FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

Gesichtserkennung in uneingeschränkten Video streams

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Medieninformatik und Visual Computing

eingereicht von

Philipp Omenitsch, B.Sc.
Matrikelnummer 1025659

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: O. Univ. Prof. Dr. Walter Kropatsch
Mitwirkung: Prof. Ivan Laptev

Wien, 24. August 2016

Philipp Omenitsch | Walter Kropatsch

FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

Face recognition in unconstrained video streams

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieur

in

Media Informatics and Visual Computing

by

Philipp Omenitsch, B.Sc.
Registration Number 1025659

to the Faculty of Informatics

at the TU Wien

Advisor: O. Univ. Prof. Dr. Walter Kropatsch
Assistance: Prof. Ivan Laptev

Vienna, 24th August, 2016

Philipp Omenitsch | Walter Kropatsch

Erklärung zur Verfassung der Arbeit

Philipp Omenitsch, B.Sc.
Untere Augartenstraße 2 / 6, 1020 Wien

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 24. August 2016

Philipp Omenitsch

Danksagung

Ich möchte mich recht herzlich bei Prof. Ivan Laptev für die großartige Betreuung und enge Kooperation bedanken. Er hat mir sehr dabei geholfen einen tiefen Einblick in die Kluft zwischen Industrie und Forschung und die domänenspezifischen Probleme der Gesichtserkennung in der echten Welt zu erlangen. Ich möchte auch Sergey Milyaev danken, der immer ein offenes Ohr für meine Ideen hatte und nie müde wurde, mir Dinge zu erklären. Ich danke Aleksander Khanin für die Möglichkeit meine Masterarbeit in Kooperation mit der Firma VisionLabs in Moskau, Russland durchzuführen. Ich bedanke mich auch bei all meinen Freunden in der Firma VisionLabs für die angenehme Arbeitsatmosphäre und die interessanten Diskussionen, ich wünsche euch alles Gute. Weiters bedanke ich mich bei Prof. Walter Kropatsch der Technischen Universität Wien für seine Betreuung und Flexibilität, er hat die Kooperation mit VisionLabs möglich gemacht und ich bin ihm dafür sehr dankbar.

Acknowledgements

I would like to thank Ivan Laptev for the great supervision and the close collaboration. He helped me get a good insight into the gap between industry and research and the domain specific problems of face recognition in real world applications. I also want to thank Sergey Milyaev for bouncing ideas and never getting tired of explaining me things. I thank Aleksander Khanin for the possibility to conduct my master thesis in cooperation with the company VisionLabs in Moscow, Russia. I also thank all my friends at VisionLabs for the good working atmosphere and interesting discussions, I wish you all the best. Furthermore, I want to thank Prof. Walter Kropatsch from the Technical University of Vienna for his supervision and flexibility, he made the cooperation with VisionLabs possible and I am very thankful for that.

Kurzfassung

Das Thema Gesichtserkennung ist gibt es schon lange in der Mustererkennung. Die jüngsten Fortschritte versprechen erfolgreiche Anwendungen für Videoüberwachung in großem Maßstab. Das Erkennen von Millionen von Menschen aufgenommen von tausenden Kameras birgt aber auch signifikante Kosten bezüglich Rechenleistung und Datenkommunikation. Diese Arbeit versucht die Komplexität von Gesichtserkennung in Videos zu reduzieren und verfolgt zwei komplementäre Strategien. Einerseits untersuchen Gesichtserkennung im "whitened"Deskriptorraum und verbessern die Komplexität der state-of-the-art Matched Background Similarity (MBGS) Methode. Andererseits untersuchen wir den Effekt den das Auswählen einer Teilmenge von Videobildern auf die Erkennungsgenauigkeit hat und zeigen, dass es möglich ist mit einem Bruchteil von Videobildern robuste Resultate zu erzielen. Für ein besseres Verständnis von Tiefen Faltenden Neuralen Netzwerken geben wir eine Einführung in das Thema und fassen aktuelle Entwicklungen zusammen, speziell im Bezug auf den weitbekannten ImageNet Wettbewerb. Wir evaluieren unsere Methoden und präsentieren konkurrenzfähige Resultate für den YTF und TSFT Benchmark mit signifikant niedrigerer Komplexität im Vergleich zu anderen Methoden.

Abstract

Face recognition is a long-standing topic in machine learning. Recent advances promise successful applications in large-scale video surveillance. Recognizing millions of people in thousands of streaming cameras, however, implies significant costs in terms of computations and communication bandwidth. This work aims to reduce the complexity of face recognition in video and pursues two complementary strategies. We first investigate face recognition in the whitened descriptor space and improve the complexity of the state-of-the-art Matched Background Similarity (MBGS) approach. Second, we study frame selection and demonstrate robust recognition performance from a small fraction of video frames. For a better understanding of the advancing Deep Convolution Neural Networks field, we give a thorough introduction to the topic and briefly summarize the recent development with a focus on the ImageNet classification challenge. We evaluate our methods and present competitive results for the YTF and TSFT benchmarks while significantly reducing the complexity of previous methods.

Contents

CHAPTER 1
Introduction

Recognizing people plays a crucial role in our daily life at work, at home, when watching a movie or when our face is checked at the passport control. Early on in our life, we learn to recognize our close family members and also to be careful with people we do not know. With advancing technology, the identification of people has become a possibility and necessity to automate and secure interactions between them. Most operations involving money, valuable assets or interaction with state organizations, require some kind of identification, authentication or verification. This is typically achieved either by knowing someone personally, for example, family members or friends, or a document that verifies your identity, like a passport. When interacting with machines, identification of a user can be crucial for some applications, face recognition is one possibility to make this process seamless. Consider passport controls at state borders, here a verification of identity is needed. The identity is established by the passport, which can be read by a machine and the person's identity can be verified with the help of face recognition [SHG12], this reduces the needed personal.

Automating mundane manual verification and authentication but also moving this process to a less corruptible (because automatic) system, makes face recognition a research topic with big opportunities. However, the classification of billions of classes, namely individual persons, is an ongoing research topic by itself. Additionally, the variability in appearances and aging makes face recognition an extraordinarily difficult problem. With the recent advances in computation power, Google has shown that this technology can also be used for organizing images and Facebook uses this technology for recognizing friends in images. These use cases show that there are many applications for face recognition in still images. However, there exists less research in the area of video face recognition. One main difference between still images and video is the higher variability in videos and the often cooperative setting in still images. When taking a picture of a person, it is more likely to wait for the best moment (i.e. someone looks towards the camera), because one wants to take images where the person is easy to recognize or more generally pictures are actively taken for some purpose.

On the other side, videos offer different viewpoints of a face, with different quality regarding lighting, size, occlusions, and emotions, making some frames more useful for face recognition than others. The goal of automatic face recognition is to track each person in a scene and use those frames of a recording that are best for face recognition. While it is possible to detect people in a scene, it is not easy to determine the best moment for face recognition, because there is no simple measure for face quality but rather it is influenced by many variables. This aspect becomes, even more, challeging for unconstrained video streams, this means that videos are made not with the same setup, but under varying, like in the real world, conditions. This thesis is going to tackle face recognition in video by analyzing current approaches and presenting an approach that not only increases qualitative performance but also reduces complexity, which is crucial in real world applications. We will discuss more challenges and give some examples in Chapter 5.

1.1 Biometrics

In general, there are three principles [O'G03], how a user can be authenticated against a system. The "what I know" principle means, I know something no one else knows, for example, a password to a user account. The "what I have" principle requires each person to possess a physical token that is itself unique, for example, an access token or credit card. The "who I am" principle is the most natural since it requires no artificial information or devices. A biometric system measures physical properties or behavior of a human body that can be used to uniquely identify a human being [O'G03]. Such measures include fingerprints, iris, hand geometry but also behavior such as gait, handwriting or typing on a keyboard. A complete list of possibilities can be found in Figure 1.1.

There are two types of purposes for biometric systems

- Verification: check if a person corresponds to a particular identity.
- Identification: determine the identity of a person from a given database.

For verification, only one trusted sample (for example face image in a passport) is needed. For identification, however, a database is needed that already contains ground truth (for example fingerprint database of criminals).

Biometric systems can be characterized by several properties, in order how useful they are in real world applications.

- **Cost** to obtain an identity, this includes the cost for a specific device (i.e. fingerprint sensor)

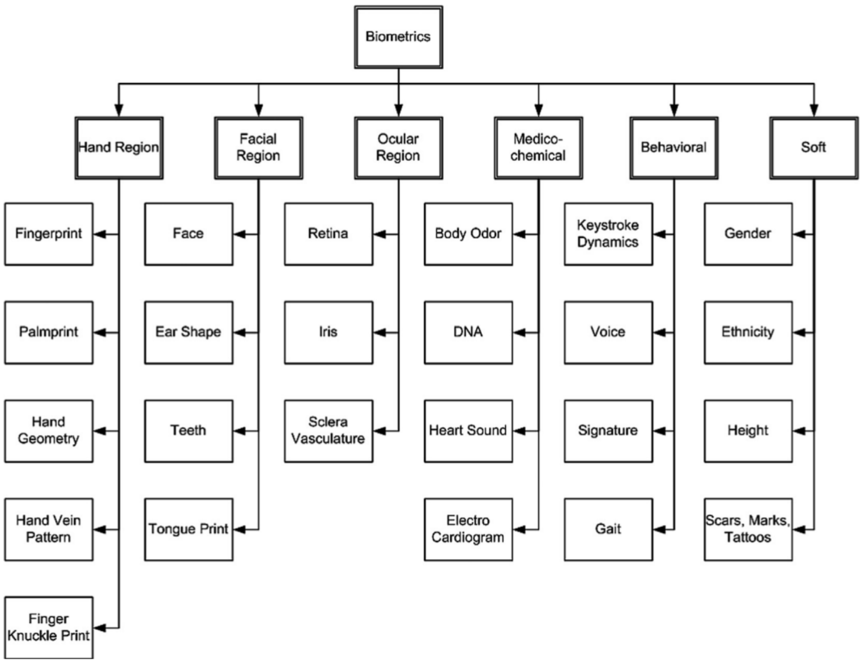


Figure 1.1: Different methods of biometric systems, categorized by body part [USA14]

- **Degree of cooperation** required to obtain an identity from uncooperative to cooperative (think people passing by a video camera versus tongue print)
- **Recognition speed**
- **Recognition accuracy**

Considering all these dimensions, face recognition is a good choice for biometric authentication. Cameras nowadays are cheap and omnipresent. Also, there already exist large databases that link the biometric identifier, the image to an identity. Some examples are government documents and social networks. The method works in an uncooperative manner, no user interaction is required, which stands in contrast to other well known biometric authentication systems such as fingerprint or iris scans, which both need the willingness and cooperation of the user to function. From a user point of view, this could also be seen as an advantage because the user knows when he is being identified. From a business or surveillance point of view, face recognition is especially interesting because the user does in many cases not know that he is being identified or tracked and the process can be fully automated on a large scale. This is, of course, the basis for a fundamental ethical debate itself, however, in this thesis, we want to focus on the technical aspects. Recent advances [PVZ15, TYRW14, SKP15] have shown face recognition rates surpassing human performance [KBBN09] in still images, which shows that the systems are capable of replacing humans for processes where verification is needed, e.g. border crossings for passports. Many institutions such as banks or governmental facilities already use video cameras for surveillance purposes, this means that the infrastructure already exists and face recognition can be employed without additional hardware costs. While face recognition in still images is a theoretical research field which is continuously moving forward, face recognition in video is also important for real world use cases and are closely related but different problem.

1.2 Goal and the structure of this thesis

The problem we want to solve in this thesis is to build the fundamentals for an automatic face recognition system for real world applications. This requires that video streams can be processed and the best possible recognition accuracy is achieved. Additionally, the system should be fast enough to recognize faces in near realtime. The goal of this thesis is to investigate methods to improve the performance of face recognition in video streams. In order to extract information from the video, first, the video has to be preprocessed to only contain faces. Furthermore, it the face has to be tracked, in case there are multiple faces in one image. The resulting set of images that contain only faces of the same person is called a face track, which consists of multiple frames. Using all frames would be the simplest solution, however, in Section 6 we will show why this is suboptimal performance wise and expensive computational wise. More precisely two methods were tried and combined. First, the selection of frames according to different criteria in order to keep the computational load reasonable and to increase recognition performance. Second, the decorrelation of features with the help of a background dataset to increase recognition performance. To evaluate results we compare our method with the original face descriptors, as well as Matched Background similarity which we explain in Section 5.1. Furthermore, we show, that with better face descriptors the effect of methods that try to model variances and subspaces created due to external factors get smaller and offer some possible explanations for this. We evaluate our approach on two databases with different face descriptors. For evaluation we define a baseline in Section 5.1 so we can compare with it. Specifically measure recognition performance as suggested by the respective databases and complexity by measuring the run time of the baseline, MBGS and our method.

This master thesis is organized as follows. First, we will explore the related work in the fields of face recognition in still images and video in Chapter 2. Next we will explain necessary concepts, Linear Discriminant models (LDA) in Section 3.1, Support Vector Machines (SVM) in Section 3.2 and Convolutional Neural Networks (CNN) in Section 3.3. We continue by sketching the pipeline for identifying a person from a raw image in Chapter 4. Furthermore, we discuss each stage of the pipeline and finally show how to improve video face recognition by frame selection and descriptor whitening in Chapter 5. Next, we report empiric results of our methods in Chapter 6 and finally reflect on and conclude the work in Chapter 7.

CHAPTER 2

Related Work

2.1 Still images

Face recognition is not a new field, rather it exists now for more than 50 years which shows the huge interest in the area. In the 1960s Woodraw Wilson Blesoe [BB91] attempted for the first time to build a semi automatic face recognition system. He soon realized that this computer vision problem was difficult to solve due to the variability in images caused by rotation, aging and facial expressions, additionally to common computer vision problems caused by lighting or focus. Early methods for still face recognition used eigenfaces and PCA [TP91, Kan77, BHK97, LVB$^+$93] on raw images or on hand-crafted features such as Scale-inariant feature transfrom (SIFT), local binary patterns (LBP), Histogram of oriented gradients (HOG) and many more for extracting a lower dimensional face representation from an image, a face descriptor. [DBSDlT11, SPVZ13, SEZ09, WHT08]. These methods were shown to work on small and constrained datasets and on aligned faces. For example, the Yale Face database [BHK97] contained 165 images of 15 people.

Simonyan et al.[SPVZ13] densely extracted SIFT features from faces and then reduced dimensionality with PCA to obtain a reasonable sized face descriptor. Sivic et al. [SEZ09] used HOG features and SVM for classification. Both works show improvements compared to earlier mentioned methods, however the features these algorithms use are hand-crafted and the recognition accuracy are not satisfying for use in real world applications.

More recently CNNs (explained in more detail in Section 3.3) have revolutionized the field of computer vision and especially image classification. Contrary earlier methods they learn features from training data and have shown substantial improvements in the field of face recognition. They do so by leveraging large training datasets with millions of images. The authors of DeepFace [TYRW14] used 3D image frontalization for face alignment, which is the input for a 8-layer CNN with 120 million parameters. FaceNet [SKP15] trained a deep CNN with 22 layers and 4.3 million parameters and a novel training method for the recognition layer, which is used without face alignment. Parkhi et al.introduced a publicly available face descriptor [PVZ15]. This neural net was trained with orders of magnitudes less training data than the afore mentioned, while achieving comparable results. We use this descriptor among others in our work. Since 2001 the evaluation dataset Labeled Faces in the Wild (LFW)[HRBLM07], has been adopted widely as defacto standard for benchmarking face recognition algorithms, it uses 13K images obtained from the internet. "In the wild", means that the pictures or videos were taken without any control over capture conditions, but on different devices in real life situations. Researchers [SWT15, TYRW14, SKP15, PVZ15] have shown 99.47%, 99.63% and 97.35% accuracy respectively. Some of these accuracies are higher than human level performance on the same dataset which was 99.2%[KBBN09]. This level of performance allows the use of face recognition in real world applications for different fields like mass surveillance, personalized marketing or fraud prevention.

2.2 Video

One of the earliest comparative studies between still image and video face recognition, was done in 2003 [PGM$^+$03], where the authors found no difference in performance between video and still image face recognition for the methods that existed at that time. Barr et al.[BBFB12] conducted an extensive inquiry in the topic of face recognition in videos. However, he reached his conclusions by analyzing easier and controlled datasets [GLLC05][LHYK05], this means they contained few people (dozens) and the images were captured under controlled conditions such as frontal pose and similar lightening.

Video face recognition can be divided into two main fields, sequence based methods that use the temporal ordering of frames [BBFB12] and set based methods, which consider the descriptors extracted from a face track as a set. Recently, set based approaches have become more popular, when looking at the literature [TYRW14, SKP15, PVZ15, BRKKJ13], which often use score fusion. By score fusion, we understand the statistical analysis of distance scores between multiple or all pairs of faces in two different video tracks. For every face pair the distance is computed and a single fused score is computed by using the average or other operators such as minimum or maximum. Score fusion is helpful at suppressing noise generated by changes of lighting, pose and face gesture in face tracks. We will call those *external factors*.

