

Face Verification and Open-Set Identification for Real-Time Video Applications

Jakob Grundström

Master's thesis
2015:E13



LUND UNIVERSITY

Faculty of Engineering
Centre for Mathematical Sciences
Mathematics

Abstract

In this work we aim to develop a video-based face verification algorithm suitable for real-time use in an embedded environment with limited space and restricted computational resources. We investigate face verification based solutions to envisioned real-time video applications that require open-set face identification. In particular, we create a prototype system for keeping track of the identities of persons currently inside a closed area. This is done by verifying face images captured at the entry with images captured at the exit.

We consider a pair-wise face verification setup that makes the binary decision if two face images are of matching or non-matching identities. The Joint Bayesian classifier was found useful for this purpose and we show that it compares well to a set of standard classifiers. Face verification is evaluated using two distinct feature types: local feature representations around landmark points and deep representations extracted from Convolutional Neural Networks (convnets) trained for generic object detection and fine-tuned on face image data. With these face verification algorithms we implement and evaluate open-set identification.

Combined with a Joint Bayesian classifier the deep representations show good accuracy applied to face verification, we reached a face verification accuracy of 91.37% on the popular Labeled Faces in the Wild benchmark. The feature extraction with the deep representations is computationally demanding but may become applicable to embedded environments in the future. The local feature representations investigated requires less computational resources but yields lower accuracy, 86.4% on the Labeled Faces in the Wild benchmark.

We identified the local features as the currently most applicable feature representation for the considered real-time video applications in the immediate future. For this reason a local feature scheme based on Local Binary Patterns (LBP) is used in our prototype system. For our demo prototype we set up a video-based face recognition pipeline where tracks of face images are matched, instead of matching singular images. Currently, the demo prototype, running on a desktop computer, process two simultaneous video streams either from recorded files or live from network video and maintain a dynamic gallery of the persons that are inside the closed area.

Contents

1	Introduction	5
1.1	Main Objective	5
1.1.1	Envisioned Real-Time Applications	5
1.2	Problem Description	6
1.2.1	Image-Based Face verification	6
1.2.2	Video-Based Face verification	6
1.2.3	Real-Time Face Recognition	6
1.2.4	Questions for Research	6
1.3	Previous Works on Real-Time Face Identification Systems	7
1.4	Restrictions	7
2	Data-Sets	8
2.1	Pubfig	8
2.2	FaceScrub	8
2.3	ChokePoint	8
2.4	Labeled Faces in the Wild	9
2.5	MSRA-CFW	9
2.6	Data Augmentation	9
2.7	Recording of Video Data	10
2.8	Comments	10
2.9	Evaluations	10
3	Methodology	11
3.1	Face Alignment	11
3.1.1	Background	11
3.1.2	Facial Landmark Detection	11
3.1.3	Fitting Transforms with Point Correspondences	12
3.2	Face Tracks from Video	14
3.2.1	Optical Flow	15
3.2.2	Feature Tracking of Point Features	15
3.2.3	Dropping Feature Points Dynamically	16
3.2.4	Face Tracks from Video Streams	16
3.3	Face Representations based on Local Features	18
3.3.1	Spatial Histograms of Local Binary Pattern	18
3.3.2	Histogram of Oriented Gradients	19
3.3.3	Framework for Extracting Local Features for Face Verification	19
3.4	Convolutional Neural Network-based Deep Face Representations	20
3.4.1	Motivation	20
3.4.2	Background: Convolutional Neural Networks	21
3.4.3	Related Work	22

3.4.4	Fine-Tuning for Face Representations	23
3.5	Linear Dimensionality Reduction Techniques	25
3.5.1	Principal Component Analysis	25
3.5.2	Linear Discriminant Analysis	26
3.6	Classification	27
3.6.1	Nearest Neighbor	27
3.6.2	Joint Bayesian Classifier	27
3.6.3	Standard Classifiers	30
3.7	Best-Shot Selection	31
3.8	Open-Set Face Identification	31
3.9	Video-Based Face Matching: Closed Area Demo Application	32
4	Experimental Evaluation	36
4.1	Face Verification	36
4.1.1	Comparison of Features	36
4.1.2	Comparison of Convnet Architectures	36
4.1.3	Effects of Larger Training Set	38
4.1.4	Classification Algorithms	38
4.1.5	LFW Evaluation	41
4.1.6	Cross-Environment Scenarios with Realistic Data	44
4.2	Open-Set Identification	44
4.2.1	Alarm Curves (Watch-list ROC)	47
4.2.2	Cumulative Match Characteristics	47
4.2.3	Increasing Gallery Size	47
4.3	Timing Experiments	47
4.4	Closed-Area Application	51
4.4.1	Memory Requirements	51
4.4.2	Time-Complexity of Face Matching	51
5	Discussion	53
5.1	Analysis	53
5.2	Future Work	55
5.3	Conclusion	57
6	References	60

Chapter 1

Introduction

As a popular topic in computer vision face recognition has received significant research efforts during the past several years (Jafri and R.Arabnia, 2009). Face recognition has many important applications in video analysis, for example access control, person identification and can constitute a key component in person re-identification systems. The primary advantage of face recognition compared to other biometrics is its non-intrusive nature and that it does not require active cooperation. Two main tasks in face recognition are *verification* (authentication) and *identification*, problems which are related. Identification concerns correctly identifying face images, usually assigning them to the identity labels in a database. The purpose of face verification is to confirm or deny a claimed identity. As a consequence verification need to explicitly consider the possibility of identities not present in the training data. Such unknown or unseen identities are called *impostors*. Identification required to handle impostors in addition to the identities present in the training database is called *open-set* identification. As opposed to *closed-set* identification which instead assumes all identities are within a finite and known set. Video-based face recognition in uncontrolled settings need to address challenges such as changes in illumination, variation in head pose, facial expression, occlusion, translation, scaling, image compression, artifacts, motion blur and out of focus effects. This is done by designing or learning invariant features for the face representation and by designing more robust face recognition algorithms.

