C_2, the aim is to find weights that satisfy the following condition:

$$\mathbf{w}^T \mathbf{x} \begin{cases} > 0 & \text{if } \mathbf{x} \in C_1 \\ < 0 & \text{if } \mathbf{x} \in C_2. \end{cases} \tag{1.14}$$

Let $\mathbf{w}(1)$ be the weight vector of arbitrary values, $\mathbf{x}(k)$ the pattern vector corresponding to step k and $\alpha > 0$ the *learning rate*. Then the training algorithm for step k of the perceptron is following [13]:

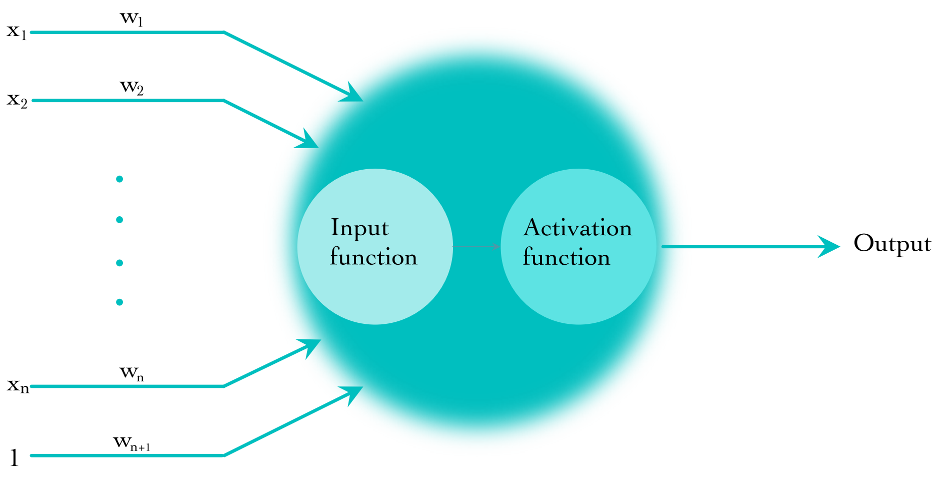


Figure 1.6: The composition of artificial neuron called perceptron.

1. if $\mathbf{x}(k) \in C_1$ and $\mathbf{w}^T\mathbf{x}(k) \leq 0$, let $\mathbf{w}(k+1) = \mathbf{w}(k) + \alpha\mathbf{x}(k)$
2. if $\mathbf{x}(k) \in C_2$ and $\mathbf{w}^T\mathbf{x}(k) \geq 0$, let $\mathbf{w}(k+1) = \mathbf{w}(k) - \alpha\mathbf{x}(k)$
3. else, let $\mathbf{w}(k+1) = \mathbf{w}(k)$.

Afterwards, *activation function* g is used to determine the class membership of given data. In this case, the activation function is a thresholding function, which assigns the pattern to class C_1 if the thresholded output is 1 and to class C_2 for this value equal to -1.

Artificial neurons in multilayer networks work in the same way as perceptron, differing mainly in activation function. Some of the most frequently used activation functions in artificial neurons are: sigmoid, rectifier linear unit (ReLU), hyperbolic tangent (see Figure 1.7) or in some instances softmax. Training of these networks is performed using the *backpropagation* algorithm.

There are many different architectures of neural networks based on the way the network is connected. Some of the most well-known are, for example, feedforward networks, convolution, radial basis function networks, and many more [13].

Feedforward Neural Network

This network usually consists of several layers of neurons. The first one (*input layer*) is comprised of the same number of neurons as is the feature vector dimensionality. The data processed in the input layer are subsequently sent to the first *hidden layer*. Each layer receives the data, performs (previously described) operations, and sends the output to the next one [7]. The number of neurons can differ in each layer. Each neuron has only one output, which is connected to all neurons in the next layer to create a fully connected network. If there is only one hidden layer in the network, it is called *shallow*, while

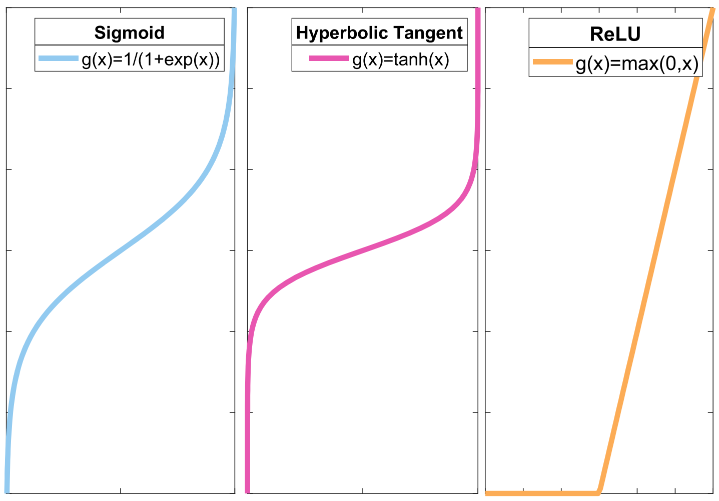


Figure 1.7: Activation functions used in artificial neuron structures.

those with two or more are labeled *deep*. The number of layers of ANN is called its *depth* [13]. The last hidden layer is connected to the *output layer*, which provides the values used for determining the final decision [7]. Examples of deep and shallow feedforward neural networks are shown in Figure 1.8.

As mentioned in the previous text, the training of ANNs is done by backpropagation. The backpropagation algorithm for these types of networks consists of four basic steps [13]:

1. input of the training data,
2. classification of the data by network and the determination of its error,
3. computation of required changes, that minimize the error, is carried out by passing the error through the network,
4. the model weights are updated, and the process is repeated until the error reaches an acceptable level.

Convolutional Neural Network

The main difference between convolutional neural networks (CNN) and feedforward networks is that they use convolution as an input function in certain layers. This allows the input of these networks to have an image format or similar grid-like form. Instead of previously extracted features, these ANNs learn the patterns directly from given images. Therefore, these types of networks are well suited for image processing applications [13].

Aside from the input and output layer, a CNN is commonly comprised of three types of layers - *convolutional*, *pooling* and *fully connected*. This structure is shown in Figure 1.10.

- **Convolutional layers** use kernels, which are convolved with the input. The kernel is the same depth as the input, has low spatial dimensionality, and is applied to the whole image. A hyper-parameter called *stride* is used to define the step size with which the kernel is moved through the input in the process of convolution. The output depth can be controlled by the number of neurons

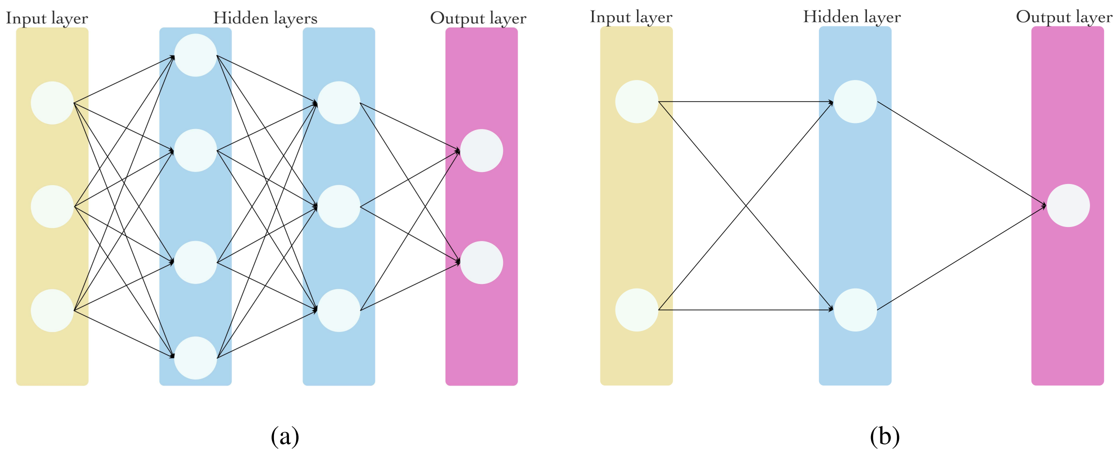


Figure 1.8: Examples of feedforward (a) deep and (b) shallow neural network structures.

in a given layer and also by using zero-padding. Same as in previously described ANNs, an activation function is applied to the outcome of convolution. An example of how this layer performs convolution on an image is shown in Figure 1.9.

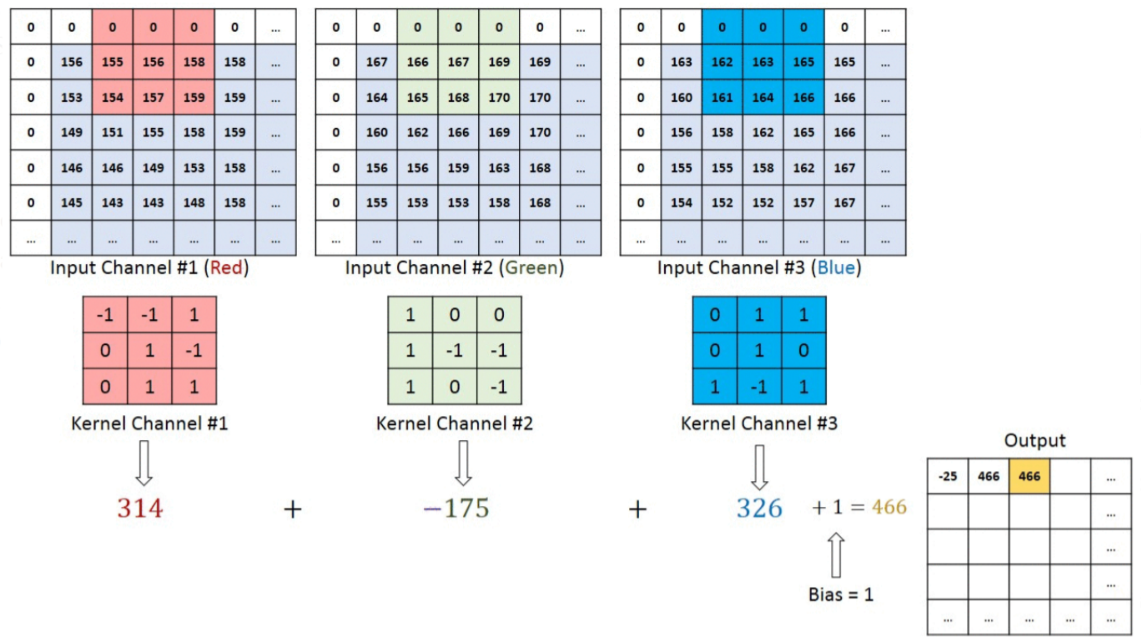


Figure 1.9: The convolution of input image with $3 \times 3 \times 3$ kernel [14].