With this method at hand, it becomes simple to use face recognition in images for videos. For example, FaceNet extended their algorithm for still images to video by averaging scores of 50 randomly sampled frames of each face track. Deepface averaged the score of the first 100 frames of a video, which their face detector detected. This method helps to even out fluctuations caused by external factors, however both methods, FaceNet and Deepface are not robust against outliers and also is computationally expansive because between 100 and 200 face descriptors have to computed for on face verification.

There are many methods aiming to extract the most information possible from the set of images that belong to the same person. This can be done by building a representation, that tries to model the subspace, which is created by the descriptors extracted of one persons face exposed to external factors. Two such methods are the matched background similarity (MBGS) [WHM11] and SVM Θ [WL13], which work on the same principle. The authors propose to use a background set in order to improve classification. MBGS similarity is a set to set similarity that uses hard negative mining on the background set and SVM for score calculation. Hard negative mining is the process of finding training data samples, that the algorithm cannot classify properly and fails to do so by a large margin. The SVM is trained at inference time in a one versus all manner, with all descriptors of a face in one class and all other face descriptors (the background set) in the other class. This greatly increases recognition accuracy for LBP features as they showed in their paper [WHM11]. SVM Θ similarity is computed by the help of an auxiliary background labeled dataset and a further optimization which decorrelates data. Both methods increase qualitative performance, however at the price of orders of magnitudes larger computational cost. This poses the question if the computationally expansive part of the MBGS approach, the SVM can be replaced by something else.

Feature whitening was applied for object classification as a replacement for linear SVM [HMR12] and was proposed earlier by Wolf et al.to replace SVM in an earlier paper called Oneshot similarity [TWH$^+$09, WHT08], which contains some components of MBGS. For comparing two face tracks with feature whitening, only 2 matrix multiplications and one dot product are required as compared to the training of a SVM and the classification with a SVM. However, this comes at the price of slightly worse performance, as we will show in section 6. In their paper Arandjelov et al.[Ara16] show an approach for set based similarity, which they call quasi-transitivity, that accounts for near duplicate and not uniformly distributed features, which can influence similarity scores to a large extent. If one considers that there are only a few samples per face extracted from each video, this seems highly likely.

For the second part of our approach, frame selection, some examples of using less frames with score fusion can be found in the literature [TYRW14, SKP15, BRKKJ13]. The authors of [BRKKJ13] did a study where the confidence score of a face detector is used to detect high quality faces and boost performance on 3 commercial-off-the-shelf still image matchers. The authors showed that by selecting the best 30 frames similar results as for selecting all frames can be achieved. They give some reason to believe that it is possible to achieve similar results with less frames than with all frames. However, to the best of our knowledge, there are no studies that perform a thorough analysis on the implications of selecting one, multiple or all frames of a face track for face recognition, across different datasets and face descriptors.

CHAPTER 3

Concepts

In this Chapter, we will introduce concepts that are crucial to understanding how our method works. A **feature** in machine learning can be understood as a distinctive attribute of a class that should be categorized.A classifier uses features and can be used to categorize data. It does so by building a model from training data which is categorized already (e.g. by hand). The trained model can then be used to classify new uncategorized data. This is a major component of machine learning and finds use in many applications.

The features important for classification are problem dependent. Consider an algorithm that should classify differently colored circles. In this case, the color is important. However, if a classifier should distinguish between circles and rectangles, the color is not important anymore, but the roundness. All features used for classification span the feature space and thus define the possible objects the algorithm is capable of differentiating. In all cases and especially for image classification, the real world object is represented by its features at a point in the feature space. Some classifiers, like the deep convolutional neural networks, define meta features, this means they compute features based on other features with the goal of finding a low dimensional representation. The final vector, with the highest level features is also called a descriptor and only this descriptor is used for classification.

In the following, we explain three classifiers, Linear Discriminant Analysis, Support Vector Machines and Deep Convolutional Neural networks. There exist many other classifiers, for example k-nearest neighbour [FN75] or decision trees[Qui86]. However, we focus on three classifiers, because they are commonly used in face recognition and are directly used by the methods we employ for face recognition in video. In Section 3.4 a summary is provided to distinguish between the advantages and disadvantages of the presented classifiers.

3.1 Linear Discriminant Analysis (LDA)

From decision theory for classification, we know that in order for optimal classification we need to know the class posteriors $Pr(G|x)$for features x of class G=k [FHT01]. For a multi class classification problem, LDA assumes that for classes with features x_k and labels $y = k$ the probability functions are normally distributed with mean μ_k and covariance matrix Σ_k.

LDA is similar to the Fisher Linear discriminant analysis [FHT01] in that both methods minimize intra-class variance and maximize inter-class variance and that they model class densities as multivariate Gaussians. The difference between them is that LDA makes two stricter assumptions about the data, namely that $\Sigma_k = \Sigma$ and classes are distributed normally, where Σ is the covariance matrix of all classes. In machine learning, LDA can be used to separate two or more classes by finding a linear combination of their features. However, LDA can also be used as a dimensionality reduction method, because it projects features with dimension d into a space with dimension $l << d$, when if one only uses the first l largest eigenvectors. For computer vision tasks an LDA model can be learned by calculating the covariance matrix over all classes from features like HOG, LBP, SIFT or a low dimensional representation of a neural network. For example, LDA is used for face recognition [OYP13].

3.2 Support Vector Machine (SVM)

In order to achieve classification, a machine learning algorithm needs training data in order to extract information to build a model with which new data can be classified. For

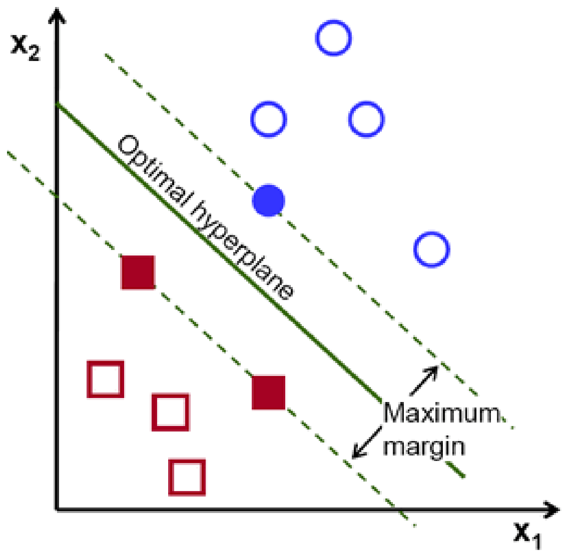


Figure 3.1: A hyperplane that maximizes margin between the two classes red and blue [svm]

the case when classes are linearly separable, either logistic regression or LDA can be used to train a classifier. They can be **optimally** separated by the perceptron (which we will discuss in Section 3.3) or by an optimally separating hyperplane [CV95]. The latter maximizes the margin between classes on training data, which will lead to better generalization on the test data [FHT01]. It does so by imposing an additional set of conditions, which ensure that all points of the training data are at least a margin M away from the decision boundary. An example can be seen in Figure 3.1. Formally, this can be stated as the optimization problem in Equation 3.1.

$$\max_{w,b,\|w\|=1} M \tag{3.1}$$

$$y_k(x_k^T w + b) \geq M \tag{3.2}$$

The SVM machine builds on the optimally separating hyperplanes but uses additional constraints to ensure that this also works on non separable problems with overlapping classes. If classes overlap, an additional slack variable ζ_k has to be added to account for points in the feature space that are now classified on the wrong side of the margin. For this to work we have to change Equation 3.1 to include the slack variable like in Equation 3.3.

$$y_k(x_k^T w + b) \geq M(1 - \zeta_k) \tag{3.3}$$

The training consists of solving the quadratic optimization problem which can for example be done with quadratic programming and Lagrangian multipliers. For real world applications, much faster implementations can be found in the literature [FCH$^+$08, CL11]. So far we only have discussed linear SVMs, however, by using the kernel trick, SVMs can also be used for learning non-linear functions[STC04]. To classify feature vectors only a dot product is required. The class can be determined by the sign of the function. Additionally, the result can be interpreted as the relative likelihood of a classification because the result is the distance from the hyperplane.

3.3 Deep convolutional neural networks (CNN)

State-of-the-art computer vision neural networks share two components. They are deep, this means they have dozens or recently up to more than 1000 layers following each other and they have convolutional layers, that use small filters applied on the whole image. The human brain consists of billions of neurons[PG88], which makes the task of understanding how it works difficult. However, there has been extensive and continued research in this field starting in the 19th century[yC88, PCP12]. Especially, the lower level sensors, for example, the eye and the visual information processing unit, the visual cortex, have been studied [PCP12] and the basic components, cells and neurons, for those systems are well understood. Artificial neural networks have a long history, which started in the 50s with the perceptron [Ros58] by Frank Rosenblatt, who defined a theoretical brain model and used it for image recognition with limited success. His model only had 400 neurons as compared to the brain which has billions. Another problem was, that a single perceptron can only solve linearly separable problems as Minsky et al.showed in 1969 [MP69]. By increasing the number of layers (hidden layer) it is possible to solve more complex problems because the model can portray more complex functions. However, at the time no algorithm existed to train a network with more than one layer. This algorithm, called back propagation was invented only in 1986 [RMG+86]. After this, the way was paved for training larger networks, the constraint now became computational power. When LeCun et al.used for the first time in 1989 convolutional neural networks [LBD+89] to recognize handwriting on letter envelopes, the networks were simple and few people expected what they were capable of. The fast recent advances in image recognition tasks were made possible by the combination of multiple factors. With the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [DDS+09] in 2010 a dataset with an order of magnitude more images became available and built the foundation for quantitative studies that could be compared easily. The dataset contains 1.4 million images split into 1000 classes. A milestone in the challenge was 2012 when for the first time a submission used a CNN for object classification [KSH12] with the name of **AlexNet** and won the challenge by a large margin. The CNN of Krizhnevsky et al.achieved a top 5 error rate of 15.3% compared to 26.2% for the second best entrance. Since 2012 all winners of the ImageNet challenge have been submissions using CNNs. The names of the winners can be found in Figure3.6.

Other advances were to a large extent driven by GPU implementation [CMM+11, KSH12], architecture design, batch normalization [IS15], Batch Stochastic Gradient Descent (batch SGD) and the introduction of the ReLU activation function [NH10].

Perceptron

The perceptron is a binary classifier for supervised learning. A perceptron can have an arbitrary amount of inputs n, which represent the features and one output representing the target. During training the weights w_i for each of inputs are adjusted in a manner that the perceptron classifies the inputs correctly. After applying the activation function, which determines the output of the perceptron based on the inputs, to the sum of all n $w_i * x_i$ and the bias, the output equals the target value for every input in the training set. The bias is used to shift the decision boundary away from the origin. The activation function is used to obtain a normalized output, normally between -1 and 1, many different activation functions exist as we shall show later. For better understanding we provide visualization of one perceptron in Figure 3.2. The activation function is used to obtain a normalized output, normally between -1 and 1. It is one main point that distinguishes neural networks from perceptrons. Neural networks are a generalization of perceptrons, where the activation function can be chosen arbitrarily and is not limited to only the sign function.

To model an even more complex behavior, neural networks can also consist of multiple layers, so that the outputs of one layer of neurons are the input for the next, higher level of neurons. In Figure 3.3 we can see an example of such a network. These classical

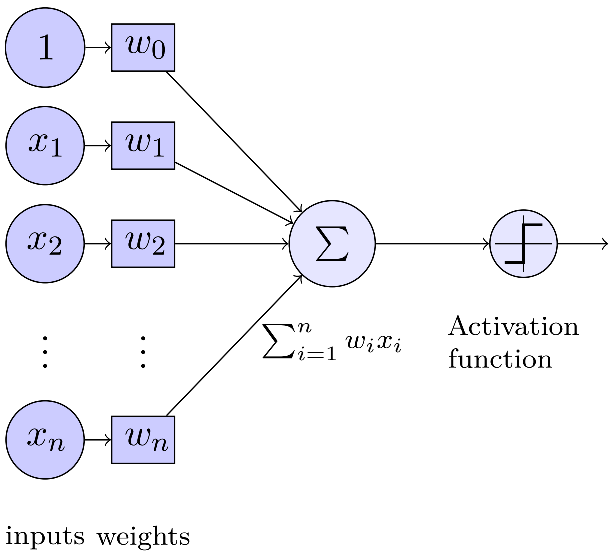


Figure 3.2: Perceptron

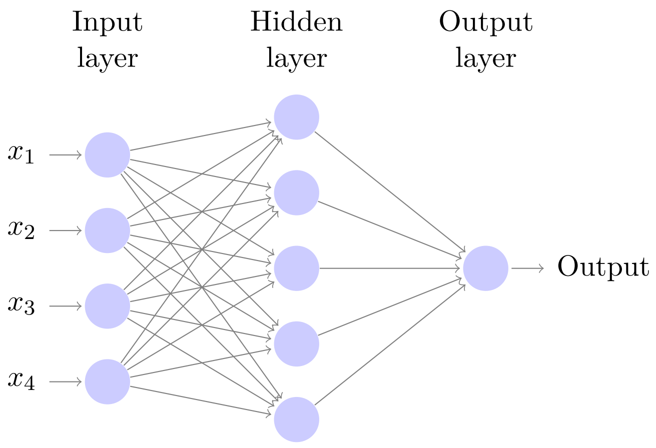


Figure 3.3: A multi layer neural network with 4 inputs and 5 neurons in the hidden layer and one neuron in the output layer

layers are also referred to as **fully connected layers** because each neuron of a layer takes all inputs from the last layer. There exist more different types of layers. They help to compute robust local features namely convolutional layers (important for images), pooling layers to reduce image dimensions. We will in the following describe the different types of layers in detail.

Fully connected layer

Fully connected layers consist of n neurons that each are connected to **all** outputs of the previous layer. If the first layer is the input layer, then they are connected to all inputs. Similar to the perceptron in Figure 3.2 each neuron learns weights w_i according to the inputs x_i and the target function. This means that each neuron produces its own output.

Convolutional layer

Convolutional layers consist of a number of filters which are learned during training. They work especially well for high dimensional inputs, for example, images. This is because they only extract local features. This is done by sharing the weights for each pixel as compared to fully connected layers, which additionally reduces the number of parameters of the layer. The main idea is that local simple features are extracted (e.g. edges, corners, ..) which can later be combined to more complex features. Since images have strong local connectivity this has proven highly effective also because the shared weights imply shift-invariance. This is similar to how the human vision works[PS94], because in the visual cortex first simple features are extracted and then combined to more and more complex features.

For the convolution three parameters need to be specified, the filter size of each neuron, the stride, which determines in which step size the pixels of the image will be convolved, usually it is 1 for every pixel. The third parameter is padding. For each pixel their neighbors are needed to compute the convolution. On pixels that have no neighbors (image border) artificial neighbors have to be added for the convolution to work. A commonly used and simple technique is to pad the borders with zeros.

The convolutions can be implemented as matrix multiplications and highly parallelized because each filter can be applied independently to each pixel of the image and its neighbors.

Pooling layer

Pooling layers help to reduce the image representation and to make features invariant to scale and translation. They do so by computing a function over a local area of their input. A common method is to use 2x2 filters with a stride of 2. This means 4 pixels of the lower layer as input produce one pixel in the output, reducing the size of the image. They can be seen as down sampling of the image in a nonlinear form. This nonlinearity works, because the assumption is that once important features are extracted, the spatial location becomes less important. The down sampling of an image can happen by combining activations of a layer by a maximum or average function. While other functions can be used, max pooling has overall shown the best results [SMB10].

Activation function

Neural networks usually use nonlinear differentiable functions like Sigmoid or Tanh instead of the signum function which is used with the percepetron. Non-linearity is important because otherwise, they could only learn linear combinations of the input data. Differentiability is important because learning algorithms that use gradients for optimization are often used. The **Sigmoid** and **Tanh** function approximate the sign function nicely and are differentiable in all points, this is the reason why they were used a lot for neural networks. However, it turned out that during training with gradient descent algorithms, the gradient of these functions would in some cases become almost zero and the network would stop learning. This is called the vanishing gradients problem [Hoc91]. This is especially troublesome if networks become deeper (more and more layers) [HBFS01, Hoc91].

The Rectified Linear Unit (ReLU) [NH10] is another activation function defined as $f(x) = max(0, x)$. It helps to avoid the vanishing gradient problem and has been shown to perform similar like Tanh but with faster convergence times during training [KSH12]. One caveat for using the ReLU function is that neurons can 'die' if the learning rate is high, because if updates are large it can happen that neurons are never activated again.