1.1 Main Objective

The aim of this work is to develop a video-based face verification algorithm suitable for real-time use in an embedded environment with limited space and restricted computational resources. Also, we want to investigate face verification based solutions to potential real-time applications (listed in 1.1.1) by creating a prototype.

1.1.1 Envisioned Real-Time Applications

Three potential applications that can be implemented with face verification are:

1. *Closed-Area Application* Keep track of identities currently inside a closed area and record identities and stay times by verifying face images of persons entering with images of the persons leaving.
2. *Multi-Camera Face Tracking* Newly observed identities are verified against previously observed identities and if the identity is recognized a common label is handed over otherwise a new label is created.
3. *Face Search* Face search in a real-time video stream based on an existing face image or face descriptor that is verified against all new faces that appear in the video stream.

As described in Section 3.9 we implement the Closed-Area Application as a demo application and to test the face verification algorithm's real-time properties. The same mechanism as in the demo application can be used to implement *Multi-Camera Face Tracking* and *Face Search*.

1.2 Problem Description

1.2.1 Image-Based Face verification

Image-based face verification can be posed as a pair-matching problem, where the verification algorithm determines if two face images belong to the same identity. This allows for a binary decision: match or non-match. A verification algorithm according to this formulation theoretically works for any pair of identities, i.e. when one or both identities are not present in the training data seen by the algorithm.

An alternative approach is to verify a single face image against a subject-specific model trained on a set of training images from a single person. The problem is whether to confirm or deny that the test image has the same identity as the model. This type of solution requires data for all persons the algorithm should be able to verify.

For both approaches the binary assignments are usually associated with a confidence score or similarity measure that is thresholded in order to get the final decision.

In this project we pursue the first problem formulation because a single trained model works for any pair of identities. This is convenient for example in the *Closed-Area Application* which requires the algorithm to keep track of a dynamic set of identities and this setup enables us to do this without online re-training. The second approach might be more suitable if all the identities are known beforehand and sufficient training data is available for each person. For a dynamic set this approach would require online re-training. Other motivations for this choice is that a pair-wise formulation makes it possible to generate a lot of data and it appears more inline with the ongoing face recognition research.

1.2.2 Video-Based Face verification

A common way to extend image-based face recognition to video is by the use of face tracking (Ramanan et al., 2007) (Everingham et al., 2009) (Fischer et al., 2011). We adopt this idea and extend the image-based face verification algorithm to video by using face tracking to temporally link face images of the same identity and then match the produced tracks using the image-based face verification.

Approaches extracting features directly from image-sequences or video exists, but is not explored in this thesis.

1.2.3 Real-Time Face Recognition

To test the real-time feasibility of the video-based face verification algorithm we place it in a real-time pipeline where face detection captures face images from the video frames. To cope with the uncontrolled settings of real-world data pre-processing and face alignment is performed. A face tracker groups the detected faces into tracks and then recognition is performed on the face tracks. In this real-time pipeline we implement the *Closed-Area* application with open-set face identification built on face verification.

1.2.4 Questions for Research

The problem description above leads to the following research questions:

- Are technologies inspired by well-performing still-image face verification algorithms applicable for real-time purposes on embedded hardware? If not, can we identify a good trade-off between accuracy and performance when it comes to developing a face verification algorithm?
- Are face recognition technology alone sufficient to achieve acceptable accuracy and performance for the target applications? This is interesting considering the research efforts put on person re-identification using multi-modal approaches.
- How much data is needed and what pre-processing does it need?
- What feature descriptors and representation are effective for face verification-based recognition? Under what circumstances are learned features be a better option than hand-crafted feature representations?
- Can we produce both a compact and accurate feature representation, achieved for example with feature selection or dimensionality reduction techniques?

1.3 Previous Works on Real-Time Face Identification Systems

The authors of (Ekenel et al., 2009) propose a visitor interface system with face verification based open-set face identification. Their real-time performing system includes face registration via face detection and face tracking. The face representation is local-feature based on the Discrete Cosine Transform (DCT). Classification is performed with linear support vector machines (SVM) and the distance to the hyperplane is used as confidence. Another real-time person identification system is presented by (Apostoloff and Zisserman, 2007). On top of the face detection, face tracking and sequence extraction it makes use of normalized intensity features extracted from facial landmark centered image patches and a random-ferns classifier. In a related system, (Sivic et al., 2009), Histogram of Oriented Gradients (HOG) features are used instead.

A key difference with this work, is that we design for a dynamic database and cannot expect the algorithm to be trained on the identities that are added to the database.

1.4 Restrictions

In this work we consider only frontal-faces, i.e. the range of face images that can be detected by a industry standard frontal-face detector. Most of the publicly available data-sets for face recognition are frontal-faces, therefore in order to have a reasonable amount of training data we consider frontal-faces. So far, recognition algorithms based on front faces have been most successful. We use an off-the-shelf solution for face detection to allow us to completely focus on the face recognition.

For the closed area demo application we assume relatively controlled settings with two cameras, the first is ceiling-mounted and centered in front of a unidirectional entrance. The second is mounted in the same way but facing a unidirectional exit. With the same functionality implemented in the closed area application it is possible to at a later time create more complex setups, such as a setup with two bi-directional doorways and four cameras.

Further, we primarily consider a short time-

span measured in terms of hours or a working day. Because this is the time-span that is relevant for the target application.