- **Pooling layers** are used to reduce the spatial dimensionality of the data, which is done by convolution with kernels. These are usually *max-pooling layers* of size 2×2 with the stride set to two [15]. This method of pooling is done by moving the kernel along the input by the given stride and returning the maximum value of the covered portion of the data.
- The **fully-connected layer** functions in the same way as a typical ANN. The image is flattened and a feedforward neural network with a soft-max activation function is used [14].

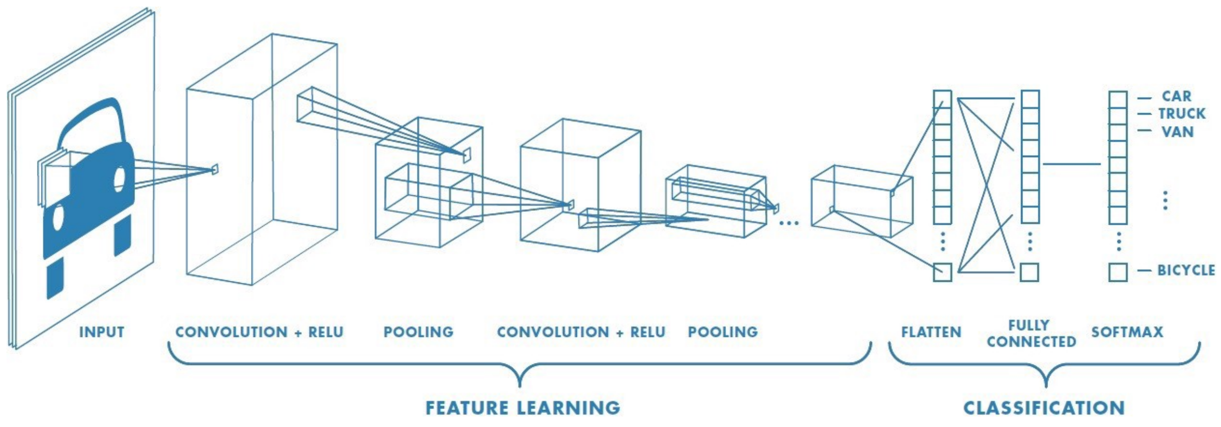


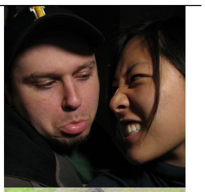
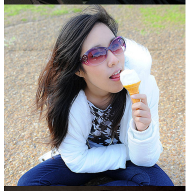
Figure 1.10: An example of CNN structure [14].

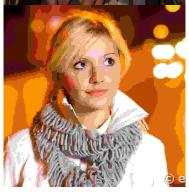
Chapter 2

Face Detection Challenges

As we learned in the previous chapter, face detection is a two-class recognition task. Therefore, the same challenges as in object-class recognition might be encountered. When dealing with faces, we observe exceptionally high intraclass variability caused by many factors such as different facial shapes, colors, makeup, facial hair, accessories, etc. [4]. Some of the main challenges that detection deals with are presented in Table 2.1. All these different scenarios will very likely occur in a real-life application, hence the need for unconstrained face detection algorithms. This term suggests that such a system should work successfully regardless of the subject conditions or how the image was captured [16].

Table 2.1: Challenges in face detection, images from dataset [17].

Challenges	Description	Example
Different expressions	Odd facial expressions might cause more within-class variation.	
Occlusion	Part of the face could be covered by an object, body part, facial hair, etc. It could reduce accuracy as well as the detection rate.	
Illumination	Illumination changes in the image might make faces less visible or reduce the appearance of certain anatomical traits.	
Complex background	With more objects present in the background, the location of the faces becomes less apparent.	

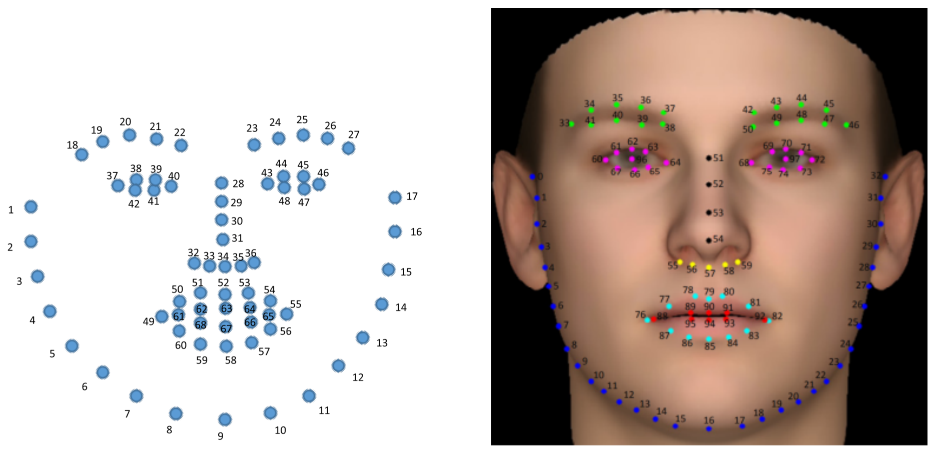
Too many faces	Too many faces present in an image takes a toll on the detection rate and accuracy.	
Resolution	Low image resolution can degrade the face, making it more difficult to detect.	
Skin color	Every human is different and so is their skin tone, therefore not all face detection algorithms work the same on all skin colors.	
Distance	Another important aspect in face detection, is the position of the captured human with regards to the camera. In some cases faces that are farther as well as faces that are too close might be more difficult to detect.	
Pose	Since not all human images are taken from the same angle in the same position, pose is one of the challenges that this process faces and has to deal with.	

Face detection datasets

When dealing with face detection, the right tools for testing and, in many cases, training are required. Therefore, several benchmark databases of images have been developed for this task. These vary in size, format, and conditions under which the images were taken. The image databases range from sets of frontal pictures taken in a controlled environment to fully unconfined settings, where all the detection challenges in Table 2.1 are present.

Aside from detecting the location of ones face, extracting its features might be valuable information. These features are called *facial landmarks* and represent facial regions such as the eyes, nose, mouth, and many others. Different face detection methods might extract different numbers of facial landmarks, so the number of these features annotated in different databases might vary. Among the facial annotations that are regularly used are, for example, the 68 point annotation shown in Figure 2.1.

For the purposes of this thesis, only datasets with annotated facial landmarks will be considered, the more suitable ones are shown in Table 2.2.



(a) (b)

Figure 2.1: Landmark annotations with: (a) 68 points, used for example in the XM2VTS database [18]; (b) 98 points in the WFLW database [19].

Table 2.2: Face detection datasets.

Database	Description
WFLW[20]	Database of color images containing annotations with 98 facial landmarks, as well as binary annotations for different face detection challenges present in images. Binary annotations include head pose, occlusion, illumination, makeup, and expression.
CelebA [21]	This dataset includes more than 200K colored images of celebrities. It contains large pose and background variations, occlusion, age, and ethnicity diversities. The database is annotated with 5 landmarks, as well as other binary facial attributes.
BioID [22]	This is a grayscale image database with front view of captured faces. They are annotated with 20 facial landmarks. The images vary slightly in pose and expression. There is some occlusion and the background is the same for all the images.

XM2VTS [23]	Dataset contains frontal images in color, annotated with 68 facial landmarks. It includes no occlusion, slight variation in ethnicity, and a uniform background under ideal lighting conditions.
MUCT [24]	This database contains color images with 76 manually annotated landmarks. The background of these images is uniform and diverse in lighting conditions. It contains a frontal view of faces, without occlusion, but considerable variation in age and ethnicity.

Chapter 3

Face Detection Approaches

As in any computer vision task, there are numerous approaches to solving the problem of face detection. These different techniques vary in robustness, execution time, accuracy as well as complexity and each is better suited for a different task. This chapter aims to introduce some of the most widely used face detection techniques.

According to [25], face detection algorithms can be divided into two main categories based on the use of prior knowledge of the human face, which are feature based and image based techniques. The classification of face detection methods is shown in Figure 3.1.

- **Feature-based approach**: This approach uses information about facial features, such as skin color or facial geometry. Detection tasks are performed by manipulating measurements of angles, distance, and area of visual features from the image. These techniques are divided into three categories which are feature analysis, low-level analysis, and active shape models.
  - The main focus of ***active shape models*** are physical, higher level appearance features [25]. These models aim to locate landmark points of an object. In an image of a face, these include features such as eyes, nose, mouth or eyebrows. Active shape models can be further classified into three subcategories, which are *Snakes*, *Point Distribution Model* and *Deformable templates* [1].
  - The first step of ***low-level analysis*** is segmentation based on pixel properties such as its color or intensity. This step generates ambiguous visual features. These are organized according to facial geometry. The faces and facial landmarks are then detected using feature analysis [25]. Low-level analysis techniques can be based on *edges*, *gray information*, *motion*, *skin color* and more.
  - ***Feature analysis*** methods focus on finding structural features invariant to changes in viewpoint, pose, or lighting conditions. These are subsequently used to localize faces in an image. Feature analysis approaches include *feature searching* algorithms, among which one of the most famous ones is the Viola-Jones face detector, and *constellation* analysis [1].
- **Image-based approach**: This approach is based on learning algorithms without the use of feature derivation and analysis. It is classified into three categories, which are linear subspace methods, neural networks, and statistical approaches [25].
  - Linear algebra defines the linear subspace as a subset of vector space, which itself is a vector space. In image processing, it is perceived as a smaller segment of a frame. Therefore, faces in images belong to a subspace of the image space. ***Linear subspace*** methods focus