Dropout

Another problem that can occur with training not only neural networks but also any other classifier is overfitting. It happens when the learned model becomes too complex for the classification problem and overfits the training data. One symptom for overfitting can be when the training data classification error is low, but the test error is high. This means that the generalization capability of the classifier is low and overfitting high. One easy way to combat overfitting in CNNs is the use of dropout layers [SHK$^+$14]. During training, some of the nodes of the network are not used for an update step (they are "dropped") with a probability p. This makes sure that no single neuron becomes too important for the network, this means that the weight vectors of the next layer are much higher for this one neuron than for others. Imagine a neuron in a CNN used for face recognition that focuses only on the nose. While it may discriminate well among the training set, it might be more stable to also take different parts of the face into account, which most likely will be represented by different neurons. Additionally, the network learns how to compensate for missing information.

Batch normalization

Neural networks are prone to initialization and can if not carefully initialized perform poorly or not converge at all. Batch normalization[IS15] is a form of normalization that forces all inputs of a network to assume a Gaussian distribution. It is usually used after fully connected and convolutional layers. This has recently shown good results and is used in many state-of-the-art architectures[HZRS15a, VTBE15], because it makes networks much more robust to bad initialization[IS15].

Regularization

Regularization with CNNs is a term used for the process of penalizing large weight vectors. It can either be applied after each or some layers or by incorporating them into the loss function. It helps to use all inputs a little rather than some a lot. Another form is weight decay, which adds an error proportional to the size of the weights directly to the error of each node. A third possibility is to enforce an upper boundary of single weights by clamping to a predetermined maximum.

Loss Functions

A loss function in machine learning describes how well a trained model maps the training data to the correct labels, the ground truth. A simple loss function would be the number of wrongly classified samples. Back propagation works by comparing the output of the neural network with the expected output and calculating a total loss. The goal of the learning algorithm, often Stochastic Gradient Descent (SGD) [Bot10], is to state the learning problem as an optimization problem and minimize the loss over the whole training set. This requires the loss function in the case of SGD to be differentiable. How the loss is calculated determines to a great deal what the network learns. For classification purposes the mean square error can be used. For multi-class problems (i.e.ImageNet) often the Softmax classifier, also named cross entropy loss, is used, because it yields an intuitive output: normalized class probabilities. The cross entropy loss L can be computed by

$$L_k = -\log \frac{e^{t_{y_k}}}{\sum_j e^{t_j}} \tag{3.4}$$

where t_j is the result of a forward pass on the sample k for class j and y_k is the ground truth label.

However, if we want to find a lower dimensional representation of an image (e.g. a face) for later comparison a representation that is highly discriminating is preferable. This is called metric learning, because the functions are used to learn a distance metric. Therefore, descriptors are embedded in the often Euclidean space so that the result has certain properties. This could be similarity (e.g. between words) transitivity or other geometric properties which are intuitive. We will discuss some important loss functions that have been recently used for face recognition and are important to this work. However, many loss functions exist for different use cases and new are proposed. Some of those are contrastive loss [SCWT14, HCL06], word2vec relationships for words[MD13] triplet loss [SKP15], Siamese loss [HCL06] and very recently centered loss [WZLQ16]. In the following, we discuss loss functions which have successfully been used for training face descriptors, like Siamese Loss[SKP15], Triplet Loss[SKP15, PVZ15].

Siamese Loss [HCL06] Two properties of a discriminating representation of features are, that descriptors for the same objects should be close to each other and different

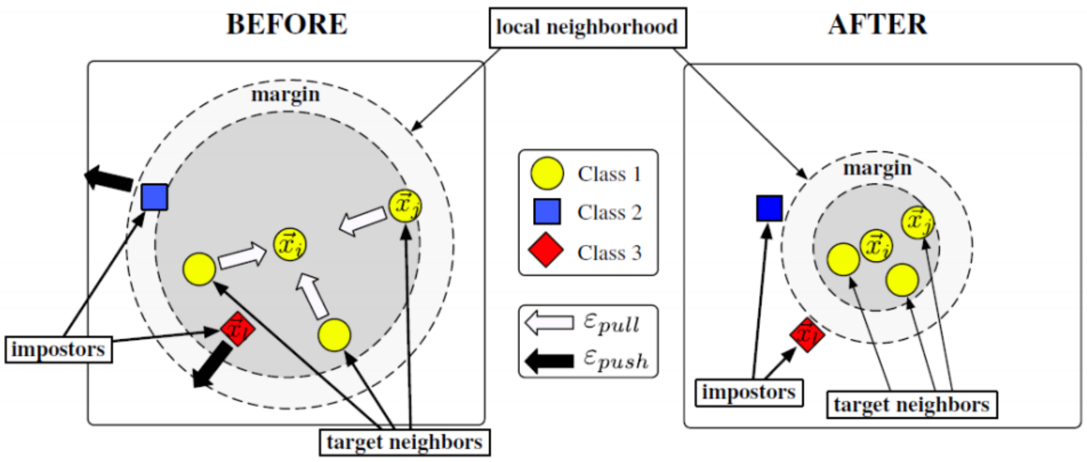


Figure 3.4: Siamese loss objective. Figure from [Kul12] adopted from [WBS05]

objects should have descriptors that are different (see Fig. 3.4. The energy is defined as

$$D = \sum_{i=1}^{N} \|G(x_1^i) - G(x_2^i)\|^2 \tag{3.5}$$

where N is the number of pairs, (x_1, x_2) is a pair and G is the function that produces the descriptor for example a CNN. For pairs where both elements belong to the same class, hence positive pairs, the distance D should be minimal and vice versa. The loss is then defined differently depending on if a positive or negative pair is learned in Equation 3.6.

$$L_{pos} = \frac{1}{2} D^2 \tag{3.6}$$

$$L_{neg} = \frac{1}{2} \max(0, D)^2 \tag{3.7}$$

$$L = L_{pos} + L_{neg} \tag{3.8}$$

During learning also a fixed margin m is added to the loss function L in order to achieve good separation. This is interesting because the error and thus the embedding of descriptors in the descriptor space does not depend on labels but is solely defined by the embedding and training data.

Triplet Loss[HA15] The triplet loss function builds on the idea of the Siamese network, however it computes the error by using not only two but three descriptors. It uses an anchor x_a, a positive sample to the anchor x_p and a negative sample x_n. Further it tries to learn not only binary relations (x_a should be close x_p and x_a should be far from x_n) but it helps to learn transitivity in the form of $d(x_a, x_p) < d(x_a, x_n) + c$ so that something is more similar than something else. This makes it also useful to image ranking. The c is

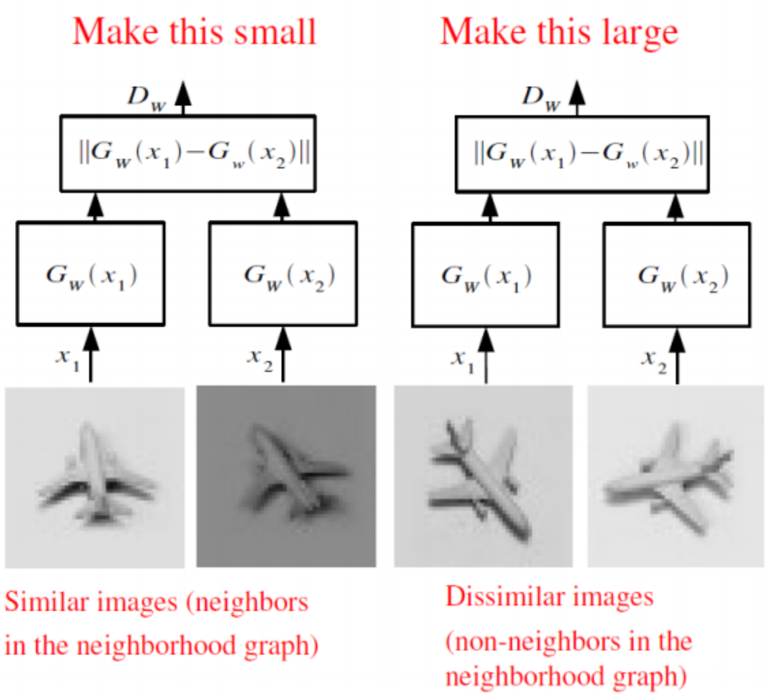

Make this small
Make this large
D_W
$\|G_W(x_1)-G_W(x_2)\|$
$G_W(x_1)$
$G_W(x_2)$
x_1
x_2
Similar images (neighbors in the neighborhood graph)
Dissimilar images (non-neighbors in the neighborhood graph)

Figure 3.5: Another visualization of the Siamese loss function. [Luo14]

called margin, similar to the Siamese margin in Figure 3.4. The loss is defined as

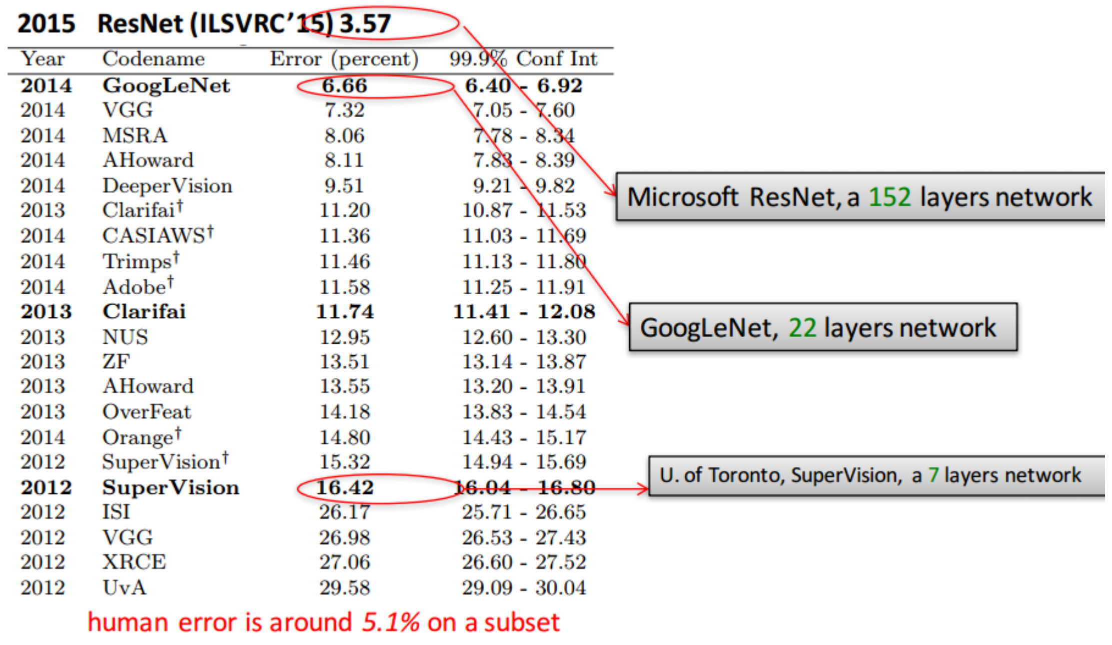
$$L = \sum_{i=1}^{N} \|G(x_a^i) - G(x_p^i)\|^2 - \|G(x_a^i) - G(x_i^n)\|^2 + \alpha \tag{3.9}$$

with N is the number of triplets.

This loss was recently used in two popular face recognition papers[SKP15, PVZ15]. Naturally, the question arises which triplets should be used for training, because with millions of training data there are more combinations of triplets than can be used for training. One option that was proposed is hard negative mining [PVZ15]. This method tries to determine difficult triplets, i.e. the triplets that violate the triplet loss margin in the current state of the network. This helps the network to concentrate on training data it cannot classify correctly yet more. However, it is not feasible only to use the most difficult samples, because samples could be labeled wrong or simply not discriminative enough by their nature. Thus a stochastic method that samples from difficult triplets is a better choice as proposed by Parkhi et al.[PVZ15].

Architectures

ImageNet is unique for the large amount of classes and images it encompasses and has become the defacto benchmark for image classification. Thus it is reasonable to use it as a proxy for which neural network architectures work well for CNNs. Lately, the trend has been going in the direction of ever deeper networks. The reasoning is that more complex and thus deeper networks can learn more complex relations. Deeper networks pose challenges because it becomes more difficult to train them due to vanishing gradients and other problems as we have mentioned before. Also deeper and larger networks are computationally more expensive. We will show some architectures to give the reader an idea of how typical, well performing networks look like.


2015 ResNet (ILSVRC'15) 3.57
Microsoft ResNet, a 152 layers network
GoogLeNet, 22 layers network
U. of Toronto, SuperVision, a 7 layers network
human error is around 5.1% on a subset

Year	Codename	Error (percent)	99.9% Conf Int
2014	**GoogLeNet**	**6.66**	**6.40 - 6.92**
2014	VGG	7.32	7.05 - 7.60
2014	MSRA	8.06	7.78 - 8.34
2014	AHoward	8.11	7.83 - 8.39
2014	DeeperVision	9.51	9.21 - 9.82
2013	Clarifai†	11.20	10.87 - 11.53
2014	CASIAWS†	11.36	11.03 - 11.69
2014	Trimps†	11.46	11.13 - 11.80
2014	Adobe†	11.58	11.25 - 11.91
2013	**Clarifai**	**11.74**	**11.41 - 12.08**
2013	NUS	12.95	12.60 - 13.30
2013	ZF	13.51	13.14 - 13.87
2013	AHoward	13.55	13.20 - 13.91
2013	OverFeat	14.18	13.83 - 14.54
2014	Orange†	14.80	14.43 - 15.17
2012	SuperVision†	15.32	14.94 - 15.69
2012	**SuperVision**	**16.42**	**16.04 - 16.80**
2012	ISI	26.17	25.71 - 26.65
2012	VGG	26.98	26.53 - 27.43
2012	XRCE	27.06	26.60 - 27.52
2012	UvA	29.58	29.09 - 30.04

Figure 3.6: An overview of how fast the ImageNet top-5 error has dropped in recent years [San]. Bold names mark winners of respective years. The error measured in the ImageNet challenge is the accuracy of assigning the right class label to an image out of 1000 classes. The utmost right column is the confidence interval calculated by bootstrap sampling of test data

In Figure 3.6 an overview of the error of classifying images of the ImageNet challenge in recent years is displayed. This proves what we have stated earlier, the advance in image classification boosted through DCNNs is clearly visible. Winners of the challenge in each year have presented deeper and more complex networks than earlier winners.

LeNet-5 [LBD+89, LBBH98] This network visualized in Figure 3.7 was one of the first networks that used Convolutional layers and performed well on the task of digit recognition on letters.

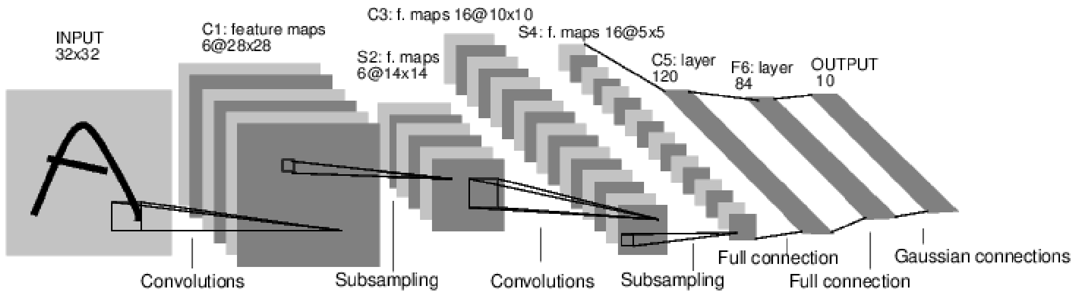


Figure 3.7: In the image one of the first convolutional neural network architectures can be seen. The network has 2 convolutional layers, 2 subsampling (pooling) layers and 2 fully connected layers and Euclidean Radial Basis Function units at the output layer (Gaussian connections) [LBBH98].

AlexNet (SuperVision) [KSH12] AlexNet, in Figure 3.8 was the first large scale trained neural network and won the ImageNet challenge in 2012.