These assumptions let us setup a general scenario using a minimum of cameras and let focus be put on the face recognition without simplifying the problem too much.

Chapter 2

Data-Sets

For the training and evaluation of this face recognition research we have identified and selected a handful of publicly available data-sets. In the listing shown below, we concisely state the purpose of each data-set within the context of this thesis. Then we proceed with describing the individual data-sets.

- **Pubfig** For development purposes and medium-scale experimental evaluations.
- **FaceScrub** For medium to large-scale training.
- **ChokePoint** For realistic test scenarios with surveillance angle and temporally ordered data.
- **Labeled Faces in the Wild (LFW)** For the formal evaluation and to enable comparison with other research efforts.
- **MSRA-CFW** To enable large-scale training even when we filter away subjects that overlap any of the data-sets we use for evaluation.

2.1 Pubfig

The Pubfig data-set ([Kumar et al., 2009](#)) contains web-images on celebrities and is provided as a list of links and checksums (checksums are used to verify if two files are identical) of the annotated images. It is partitioned in two identity disjoint sets: *eval* and *dev*, with 140 and 60 database subjects respectively.

In order to retrieve the images we created a couple of download scripts verifying the checksums and checking that the JPEG files are not corrupt. However, a substantial amount of the links were broken or returned an image not

identical to the annotated one according to the checksums. The provided face crop annotations then used a set of padded face images with approximately centered faces and padding relative to face width. The cropped images were further processed with face alignment Section 3.1.

This resulted in a downloaded data-set of wide face crops 15,617 and 5,578 images for the *eval* and *dev* partitions and for each crop type.

2.2 FaceScrub

FaceScrub database ([Ng and Winkler, 2014](#)) contain images of 530 actors and actresses retrieved from the web and is approximately balanced with respect to gender. It is provided with annotations in almost the same format as Pubfig, so we could use almost identical retrieval scripts to request and process each of the 107,818 image URLs. The slightly more than 15GB worth of annotated JPEGs were processed in the same manner as Pubfig and resulted in the final image count of 77,323. Here as well, some of the image links have decayed.

2.3 ChokePoint

The ChokePoint data-set ([Wong et al., 2011](#)) was selected to model realistic video-based application scenarios with recordings made from surveillance view. The ChokePoint videos were captured in two sessions with 25 and 29 subjects, one month apart and each at a different portal. At a portal, the setup has 3 cameras mounted in either direction (centered, ~ 30 and ~ 45 degrees) and there are 4 sequences of recorded video in either direction with the subjects sequentially walking through the scene.

Since the preprocessed, cropped images are only provided in gray-scale we retrieved the approximately 12GB of raw JPEG frames and performed cropping ourselves. Cropped face images were created from originals with a face detector using a minimum resolution of 40×40 and by first creating regions-of-interest (ROI) guided by the XML-encoded eye coordinate ground truths of the data-set and then running the face detector. This strategy gave few mistakes and but introduced small miss-alignments coming from the face detector. For each video sequence the images were grouped by the subject labels also provided in the annotations (this removes the need for face tracking before performing realistic image-set based face recognition).

2.4 Labeled Faces in the Wild

Labeled Faces in The Wild (LFW) (Huang et al., 2007) has become something of a standard benchmark for image-based face verification in uncontrolled settings. It contains web-images of 5,749 celebrities, where only about half of the subjects have more than one image.

The benchmark consists of independent 10-folds with 600 pairs each, 300 matching and 300 non-matching. The testing follows 10-fold cross-validation scheme, where 10 test cases is created by selecting 1 fold as test set and leave the other 9 folds for training. The reported results are the average verification accuracy and its standard deviation and the average Receiver Operating Characteristic (ROC) curve.

LFW has a couple of evaluation protocols depending on if unsupervised or supervised learning is allowed, if only the binary match-no match labels are allowed or if the provided identities is allowed to be used to create more pairs and if labeled or non-labeled outside data are allowed. In this project the relevant protocol is *Unrestricted, Labeled Outside Data*, since we use labeled data from other data-sets to train the face verification algorithms. It also contains a category for the estimated Human Performance on face verification.

The data-set is provided as widely padded face crops that we use directly as input to the face aligner.

2.5 MSRA-CFW

MSRA-CFW (Zhang et al., 2012) is a large data-set with 1,583 identities and it is provided as 202,792 image URLs. Instead of checksums a set of backup links to identical images has been provided. This is a crude solution, as there is no way to validate if the image downloaded is identical to the image that was annotated (except maybe for the provided thumbnails).

The here used retrieval strategy for this database is therefore to run a face detector on the downloaded JPEG images and accept the noise introduced by assuming the both the face detection and the person identity is correct.

Because of the limited time we only expected to be able to download parts of this data-set, subjects were downloaded ordered by decreasing number of samples and the number of samples per subject were limited to 400.

2.6 Data Augmentation

Data augmentation is a procedure to generate extra, synthetic training data from a data-set by applying different transformations. The reason to do this is to obtain more densely sampled underlying class distributions and thereby achieve better classification.

Noise transformations, such as Gaussian and Salt and Pepper noise, can give increased robustness by decreasing an algorithms dependency on individual pixel values.

Color augmentations, like in (Krizhevsky et al., 2012) by adding to each image pixel multiples of the principal components of color sample data and with magnitudes proportional to the eigenvalues multiplied with a Gaussian distributed scale factor. This has been shown to increase performance in generic object detection. It can be used to increase the performance in gray-scale based algorithms too (Chatfield et al., 2014), applying it first to color images and then convert to gray-scale.