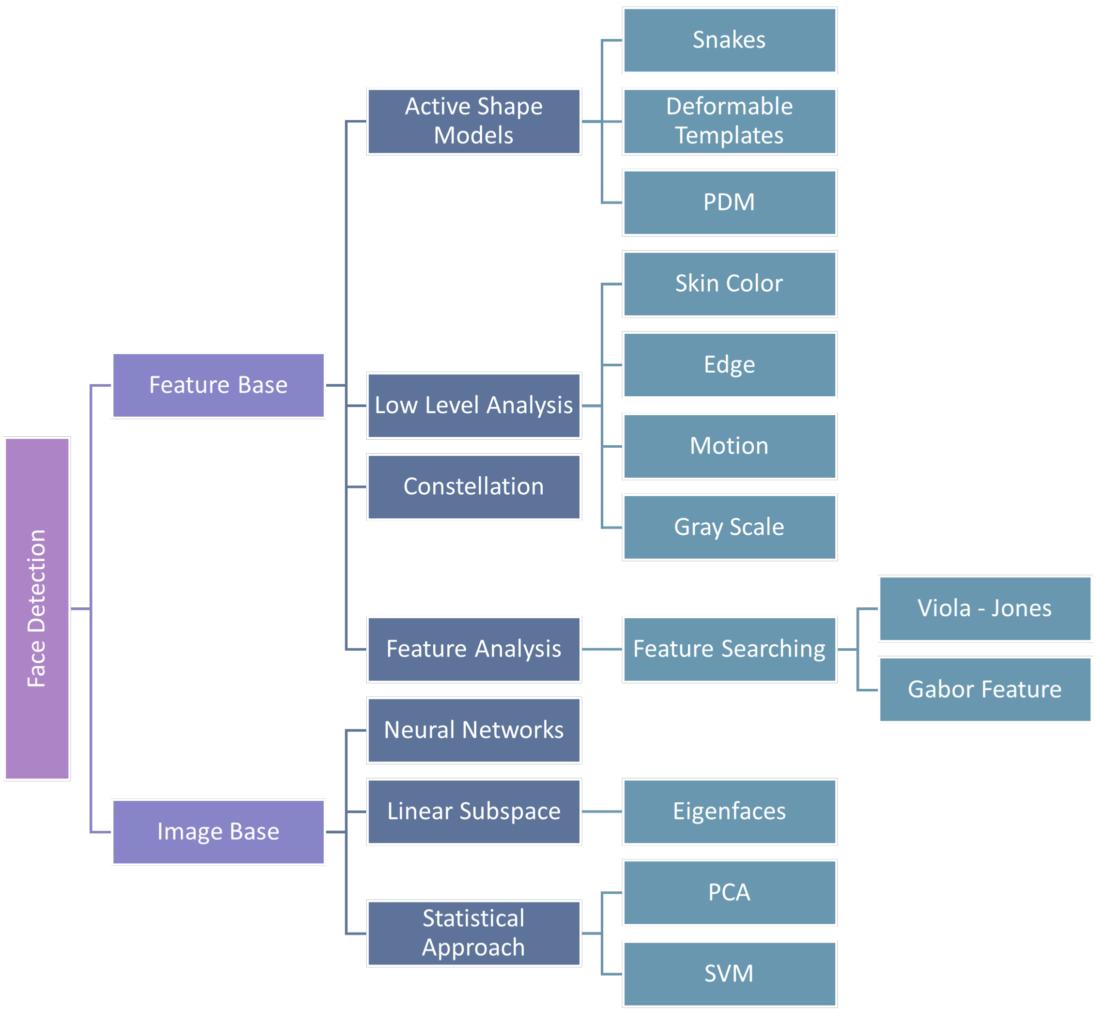


Figure 3.1: Approaches in face detection.

on the representation of this subspace of facial images. These techniques include *eigenfaces*, *Fisherfaces* or *tensorfaces*.

– ***Neural networks*** are series of algorithms based on human biology. Mimicking the structure of neural networks in the brain, they receive data and learn to recognize different patterns [26]. They are frequently used in pattern recognition tasks and face detection can be viewed as a two-class recognition problem. Thus, numerous neural network architectures were designed specifically for this task [4]. An introduction to these algorithms was already given in Chapter 1.

– ***Statistical approaches*** make use of statistics in face detection tasks. Statistical algorithms used for face detection are, for example, *Principal Component Analysis* or *Support Vector Machine*, which were described in Chapter 1 .

The next part of this chapter is focused on some of the most common face detection techniques.

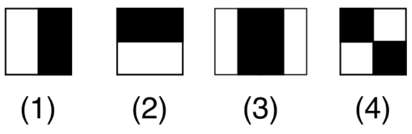
3.1 Viola-Jones Detector

This algorithm is one of the *feature searching* methods used in detection. Its main advantage is the fast computation time and fairly high precision [1].

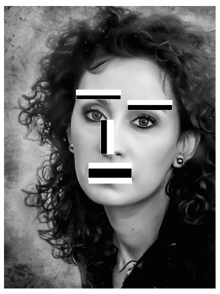
The Viola-Jones detector consists of three steps: the computation of *integral image*, using AdaBoost for training strong classifiers and the construction of cascade classifier. The integral image is computed from gray-level images. It is a fast and efficient algorithm used to compute the sum of the pixel values in a rectangular window of the image. Its pixel at position (x, y) is given by the equation:

$$\mathrm{ii}(x, y) = \sum_{x' \leq x, y' \leq y} \mathrm{i}(x', y'), \tag{3.1}$$

where $\mathrm{i}(x, y)$ denotes a pixel of the original image at a given location. Afterwards, Haar-like features are computed from this integral image. There are several types of Haar-like features (see Figure 3.2b), for this algorithm, *horizontal* and *vertical* are used [27]. Vertical detect face parts such as forehead or eyebrows, horizontal detect nose, for example (see Figure 3.2b). These features are created by calculating the mean intensities in dark windows subtracted by the mean intensities of light windows. A Haar-like feature is present in the image if this difference surpasses the given threshold [28].



(a)



(b)

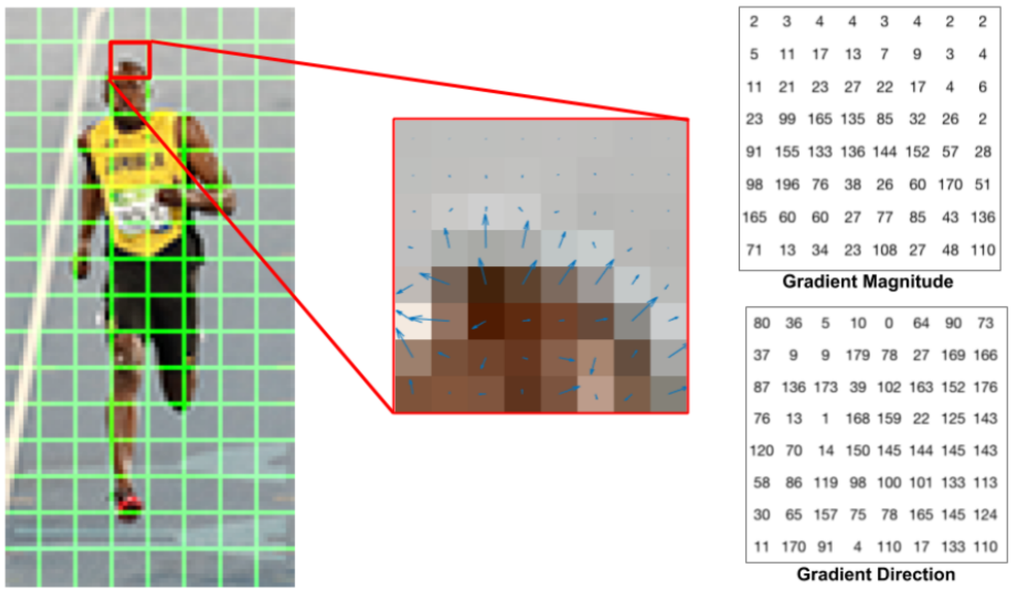
Figure 3.2: (a) Types of Haar-like features: (1) horizontal, (2) vertical edge features, (3) vertical line feature, (4) diagonal line feature [28], (b) Illustration of what facial regions given haar-like features symbolize in face image [28].

The AdaBoost algorithm (which was introduced in Chapter 1 Section 1.3) is applied to the extracted features to select only the most critical ones, which form a classifier and the rest are discarded [1]. In the last step, sub-windows of the image are categorized by a cascading classifiers. If the result of one of these classifiers is negative (meaning image does not contain a face) for given sub-window, it is rejected and the processing of this window ends [29]. The likelihood of each sub-window containing a face is determined by the number of stages passed in this classification process [27]. Implementation of this face detection algorithm can be found in the OpenCV library.

3.2 Histogram of Oriented Gradients and Linear SVM Detector

This approach combines feature extraction technique called histogram of oriented gradients with linear support vector machine classifier (HOG+SVM). This method is invariant with respect to changes in illumination and is capable to extract more complex facial features than linear filters used in the previous algorithm [27]. The HOG features are extracted in following steps [30]:

1. An image is split into blocks, where histograms are calculated (see Figure 3.3) by filtering these regions with kernels: $[-1, 0, 1]$ and $[-1, 0, 1]^T$.



2	3	4	4	3	4	2	2
5	11	17	13	7	9	3	4
11	21	23	27	22	17	4	6
23	99	165	135	85	32	26	2
91	155	133	136	144	152	57	28
98	196	76	38	26	60	170	51
165	60	60	27	77	85	43	136
71	13	34	23	108	27	48	110

80	36	5	10	0	64	90	73
37	9	9	179	78	27	169	166
87	136	173	39	102	163	152	176
76	13	1	168	159	22	125	143
120	70	14	150	145	144	145	143
58	86	119	98	100	101	133	113
30	65	157	75	78	165	145	124
11	170	91	4	110	17	133	110

Figure 3.3: Image gradients in given block calculated in the first step of HOG feature extraction algorithm [31].

2. A histogram using previously computed gradients is created based on their orientation and magnitude.

3. Overall histogram from the previous marginal histograms is formed, creating a feature vector.

4. Normalization to lighting changes over grouped blocks is carried out. Afterwards, the normalized vectors are concentrated into a feature vector for the given image.

After extracting features using HOG algorithm, a simple linear SVM (described in Chapter 1) is trained with these data [27]. This face detector is implemented an available in dlib library.

3.3 Multitask Cascaded Convolutional Networks (MTCNN)

This approach uses cascaded convolutional neural networks for face and landmark detection. It consists of three stages (see Figure 3.4) [32]:

1. A pyramid is created by resizing the input image. This pyramid is passed through a fully convolutional neural network (a CNN without the fully connected layers) - proposal network (P-Net). The output of this network is windows that might obtain faces, as well as their bounding boxes. These candidates are calibrated and if overlapping, merged using non-maximum suppression.

2. Previously extracted facial region candidates are fed to different CNN - refine network (R-Net). This network discards number of false positives and, same as in the previous step, calibration followed by non-maximum suppression is carried out.

3. Last stage of this process uses output network (O-Net). It is similar to the previous stage, except in this stage, the output are facial landmarks.

The structure, layers and outputs of the previously mentioned CNNs are shown in Figure 3.5.