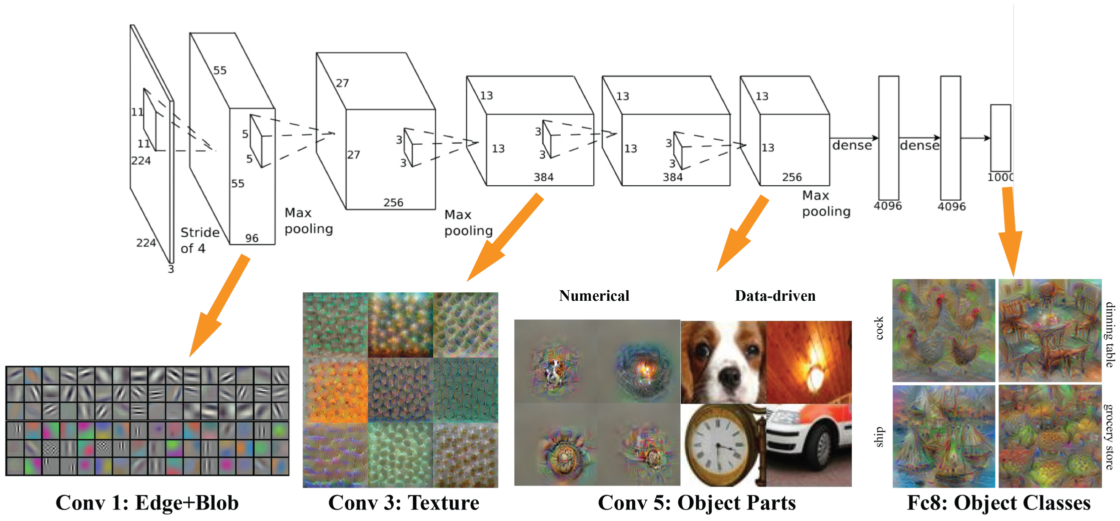


Figure 3.8: AlexNet architecture and visualization of some stages [WZTF15]

ResNet [HZRS15a] This method called Residual networks can be interpreted as an ensemble of networks [VWB16] which learn residuals, which helps make much deeper networks possible. The authors report networks with up to 150 layers and won the ImageNet challenge in 2015. In Figure 3.9 we show three different architectures. The first, VGG-19 [SZ14] is a popular network and has had aninfluence on the development of new architectures, also the ResNet architecture. We can observe similarities between VGG-19 and ResNet-34-layer plain architectures. However, new architectures tend to have even more layers. Deeper networks add more complexity and often do not converge. With the residual architecture of ResNet, this problem has been reduced. This an be seen in the Figure as the rightmost 34-layer residual network. Networks as deep as 1001 layers have been trained, shown to converge and increase performance [HZRS15a].

Weight initialization If we would initialize all weights to zero, when using the backpropagation algorithm all neurons would update in the same fashion because they have the same gradients and thus we would have identical neurons. Hence we have to introduce a random component, commonly referred to as symmetry breaking. Weight initialization is important because it determines how the network will converge, there exist different possibilities in the literature [GB10, HZRS15b]. Recently, best results have been shown by initializing weights with $\frac{rand()}{2\sqrt{n}}$ [HZRS15b] where $rand()$ returns a random number sampled from a standard normal distribution and n is the number of inputs of a neuron. However, with ReLU activations units this has become less of a problem, but still needs proper consideration.

Finetuning Neural networks need large amounts of data to work well. On the one hand, they can represent many different classes, on the other hand, they are prone to overfitting and have to be trained carefully so training data is not overfitted. Additionally, training a DCNN can take weeks, even with newest hardware [HZRS15a]. In practice, if one has only a small dataset it might be preferable to finetune a neural network rather than to train it from scratch. Finetuning neural networks is a fast way to train a classifier. One should use a network that was trained on a preferably large dataset[SRASC14, YCBL14] as initialization and continue training with one's own dataset. This works, because lower layers of CNNs contain more generic features and the assumption is that those features if they were learned on big datasets, will tend to be better and more generic than lower level features trained on a small dataset. Additionally, using only a small dataset for training a network from scratch might lead to overfitting.

In practice, it is sound advice for small datasets (in the order of thousands of images) to retrain a network that was trained on the ImageNet dataset, remove the classification layers and finetune the network on the small dataset. One could either freeze lower layers or learn a representation based on the output of the pretrained network.

Data augmentation Data augmentation is the process of generating artificial mutations of original data. Since neural networks are prone to overfitting, data augmentation is a good way to make them more generic [SLJ$^+$15]. Data can be augmented by simple geometrical transformations, rotation, cropping of different scales, change of aspect ratio or small color variations. For example, if one wants to train a face recognition algorithm, one could randomly crop the face of the images so the sizes will be different and the crop is jittery around the head. This will teach the network to become more invariant to the exact input and might help a lot for practical problems during inference.

It seems that neural networks can leverage the growing size of datasets better than any other algorithm available. However, large datasets might not be available to individuals or research facilities but only to large companies. To work around this, recently some researchers have tried to train neural networks on synthetic data which was generated by rendering 3D models of objects [PSAS15] or e.g. faces [MaTaH^{+}16]. This could lead to interesting possibilities.

Hyperparameters Learning rate, weight decay, weight initialization, batch size, number of epochs as well as parameters that influence data augmentation are all considered as hyperparameters. Tuning hyperparameters is an important part of training neural networks. Ideally, we would have a systematic approach to conquer this problem, but often not all parameter combinations can be tried because of long training times. For example, the 150 layer ResNet network took 3 weeks for training on multiple GPUs [HZRS15a]. While batch SGD, ReLU and batch normalization have helped to reduce convergence problems of networks during training and accelerate the convergence, the learning rate is still crucial for learning. A common practice is to set the learning rate to 0.1 for the start and decrease it by a factor of 10, once the network performance does not increase anymore. For finetuning a lower learning rate should be used. In general, there are no guidelines on how hyperparameters should be chosen, since the whole field is still young and changes all the time.

Training

Training neural networks consists of four important parts. Training data, neural network architecture, loss function and hyperparameters. To the hyperparameters we count all free parameters that have to be set for the training. All parameters influence how the network learns and consequently the quality of the resulting network. Training networks is not straightforward and requires a systematic approach, some parts of which we tried to outline already.

3.4 Summary

SVM and LDA are both classifiers that are to be used with handcrafted features. For computer vision tasks, such features can be of various form, but have historically been features like SIFT, HOG or LBP. While this works for simple tasks, more complicated tasks like object recognition have tended to be too complex and different. Convolutional neural networks approach this problem by replacing the handcrafted features with learned features from large datasets and combining them to more abstract ones in multiple stages with large success, if one uses the ImageNet challenge as a benchmark. This last point makes the way convolutional neural networks work similar to the human visual cortex [LP16].

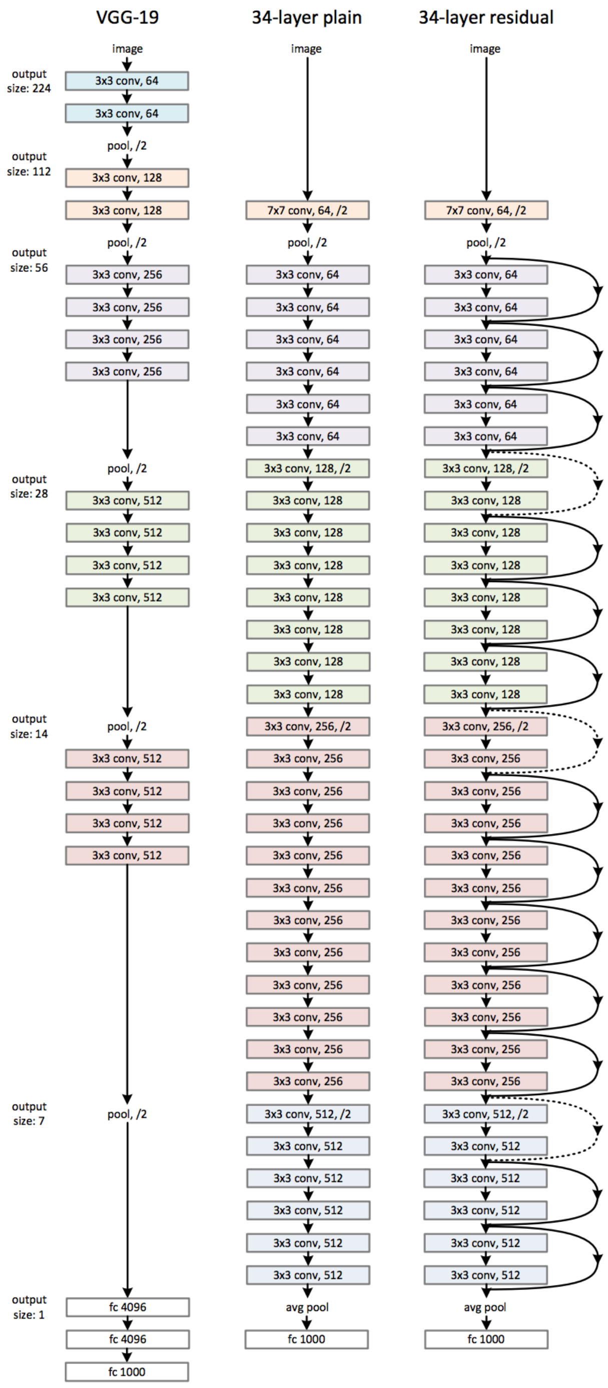


Figure 3.9: VGG-19 [SZ14], plain and residual [HZRS15a] network architectures. The arrows in the right figure show how the input of preceding layers is used to obtain residuals

CHAPTER 4

Face recognition

The main problems face recognition are face verification and face identification. This means given two faces to conclude if they are the same or different persons. Identification is the task of determining the identity of a person from a single image assuming other images of him are stored in the database. Both applications require very similar steps and only differ in the last stage of the pipeline. Face recognition requires multiple preprocessing steps. How many and which has recently, with the advance of deep CNNs, become a hot topic. While some use very refined preprocessing such as 3D alignment [TYRW14] or 2D alignment [PVZ15, SWT15] other methods with comparable performance do not use any alignment [SKP15]. Nevertheless, the preprocessing steps can be divided into face detection and some or no normalization. The latter of which is disputed. We will discuss commonly used important processing steps and give an outlook on what is important, especially with regards to deep CNNs.

4.1 Detection

Face detection is the first step to face recognition. Given an image, we want to detect all faces of humans in the image. This is a long standing research field, which is still advancing, however lately face detectors have become both, fast and accurate enough to use them in real time and real world environments. Covering the whole field of face detection alone could fill books because of extensive literature available, this is why we will only look at the Viola Jones and Deformable Part Model (DPM) face detection algorithms and explain its basics. The latter is state-of-the-art and relevant to us because it was used in our pipeline. For a more detailed review of the field we refer to two excellent reviews et al. [ZZ10, ZZZ15]. It is essential because neural networks have the ability to highly overfit training data and this step helps focus the algorithm only on the face. Without face localization the network could be tempted to learn features from the background.

Problem One of the difficult problems for object detection in general is, that an object can be located at any given point in an image at an unknown scale. In order to be shape independent almost always the goal is not to find the exact contour of the image but a bounding box. For a brute force approach of localizing objects in an image, this means one would need to evaluate all possible positions and scales of bounding boxes in an image and compute features for each and every one of them. This could possibly amount to hundreds of thousands of classifications for a single image depending on the size of the image. This is not a good idea. One more challenge for face detection is the scarcity of faces as compared to the possible locations, which makes a very low false positive rate necessary in order for a face detector to be useful.

The Viola Jones object detection algorithm [VJ01] was one of the first to have an efficient answer to this problem. Three key contributions that accelerated the filed, were integral images for fast feature evaluation, boosting for feature selection and cascading to use complex (slow) classifiers only on prospective candidates. It makes use of an integral image to compute Haar-like features over a gray scale image and uses these features as input for boosted weak classifiers. Additionally, it uses a cascaded architecture that employs simple and fast classifiers in the first stages and more complex ones in later cascades. The key idea here is that simple true negatives are filtered in early, computationally cheap cascades.

Another improvement came with the advance of DPM [FGMR10] detectors for object detection, which can be considered defacto standard nowadays for high speed face detection [SF14]. CNN based object detection is advancing fast with recent state-of-the-art results. There are Recurrent-CNN (R-CNN) [SR15] and cascaded CNN [LLS$^+$15, QYLH16]based approaches which have received some attention. However, most are still orders of magnitudes slower[SF14, GDDM14] than traditional simple feature based approaches. Very recently, the tendency has shifted from evaluating many different parts of an image to CNNs that directly predict interesting regions from the whole image [LAE$^+$15, RDGF15], which performs comparably to R-CNNs but is faster[SR15].

4.2 Alignment

Face alignment is used by some[TYRW14, SKP15] state-of-the-art approaches but not by others [SWT15]. More abstractly, one can justify the use of alignment for different computer vision tasks with the way machine learning algorithms work. The idea is to bring input images into a normalized format. Especially for non-rigid objects a CNN or other feature extractor, has to learn two things, how to find a representation of the same object or face and how to compensate for external factors. It does so by trying to find similarities between different depictions of the same image. This also implies, that it has to learn that for example the same face can appear in different places. It has to learn translation invariance but also scale, rotation, lighting and in the case of faces age, emotion and other invariances. To put it bluntly, the idea is, that the better we normalize images for position, scale and rotation the less the algorithm has to learn

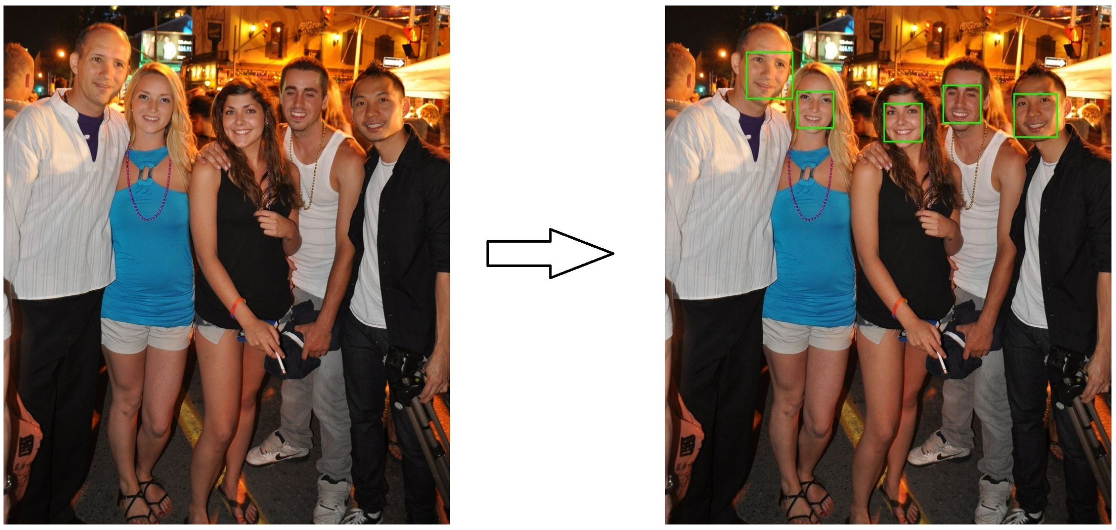


Figure 4.1: Face detection step. Image from AFLW database [KWRB11]

them and can instead focus on other aspects. Because faces are three-dimensional and images 2-dimensional projections of them, information is lost. This makes out of plane, 3D-rotations difficult, because they are not linear in 2 dimensions, also occluded regions have to be extrapolated.

The human brain is incredibly good at predicting the near-term future and extrapolating missing information [SAB07], our models and artificial intelligence applications so far lack exactly this ability. In order to account for this, we can try to 3D align a face and bring it into a neutral, mostly frontal, position. Both 3D and 2D alignment work by detecting landmarks on the face. There are many different possibilities, as this is a big field of research by itself [ESZ06, KS14, BRM13, YDZL15, XYK15]. Most state-of-the-art approaches work either with the use of cascaded regression trees and simple fast features (HOG, LBP or pixel differences) or are CNN based, for our experiments we use [ESZ06] whenever we do 2D alignment. In Figure 4.2 we show one example of our preprocessing pipeline.

One argument why alignment might not be necessary and could even have negative effects is a possible alignment error. Errors in the landmark detection process can result in wrong face alignments and thus are sources for errors if the face feature extractor was only trained on aligned faces. Another argument is that resampling causes discretization errors.

4.3 Normalization

Many face recognition methods, especially such that are based on handcrafted features, include multiple data normalization steps additional to the already mentioned. This can include

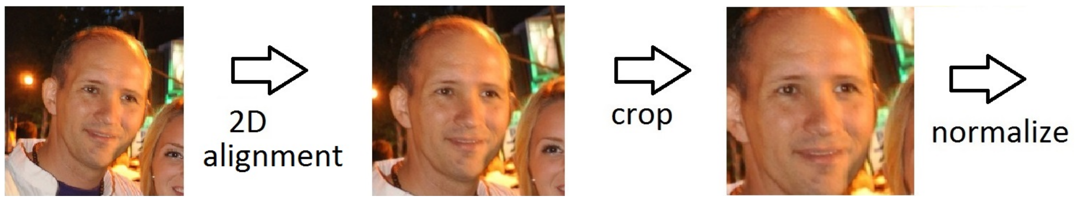


Figure 4.2: Preprocessing of image from AFLW database [KWRB11]

- removal of background;
- using specific parts of the face only;
- removal of camera artifacts (including lens distortion and noise);
- pixelwise or color channel mean subtraction and normalization;
- grayscale or different colorspace conversions

Interestingly, CNN state-of-the-art approaches that have been shown to work well on in the wild data, follow the exactly opposite approach. While normalization tries to bring data to a similar, normalized format, data augmentation creates more not normalized versions of the original data. Those approaches augment data at training as we saw in Section 3.3 in order to make features more invariant to changes which are not important for the recognition task. This helps counter overfitting. For example, with normalized faces if a woman always wears the same earrings in the training data and features only for the earrings are learned because they differentiate her best from other faces. However, if the face is randomly cropped and also some crops in the training show the face without ear rings, these features will become less important.