Inspired by the *Dropout* (Hinton et al., 2012) used in training of neural networks, preventing co-adaption of neurons by randomly inhibiting neuron activations. We do this on the input images in a localized manner, placing a rectangle of zeros with the size of a typical occlusion at randomized locations. The idea here is to reduce the co-adaption of features from local

parts of a face, for example if one of the eyes, the mouth or the nose is occluded. We call this procedure *Blackout*.

Foveation, inspired by the Fovea in the retina, is a technique of creating multi-resolution images where content is sharp concentrated around a focus point while the peripheral parts are blurred proportional to the distance from to the focus point. Foveation has been used as data augmentation in pixel-classifiers for segmenting neuronal membranes in electron microscopy images (Ciresan et al., 2012). We use foveation in a slightly different manner similar to the sub-crops with randomized offsets used in (Krizhevsky et al., 2012), randomly generating the focus point location as the sum of the center point and a random offset drawn from a Gaussian distribution. The aim is to reduce co-adaption of localized high-frequency content in the images.

We use noise, color augmentation, blackout, foveation, aligned and un-aligned images in the following text whenever we say that data augmentation is applied.

2.7 Recording of Video Data

As part of the developed Closed-Area demo application we recorded video sequences from a typical application scenario. A single entry and a single exit in office corridor environment. Both cameras are ceiling mounted and approximately centered with the aim to capture frontal-faces from surveillance view. The illumination conditions are slightly different as one of the camera faces large windows while the other is directed inwards. Figure 3.10a shows an overview of the demo application including the camera arrangements.

2.8 Comments

Several face verification algorithms are reported to perform significantly better if the training data have both large *width*, number of examples per subjects, and *depth*, number of unique identities (Chen et al., 2012) (Chen et al., 2013). For this reason neither ChokePoint or LFW are not well suited for training. Therefore, during the development process we have primarily used a balanced version of the Pubfig data-set with 40 uniformly sampled images per subject, dis-

carding subjects with too few images and partitioned into a training set of 140 subjects and a test set with 60 identities disjoint to the training set.

2.9 Evaluations

We have performed the following evaluations leading from face verification algorithm and progressively towards the implemented closed area demo application:

1. **Verification Test** Evaluate and analyze the verification performance on test databases.
2. **Open-Set Identification Test** Evaluate performance of verification based setups implementing open-set identification on test databases. This is highly relevant because the application Section 1.1.1 can be modeled with one (3) or multiple instances (1,2) of open-set face identification.
3. **Simulation of Application with Recorded Video Clips** Simulate application scenarios with recorded video clips, with one or multiple cameras.
4. **Physical Demo Setup** Implement as a demo application and run with live video streams from cameras.

Chapter 3

Methodology

3.1 Face Alignment

Face alignment has shown to improve face recognition performance, for example in (Huang et al., 2012). The goal is to reduce variation in the data that is not discriminative when it comes to identities, such as variation due to head pose and the scaling and translation effects introduced by many face detectors. As part of this project we investigate face alignment based on 2D point correspondences fitting wither a similarity transform or an affine transform.

3.1.1 Background

2D point correspondences can be used to estimate transforms between the image subject to alignment and a rectified reference image. This was the approach taken by (Chen et al., 2012) who use an affine transformation estimated from five landmarks and (Sun et al., 2014b) who fits a similarity transform based on three landmark points, both achieves great results for face verification.

More sophisticated approaches include face alignment based on active shapes. For example (Cao et al., 2013) who claims real-time performance and was used in the face verification method of (Chen et al., 2013).

State of the art methods, such as (Taigman et al., 2014), exploits the 2D-3D correspondences between 67 detected landmarks of a 2D face and corresponding points on a reference 3D shape. A 3D-2D camera is then fitted with generalized least squares using the point correspondences. The points of the 2D image are connected into triangles with Delaunay triangulation and this triangulation is used to direct a piece-wise affine transformation onto the 3D face. Finally, less

visible triangles as seen from the fitted camera are filled in with color values from their symmetrical counterparts and a frontal-face view is rendered. The authors dub their rigorous alignment process *frontalization*. An even more recent but similar idea is presented by (Hassner et al., 2014). Generally, methods incorporating 3D models should be able to compensate for out-of-plane rotations better.

Other approaches make use of neural networks to estimate identity-preserving transformations (Zhu et al., 2013). Recent results (Razavian et al., 2014a) suggests that the first hidden layer of a generic convolutional neural network can be used as features for linear regression localizing 2D-landmark points.

3.1.2 Facial Landmark Detection

An important step in alignment methods based on point correspondences is being able to detect stable point features in the image category subject to the alignment. In face recognition, it is fairly common to use points located at *facial landmarks* such as the nose tip and the corners of the eyes and the mouth. To find such points we need a *facial landmark detector*.

Flanmark

One way to create such a detector is to maximize a score function based on the summation of local appearance fit and deformation cost for each landmark of a suggested configuration. This can be achieved with a Deformable Parts Model (DPM), as in (Uříčář et al., 2012), where the location of each facial landmark is connected via a directed acyclic graph to the fixed face center. A penalty is then introduced on graph deformations away from the initial positions. The local appearance around

the landmarks is described with Local Binary Pattern (LBP) features and used to discriminatively score the appearance fit. The parameters for the appearance fit and deformation cost functions are learned from annotated examples using a structured Support Vector Machine (SVM) (taking structured output labels not just scalars). By modeling landmark positions on a directed acyclic graph the score function can be formulated as a max-sum problem and maximized with dynamic programming. The publicly available implementation is efficient and is what we have consistently used when detecting facial landmarks throughout this project.