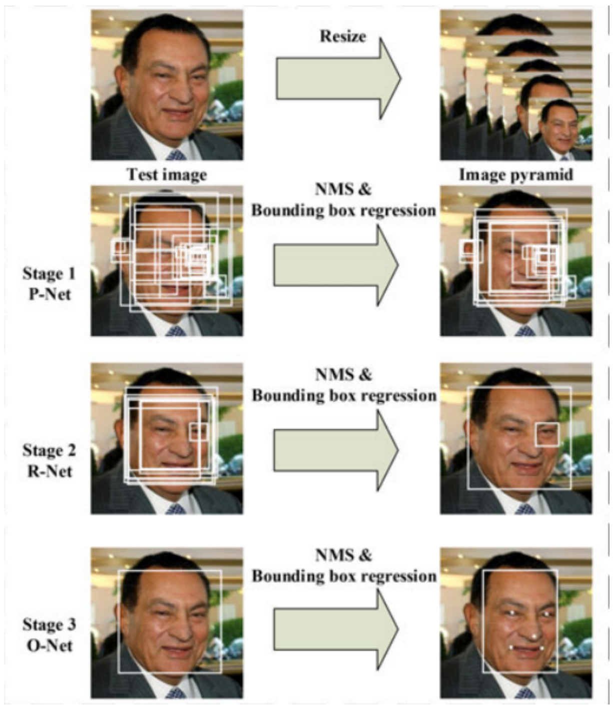


Figure 3.4: Pipeline of MTCNN face detection technique [32]

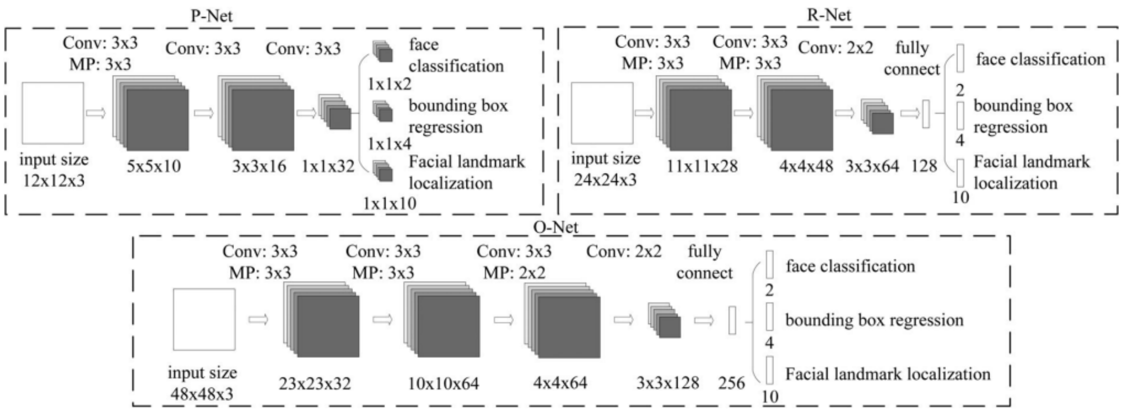


Figure 3.5: Composition of the three types of networks used in MTCNN pipeline [32]

3.4 RetinaFace

This approach simultaneously detects a bounding box as well as facial landmarks (eyes, corners of the mouth and nose) and $3D$ vertices for each facial region. Joint learning process is used to achieve this goal. Three main components of RetinaFace approach to face detection are: the feature pyramid network, the context module and the cascade multi-task loss [33](Figure 3.6 and Figure 3.7).

1. The first step consists of generating a **feature pyramid network** (FPN). These networks are used for detection in varying scales of input images and further information about them can be found in [34]. In this particular case, the first four levels of the pyramid are computed using a Residual Network (ResNet). ResNet is a neural network that skips one or more layers, allowing the training

of very deep networks [35]. The last (fifth) level of the FPN is computed through convolution on the previous level.

2. **Context module** is used to improve detection by using context. In this pipeline, deformable convolutional networks (DCNs) are used on five levels of the pyramid [33]. DCN is a convolutional neural network designed to overcome the inability of CNNs to learn geometric transformations from given data [36].

3. The output of the context module is a **multitask-loss**. This loss function is defined for every training anchor i as:

$$L = L_{cls}(p_i, p_i^*) + \lambda_1 p_i^* L_{box}(t_i, t_i^*) + \lambda_2 p_i^* L_{pts}(l_i, l_i^*) + \lambda_3 p_i^* L_{mesh}(v_i, v_i*), \tag{3.2}$$

where t_i, l_i, v_i are box, landmarks and vertices predictions for facial region corresponding to the actual t_i^*, l_i^*, v_i^*. Symbol p_i is the probability of each anchor being a face with its corresponding ground truth being p_i^*. Loss functions L_{cls}, L_{box}, L_{pts}, L_{mesh} are classification, box, points and mesh regression loss. Parameters λ_1, λ_2 and λ_3 are called loss-balancing parameters [33].

Implementation of this approach can be found on GitHub [37].

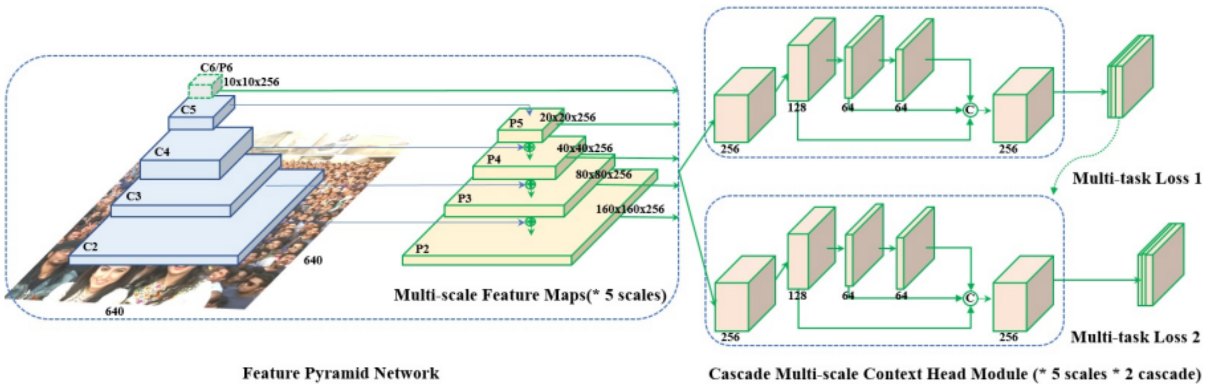


Figure 3.6: The composition of RetinaFace network structure [33].

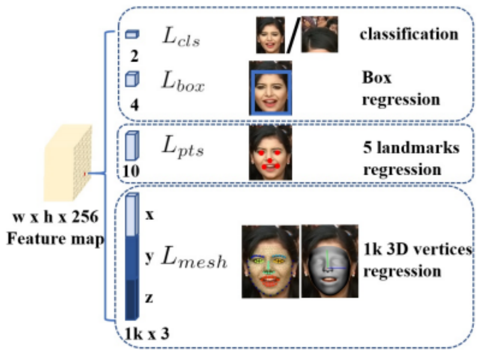


Figure 3.7: RetinaFace multitask loss [33].

Chapter 4

Experiments and Evaluation

The previous chapter introduces some of the methods commonly used for face detection, which will be evaluated in this chapter. Experiments were carried out using MTCNN (implemented on GitHub [38]), RetinaFace (from GitHub [39]), dlib implementation of the histogram of oriented gradients combined with linear SVM detector, and OpenCV implementation of the Viola-Jones detector. These methods were chosen because of their effectiveness, accessibility, and ability to detect faces, as well as some facial landmarks. All the detectors were tested using Python. For each approach, a set of five facial landmarks (left eye centre, right eye centre, tip of the nose, left mouth corner, and right mouth corner) was extracted along with a bounding box for the given facial region. Both results for RetinaFace and MTCNN are in form of these five landmarks and detection box. However, the landmark annotations for HOG+SVM and Viola-Jones do not contain the location of the eye centers, just the eye corners. Therefore, the location of each eye was calculated as the middle of the segment connecting these corner points.

4.1 Datasets

Datasets chosen for the evaluation are WFLW and CelebA test sets (a further description of the images included in these collections was given in Chapter 2). Both collections of face images are set in unconstrained conditions and contain facial landmark annotations important for testing algorithms ability to detect the chosen regions that the methods are trained to label. Another requirement for these sets of images was that they were not used for training of the chosen face detection approaches. This ensures that the results are unbiased. CelebA and WFLW datasets contain binary annotations for different image and face attributes. WFLW, for example, has images of faces with occlusion, heavy makeup (over 200 images), illumination (almost 700 images), more expressive facial expressions (around 300), or poses (over 300) labeled (see examples in Figure 4.1). In CelebA, image annotations include faces with glasses (approximately 1300 images) and blurry images (over a 1000).

Another dataset used for the evaluation was created from the Vienna City Library's collection of posters. It is a set of more than 5000 posters scanned in high resolution. These include, for example, different advertisements, documents, and political messages. For the experiments, I chose and labeled almost 130 images with a bounding box containing each face and five landmark points representing the center of each eye, nose, and mouth corners (see example of these images in Figure 4.2). Among these are images containing multiple faces, faces with occlusion, variable background, age, ethnicity, and lighting conditions. These images were labeled using Label Studio. This open source labelling tool, however, does not allow to do both bounding box and keypoint labelling simultaneously. Therefore, I

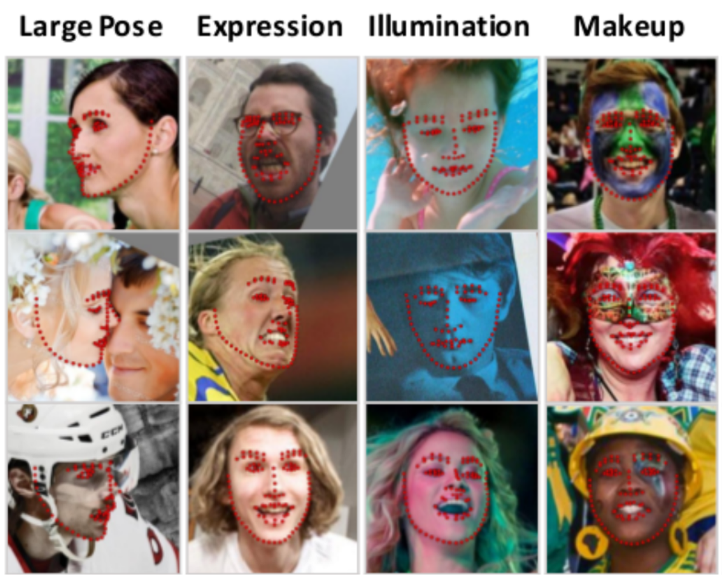


Figure 4.1: Examples of images in challenging subsets of WFLW [40].