4.4 Descriptors

In this step the preprocessed image, which contains a face, is converted to a descriptor. The descriptor is computed by using the respective algorithm. For CNNs the images are fed directly into the network and a forward pass is performed and the output is used as the descriptor for the face. The representation of the image in lower dimensional descriptor space is important for the further steps, face recognition, verification or clustering, which we discuss in Section 4.5. The descriptor is a representative for each person in a way that images of the same person produce similar descriptors and pictures of different persons should produce different descriptors. Similarity can be expressed in terms of distance. How this distance is defined depends for CNNs on the descriptors and how they were trained. This is determined mainly by the loss that is minimized. Often the Euclidean distance is chosen because it is simple. In these terms the same face should be closer and different faces should be further apart. This makes clustering and verification very intuitive, because simply a threshold can be found on a training set in order to determine if two descriptors represent the same or different faces depending on their distance. Other important aspects for face descriptors in real applications are their size in terms of dimensions, because of storage and speed with which they can be computed and compared with each other.

We explain how we computed three different descriptors and use them in Section 5 to conduct experiments for video face recognition. We use these descriptors to try the generalization ability of our method. We use handcrafted features (LBP), an early neural network architecture and a state-of-the-art DCNN architecture.

Local Binary Patterns (LBP) [OPH96]. Local binary patterns are texture descriptors which encode changes of light intensity in a fixed neighborhood around a pixel. The faces are converted to gray scale, 2D aligned and cropped centered according to extracted landmarks. The descriptors are computed by extracting the LBP for each pixel and concatenating them into a single high-dimensional vector.

AlexNet CNN face descriptors (AlexNet). The AlexNet[KSH12] CNN architecture used in this work, has been pre-trained on ImageNet and fine-tuned on the face recognition task using a proprietary dataset of frontal face images. The CNN is trained by considering a classification problem with n classes with cross entropy loss. A class is a person and n is the number of individuals. To compute descriptors we remove the last fully connected layer, do a forward pass on an input image and receive the 160-dimensional feature descriptor.

Deep CNN face descriptors (VGG) [PVZ15]. The deep CNN was trained on 2.6 million pictures of 2622 unique individuals downloaded from the internet. We use the publicly available network [PVZ15] and remove the classification to obtain 4096-dimensional face descriptors. The dissimilarity of faces is measured by the Euclidean distance computed for L2-normalized descriptors. Prior to descriptor extraction we align faces using a 2D similarity transformation as suggested in [PVZ15].

4.5 Identification and Verification

In face recognition, there are two main use cases for descriptors. For surveillance, it is person identification. Given a database with already computed descriptors for known persons, we can compute the distances and identify the person, if the descriptor was learned with a meaningful distance metric. Another possibility would be to use a k-nearest neighbor [FN75] classifier if the distance metric of the descriptors is appropriate. For otherwise trained descriptors (i.e. LBP), one could either train a classifier like a SVM on a training dataset that contains labeled faces. If the descriptor is computed by a CNN, a fully connected layer can be added and only this single layer can be trained on training data for face recognition with a Softmax classifier (Section 3.3). Of course CNN can also be trained to learn feature encoding and recognition together in one step.

The second task is face verification. When a person wants to interact with a system, the person will tell the system who he is and this can be verified with an image. In technical terms this can be accomplished by storing a descriptor for each identity in the database. For verification the distance between the stored descriptor and the newly calculated descriptor is calculated. The distance in which faces are still classified as the same can be either obtained by using the margin (i.e.triplet loss, Siamese loss for CNN) or by learning it from a training dataset.

4.6 Liveness Detection

Face recognition can have many different applications. It can be used to identify people for commercial purposes such as identifying customers for business in order to deliver tailor-made advertisement or services. Another use case are security related applications such as access control or authentication of financial transactions. However, it is easy and inexpensive to fool such a system with a simple printed image of a face. While for the business related use cases this is not an important problem it is even more so for security related applications.

Due to this fact, there has been a lot of research in the area of face spoof detection as well as some advances.

Generally one can divide attacks against face recognition systems into four classes:

1. print a picture
2. static image on phone
3. replay a video
4. 3D Mask

Print and Replay are the easiest for an attacker, as resources can in some cases even be found on the internet, and are cheap and fast to deploy. In any case, they are easy to acquire, since an image or a video can be taken of another person without his knowledge.

One kind of features used for spoof detection that has shown state-of-the-art results are texture based features such as Local Binary Patterns (LBP)[dFPADMM12], Binarized Statistical Image Features (BSIF)[AKC15] and Local Phase Quantization (LPQ) [BSO$^+$15]. Other authors have tried to exploit the fact that when taking an image of an image certain properties might be magnified or absent. For example when taking a high resolution image of a low resolution image, Moire patterns can be detected under certain conditions [PHJO15, GdQ15]. Other properties include frequency distribution skewed towards higher frequencies[LWTJ04, Kan15], imperfect representation of colors as they might be distorted [WHJ15] (i.e. a cheap LCD display can neither display all colors nor display colors correctly), blurry (out of focus due to proximity) or have reflections due to ink or display. Again others try to reconstruct 3D information from images, either through depth of field [KBL15] or by multiple view angles [WYL+13]. Some of these methods propose special equipment (Infrared camera, light-field camera [KBL15] or 3D sensors) as possible solutions, while others work only under constrained conditions (i.e. Moire patterns). Or require the user's cooperation for example eye blinking, head rotation or smiling. However, ideally the liveness detection method should work in an unconstrained environment, under all conditions.

CHAPTER 5

Face recognition in video

Video is becoming increasingly the format of transmitting information around the internet. According to a white paper from Cisco [Ind16] video accounted for 70 percent of all internet protocol based internet traffic in 2015, with video surveillance being 516 Petabytes a month. Furthermore, the growth of video traffic on the internet is expected to be more than 10% until 2020 every year. More and more video data offers big possibilities for real-time face recognition and face recognition in videos in general. Be it for actor recognition in movies, surveillance purposes or customer identification for retail. While technically face recognition in video is similar to face recognition in still images, the amount of data to be handled and the often different setting, need to be taken into account, in order for a real world application to work well.

Cooperative vs uncooperative Still images of people are often taken with additional subjective knowledge of the photographer of what a good image is. There are many factors which play a role not only for the subjective quality of an image but which also have an influence on the objective recognition performance of faces. These include head pose, lighting focus, emotions (i.e. open mouth, closed eyes). There exist also formal norms, how an image of a person's face has to look like in order to be considered of high quality for face recognition [ICA06]. This is used for document verification from passports at airports. One such example can be seen at a Japanese airport in Figure 5.1. Here lighting, distance to the camera, head pose and emotions are controlled.

When taking a video, a sequence of images over time are captured. If the external factors change over time, it is likely that some of the captured images will be better than others, some examples can be seen in Figure 5.2.

Problem definition In order to be able to recognize faces in video from a real world scenario, we have to carefully examine 2 criteria

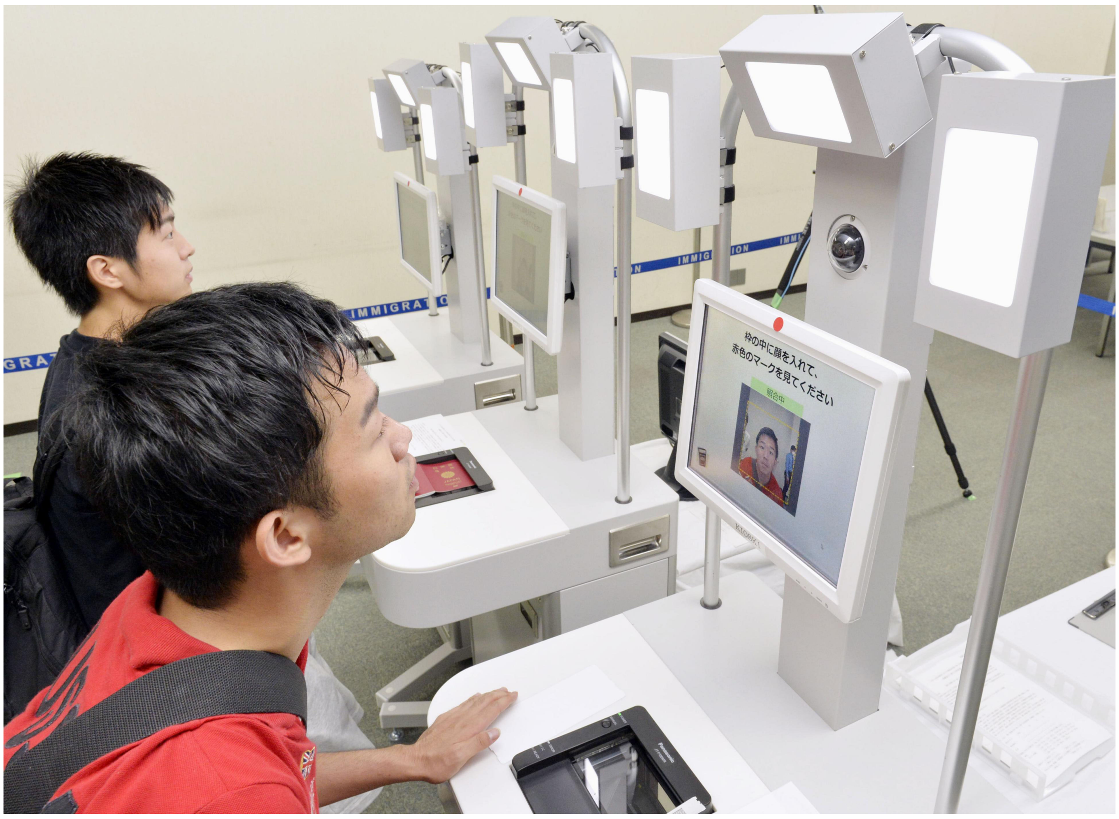


Figure 5.1: Japanese airport face recognition system [OSA14]. A typical cooperative system

- computation speed

- recognition accuracy

In Section 5.1 we explain MBGS and feature whitening. Furthermore, we present the influences of both methods on recognition performance and computational complexity for multiple descriptors and datasets. Our work shows that whitening can replace SVM and studies the exact effects regarding speed and performance. In Section 5.2, we study how we can select frames from a sequence in order to maximize recognition accuracy. The novelty is our systematic study of different frame selection methods with different descriptors on multiple datasets. Furthermore, we study the effect of taking a subset of frames per video, to decrease computation complexity. Next, we present details for the used databases.

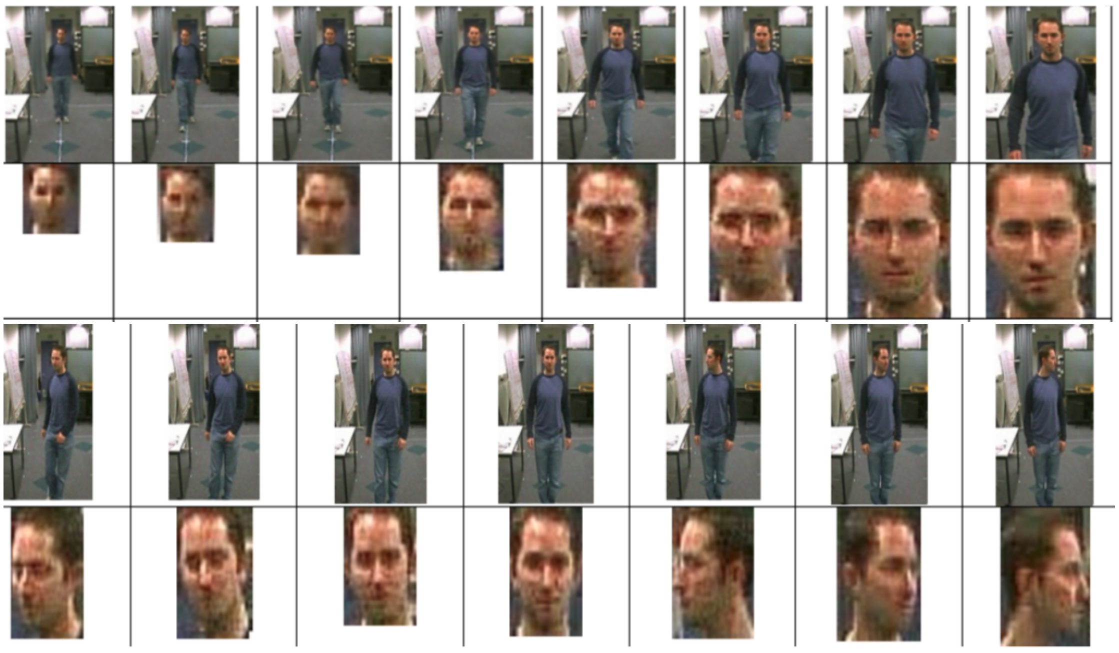


Figure 5.2: Top: different distance of the face and thus different size in pixels over time; bottom different head poses[NM08]

5.1 Set-based similarity scores

Calculating the similarity between two faces can be done with the help of calculating the corresponding face descriptors and the distance between them. If one wants to extend this concept to two sets of faces the simplest way is to calculate face descriptors for all faces in the first step. In the second step, one can calculate the average distance between all pairs of faces which belong to different sets and obtain a single score that reflects the similarity between two sets. The average function seems like an arbitrary choice as we have seen in Chapter 2 there are better options.

Baseline

In order to be able to compare the face recognition rate of different approaches for face recognition in video, it makes sense to establish a baseline and use a simple one. Three different face descriptors are used as described in Section 4.4, VGG, ResNet and LBP and two databases YTF and TSFT. As a baseline, we choose a set-based comparison method of all frames. We assume each video to be pre-processed by a face tracker generating a temporal sequence of face detections for each person in a video. Face detections $i = 1, .., n$ in a track are associated with *face descriptors* $X = \{x_i\}$. In this work we compute the similarity $\mathcal{S}$ between two tracks X, Y as the average similarity between corresponding descriptors for some similarity function f as

$$\mathcal{S}(X,Y) = \frac{1}{|X||Y|}\sum_{x\in X}\sum_{y\in Y} f(x,y) \tag{5.1}$$

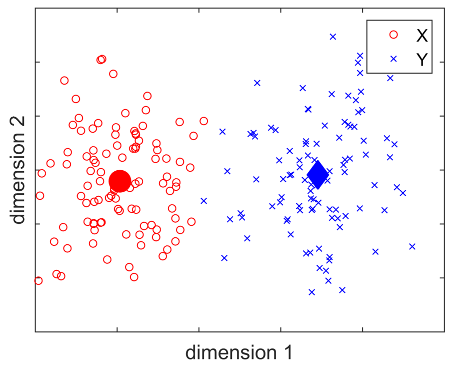
For our baseline, we choose the Euclidean distance for computing the distance between any two descriptors and the mean of distances between all frame pairs like in Equation 5.1. It can easily be seen that this is not optimal, because calculating the average of all distances between frame pairs, weights frames suited well for face recognition and bad ones equally, however imagine the case, that there is only one frame in each video sequence that produces a descriptor which will result in the correct classification. In this case, the "wrong" descriptors would falsify the results, even though the right information is available. If there was a way to determine the frame which the highest likelihood of correct classification, this problem could be solved. We will present some possibilities in Section 5.2.

Example To better understand, how set-based similarity scores work, let us consider an example. When we compare two videos X and Y, we first compute for each frame in each face track a descriptor. The descriptor is computed in a way, to project the high dimensional image of a face ($width * height * colors$) in a lower dimensional space with the number of dimensions d, more details can be found in Section 4.4. Ideally, this representation would project each face to exactly one unique point in the d dimensional space and would ignore external factors. In practice, we try to train a feature extractor that will provide a descriptor that projects the same or very similar faces (different only by external factors) to a subspace of the descriptor space. Another important attribute to measure the similarity between faces is that similar faces should be close together and not similar faces should be far apart in the descriptor space. Of course, also other similarity measures are possible.

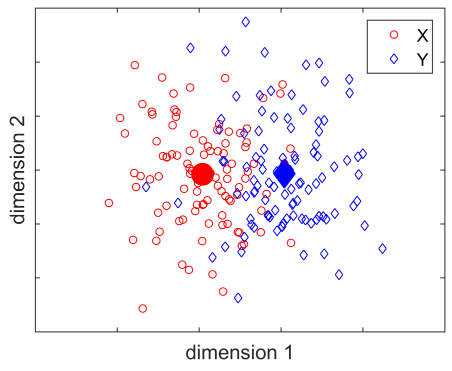
In order to visualize our example, we assume we have a feature extractor, that projects all faces into a space with $d = 2$. Let us first consider the case that we want to compare two face tracks of different people, each with one hundred frames, thus 100 descriptors each. If we assume that external factors let the descriptors have a near normal distribution we can imagine an image close to Figure 5.3a. In this case using the average of all descriptors for each face will give a reliable distance between different faces.

If we look at Figure 5.3b we see what could happen if a different descriptor is not so good at projecting faces to their own subspace, but if the subspaces overlap. In this case, the simple average of descriptors becomes much less reliable. This can have multiple reasons, one could be that the descriptor was only trained with frontal, optimally lit faces and cannot project faces with different external factors to the same subspace.