3.1.3 Fitting Transforms with Point Correspondences

A reasonable approach to face alignment is to use 2D point correspondences to fit a transform between the detected landmarks in the image to be aligned and a set of reference points in a rectified frame. The implementation of such an approach, fitting either a *similarity transform* or an *affine transformation*, is described in the following sections.

Reference Points

In our case, statistics on detected landmarks position were collected from a data-set automatically aligned with commercial techniques (Huang et al., 2012). The process involved the usage of a face detector in order to express landmark coordinates relative to the face detection bounding boxes. The average positions of the landmarks was used as reference points in all of the experiments described in the following subsections. The landmark statistics are visualized with the 2D-histogram in Figure 3.2a, the reference points are located at the close to the tops.

Direct Linear Transformation

Direct Linear Transformation is a popular algorithm in computer vision and can be used to estimate a *homography* from point correspondences. A *homography* is a transformation H , a 3×3 matrix, mapping point correspondences expressed with homogeneous coordinates as follows

$$\hat{\mathbf{x}}_H = H\mathbf{x}_H. \quad (3.1)$$

In a homography representing a full projective transform we have 9 unknowns and as described in (Solem, 2012) and (Hartley and Zisserman, 2004) we state the problem

$$\begin{pmatrix} \hat{x}_i \\ \hat{y}_i \\ \vdots \end{pmatrix} = M \begin{pmatrix} h_{1,1} \\ h_{1,2} \\ h_{1,3} \\ h_{2,1} \\ h_{2,2} \\ h_{2,3} \\ h_{3,1} \\ h_{3,2} \\ h_{3,3} \end{pmatrix},$$

$$M = \begin{pmatrix} -x_i & -y_i & -1 & 0 & 0 & 0 & x_i\hat{x}_i & y_i\hat{x}_i & \hat{x}_i \\ 0 & 0 & 0 & -x_i & -y_i & -1 & x_i\hat{y}_i & y_i\hat{y}_i & \hat{y}_i \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{pmatrix},$$

$$i \in [1, n], \quad (3.2)$$

this formulation is called *Direct Linear Transformation* and can be solved with least squares. Practically, the least squares solution is often obtained with *Singular Value Decomposition* (SVD).

Next we describe how we used the DLT algorithm to fit *similarity transforms* and *affine transforms*.

Similarity Transform

Expressed here in homogeneous coordinates a similarity transform maps 2D points according to

$$\hat{\mathbf{x}}_H = \begin{pmatrix} sR & t \\ 0 & 1 \end{pmatrix} \mathbf{x}_H \quad (3.3)$$

and performs rotation R , uniform scaling s and translation t .

As in (Solem, 2012) we can write a similarity transform for a single point correspondence as

$$\hat{\mathbf{x}} = \begin{pmatrix} a & -b \\ b & a \end{pmatrix} \mathbf{x} + \begin{pmatrix} t_x \\ t_y \end{pmatrix}, \quad (3.4)$$

with

$$sR = \begin{pmatrix} a & -b \\ b & a \end{pmatrix}, s = \sqrt{a^2 + b^2}.$$

With this parameterization in unknowns

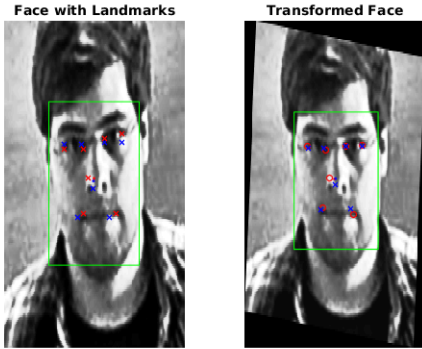


Figure 3.1: The input face image (left) with detected facial landmark points (red) and reference points (green) is aligned with a similarity transform estimated from point correspondences. The transformed image (right) shows that the transformed landmarks points (red) are close to the reference points and the bounding box (green) contains a registered face image.

a, b, t_x and t_y the DLT equations simplify to

$$\begin{pmatrix} \hat{x}_i \\ \hat{y}_i \\ \vdots \end{pmatrix} = \begin{pmatrix} x_i & -y_i & 1 & 0 \\ x_i & y_i & 0 & 1 \\ \vdots & \vdots & \vdots & \vdots \end{pmatrix} \begin{pmatrix} a \\ b \\ t_x \\ t_y \end{pmatrix}, \quad i \in [1, n] \quad (3.5)$$

and we can solve it as mentioned in Section 3.1.3.

In our implementation, we detect 8 landmark points for each face with Flandmark (Uříčář et al., 2012), consequently we have 8 points correspondences at our disposal. We create an overdetermined equation system on same form as Equation 3.5 and solve it using least squares. The solution minimizes the error $\|\hat{\mathbf{x}} - T\mathbf{x}\|_2^2$ where T is the estimated transform.

Affine Transform

An *affine transformation* allows for more deformation than a *similarity transform*. According to

$$\hat{\mathbf{x}}_H = \begin{pmatrix} A & t \\ 0 & 1 \end{pmatrix} \mathbf{x}_H \quad (3.6)$$

it performs an affine mapping A (including scaling, rotation, squeeze and shear mapping) and a translation t while preserves parallel lines.

An affine transformation simplifies the DLT algorithm 3.1.3 as it comes with the constraints $h_{3,1} = h_{3,2} = 0$ and $h_{3,3} = 1$ on the homography. We solve the simplified system

$$\begin{pmatrix} \hat{x}_i \\ \hat{y}_i \\ \vdots \end{pmatrix} = \begin{pmatrix} -x_i & -y_i & -1 & 0 & 0 & 0 & \hat{x}_i \\ 0 & 0 & 0 & -x_i & -y_i & -1 & \hat{y}_i \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{pmatrix} \begin{pmatrix} h_{1,1} \\ h_{1,2} \\ h_{1,3} \\ h_{2,1} \\ h_{2,2} \\ h_{2,3} \\ h_{3,3} \end{pmatrix}, \quad i \in [1, n] \quad (3.7)$$

and then normalize H by dividing with $h_{3,3}$.