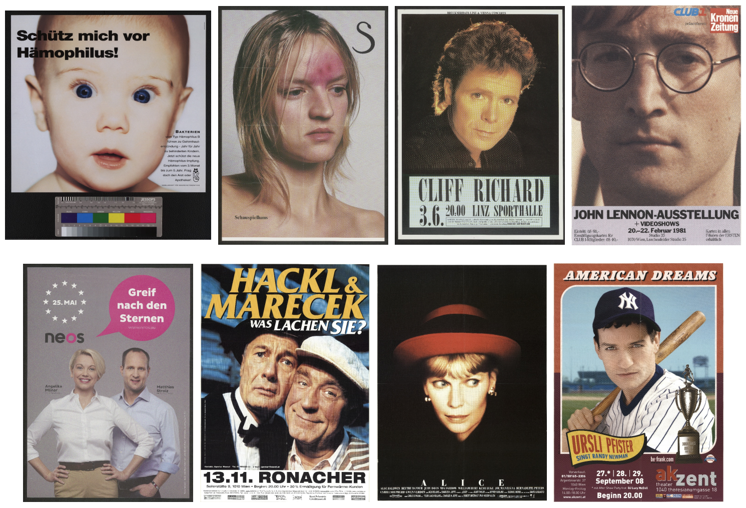


Figure 4.2: Example of posters from the Vienna City Library, which were used to create a dataset for experiments.

used boxes to mark facial landmarks and extracted only the left corner point of the rectangle which was placed in the sought after keypoint (see example in Figure 4.3).

4.2 Evaluation metrics

One of the most common metrics used for the evaluation of face and facial landmark detection is *normalized mean squared error* (NME) defined as:

$$\mathrm{NME} = \frac{1}{K}\sum_{k=1}^{K}\mathrm{NME}_k, \tag{4.1}$$

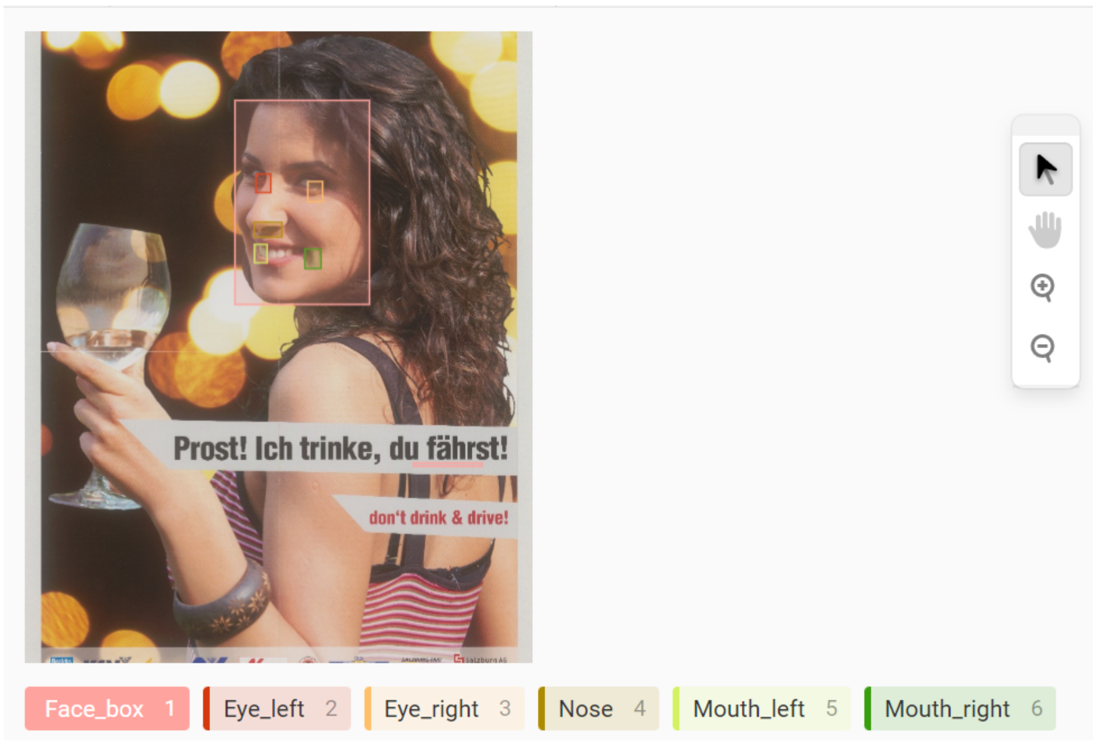


Figure 4.3: Labelling setup for the Vienna City Library dataset.

where

$$\mathrm{NME}_k = \frac{1}{N_L} \sum_{i=1}^{N_L} \frac{\left\| Y_i - \hat{Y}_i \right\|}{d} \times 100. \tag{4.2}$$

K denotes the overall number of images in dataset, $\hat{Y}$ and Y are experimental and the ground truth coordinates of given landmarks, d is a constant for each set called normalization coefficient and N_L is the number of landmarks for given facial region.

Another useful metric showing the number of images where the NME exceeds 10 % is called the *failure rate* (FR) and is defined as:

$$FR = \frac{1}{K} \sum_{k=1}^{K} [NME_k \geq 10\ \%], \tag{4.3}$$

using the same notation as (4.1) [41].

The *intersection over union* (IoU) is calculated from the ground truth box and the bounding box produced by experimental results. It is used in object detection to determine whether boxes overlap (see Figure 4.4) and is defined as [42]:

$$\mathrm{IoU} = \frac{\text{predicted} \cap \text{ground truth}}{\text{predicted} \cup \text{ground truth}}. \tag{4.4}$$

I use this ratio to conclude if the method correctly identified the facial region. While evaluating the face detection accuracy, the IoU is first calculated and then based on a minimum threshold, which was set for this metric, true positives (tp), false positives (fp) and false negatives (fn) in face detection are

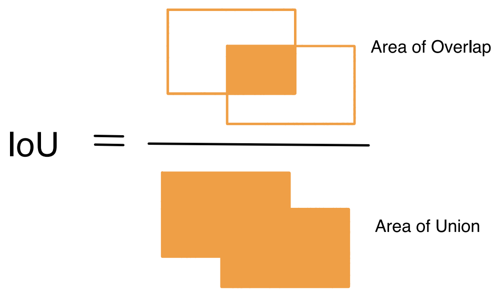


Figure 4.4: Visualization of intersection over union of two bounding boxes.

determined. Using this information, precision, recall and F1 score metrics can be calculated according to following equations [43]:

$$\text{Precision} = \frac{tp}{tp + fp}, \tag{4.5}$$

$$\text{Recall} = \frac{tp}{tp + fn}, \tag{4.6}$$

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{4.7}$$

4.3 Results

Facial Landmark Detection

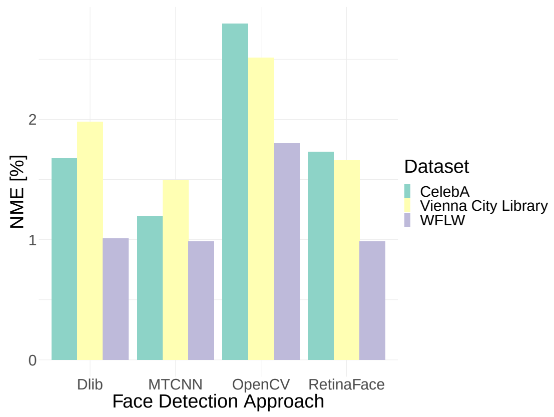
The NME (with normalization coefficient d in Equation (4.2) set to the distance between the eye centres) was calculated from correctly detected faces according to IoU (at least 50%) and the results for different datasets are shown in Table 2.2.

Table 4.1: NME [%] for tested face detection approaches on given datasets.

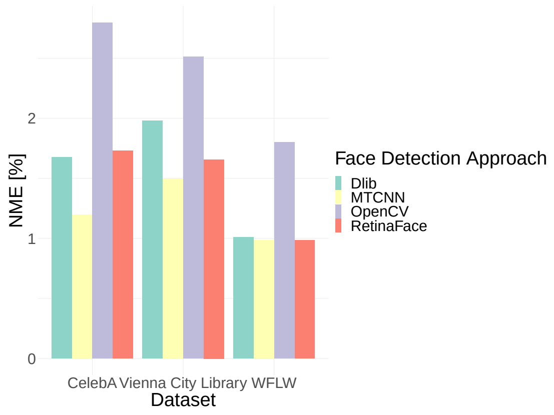
Face Detection	**Dataset**		
Approach	*WFLW*	*CelebA*	*Vienna City Library*
RetinaFace	0.9843	1.7279	1.657
MTCNN	0.985	1.1971	1.4929
Viola-Jones	1.8003	2.7953	2.5119
HOG+SVM	1.0089	1.6744	1.9811

Table 4.1 and graphs in Figure 4.5 show that for almost every dataset, the MTCNN approach yields the best results from the NME metric. Only RetinaFace was able to achieve a lower score in the WFLW dataset, differing only in the thousandth of a percent. Thus, MTCNN is the best performing method for a given task. The worst performing approach according to NME is the Viola-Jones detector yielding by far the worst score on all datasets.

The ability of chosen face detection methods to deal with different detection challenges was tested on the subsets of WFLW and CelebA, and the resulting NME metric for each approach is shown in Table 4.2.



(a) Bar graph showing how the NME metric differs for each method on chosen datasets.



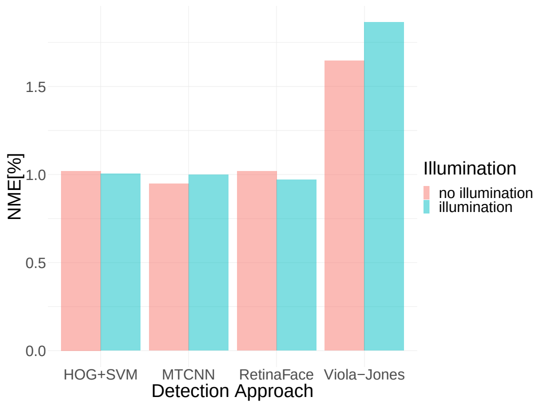
(b) Bar graph showing how the NME metric differs between chosen face detection approaches on each dataset.

Figure 4.5: Visualization of data from Table 4.1 in bar graphs.

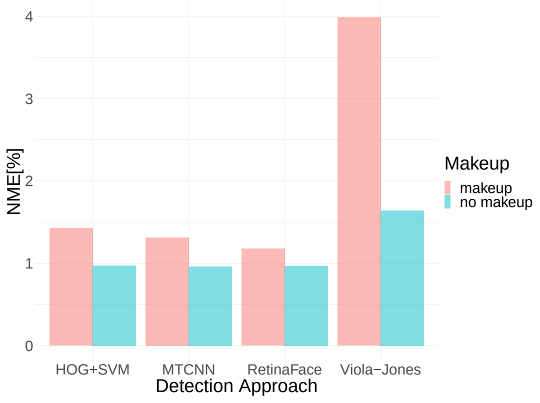
Table 4.2: NME [%] for detection on more challenging images from datasets (WFLW, CelebA) facing less ideal conditions.