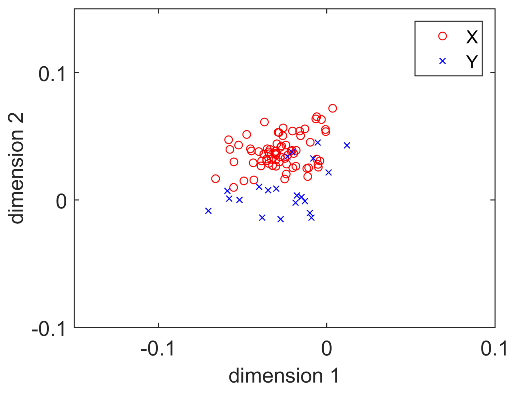
Until now our example was only theoretical, in reality, the distribution of descriptors for the same face but different external factors can be different and again depends on the algorithm that produces the descriptors, how it was trained (what it should optimize, see



(a) Ideal distribution of descriptors for two different people

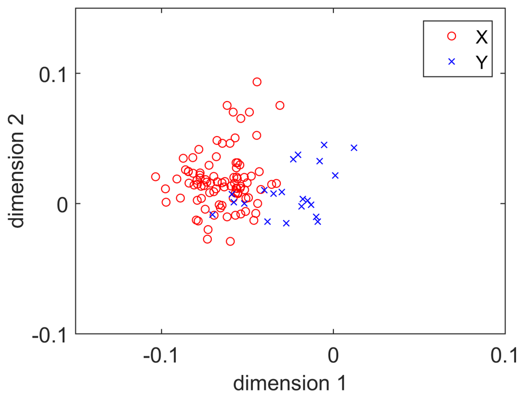


(b) Overlapping distribution for two different people

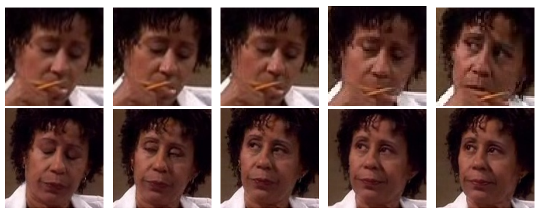


(c) First 2 dimensions of AlexNet descriptor for track 0004 and track 0005 of TSFT database

(d) Top row: track 0004 (red in figure 5.3c) Bottom row: track 0005 (blue in figure 5.3c)



(e) First 2 dimensions of AlexNet descriptor for track 0003 and track 0005 of TSFT database



(f) Top row: track 0003 (red in figure 5.3e) Bottom row: track 0005 (blue in figure 5.3e)

Figure 5.3: The average and standard deviation of critical parameters: Region R4

3.3) and the training data. In Figure 5.3c we see distributions for two dimensions of the descriptor produced by the AlexNet descriptor described in Section 4.4 for two different faces. Clearly, it would be positive to find a way to choose such descriptors of faces, that maximize the distance between descriptors for different faces, and minimizes it for same faces. Another way to approach this problem would be to model the distribution of descriptors, that is induced by external factors for the same face. In the following, we will discuss both methods.

Matched Background similarity (MBGS)

Matched background similarity (MBGS) [WHM11] is a set-to-set similarity measure. Computation of MBGS requires a *background* set of face descriptors B used as negative data to train a 1-vs-all SVM classifier for each face track. This is done to model the distribution of different descriptors for the same face by external factors. Given two face tracks X, Y, a linear SVM is first trained to discriminate X from B and is then applied to classify Y. Similarly, another SVM is trained for Y and is used to classify X. The final similarity score is defined as an average of classification scores for X and Y. The MBGS algorithm is summarized in Algorithm 5.1 and further details can be found in [WHM11]. At test time, a given sequence X is matched to the most similar sequence in the database by maximizing the similarity score of MBGS(X, Y, B) for all database sequences Y.

Algorithm 5.1: MBGS algorithm.

```
input  : $X$, $Y$, B
output : similarity
1 $B_1$ = NearestNeighbours($X, B$) ;
2 model1 =trainSVM($X, B_1$);
3 conf1 = classify($Y$, model1);
4 sim1 = average(conf1);
5 $B_2$ = NearestNeighbours($Y$, $B$);
6 model2 = trainSVM($Y$, $B_2$);
7 conf2 = classify($X$, model2);
8 sim2 = average(conf2);
9 similarity = (sim1 + sim2) / 2;
```

Whitening

MBGS provides an increase in recognition performance as compared to the baseline but involves costly SVM training at test time. To reduce computational complexity, we propose to use a simpler LDA model [HTF09]. Replacing the SVM with a LDA model has been suggested by the authors of a similar method to MBGS namely the One Shot similarity [TWH$^+$09, WHT08]. However, the novelty of our approach is that we use whitening for MBGS in order to reduce complexity and we study the systematic reduction of used frames. Our hypothesis is, that with fewer data for training a SVM model should perform worse than whitening because its model is independent of the number of frames at test time. We assume that features of a class, in our case face descriptors of a person, have similar covariance matrices Σ [FHT01]. We make this assumption because differences in descriptors of the same person are expected to be person-agnostic and mainly caused by variations in lighting, pose changes and other common factors. This idea benefits from lots of training data and it is important that the data from which the covariance matrices are computed have a similar distribution according to external factors as the test data.

LDA decorrelates features by transforming them to have an identity covariance matrix [FHT01] before classification, this part of LDA is called whitening. Whitening maximizes the ratio between inter-class and intra-class variance, thus making similarities more meaningful. For further details of the relation between whitening and LDA we refer to the work of Hariharan et al. [HMR12].

We compute the covariance matrix Σ for face descriptors in the background set. The whitening matrix W is then defined in terms of the eigenvalues D and the eigenvectors U of Σ as

$$W = D^{-1/2}U^T \quad with \quad \Sigma = UDU^T. \tag{5.2}$$

Since Σ is often sparse and singular, we add a small value $\lambda = 0.01$ to the diagonal matrix D. We found that the choice of λ is important and depends on the type of the descriptor as well as the rank of Σ. The computation of the whitening matrix can be done off-line. To compute similarity between descriptors x_i, y_j, we compute the inner product of whitened and centered descriptors as

$$s_{i,j} = < W(x_i - \mu_0), W(y_j - \mu_0) >, \tag{5.3}$$

where μ_0 is the mean descriptor of the background set. The operation in (5.3) can be computed efficiently for all descriptors using a single matrix multiplication.

To summarize, while MBGS builds one model for each face with the help of a background set, we use the background set in order to obtain a model and transform our descriptor space to a decorrelated space. This space can be better for differentiating between classes than the original Euclidean, because it maximizes inter-class and reduces intra-class variance. As mentioned before this method is simple and fast and is expected to bring most improvement when the distributions regarding external factors for training and test data are similar.

5.2 Frame selection

From a computation complexity point of view, we would like to use as few frames of a video as possible, while retaining or even increasing recognition performance. In the best case, we would only choose two frames, $x_1 \in X$ and $y_1 \in Y$ which either minimize or maximize the distance s depending on whether the two tracks are of the same person or not. In order to account for noise we can also choose to select only a few frames and empirically test different frame selection strategies for different numbers of frames. Each of our strategies should produce a list of n best face candidates from a face track in a video. The main motivation for this is, to find biases for each face descriptor algorithm for which they produce better results. If for example a CNN was trained only on high-quality frontal faces, it would make sense, given the choice, to also use similar images for face recognition, as the CNN will work best on data which is similar to the data it was trained with.

Baseline - Random

As a baseline to compare our selection strategies against, we shuffle the frames and take n frames from two videos X and Y. To calculate a final score between two face tracks, distances are computed exhaustively and averaged like in Equation 5.1.

Medoid

The medoid is the point of a cluster, which has the minimal average distance to all other points in the cluster. It is similar to the average of all descriptors but is more robust to outliers and is always a point of the dataset. One of our main assumptions is that variations due to external factors in videos will lead to a roughly normal distribution of descriptors of one face track. We compute the medoid score for each frame of a video as the sum of distances from one descriptor to all other descriptors. Thus the medoid should, if not produce an optimal ordering, at least rank outliers low. If the whitening is used, we use the dot product as a similarity metric, otherwise the Euclidean distance.

Face detector score [MBPV14]

A face detector (see Section 4.1), additionally to detecting faces, assigns a score to each detection. This score can be interpreted as a confidence that the detection is indeed a face and not a false positive. There could be different options, we are using a state-of-the-art deformable part model detector from [MBPV14], which operates based on the principles outlined in [FGMR10] in more detail. We choose this detector because it could be used in real world applications due to its speed and low false positive rate.

Closed eyes score

One difficulty in face recognition is, that the face is not rigid. Emotions, aging and eye blinking, can change the appearance of a face. Our assumption is that face recognition works better on faces with either open or closed eyes, because this reduces the complexity an algorithm has to learn. First landmarks are extracted according to [ESZ06] to find the position of the eyes in a face. Then a square is cropped with help of the detected eye corners in 2*(distance between eye corners) size and the crop is resized to 24x24 like in [STLC14].

Similar to the authors of [STLC14] who use Histograms of Principal Oriented Gradients (HPOG), in this work Pyramid Histogram of Gradients (PHOG) [BZM07] are as descriptors.

In the next step a binary SVM is trained to recognize closed eyes from the previously extracted PHOG features from the publicly available closed eye database [STLC14]. As a score for ordering the frames of a video the mean confidence of the classification score of both eyes from the trained SVM is used.

The method was evaluated with the CEW guidelines: training 10 times with random 100 open and 100 closed eye patches and using the rest for testing.

Additionally, we present results for 10 fold cross-validation, because for the application of face recognition in video it is more interesting to train on more data and have a more robust classifier. We report accuracy and standard deviation for 10 runs like the database authors suggest [STLC14].

Evaluation	HPOG+SVM [STLC14]	MHPOG+SVM [STLC14]	PHOG (ours)
CEW	93.13 ± 0.45	93.51 ± 0.36	85.57 ± 1.1
10-fold	n.A	n.A	90.71 ± 1.1

Face pose

The yaw angle of a face is estimated from face landmarks [ESZ06]. The ranking of faces is done by minimizing the absolute value of the yaw.

In the following Section we present our results with the presented methods and compare them with the state of the art for both speed and recognition performance.

CHAPTER 6

Results

We set out with the goal, to study how face recognition in video can be improved, both in accuracy but importantly also in complexity, since this is crucial for real-life scenarios. We present our results and compare them to other methods. In our work we use two databases with different evaluation methods and three different face descriptors as described in Section 4.4 to show that our methods work.

6.1 Evaluation

In this section we explain the methods we use to evaluate our approach in detail. We describe the databases we used and some crucial experimentation detail.

YouTube Faces Database (YTF) [WHM11].

This dataset contains a total of 3425 videos of 1595 subjects. The average length of a video is 181.3 frames. The clips were downloaded from YouTube by searching for individuals in the LFW dataset [HRBLM07]. The authors provide 10 folds for face verification. Each fold is independent, this means no person in one fold appears again in another fold. Each fold has 250 pairs of different persons and 250 of same persons. This sums up to 5000 video pairs in total. We follow the YTF restricted evaluation setup for frame selection and score calculations. Face recognition performance is measured as the mean accuracy over the 10 training - test folds of the dataset.

TV Series Face Tubes (TSFT) [SP15].

This dataset consists of 589 face tracks, with an average length of 55 frames. Face detections were manually annotated for every 10 frames and linearly interpolated. The database contains some challenging tracks, due to rapid movements and different appearances of characters (see figure 6.1). The tracks are from 8 different series and of 94

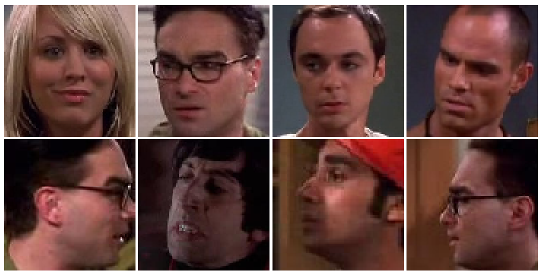


Figure 6.1: Examples from the TSFT database. Top row: high quality faces. Bottom row: faces that are difficult to recognize [SP15]

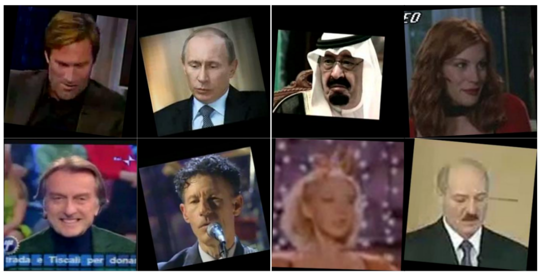


Figure 6.2: Some already aligned faces from the YTF database [WHM11]

subjects. We use the proposed unknown persons evaluation split of the authors. We use the training set for background generation. The test set contains 414 positive pairs and 6141 negative pairs for face verification. We follow the authors of [SP15] and measure Average Precision for evaluation and comparison purposes.

Experimentation Details

The CNN descriptors were computed with Torch accelerated by GPU. All descriptor comparison, whitening and MBGS computations were done with MATLAB. It is notable, that the training performance of SVM for MATLAB can be accelerated immensely by using an implementation that solves another formulation of the SVM problem more efficiently. Differences for different implementations can be seen in a benchmark in Figure6.3. More details can be found in the Liblinear paper [FCH+08].

YTF database Faces were detected with the Viola Jones face detector. The bounding box of faces were expanded by 2.2 and then resized to 200 x 200 pixels. In the next step the authors cropped a 100 x 100 window at the face center. Faces are 2D aligned by facial feature points with the help of [ESZ06]. LBP descriptors were provided by the authors of [WHM11], they convert the images to grayscale and compute LBP features for the YTF dataset. For computing CNN AlexNet and CNN VGG descriptors, the already 2D aligned and cropped faces provided by the YTF dataset were used.

TSFT database We used a DPM model for face registration and extract CNN AlexNet and CNN VGG features. For the frames the detector did not find a face for, we selected a centered crop for faces with width and height bigger than 100 pixel. For smaller images we do not crop, but use the whole image. In total the DPM face detector did not find faces for 2478 out of a total of 32209 frames.

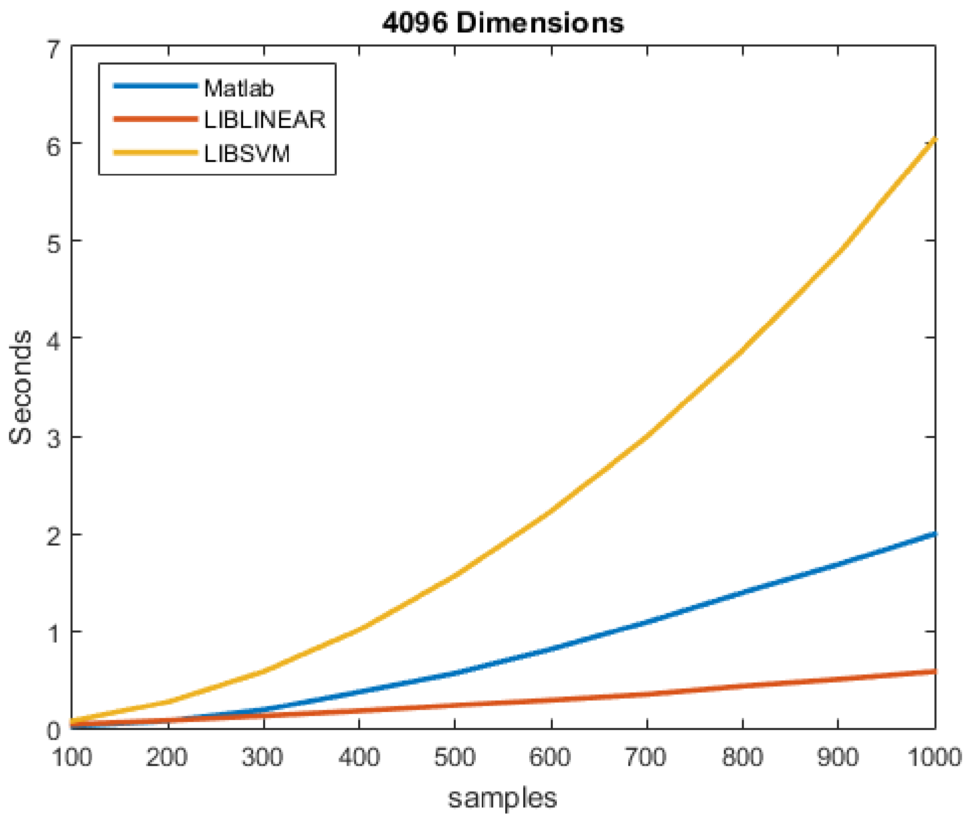


Figure 6.3: Benchmark of Matlab, Liblinear and Libsvm SVM implementations

6.2 Face recognition using all frames

In this section we show that feature whitening is competitive with other approaches using all frames of a video for face verification. We compare against the average of all Euclidean distances between frame pairs as our baseline described in Section 5.1 and MBGS from Section 5.1.