Weighted Least Squares

As mentioned in 3.1.3, we extracted statistics of landmark points. Investigation of those led to the conclusion that not all detected landmark points are equally stationary. For example the position of the mouth corners move around much more more than the eye positions 3.2a. This led to the idea to trust landmarks inversely proportional to their associated uncertainty (i.e. variance) while fitting the affine transformations.

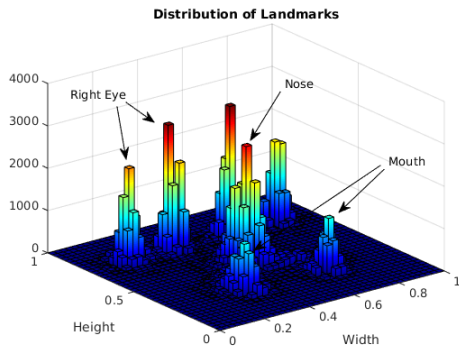
This serves as motivation for experiments with *weighted least squares* with the reciprocal variance of the landmark points as weights.

Eigen-Analysis of Facial Landmarks - Generalized Least Squares

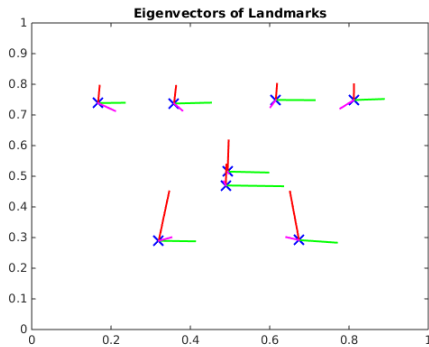
The idea of 3.1.3 can be taken a step further by also considering the covariance between the landmark points in the configuration. The covariance matrix can be estimated from the landmark statistics and we can have a look at its eigenvectors and eigenvalues.

As seen, in figure 3.2b, from the two largest eigenvectors (red and green) with respect to each landmark point, there is a lot of translation both vertically and horizontally. This is possibly due to miss-alignments by the face detector. The third largest eigenvector (magenta) indicate motion due to scaling. We can also observe that the magnitudes of the eigenvalues are larger for the mouth and nose points compared to the other landmarks.

With the estimated covariance matrix we tried fitting affine transforms with *generalized*



(a) Distribution of detected landmarks positions for a machine-aligned data-set.



(b) Eigenvectors of the estimated landmark position covariance matrix scaled in proportion to eigenvalue magnitude.

Figure 3.2: Analysis of facial landmark data.

least squares. We noted that the landmark points were transformed to configurations with much more dependency, e.g. the mouth corners were transformed symmetrically with respect to each other. However, the approach often finds to bad transforms. Perhaps, this is due to the automatic nature of the landmark statistics leading to accumulated noise from first the face detector and then the landmark detector or that the uncertainty is not close to identical across the observations. It could be interesting to see if this works better with manually annotated landmark data.

3.2 Face Tracks from Video

From a single video stream it is possible to create sequences of face images that with high confidence belong to the same identity by exploiting the spatio-temporal relations between consecutive frames. These sequences, called *face tracks*, can then be used for identity-matching between separate video sources.

To achieve this, we have implemented a detection-based face tracker. The main motivation for using a face tracker is to reduce the computational complexity of the identity-matching problem by considering face tracks in contrast to directly matching the detections of individual frames.

Some researchers, such as (Fischer et al., 2011), stress the advantage of using a face tracker in conjunction with a face detector, as this enables capturing a wider range of head poses compared to the frontal-faces given by most face detectors. All the same, we chose to implement a detection-based tracking solution, similar to (Everingham et al., 2009), since it does not introduce any new false detections in addition to those caused by the face detector. Other reasons include that the main focus of this project is face recognition constrained to frontal-faces and that most publicly available data-sets of reasonable size are restricted to frontal-faces.

Several research efforts have used face tracker approaches, notably: real-time person identification (Apostoloff and Zisserman, 2007), person re-identification in TV-series (Fischer et al., 2011), automatic naming of TV-series characters (Everingham et al., 2009), human action localization in video (Kläser et al., 2012)

and semi-supervised creation of labeled datasets (Ramanan et al., 2007). This suggests that it is a fairly common setup.

This chapter starts off with giving a technological background for optical flow and point-features tracking, before going into the details of our tracker implementation.

3.2.1 Optical Flow

Optical flow refers to the field of velocity vectors describing the relative motion between a camera and the scene it captures (Wikimedia Foundation, 2015c). For video imaging with a static camera the optical flow can be estimated from two or more consecutive video frames. Under the assumption that the optical flow is constant in a pixel's local spatial neighborhood this can be done with differential methods based on Taylor series expansion. One such method is the *Lucas-Kanade method* (Lucas and Kanade, 1981), which is briefly described below.

Estimating Optical Flow: Lucas-Kanade Method

Let $I(\mathbf{r})$ be the intensity of a single video pixel located at $\mathbf{r} = (x, y, t)^T$ and $I(\mathbf{r} + \mathbf{d})$ be the intensity of the same image content but displaced by a vector $\mathbf{d} = (\Delta x, \Delta y, \Delta t)^T$ to a nearby spatial position in another frame at the temporal distance Δt .