Face Detection Approach	**Challenge** *Illumination*	*Makeup*	*Expression*	*Pose*	*Glasses*	*Blur*
RetinaFace	1.0191	1.1757	1.10157	1.1596	2.1778	1.5798
MTCNN	0.9475	1.3133	1.0216	1.7556	1.6834	1.1619
Viola-Jones	1.6456	3.9853	1.526	4.184	3.6929	2.1528
HOG+SVM	1.0194	1.4246	1.0828	1.5974	2.2345	1.5681

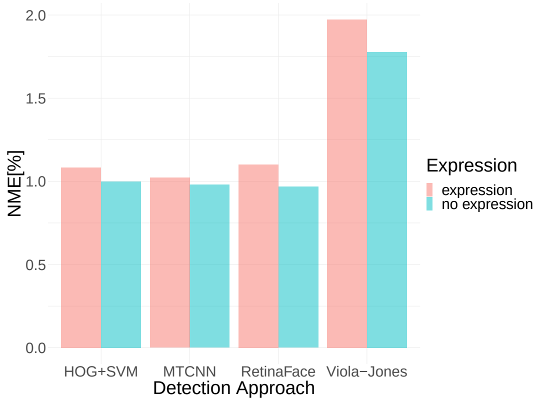
Figure 4.6 shows the NME in Table 4.2 and the same metric on the datasets without the images used for these calculations to visualize their difference. As can be seen from these results, illumination has almost no impact on any of these landmark detection approaches. Faces with heavy makeup pose a challenge for the Viola-Jones detector, causing the error to increase by more than 2%. Unusual facial expressions cause very little increase in NME for all the detectors, making them very well suited for detecting facial landmarks in these conditions. Viola - Jones detector appears to be less suited for detecting these keypoints on non-frontal faces. This condition causes its NME to increase by approximately 4% compared to the only frontal face dataset, which might be because of its feature based character. MTCNN, RetinaFace and HOG+SVM have the biggest rise in NME when faces are partially covered with glasses. This error drops to almost zero percent for these methods, when images containing the mentioned attributes are excluded. Although these methods work better in images where faces are not covered with eyewear, their NME is very low even when such situations occur. Experiments with blurry images showed that this condition is less likely to cause problems in facial landmark detection using chosen methods. The NME in Figure 4.8f appears slightly higher for images that are not blurry for some methods. This might be due to the higher complexity of this subset compared to the rest of the images.



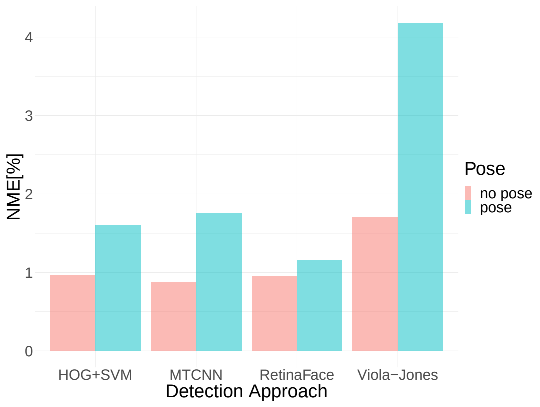
(a) NME calculated on WFLW dataset images with and without illumination.



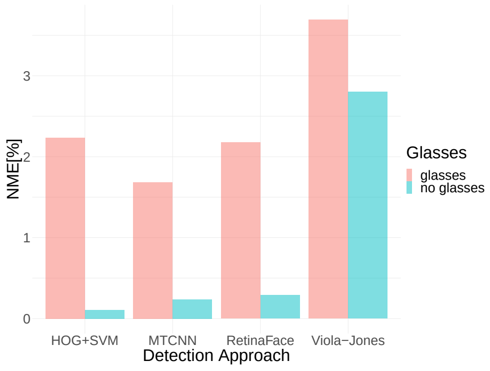
(b) NME calculated on WFLW dataset images containing faces with and without heavy makeup.



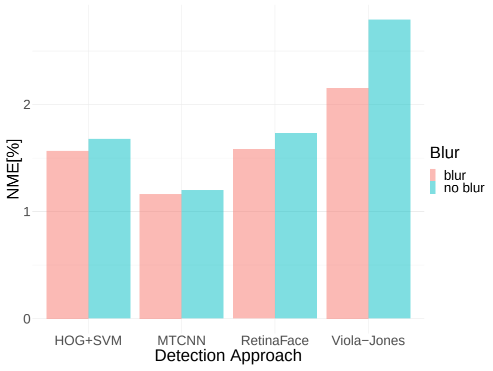
(c) NME on images with unusual and more natural facial expressions from WFLW dataset.



(d) NME on images with frontal view against subset with more complicated poses.



(e) NME calculated on CelebA dataset images containing faces with and without.



(f) NME calculated on blurry and clear images from CelebA dataset.

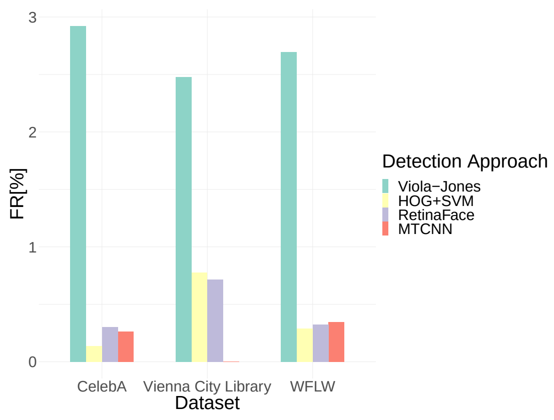
Figure 4.6: NME metric for different approaches on subset of given datasets with and without common face detection challenges.

With the use of NME, the failure rate was also calculated on given datasets corresponding to Equation (4.3), and is shown in Table 4.3. HOG+SVM outperformed other approaches in the FR metric on almost all the datasets. This shows that while this approach is not the most effective overall, it is the

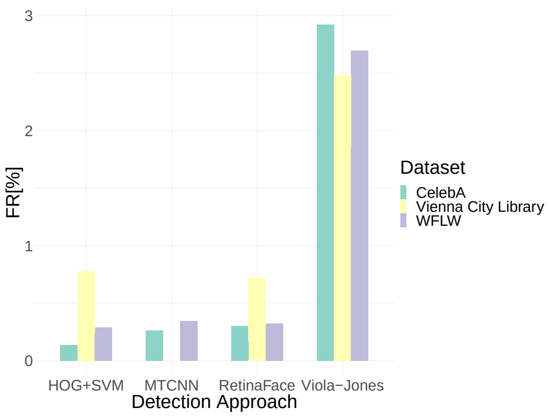
Table 4.3: FR [%] calculated using the NME for each method on all datasets.

Face Detection	**Dataset**		
Approach	*WFLW*	*CelebA*	*Vienna City Library*
RetinaFace	0.3238	0.30122	0.7143
MTCNN	0.3454	0.2621	0
Viola-Jones	2.6946	2.9215	2.4793
HOG+SVM	0.2894	0.1374	0.7752

least likely to fail to detect the approximate location of facial landmarks. However, the failure rate for MTCNN and RetinaFace was very close to HOG+SVM, showing very good results for these methods.



(a) Bar graph showing how the FR metric differs for each method on chosen datasets.



(b) Bar graph showing how the FR metric differs between chosen face detection approaches on each dataset.

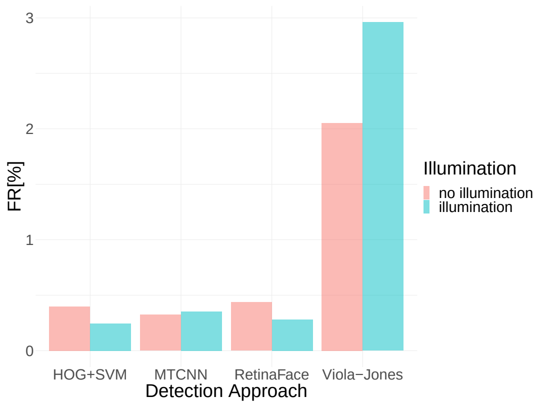
Figure 4.7: Visualization of data from Table 4.3 in bar graphs.

FR was also calculated for all challenging subsets and the results are shown in Table 4.4 and Figure 4.8.

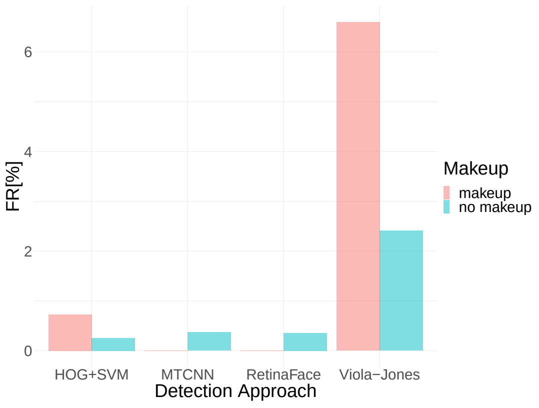
Table 4.4: FR [%] for detection on more challenging images from datasests (WFLW, CelebA) facing less ideal conditions.

Face Detection	**Challenge**					
Approach	*Illumination*	*Makeup*	*Expression*	*Pose*	*Glasses*	*Blur*
MTCNN	0.3236	0	0.3378	1.7301	0.6275	0.101
RetinaFace	0.4367	0	0.6472	0.6211	0.4691	0.1998
HOG+SVM	0.3976	0.7246	0	1.9802	0.5978	0.1529
Viola-Jones	2.0513	6.5934	2.5477	11.7647	4.3967	1.6949

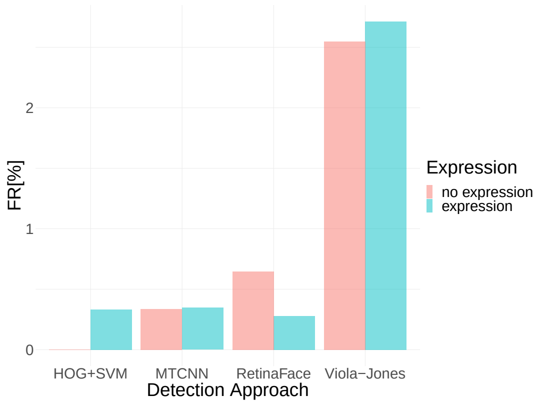
These results show again that Viola-Jones fails in detecting facial landmarks much more often for images containing faces with makeup, glasses, and non-frontal poses. According to this metric, other approaches are better suited to deal with these conditions.



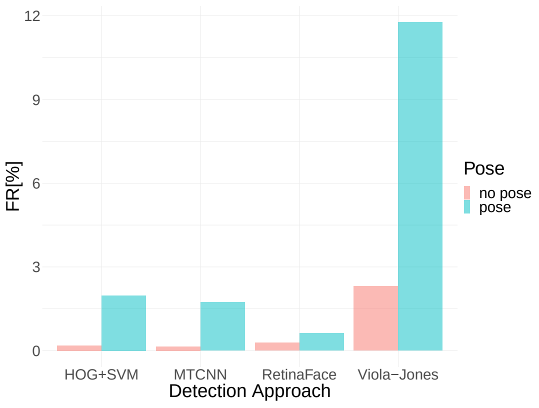
(a) FR calculated on WFLW dataset images with and without illumination.