YTF dataset. For all three face descriptors, MGBS and feature whitening both show a clear improvement compared to the original descriptors and score fusion in Figure 6.5. We show that whitening performs competitively with the state-of-the-art methods when applied to LBP descriptors, for example Latent max margin ML [SP15] reaches 0.86 Area Under Curve (AUC) whereas whitening reaches 0.84 AUC.

TSFT dataset. Results for TSFT are shown in Figure 6.7. MBGS and feature whitening have a larger effect on the AlexNet descriptor than on the VGG descriptor. The random chance (assigigning random labels to face pairs for verification) for the TSFT dataset is 6.3% average precision, because the dataset has much more negative pairs than

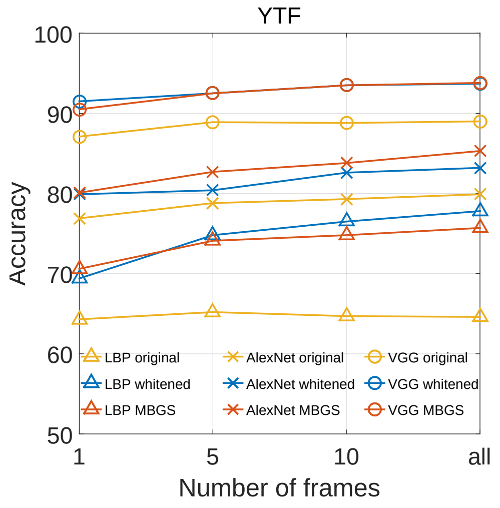


Figure 6.4: **Results for YTF:** Frames sorted by face detector confidence score. Original are the face descriptors with all to all set-based similarity

LBP	Original	MBGS	Whitening
Wolf et al.[WHM11]	63.7	76.4	-
MBGS + SVMΘ [WL13]	79.4	-	-
DDML [HLT14]	81.2	-	-
LBP all frames	64.6	75.7	**77.8**
LBP 5 frames	65.2	74.8	74.1
LBP 1 frame	64.3	70.6	**69.4**
CNN	Original	MBGS	Whitening
FaceNet [SKP15]	95.1	-	-
DeepFace [TYRW14]	92.5	-	-
DeepId2+ [SWT15]	93.2	-	-
VGG [PVZ15]	91.6	-	-
VGG + emb [PVZ15]	97.3	-	-
AlexNet all frames	79.9	85.3	**83.2**
AlexNet 5 frames	78.8	80.5	82.7
AlexNet 1 frame	76.9	80.1	**79.9**
VGG all frames	89.0	93.8	**93.7**
VGG 5 frames	88.9	92.5	92.5
VGG 1 frame	87.1	90.5	**91.5**

Figure 6.5: **Results for YTF:** Recognition accuracy (%) for the YTF dataset.

positive. By using VGG descriptors, we outperform the state-of-the-art for the TFST database by a large margin, improving average precision from 38.8[SP15] to 88.6 percent.

MBGS shows a clear improvement compared to the original baselines on both datasets and for all three tested descriptors. Notably, our proposed feature whitening demonstrates similar improvements despite requiring fewer computations. As expected, we also note that both types of CNN descriptors outperform methods using the LBP descriptor.

VGG outperforms AlexNet due to the more advanced CNN architecture and more variation in the training data. Moreover, even for VGG, which already has high accuracy, whitening provides significant improvements. This shows that whitening works for multiple descriptors on multiple datasets which is strong empirical evidence.

6.3 Frame selection

Figure 6.9 presents results for different methods of frame selection using 1 frame as described in Section 5.2. We note that frame selection based on head pose and open eyes performs close to random for both datasets. Face detector confidence and medoid methods clearly outperform the random baseline for all three descriptors and two datasets. These results demonstrate that the choice of frame selection has a clear impact on the

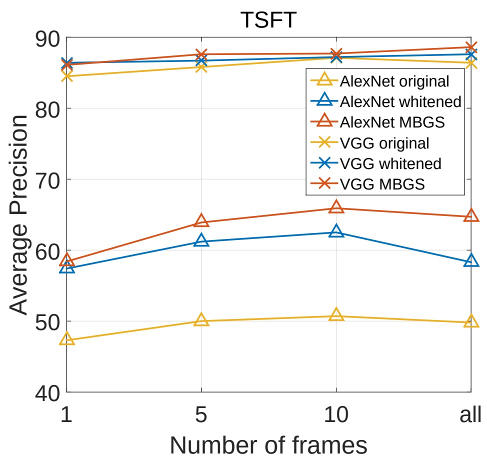


Figure 6.6: **Results for TSFT:** Frames sorted by face detector confidence score. Original are the face descriptors with all to all set-based similarity

Method	Original	MBGS	Whitening
Chance	6.3	6.3	6.3
Latent ML [SP15]	38.8	-	-
AlexNet all frames	49.8	64.7	**58.3**
AlexNet 5 frames	50.0	61.2	63.9
AlexNet 1 frame	47.3	58.4	**57.4**
VGG all frames	86.4	88.6	**87.6**
VGG 5 frames	85.8	86.7	87.6
VGG 1 frame	84.5	86.1	**86.4**

Figure 6.7: **Results for TSFT:** Average precision (%) for TSFT dataset.

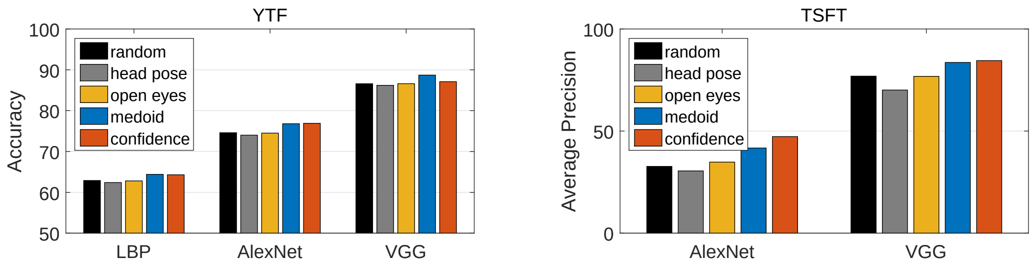
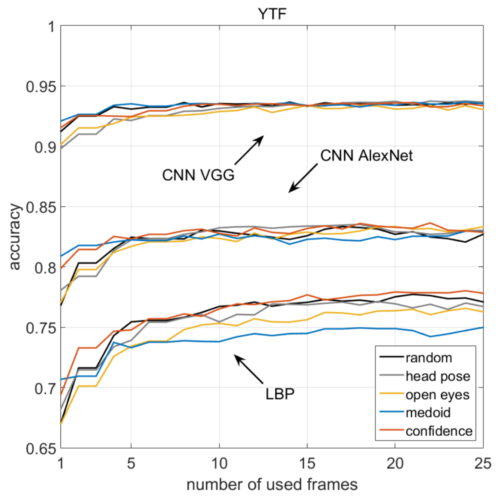


Figure 6.8: **Best frame selection:** Results for different methods using 1 frame from Section 5.2 for YTF and TSFT.

recognition performance. From the tables in Figures 6.5 and 6.7 we conclude that only the best frame according to confidence and using whitening, is better or equal than using all frames with the base line method, on all datasets and with all descriptors. In other words, we see evidence over different descriptors and datasets that whitening can be used to reduce the number of frames by orders of magnitudes while maintaining the same recognition accuracy. For real world applications this is an interesting trade off, because face recognition can be tuned based on the applications need either for speed or accuracy.



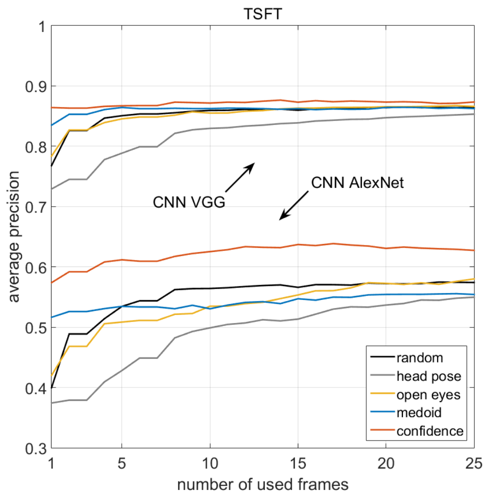


Figure 6.9: **Frame selection:** Results for different methods using multiple frames with whitening from Section 5.2 for YTF and TSFT.

6.4 Trade-off regarding number of frames

In some use cases it might be more important to use as few frames as possible while sacrificing some recognition performance, for example when a system is under high load but wants to stay responsive. Or if the recognition speed is more important than accuracy. In Tables 6.4 and 6.6 we compare results for different numbers of frames and show how the number of frames and performance relate to each other.

YTF dataset For the Youtube dataset in Table 6.4 we can observe that using 10 frames from each face track results in almost the same performance as when using all frames. For LBP and AlexNet the effect is even more evident, while using one frame for LBP and whitening results in 69.4 percent accuracy using 10 frames results already in 76.5%. Using all frames yields a further but relatively smaller improvement to 77.8% accuracy.

TSFT dataset Results for the TSFT database in Table 6.6 are similar to YTF when using VGG descriptors. For AlexNet descriptors using 10 frames performs better than using all frames. One possible reason for this is that the database contains more challenging, hand annotated face tracks, whereas YTF includes only faces that were detected by a face detector. Some examples for challenging faces can be seen in Figure 6.1.

Overall we observe that using the first 10 frames sorted by the face detector confidence, always performs better than using less frames. On the other hand the gain in performance from using all frames compared to 10 frames is marginal or none. We conclude that

Table 6.1: **Video face recognition benchmark:** Average runtimes (sec.) over 50 runs for comparing one test video with 500 videos in the database. The runtimes are measured for Intel Core i5-4460 3.2 GHz CPU with 8 GB memory.

Method	Original	MBGS	Whitening
AlexNet 1 frame	0.023	3.5	0.013
AlexNet 5 frames	0.032	3.7	0.016
AlexNet 10 frames	0.035	3.8	0.018
AlexNet all TSFT	0.087 ± 0.02	4.9 ± 0.5	0.039 ± 0.015
AlexNet all YTF	0.464 ± 0.33	9.5 ± 3.1	0.19 ± 0.14
VGG 1 frame	0.04	126	0.025
VGG 5 frames	0.11	129	0.05
VGG 10 frames	0.16	134	0.065
VGG all TSFT	1.9 ± 0.65	160 ± 11	0.29 ± 0.12
VGG all YTF	9.6 ± 15.7	242 ± 52	4.1 ± 14.8

selecting a small number of frames for face recognition in video can be used without significantly degrading its performance.

6.5 Computational complexity and run time

In this section we investigate the trade-off between the quality and the speed for different methods. Let k be the number of dimensions of the face descriptor and n the number of frames of a face track which are used for comparison. Testing with feature whitening needs only one matrix multiplication (inner product) in $O(k^2n)$, because whitened descriptors only have to be computed once. MBGS needs $O(kn + (Cn))$ for the background set of size $C = 1000$ using Liblinear SVM implementation [FCH+08] and the nearest neighbor search. We benchmark runtimes for the following setup: we compare 1 video with 500 different videos, this setup is relevant for real world use cases. In Table 6.1 we show results for the AlexNet (k=160) and VGG (k=4096) descriptors. The differences in runtimes for TSFT and YTF with all frames can be explained with different average tracks lengths, 181 frames for YTF and 55 frames for TSFT. In practice the number of dimensions is important as it has linear impact on runtime. Overall we observe that whitening is several orders of magnitude faster compared to MBGS for both types of face descriptors, while the recognition performance for whitening and MBGS is similar.

CHAPTER 7

Conclusion

Our contributions shown in this thesis are numerous. We show that whitening can be used to replace matched background similarity with comparable qualitative performance. We do so by comparing results for three descriptors and two databases. One advantage when using whitening as compared to MBGS is the orders of magnitude faster computation of distances for face verification. This result can be generalized and used for other domains that make use of set based similarities with SVMs.

We show that by selecting descriptors used for distance calculation based on domain specific criteria (face detector score), the recognition performance can be increased or kept the same, while reducing the number of descriptors used by up to 90 percent and more. While this effect is domain specific and has to be tuned for specific applications, in real world applications this is justifiable by the large reduction of the number of samples needed for stable verification.

Summary

In this work we explain why face recognition is a promising biometric verification method and why it is important to study how it can be used in real world scenarios, for example with video cameras. We explain three important classifiers, linear discriminant analysis, support vector machines and convolutional neural networks. The latter have changed the computer vision landscape in the last few years by dramatically improving results on the renowned large scale object recognition challenge ImageNet. We explain how CNNs work, starting from the perceptron to deep convolutional networks in detail and further describe important parts such as Batch normalization, ReLU, different loss functions and data augmentation. Next we outline the pipeline necessary for face recognition in still images and give a high level overview. We explain each stage of this pipeline beginning at face detection and object detection in general over face alignment and normalization and when it can be useful. We end by explaining the core piece of the pipeline, how descriptor training and computation is done, by the example of three different face descriptors. Next, we explain the differences of video face recognition to face recognition in still images. We present a method, whitening, which increases recognition performance similar to other methods like Matched background similarity, but can be computed orders of magnitudes faster. Furthermore, we define different methods for choosing a subset of frames of a face track that in some cases increase recognition performance, but in all cases reduces computational complexity. We then present empirical results of the aforementioned methods for the famous YouTube Faces Database and the new TV Series Face Tubes database. We also show the trade-off between the computational complexity of our method and recognition performance.

Discussion

We have shown that whitening improves recognition performance to a similar extent as MBGS while being orders of magnitudes faster. The performance is increased not only when used for all frames but also for less and even a single frame, however, the choice of frames is important. We have investigated different methods for frame selection. Medoid and face detector confidence have shown the overall best performance, while head pose and open eyes give no improvement. Furthermore, we have compared the gain of selecting only one frame versus the gain of selecting multiple or all frames from a video. If possible, selecting more frames will be advantageous for recognition performance, however, this depends also on the used descriptor, as for some descriptors which cannot account for external factors may result in very different descriptors on frames that are very diverse.

In practice, while there is less difference between using one or all frames in computational cost (because of high constant costs), the number of frames that have to be transmitted over a network, if capture and recognition are not done on the same processing unit, can be greatly reduced by employing whitening or MBGS and using fewer frames. Once again it depends on the application to decide between the trade-off of lower bandwidth (fewer frames), lower computational cost (whitening) and recognition performance.

The results, especially regarding whitening, cannot only be used for faces but more general also for other applications for computer vision where some effects of external factors can be diminished, if multiple different depictions of an object are available, which is particularly interesting for videos. While other methods exist that outperform ours in quality, whitening is computationally efficient. In our experiments, it was even faster than a method using the Euclidean distance as similarity measure for comparing one with many descriptors. This paired with the fact that it increases recognition performance drastically makes the method interesting for many use cases.

Future work

With our work we showed that it is possible to reduce the number of frames used for face recognition, to use only a fraction of frames without losing any recognition performance, which can mainly be explained through the high redundancy of information in videos. We studied several criteria for selecting frames. One possibility to improve upon the criteria would be to learn a regressor on the descriptors of faces or directly on face images, that predicts the face quality. However, it might be more useful in a real world application to either improve the face detector score training, so that the score not only represents the probability of a face but the quality of a face, as the face detector is in many cases applied directly before face recognition. As annotation for face quality, one could try to find those frames in a video, that increase recognition performance. This would be an exhaustive procedure and would require a lot of processing time, however, it poses a way to directly create annotations for a specific face descriptor.

Another interesting approach is to try how CNN based descriptors trained with different loss functions, especially triplet loss [HA15], Siamese loss [HCL06] and the new centered loss [WZLQ16] influence the performance of whitening in order to better understand and test the hypothesis of our whitening approach.

Bibliography

[AKC15] Shervin Rahimzadeh Arashloo, Josef Kittler, and William Christmas. Face spoofing detection based on multiple descriptor fusion using multiscale dynamic binarized statistical image features. *IEEE Transactions on Information Forensics and Security*, 10(11):2396–2407, 2015.

[Ara16] Ognjen Arandjelovic. Learnt quasi-transitive similarity for retrieval from large collections of faces. *arXiv preprint arXiv:1603.00560*, 2016.

[BB91] Anne Olivia Boyer and Robert S. Boyer. *A Biographical Sketch of W. W. Bledsoe*, pages 1–29. Springer Netherlands, Dordrecht, 1991.

[BBFB12] Jeremiah R Barr, Kevin W Bowyer, Patrick J Flynn, and Soma Biswas. Face recognition from video: A review. *International Journal of Pattern Recognition and Artificial Intelligence*, 26(05):1266002, 2012.

[BHK97] Peter N Belhumeur, João P Hespanha, and David J Kriegman. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 19(7):711–720, 1997.

[Bot10] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer, 2010.

[BRKKJ13] Lacey Best-Rowden, Brendan Klare, Joshua Klontz, and Anubhav K Jain. Video-to-video face matching: Establishing a baseline for unconstrained face recognition. In *IEEE Sixth International Conference on Biometrics: Theory, Applications and Systems (BTAS),*, pages 1–8. IEEE, 2013.