The main idea in optical flow estimation, as understood from (Wikimedia Foundation, 2015b), is that the pixel intensity remains the same (or changes minimally) for the same image content when assuming small displacement d . This is called the *brightness constancy constraint* and can be formulated as

$$I(\mathbf{r}) = I(\mathbf{r} + \mathbf{d}) \approx I(\mathbf{r}) + \nabla_r I(\mathbf{r}) \cdot \mathbf{d}, \quad (3.8)$$

where the last equation is obtained by applying linear Taylor series approximation, i.e. taking a step \mathbf{d} in the direction of the gradient with respect to \mathbf{r} . This in turn leads to

$$\nabla_r I(\mathbf{r}) \cdot \mathbf{d} = 0 \quad (3.9)$$

and dividing with Δt we get

$$\nabla_r I(\mathbf{r}) \cdot (V_x, V_y, 1)^T = I_x V_x + I_y V_y + I_t = 0, \quad (3.10)$$

where I_x, I_y and I_t are partial derivatives of the image intensity with respect to pixel r and $V_x = \frac{\Delta x}{\Delta t}$ and $V_y = \frac{\Delta y}{\Delta t}$ are the components of the velocity vector we want to estimate for each pixel.

Assuming that the optical flow is approximately constant locally, each neighborhood pixel $\mathbf{n}_k, k \in [1, n]$ (including the center pixel) should satisfy the *brightness constancy constraint* and we obtain one equation 3.10 per neighborhood pixel. This results in an overdetermined equation system

$$Av = b, \quad (3.11)$$

where

$$A = \begin{pmatrix} \vdots & \vdots \\ I_x(\mathbf{n}_k) & I_y(\mathbf{n}_k) \\ \vdots & \vdots \end{pmatrix},$$

$$v = \begin{pmatrix} V_x \\ V_y \end{pmatrix}, b = \begin{pmatrix} \vdots \\ -I_t(\mathbf{n}_k) \\ \vdots \end{pmatrix},$$

with partial derivatives I_x, I_y and I_t with respect to pixel r .

Multiplying the system with A^T we end up with the *normal equations*

$$A^T A v = A^T b \quad (3.12)$$

and iff the *structure tensor* $A^T A$ is invertible we can find the least-squares solution, which minimize the residual $\|Av - b\|_2^2$, via the *Moore-Penrose pseudo-inverse*

$$v = (A^T A)^{-1} A^T b. \quad (3.13)$$

For $A^T A$ to be invertible positive eigenvalues λ_1 and λ_2 are required, from numerical perspective preferably with some margin.

Moreover, the weighted least squares solution $v = (A^T W A)^{-1} A^T W b$ can be used to assign more weight to the closest neighbors, e.g. solving for a neighborhood weighted by a Gaussian window.

3.2.2 Feature Tracking of Point Features

Tomasi and Kanade (Tomasi and Kanade, 1991) describes a method for point feature

tracking on top of the Lucas-Kanade estimated optical flow 3.2.1 and investigate what point features are best suited for tracking. The argument about feature selection involves the two eigenvalues of the structure tensor $A^T A$. If $\lambda_1 \gg \lambda_2$ the pixel is located on an edge the pixel position is not well described by its spatial neighborhood since there is large uncertainty along the direction of the edge tangent. On the other hand, if $\lambda_1 \sim \lambda_2 \gg 0$ the pixel is a corner point or a local extremum and the pixel position is well defined by its neighborhood.

The conclusion from (Tomasi and Kanade, 1991) is that corner points are better suited to be tracked with search directed by the trajectories of the estimated optical flow. The point feature detector they present, which was later termed *Good Features to Track* (Tomasi and Shi, 1994), is very similar to the *Harris Corner Detector* (Harris and Stephens, 1988). In both methods points are detected if certain conditions on the structure tensor eigenvalues are satisfied. In the case of *Good Features to Track* the condition is

$$\min_{k \in \{1,2\}} \lambda_k \geq \tau > 0 \quad (3.14)$$

for a predefined threshold τ , while the condition for the *Harris Corner Detector* is

$$\det(A) - \kappa \text{Tr}(A)^2 > \tau \quad (3.15)$$

for a sensitivity parameter κ . Being formulated in $\det(A)$ and $\text{Tr}(A)$ there is no need to explicitly compute the eigenvalues of A .

Combining such a point feature detector with the Lucas-Kanade method (Lucas and Kanade, 1981) to track point features with search directed by optical flow results in something generally called a *Kanade-Lucas-Tomasi tracker* (KLT-tracker) (Wikimedia Foundation, 2015a) and the points being tracked are often called *interest points*.

More robust versions of KLT-tracking use a scale-space pyramid implementation, as in (Bouguet, 2000), initializing the search in lower resolutions and then continuing in the increasingly larger resolutions. This is a trick to be able to track longer pixel-wise distances than the small displacement assumption otherwise allows.

3.2.3 Dropping Feature Points Dynamically

Shi-Tomasi (Tomasi and Shi, 1994) introduced a way to discard inefficient interest points. The first step includes fitting an affine transformation between the neighborhood of the tracked point and the location suggested by the KLT-tracker in a non-consecutive frame. Then, if the re-projected neighborhood is too dissimilar from the original one the interest point is discarded. This is a way to maintain a good set of point features, features with a chance of being valid between more than just consecutive frames.

3.2.4 Face Tracks from Video Streams

With the background given above, this section describes the implemented solution for how to group face detections from a single video stream by identity. These temporally ordered sequences of face images assumed to have the same identity are called *face tracks*.

Inspiration: Naming Characters in TV-Series

Our solution is inspired by (Everingham et al., 2009) but differs in both implementation and the decision process. Everingham (Everingham et al., 2009) implemented a KLT-based face tracking scheme as a module in a system for naming characters in TV-series by connecting the names in the subtitles with detected *face tracks* of the named character. After they learned these associations offline face recognition was applied to a TV show episode (*Buffy the Vampire Slayer* to be precise) to name known characters and register timestamps of appearances.