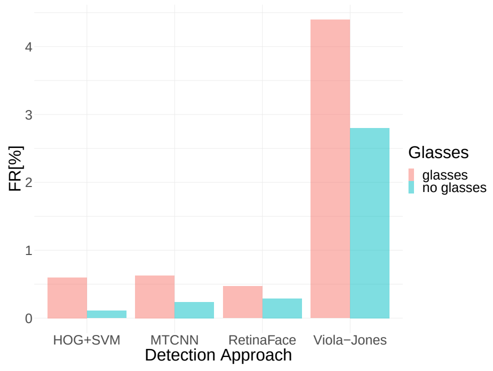
(b) FR calculated on WFLW dataset images containing faces with and without heavy makeup.



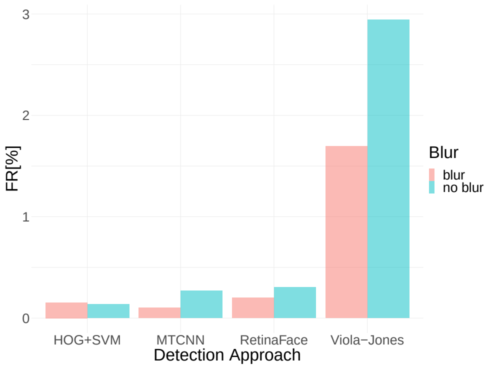
(c) FR on images with unusual and more natural facial expressions from WFLW dataset.



(d) FR on images with frontal view against subset with more complicated poses.



(e) FR calculated on CelebA dataset images containing faces with and without.



(f) FR calculated on blurry and clear images from CelebA dataset.

Figure 4.8: FR metric for different approaches on subset of given datasets with and without common face detection challenges.

Face Box Detection

Intersection over union was calculated for each facial region to determine the number of true positives, false positives, and false negatives in face detection. The threshold for this was set at least 50% for acceptable detection and then 75% (more precise detection). Both percentages of IoU were used to test the algorithms precision in face detection. By observing the WFLW dataset, I found that it included a larger number of unlabeled faces. Therefore, I decided to calculate the following metrics only on the Vienna City Library and CelebA datasets. Table 4.5 shows the precision, recall, and F1 score metrics for the face detection approaches tested. The results show that the HOG+SVM method outperformed all

Table 4.5: Precision, recall and F1 score for bounding box detection on Vienna City Library and CelebA datasets.

Face Detection Approach	minimum IoU	Dataset	Precision	Recall	F1 score
MTCNN	0.5	CelebA	96.45%	99.40%	97.90%
RetinaFace	0.5	CelebA	95.04%	99.79%	97.36%
HOG+SVM	0.5	CelebA	99.27%	94.80%	96.98%
Viola-Jones	0.5	CelebA	91.24%	87.06%	89.10%
MTCNN	0.5	Vienna City Library	97.74%	94.89%	96.30%
RetinaFace	0.5	Vienna City Library	88.05%	97.90%	92.72%
HOG+SVM	0.5	Vienna City Library	100.00%	95.56%	97.73%
Viola-Jones	0.5	Vienna City Library	88.32%	91.67%	89.96%
MTCNN	0.75	CelebA	96.03%	99.39%	97.68%
RetinaFace	0.75	CelebA	94.60%	99.78%	97.12%
HOG+SVM	0.75	CelebA	99.22%	94.79%	96.96%
Viola-Jones	0.75	CelebA	89.80%	86.88%	88.32%
MTCNN	0.75	Vienna City Library	97.74%	94.89%	96.30%
RetinaFace	0.75	Vienna City Library	87.42%	97.89%	92.36%
HOG+SVM	0.75	Vienna City Library	100.00%	95.56%	97.73%
Viola-Jones	0.75	Vienna City Library	88.32%	91.67%	89.96%

the other methods in the precision metric on the remaining datasets. Thus, this experiment demonstrated that the method is the most precise of all models in detecting faces, producing the smallest number of false positives. RetinaFace had lower precision on the created Vienna City Library dataset compared to CelebA. This decline was caused by images that contained very small faces, which were not even noticed during the labeling process, only upon closer inspection during evaluation. This dataset is also the smallest in number of images, thus these errors might be more apparent in the results. As it can be seen from Table 4.5, the recall for this method is the highest on every dataset. Thus, this approach is the least likely of all to miss a face in an image. HOG+SVM had a worse recall metric than precision, meaning that it is more likely to not detect a face than produce a false positive. Although RetinaFace and HOG+SVM were the best preforming detectors in given metrics, MTCNN was very close in both metrics, showing excellent results in face box detection. The Viola - Jones detector had the lowest precision as well as recall from all the methods. This means that it had the highest rate of false detections as well as the number of faces that were not detected, making this method the least effective. The F1 score metric combines both precision and recall. From this metric, it is evident that the MTCNN and HOG+SVM are the best performing approaches in face detection on both datasets with RetinaFace outperforming them on the CelebA set.

Precision, recall, and F1 metrics were also calculated for CelebA subsets containing faces with glasses and blurry images alone. This experiment aims to determine whether these conditions affect face box detection. The results are shown in Tables 4.6 and 4.7

Table 4.6: Performance metrics calculated on the subset of images with faces wearing glasses from the CelebA dataset.

Face Detection Approach	minimum IoU	Precision	Recall	F1 score
MTCNN	0.5	96.89%	98.92%	97.90%
RetinaFace	0.5	95.61%	99.23%	97.39%
HOG+SVM	0.5	99.40%	90.83%	94.92%
Viola-Jones	0.5	91.88%	75.78%	83.06%
MTCNN	0.75	96.06%	98.91%	97.46%
RetinaFace	0.75	94.87%	99.22%	97.00%
HOG+SVM	0.75	99.40%	90.83%	94.92%
Viola-Jones	0.75	90.46%	75.49%	82.30%

Table 4.7: Performance metrics calculated on the subset of blurred images from the CelebA dataset.

Face Detection Approach	minimum IoU	Precision	Recall	F1 score
MTCNN	0.5	94.28%	98.02%	96.11%
RetinaFace	0.5	91.01%	99.11%	94.89%
HOG+SVM	0.5	99.39%	64.68%	78.37%
Viola-Jones	0.5	93.95%	75.80%	83.90%
MTCNN	0.75	93.61%	98.00%	95.76%
RetinaFace	0.75	90.10%	99.10%	94.39%
HOG+SVM	0.75	99.39%	64.68%	78.37%
Viola-Jones	0.75	92.59%	75.53%	83.19%

These experiments revealed that HOG+SVM and Viola-Jones are more likely to fail at detecting faces partially covered with glasses and faces in blurry images. The remaining methods are unaffected by these conditions in terms of face box detection, showing very high values in all performance metrics.

4.4 Experiment Evaluation

Publicly availble implementations of four face detection approaches (MTCNN, RetinaFace, Viola-Jones, HOG+SVM) that produced (at least) five facial landmarks and bounding boxes were tested on three different datasets (CelebA, WFLW, Vienna City Library). The normalized mean error, failure rate, precision, recall, and F1 score metric were used to evaluate the performance of these methods on all image sets. Subsequently, CelebA and WFLW subsets containing images with common face detection challenges were extracted and used to evaluate these methods under more complicated conditions. MTCNN outperformed other methods in NME with RetinaFace and HOG+SVM yielding very similar results. HOG+SVM achieved the lowest failure rate in almost all datasets, showing that it rarely fails (has NME higher than 10%) in landmark detection. Both the RetinaFaces and MTCNNs results were in close proximity to HOG+SVM with Viola-Jones again having the worst performance. Testing facial landmark detection on more challenging subsets showed that Viola-Jones is highly sensitive to face pose changes, faces with heavy makeup, and faces wearing glasses. RetinaFace, MTCNN and HOG+SVM achieved very low NME values and FR metric values on all of these subsets, only showing a little higher NME values on subset with glasses present. For face box detection, IoU was calculated to determine the precision of the facial bounding box. Setting the threshold for IoU to 50% and 75% allowed me to determine the number of true positives, false positives and false negatives, which led to the calculation of precision, recall, and F1 score. These results showed that HOG+SVM was most precise while RetinaFace achieved the best recall values. Overall MTCNN, RetinaFace and HOG+SVM were the best at detecting faces in image, while Viola-Jones showed to be less successful at this task with F1 score on all datasets under 90%. These metrics evaluated on challenging subsets remained high for all of the approaches except Viola-Jones, which turned out to be more sensitive to blur and people wearing glasses when detecting faces. HOG+SVM recall also seemed to decrease to little over 90% when the faces were partially covered with glasses and to approximately 64% for blurry images. These results show that the best-suited approaches for face detection in an unconstrained environment were MTCNN and RetinaFace. HOG+SVM would be best suited for face detection on images that are not blurred, and Viola-Jones is the most likely to fail at this task in more challenging conditions.

Conclusion

The goal of this research thesis was to describe common face detection approaches, choose the most suitable ones for detection under unconstrained conditions, and test them on appropriate datasets. For the experiments, I selected publicly available implementations of MTCNN, RetinaFace, histogram of oriented gradients and linear SVM detector (implemented in dlib) and the Viola-Jones detector (OpenCV implementation). Datasets selected for this task were the testing sets from WFLW, CelebA, and a dataset created from the Vienna City Library collection of posters, which I manually labeled with face bounding box and five facial landmark coordinates (representing the centre of left eye, the centre of right eye, tip of the nose, left mouth corner, and right mouth corner). Subsets from the CelebA and WFLW sets containing more challenging images (with blur, illumination, heavy makeup, non-frontal poses, and glasses present) were extracted to determine the capability of chosen approaches to deal with these conditions. All these publicly available methods were tested on almost 22 630 images using Python.

The results show that most of these detectors perform well under various conditions. MTCNN, RetinaFace and HOG+SVM achieved high accuracy in bounding box detection on CelebA and Vienna City Library datasets, with Viola-Jones reaching the lowest precision, recall, and F1 score metrics on almost all the sets. Lower precision values for RetinaFace on the Vienna City Library set was caused by this method being able to detect very small faces, which were hardly noticeable by the human eye and therefore remained unlabelled. On more challenging subsets of CelebA, RetinaFace and MTCNN performed very well, achieving high performance metric values for bounding box detection. Viola-Jones reached low recall values for both the subset with blurred images and the subset containing images of people with glasses. This means that these conditions cause the detector to fail in finding faces. This shortcoming for the Viola-Jones detector was however anticipated, since the method is feature-based. HOG+SVM also performed poorly in terms of recall on these subsets, which might also be a result of its feature-based character.