[BRM13] Tadas Baltrusaitis, Peter Robinson, and Louis-Philippe Morency. Constrained local neural fields for robust facial landmark detection in the wild. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 354–361, 2013.

[BSO$^+$15] A Benlamoudi, D Samai, A Ouafi, SE Bekhouche, A Taleb-Ahmed, and A Hadid. Face spoofing detection using multi-level local phase quantization (ml-lpq). 2015.

[BZM07] Anna Bosch, Andrew Zisserman, and Xavier Munoz. Representing shape with a spatial pyramid kernel. In *Proceedings of the 6th ACM international conference on Image and video retrieval*, pages 401–408. ACM, 2007.

[CL11] Chih-Chung Chang and Chih-Jen Lin. Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011.

[CMM$^+$11] Dan C Ciresan, Ueli Meier, Jonathan Masci, Luca Maria Gambardella, and Jürgen Schmidhuber. Flexible, high performance convolutional neural networks for image classification. In *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*, volume 22, page 1237, 2011.

[CV95] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[DBSDlT11] Oscar Déniz, Gloria Bueno, Jesús Salido, and Fernando De la Torre. Face recognition using histograms of oriented gradients. *Pattern Recognition Letters*, 32(12):1598–1603, 2011.

[DDS$^+$09] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE, 2009.

[dFPADMM12] Tiago de Freitas Pereira, André Anjos, José Mario De Martino, and Sébastien Marcel. Lbp- top based countermeasure against face spoofing attacks. In *Asian Conference on Computer Vision*, pages 121–132. Springer, 2012.

[ESZ06] Mark Everingham, Josef Sivic, and Andrew Zisserman. Hello! my name is... buffy –automatic naming of characters in tv video. In *BMVC*, volume 2, page 6, 2006.

[FCH$^+$08] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. Liblinear: A library for large linear classification. *The Journal of Machine Learning Research*, 9:1871–1874, 2008.

[FGMR10] Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, 2010.

[FHT01] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics Springer, Berlin, 2001.

[FN75] Keinosuke Fukunaga and Patrenahalli M. Narendra. A branch and bound algorithm for computing k-nearest neighbors. *IEEE transactions on computers*, 100(7):750–753, 1975.

[GB10] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Aistats*, volume 9, pages 249–256, 2010.

[GDDM14] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

[GdQ15] Diogo Caetano Garcia and Ricardo L de Queiroz. Face-spoofing 2d-detection based on moiré-pattern analysis. *IEEE Transactions on Information Forensics and Security*, 10(4):778–786, 2015.

[GLLC05] Rodney Goh, Lihao Liu, Xiaoming Liu, and Tsuhan Chen. The cmu face in action (fia) database. In *Analysis and Modelling of Faces and Gestures*, pages 255–263. Springer, 2005.

[HA15] Elad Hoffer and Nir Ailon. Deep metric learning using triplet network. In *International Workshop on Similarity-Based Pattern Recognition*, pages 84–92. Springer, 2015.

[HBFS01] Sepp Hochreiter, Yoshua Bengio, Paolo Frasconi, and Jürgen Schmidhuber. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies, 2001.

[HCL06] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742. IEEE, 2006.

[HLT14] Junlin Hu, Jiwen Lu, and Yap-Peng Tan. Discriminative deep metric learning for face verification in the wild. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

[HMR12] Bharath Hariharan, Jitendra Malik, and Deva Ramanan. Discriminative decorrelation for clustering and classification. In *Computer Vision–ECCV 2012*, pages 459–472. Springer, 2012.

[Hoc91] Sepp Hochreiter. Untersuchungen zu dynamischen neuronalen netzen. *Diploma, Technische Universität München*, page 91, 1991.

[HRBLM07] Gary B Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical report, Technical Report 07-49, University of Massachusetts, Amherst, 2007.

[HTF09] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *Unsupervised learning*. Springer, 2009.

[HZRS15a] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015.

[HZRS15b] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1026–1034, 2015.

[ICA06] ICAO. 9303-5 machine readable travel documents-part 1-2. Technical report, International Civil Aviation Organization, 2006.

[Ind16] Cisco Visual Networking Index. The zettabyte era—trends and analysis. *Cisco white paper*, 2016.

[IS15] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[Kan77] Takeo Kanade. *Computer recognition of human faces*, volume 47. Birkhäuser, 1977.

[Kan15] Hyunho Kang. Face anti-spoofing based on image block difference and logistic regression analysis. In *Consumer Electronics-Berlin (ICCE-Berlin), 2015 IEEE 5th International Conference on*, pages 493–495. IEEE, 2015.

[KBBN09] N. Kumar, A. C. Berg, P. N. Belhumeur, and S. K. Nayar. Attribute and Simile Classifiers for Face Verification. In *IEEE International Conference on Computer Vision (ICCV)*, Oct 2009.

[KBL15] Sooyeon Kim, Yuseok Ban, and Sangyoun Lee. Face liveness detection using defocus. *Sensors*, 15(1):1537–1563, 2015.

[KS14] Vahid Kazemi and Josephine Sullivan. One millisecond face alignment with an ensemble of regression trees. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1867–1874, 2014.

[KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[Kul12] Brian Kulis. Metric learning: A survey. *Foundations and Trends in Machine Learning*, 5(4):287–364, 2012.

[KWRB11] Martin Koestinger, Paul Wohlhart, Peter M. Roth, and Horst Bischof. Annotated facial landmarks in the wild: A large-scale, real-world database for facial landmark localization. In *First IEEE International Workshop on Benchmarking Facial Image Analysis Technologies*, 2011.

[LAE$^+$15] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. SSD: Single shot multibox detector. *arXiv preprint arXiv:1512.02325*, 2015.

[LBBH98] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[LBD$^+$89] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

[LHYK05] K.C. Lee, J. Ho, M.H. Yang, and D. Kriegman. Visual tracking and recognition using probabilistic appearance manifolds. *Computer Vision and Image Understanding*, 2005.

[LLS$^+$15] Haoxiang Li, Zhe Lin, Xiaohui Shen, Jonathan Brandt, and Gang Hua. A convolutional neural network cascade for face detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5325–5334, 2015.

[LP16] Qianli Liao and Tomaso Poggio. Bridging the gaps between residual learning, recurrent neural networks and visual cortex. *arXiv preprint arXiv:1604.03640*, 2016.

[Luo14] Hengliang Luo. Siamese network architecture and applications in computer vision. 2014.

[LVB$^+$93] Martin Lades, Jan C Vorbrüggen, Joachim Buhmann, Jörg Lange, Christoph V d Malsburg, Rolf Wurtz, and Wolfgang Konen. Distortion invariant object recognition in the dynamic link architecture. *Computers, IEEE Transactions on*, 42(3):300–311, 1993.

[LWTJ04] Jiangwei Li, Yunhong Wang, Tieniu Tan, and Anil K Jain. Live face detection based on the analysis of fourier spectra. In *Defense and Security*, pages 296–303. International Society for Optics and Photonics, 2004.

[MaTaH$^+$16] Iacopo Masi, Anh Tu an Trãn, Tal Hassner, Jatuporn Toy Leksut, and Gérard Medioni. Do we really need to collect millions of faces for effective face recognition? In *European Conference on Computer Vision (ECCV)*, October 2016.

[MBPV14] M. Mathias, R. Benenson, M. Pedersoli, and L. Van Gool. Face detection without bells and whistles. In *ECCV*, 2014.

[MD13] T Mikolov and J Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 2013.

[MP69] Marvin Minsky and Seymour Papert. Perceptrons. 1969.

[NH10] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 807–814, 2010.

[NM08] Kamal Nasrollahi and Thomas B Moeslund. Face quality assessment system in video sequences. In *Biometrics and Identity Management*, pages 10–18. Springer, 2008.

[O'G03] Lawrence O'Gorman. Comparing passwords, tokens, and biometrics for user authentication. *Proceedings of the IEEE*, 91(12):2021–2040, 2003.

[OPH96] Timo Ojala, Matti Pietikäinen, and David Harwood. A comparative study of texture measures with classification based on featured distributions. *Pattern recognition*, 29(1):51–59, 1996.

[OSA14] TOMOHIRO OSAKI. Airport facial recognition system eyed but only for japanese. Technical report, The Japan Times, 2014.

[OYP13] Sung-Kwun Oh, Sung-Hoon Yoo, and Witold Pedrycz. Design of face recognition algorithm using pca-lda combined for hybrid data preprocessing and polynomial-based rbf neural networks: Design and its application. *Expert Systems with Applications*, 40(5):1451–1466, 2013.

[PCP12] Sanford L Palay and Victoria Chan-Palay. *Cerebellar cortex: cytology and organization.* Springer Science & Business Media, 2012.

[PG88] B Pakkenberg and HJG Gundersen. Total number of neurons and glial cells in human brain nuclei estimated by the disector and the fractionator. *Journal of microscopy*, 150(1):1–20, 1988.

[PGM$^+$03] P Jonathon Phillips, Patrick Grother, Ross Micheals, Duane M Blackburn, Elham Tabassi, and Mike Bone. Face recognition vendor test 2002. In *Analysis and Modeling of Faces and Gestures, 2003. AMFG 2003. IEEE International Workshop on*, page 44. IEEE, 2003.

[PHJO15] Keyurkumar Patel, Hu Han, Anil K Jain, and Greg Ott. Live face video vs. spoof face video: Use of moiré patterns to detect replay video attacks. In *2015 International Conference on Biometrics (ICB)*, pages 98–105. IEEE, 2015.

[PS94] Uri Polat and Dov Sagi. Spatial interactions in human vision: from near to far via experience-dependent cascades of connections. *Proceedings of the National Academy of Sciences*, 91(4):1206–1209, 1994.

[PSAS15] Xingchao Peng, Baochen Sun, Karim Ali, and Kate Saenko. Learning deep object detectors from 3d models. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1278–1286, 2015.

[PVZ15] Omkar M Parkhi, Andrea Vedaldi, and Andrew Zisserman. Deep face recognition. *Proceedings of the British Machine Vision*, 1(3):6, 2015.

[Qui86] J. Ross Quinlan. Induction of decision trees. *Machine learning*, 1(1):81–106, 1986.

[QYLH16] Hongwei Qin, Junjie Yan, Xiu Li, and Xiaolin Hu. Joint training of cascaded cnn for face detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3456–3465, 2016.

[RDGF15] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. *arXiv preprint arXiv:1506.02640*, 2015.

[RMG$^+$86] David E Rummelhart, James L McClelland, PDP Research Group, et al. Parallel distributed processing. explorations in the microstructure of cognition. volume 1: Foundations, 1986.

[Ros58] Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.

[SAB07] Daniel L Schacter, Donna Rose Addis, and Randy L Buckner. Remembering the past to imagine the future: the prospective brain. *Nature Reviews Neuroscience*, 8(9):657–661, 2007.

[San] Leonardo Araujo Santo. Artificial Intelligence googlenet introduction. `http://web.archive.org/web/20160810093935/https://leonardoaraujosantos.gitbooks.io/artificial-inteligence/content/googlenet.html`. Accessed: 2016-08-10.

[SCWT14] Yi Sun, Yuheng Chen, Xiaogang Wang, and Xiaoou Tang. Deep learning face representation by joint identification-verification. In *Advances in Neural Information Processing Systems*, pages 1988–1996, 2014.

[SEZ09] Josef Sivic, Mark Everingham, and Andrew Zisserman. “who are you?”-learning person specific classifiers from video. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 1145–1152. IEEE, 2009.

[SF14] Mohammad Amin Sadeghi and David Forsyth. 30hz object detection with dpm v5. In *European Conference on Computer Vision*, pages 65–79. Springer, 2014.

[SHG12] Luuk J Spreeuwers, AJ Hendrikse, and KJ Gerritsen. Evaluation of automatic face recognition for automatic border control on actual data recorded of travellers at schiphol airport. In *Biometrics Special Interest Group (BIOSIG), 2012 BIOSIG-Proceedings of the International Conference of the*, pages 1–6. IEEE, 2012.

[SHK$^+$14] Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[SKP15] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[SLJ$^+$15] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–9, 2015.

[SMB10] Dominik Scherer, Andreas Müller, and Sven Behnke. Evaluation of pooling operations in convolutional architectures for object recognition. In *International Conference on Artificial Neural Networks*, pages 92–101. Springer, 2010.

[SP15] Gaurav Sharma and Patrick Pérez. Latent max-margin metric learning for comparing video face tubes. In *Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2015.

[SPVZ13] Karen Simonyan, Omkar M Parkhi, Andrea Vedaldi, and Andrew Zisserman. Fisher vector faces in the wild. In *BMVC*, volume 5, page 11, 2013.

[SR15] Ross Girshick Jian Sun Shaoqing Ren, Kaiming He. Faster R-CNN: Towards real-time object detection with region proposal networks. *arXiv preprint arXiv:1506.01497*, 2015.

[SRASC14] Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. Cnn features off-the-shelf: an astounding baseline for recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 806–813, 2014.

[STC04] John Shawe-Taylor and Nello Cristianini. *Kernel methods for pattern analysis*. Cambridge university press, 2004.

[STLC14] Fengyi Song, Xiaoyang Tan, Xue Liu, and Songcan Chen. Eyes closeness detection from still images with multi-scale histograms of principal oriented gradients. *Pattern Recognition*, 47(9):2825–2838, 2014.

[svm] Opencv introduction to svm. `http://docs.opencv.org/2.4/doc/tutorials/ml/introduction_to_svm/introduction_to_svm.html`. Accessed: 2016-08-10.

[SWT15] Yi Sun, Xiaogang Wang, and Xiaoou Tang. Deeply learned face representations are sparse, selective, and robust. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[SZ14] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[TP91] Matthew Turk and Alex Pentland. Eigenfaces for recognition. *Journal of cognitive neuroscience*, 3(1):71–86, 1991.

[TWH$^+$09] Yaniv Taigman, Lior Wolf, Tal Hassner, et al. Multiple one-shots for utilizing class label information. In *BMVC*, pages 1–12, 2009.

[TYRW14] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lars Wolf. Deepface: Closing the gap to human-level performance in face verification. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1701–1708. IEEE, 2014.

[USA14] JA Unar, Woo Chaw Seng, and Almas Abbasi. A review of biometric technology along with trends and prospects. *Pattern recognition*, 47(8):2673–2688, 2014.

[VJ01] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I–511. IEEE, 2001.

[VTBE15] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[VWB16] Andreas Veit, Michael Wilber, and Serge Belongie. Residual networks are exponential ensembles of relatively shallow networks. *arXiv preprint arXiv:1605.06431*, 2016.

[WBS05] Kilian Q Weinberger, John Blitzer, and Lawrence K Saul. Distance metric learning for large margin nearest neighbor classification. In *Advances in neural information processing systems*, pages 1473–1480, 2005.

[WHJ15] Di Wen, Hu Han, and Anil K Jain. Face spoof detection with image distortion analysis. *IEEE Transactions on Information Forensics and Security*, 10(4):746–761, 2015.

[WHM11] Lior Wolf, Tal Hassner, and Itay Maoz. Face recognition in unconstrained videos with matched background similarity. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 529–534. IEEE, 2011.

[WHT08] Lior Wolf, Tal Hassner, and Yaniv Taigman. Descriptor based methods in the wild. In *Workshop on faces in'real-life'images: Detection, alignment, and recognition*, 2008.

[WL13] Lior Wolf and Noga Levy. The svm-minus similarity score for video face recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3523–3530, 2013.

[WYL$^+$13] Tao Wang, Jianwei Yang, Zhen Lei, Shengcai Liao, and Stan Z Li. Face liveness detection using 3d structure recovered from a single camera. In *2013 International Conference on Biometrics (ICB)*, pages 1–6. IEEE, 2013.

[WZLQ16] Yandong Wen, Kaipeng Zhang, Zhifeng Li, and Yu Qiao. A discriminative feature learning approach for deep face recognition. In *ECCV*, 2016.

[WZTF15] Donglai Wei, Bolei Zhou, Antonio Torrabla, and William Freeman. Understanding intra-class knowledge inside cnn. *arXiv preprint arXiv:1507.02379*, 2015.

[XYK15] Shengtao Xiao, Shuicheng Yan, and Ashraf A Kassim. Facial landmark detection via progressive initialization. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 33–40, 2015.

[yC88] Santiago Ramón y Cajal. *Estructura de los centros nerviosos de las aves.* 1888.

[YCBL14] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328, 2014.

[YDZL15] Jing Yang, Jiankang Deng, Kaihua Zhang, and Qingshan Liu. Facial shape tracking via spatio-temporal cascade shape regression. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 41–49, 2015.

[ZZ10] Cha Zhang and Zhengyou Zhang. A survey of recent advances in face detection, 2010.

[ZZZ15] Stefanos Zafeiriou, Cha Zhang, and Zhengyou Zhang. A survey on face detection in the wild: past, present and future. *Computer Vision and Image Understanding*, 138:1–24, 2015.