The detection of facial landmarks was tested using the NME and FR metrics. Both were calculated from images where the detectors were able to correctly identify facial regions and normalized by the ground truth distance between eye centres. All methods achieved a relatively low NME on the datasets, with Viola-Jones having the highest NME of all on the CelebA dataset of almost 3%. NME for the subsets of images showed that all the detectors were successful in accurately locating facial landmarks under difficult conditions. Viola-Jones again reached highest NME on dataset containing faces with heavy makeup, dataset of faces with glasses and non-frontal faces. According to the failure rate calculations, it is rare for the chosen face detection approaches to fail at locating facial landmarks even in the more difficult conditions. Viola-Jones was found to have the highest failure rate of all detectors. It was more prone to fail to detect facial landmarks when difficult poses or heavy makeup was present.

MTCNN, RetinaFace and HOG+SVM methods were found to be well suited for facial landmark localization even under challenging conditions. However, HOG+SVM struggled in face detection under less ideal circumstances. Even though Viola-Jones was the worst performing method, results show that it would be a good option for face detection and landmark localization under more optimal conditions. The experiments demonstrated that in an unconstrained environment, neural network approaches (MTCNN, RetinaFace), work the best for face detection and landmark localization.

Results show that RetinaFace and MTCNN are very well suited for face detection under unconstrained conditions, meaning that these methods would have the potential to be successful in real world applications. However, in some applications, computing time of these methods might be of essence. This thesis does not contain experiments that compare said characteristic which might alter the suitability of tested approaches for some tasks.

Face detection is a stepping stone to a lot of face-related computer vision application. These methods might be further used in tasks such as face recognition, authentication or facial analysis in the real world conditions.

Bibliography

[1] Ashu Kumar, Amandeep Kaur, and Munish Kumar. “Face detection techniques: a review.” In: *Artificial Intelligence Review* 52.2 (2019), pp. 927–948.

[2] Stefanos Zafeiriou, Cha Zhang, and Zhengyou Zhang. “A survey on face detection in the wild: past, present and future.” In: *Computer Vision and Image Understanding* 138 (2015), pp. 1–24.

[3] Mayank Chauhan and Mukesh Sakle. “Study & analysis of different face detection techniques.” In: *International Journal of Computer Science and Information Technologies* 5.2 (2014), pp. 1615–1618.

[4] Ming-Hsuan Yang, David J Kriegman, and Narendra Ahuja. “Detecting faces in images: A survey.” In: *IEEE Transactions on pattern analysis and machine intelligence* 24.1 (2002), pp. 34–58.

[5] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[6] Javaid Nabi. *Machine Learning —Fundamentals*. en. May 2019. URL: `https://towardsdatascience.com/machine-learning-basics-part-1-a36d38c7916` (visited on 07/01/2022).

[7] Jan Flusser, Tomáš Suk, and Barbara Zitová. *2D and 3D Image Analysis by Moments*. English. 1st edition. Chichester, West Sussex, United Kingdom ; Hoboken, NJ: Wiley, Dec. 2016. ISBN: 978-1-119-03935-8.

[8] Milan Šonka, Václav Hlaváč, and Roger Boyle. *Image Processing, Analysis, and Machine Vision*. English. 4th edition. Stamford, CT, USA: Cengage Learning, Jan. 2014. ISBN: 978-1-133-59360-7.

[9] David G. Stork Richard O. Duda Peter E. Hart. *Pattern classification*. Wiley Hoboken, 2000.

[10] Matt Brems. *A one-stop shop for principal component analysis*. Jan. 2022. URL: `https://towardsdatascience.com/a-one-stop-shop-for-principal-component-analysis-5582fb7e0a9c` (visited on 08/20/2022).

[11] Casey Cheng. *Principal Component Analysis (PCA) explained visually with Zero math*. Mar. 2022. URL: `https://towardsdatascience.com/principal-component-analysis-pca-explained-visually-with-zero-math-1cbf392b9e7d` (visited on 08/20/2022).

[12] Aaron Hertzmann, David J. Fleet, and Marcus Brubaker. *AdaBoost - Department of Computer Science, University of Toronto*. 2015. URL: `https://www.cs.toronto.edu/~mbrubake/teaching/C11/Handouts/AdaBoost.pdf` (visited on 08/20/2022).

[13] Rafael Gonzalez and Richard Woods. *Digital Image Processing*. English. 4th edition. New York, NY: Pearson, Mar. 2017. ISBN: 978-0-13-335672-4.

[14] Sumit Saha. *A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way*. en. Dec. 2018. URL: `https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53` (visited on 05/03/2022).

[15] Keiron O'Shea and Ryan Nash. "An introduction to convolutional neural networks." In: *arXiv preprint arXiv:1511.08458* (2015).

[16] Brendan F. Klare et al. "Pushing the frontiers of unconstrained face detection and recognition: Iarpa janus benchmark a." In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 1931–1939.

[17] Martin Koestinger et al. "Annotated Facial Landmarks in the Wild: A Large-scale, Real-world Database for Facial Landmark Localization." In: *Proc. First IEEE International Workshop on Benchmarking Facial Image Analysis Technologies*. 2011.

[18] Vignesh Suresh. *Face detection and landmarks using dlib and opencv*. June 2021. URL: `https://vigneshs4499.medium.com/face-detection-and-landmarks-using-dlib-and-opencv-8c824f50cc78` (visited on 08/20/2022).

[19] Jimiama M. Mase et al. "Towards Privacy-Preserving Affect Recognition: A Two-Level Deep Learning Architecture." In: *arXiv preprint arXiv:2111.07344* (2021).

[20] Arnaud Dapogny, Kevin Bailly, and Matthieu Cord. "Deep Entwined Learning Head Pose and Face Alignment Inside an Attentional Cascade with Doubly-Conditional fusion." In: *2020 15th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2020)*. IEEE. 2020, pp. 192–198.

[21] Ziwei Liu et al. "Deep Learning Face Attributes in the Wild." In: *Proceedings of International Conference on Computer Vision (ICCV)*. Dec. 2015.

[22] Oliver Jesorsky, Klaus J. Kirchberg, and Robert W. Frischholz. "Robust face detection using the hausdorff distance." In: *International conference on audio-and video-based biometric person authentication*. Springer. 2001, pp. 90–95.

[23] Kieron Messer et al. "XM2VTSDB: The extended M2VTS database." In: *Second international conference on audio and video-based biometric person authentication*. Vol. 964. Citeseer. 1999, pp. 965–966.

[24] Stephen Milborrow, John Morkel, and Fred Nicolls. "The MUCT landmarked face database." In: *Pattern recognition association of South Africa* 201.0 (2010).

[25] Erik Hjelmås and Boon Kee Low. "Face detection: A survey." In: *Computer vision and image understanding* 83.3 (2001), pp. 236–274.

[26] Md Khaled Hasan et al. "Human face detection techniques: A comprehensive review and future research directions." In: *Electronics* 10.19 (2021), p. 2354.

[27] Kaiqi Cen. *Study of Viola-Jones Real Time Face Detector*. `https://web.stanford.edu/class/cs231a/prev_projects_2016/cs231a_final_report.pdf`. (Visited on 08/20/2022).

[28] Darshan Adakane. *What are Haar features used in face detection?* Nov. 2019. URL: `https://medium.com/analytics-vidhya/what-is-haar-features-used-in-face-detection-a7e531c8332b` (visited on 08/20/2022).

[29] Ankit Srivastava et al. "A survey of face detection algorithms." In: *2017 International Conference on Inventive Systems and Control (ICISC)*. IEEE. 2017, pp. 1–4.

[30] Sriman K. P. et al. "Comparison of Paul Viola–Michael Jones algorithm and HOG algorithm for Face Detection." In: *IOP Conference Series: Materials Science and Engineering*. Vol. 1084. 1. IOP Publishing. 2021, p. 012014.

[31] Satya Mallick. *Histogram of oriented gradients explained using opencv*. Nov. 2021. URL: `https://learnopencv.com/histogram-of-oriented-gradients/`.

[32] Kaipeng Zhang et al. “Joint face detection and alignment using multitask cascaded convolutional networks.” In: *IEEE signal processing letters* 23.10 (2016), pp. 1499–1503.

[33] Jiankang Deng et al. “Retinaface: Single-shot multi-level face localisation in the wild.” In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 5203–5212.

[34] Jonathan Hui. *Understanding Feature Pyramid Networks for Object Detection (FPN)*. Apr. 2020. URL: `https://jonathan-hui.medium.com/understanding-feature-pyramid-networks-for-object-detection-fpn-45b227b9106c` (visited on 08/20/2022).

[35] Vincent Feng. *An overview of ResNet and its variants*. July 2017. URL: `https://towardsdatascience.com/an-overview-of-resnet-and-its-variants-5281e2f56035` (visited on 08/20/2022).

[36] Divyanshu Mishra. *Deformable convolutions demystified*. July 2021. URL: `https://towardsdatascience.com/deformable-convolutions-demystified-2a77498699e8` (visited on 08/20/2022).

[37] Sefik Ilkin Serengil and Alper Ozpinar. “HyperExtended LightFace: A Facial Attribute Analysis Framework.” In: *2021 International Conference on Engineering and Emerging Technologies (ICEET)*. IEEE. 2021, pp. 1–4. DOI: `10.1109/ICEET53442.2021.9659697`. URL: `https://doi.org/10.1109/ICEET53442.2021.9659697`.

[38] Ipazc. *Ipazc/MTCNN: MTCNN face detection implementation for tensorflow, as a pip package.* URL: `https://github.com/ipazc/mtcnn`.

[39] Serengil. *Serengil/Retinaface: RetinaFace: Deep Face Detection Library for python*. URL: `https://github.com/serengil/retinaface` (visited on 08/20/2022).

[40] Wayne Wu et al. “Look at Boundary: A Boundary-Aware Face Alignment Algorithm.” In: *CVPR*. June 2018.

[41] Kostiantyn Khabarlak and Larysa Koriashkina. “Fast Facial Landmark Detection and Applications: A Survey.” In: *Journal of Computer Science and Technology* 22.1 (Apr. 2022), p. 30. DOI: `10.24215/16666038.22.e02`. URL: `https://arxiv.org/pdf/2101.10808.pdf`.

[42] Ziyuan Yang et al. “Real-Time Pre-Identification and Cascaded Detection for Tiny Faces.” In: *Applied Sciences* 9.20 (2019). ISSN: 2076-3417. DOI: `10.3390/app9204344`. URL: `https://www.mdpi.com/2076-3417/9/20/4344`.

[43] Harikrishnan N B. *Confusion matrix, accuracy, precision, recall, F1 score*. June 2020. URL: `https://medium.com/analytics-vidhya/confusion-matrix-accuracy-precision-recall-f1-score-ade299cf63cd` (visited on 08/20/2022).