To create their face tracks they track interest points forward and backwards through whole scenes of the TV episode and count interest point trajectories that intersect each of the detection bounding boxes. From this they formulate a pair-wise confidence that detections from two separate frames are connected.

Let F_A and F_B be face detections in two separate frames and A and B be the sets of interest point trajectories that intersect the respective bounding boxes. Then the confidence is defined as

$$g(A, B) = \frac{|A \cap B|}{|A \cup B \setminus A \cap B|} \quad (3.16)$$

that is the number of point-tracks the detections have in common divided by the number of disjoint ones.

Connecting Face Detections to Face Tracks

In **Algorithm 1** implementing a point feature-based face tracking on detections, interest points are tracked forward and backward between the current and a previous *detection-frame*. Just as in (Everingham et al., 2009) detections in the same frame are assumed to be of disjoint identities. For efficiency, only two frames at the time are considered: the current frame with a newly detected face and the latest frame of an *open track* candidating to accumulate the detection.

Let F_A and F_B be face detections in current and the previous frame respectively. Between 30 and 50 *Good Feature to Track*-interest points are detected for each face image, sets A and B . Interest points are then tracked both forwards and backwards with the pyramid implementation of Lucas-Kanade optical flow tracking described in (Bouguet, 2000) which is part of the well-known OpenCV library. This results in two sets A_{inlier} and B_{inlier} of tracked interest points ending up inside the respective bounding boxes and that are not dropped according to the *Shi-Tomasi verification* (see Section 3.2.3). We consider these two sets as *inliers* and since we know the number of interest points with which we initialized the searches, $|A|$ and $|B|$, we compute the *forward* and *backward inlier ratios* defined as

$$r_{fw} = \frac{|A_{inlier}|}{|A|}, r_{bw} = \frac{|B_{inlier}|}{|B|}. \quad (3.17)$$

Finally, a comfortable confidence measure is derived by the *harmonic mean* of the two inlier ratios

$$f(r_{fw}, r_{bw}) = \frac{2r_{fw}r_{bw}}{r_{fw} + r_{bw}}. \quad (3.18)$$

This means that when both forward and backward inlier ratios are high the confidence is high, close to 1, and it is sufficient that one is small to significantly lower the confidence.

The algorithmic core of our detection-based tracking is concisely summarized in Algorithm

1.

Data: F_A, F_B face detections from nearby frames.

Data: A, B detected interest points.

Result: Confidence $f(r_{fw}, r_{bw})$ of F_A, F_B belonging to same identity.

1. Track A backward.
2. Track B forward.
3. $B_{inlier} \leftarrow$ non-dropped forward-tracked points intersecting F_A .
4. $A_{inlier} \leftarrow$ non-dropped backward-tracked points intersecting F_B .
5. Compute inlier-ratios
 $r_{fw} \leftarrow \frac{|A_{inlier}|}{|A|}, r_{bw} \leftarrow \frac{|B_{inlier}|}{|B|}$.
6. $f(r_{fw}, r_{bw}) \leftarrow \frac{2r_{fw}r_{bw}}{r_{fw} + r_{bw} + \epsilon}$.

Algorithm 1: Confidence measure for inferring identity information on face detections in nearby frames from a single video source.

Samples from a captured face track is shown in Figure 3.6a.

Managing Face Tracks

Another problem considered in the tracker is how to manage face tracks: when new tracks should be created, when *open tracks* can be closed and if a closed track is worth keeping.

From Section 3.2.4 we know how to register detections, i.e. connect new detections to any of the open tracks. If the confidence Equation 3.18 is below a set threshold then it is stipulated that no good candidate has been found and the algorithm starts to consider opening a new track. Another condition for a track to be opened is that the *maximum recent overlap* between the suggested track and any open track is small. In our case this means less than 25% overlap. This is primarily to avoid corrupted interest points for the new detection, corrupted in the sense that the interest points are associated with another identity. With recent we mean that the overlap occurred within a few past seconds.

Face tracks are simply closed when a maximum time of inactivity has passed. Hence the

tracks keep the absolute time of when it accepted its last member.

As a way to reduce incorrect face tracks caused by sporadic face detector mistakes, a track is required to have a certain minimum number of detections. This increase the quality of the face tracks ensuring a minimum length of the sequences.

To further increase the robustness to false detections done by the face detector, such as large detections holding a smaller correct open track inside we introduced a *minimum scale change condition* on the bounding boxes for new detections to be accumulated by an open track.

3.3 Face Representations based on Local Features

3.3.1 Spatial Histograms of Local Binary Pattern

Spatial histograms of Local Binary Pattern (LBP) is a widely used local image feature. It has found applications in many face recognition tasks, for example face identification, face verification, facial expression recognition and gender classification.

Local Binary Pattern

Local Binary Pattern is a local operator that was originally designed for texture analysis (Ojala et al., 2002). The center pixel value is thresholded against the samples from its local neighborhood. Each thresholded sample account for a single bit in the bit-sequence describing and characterizing the neighborhood. The bit-sequence constitute a code, for a neighborhood of eight an 8-bit integer in $[0, 255]$, that is called an LBP-code or an LBP-pattern.

Similarly to (Pietikäinen et al., 2011), the extended LBP-operator can be defined for neighborhood of size P (number of sample points) and radius R as

$$\text{LBP}_{P,R}(\mathbf{x}_c) = \sum_{p=0}^{P-1} \theta(I(\mathbf{x}_p) - I(\mathbf{x}_c))2^p \quad (3.19)$$

where θ is the unit step function and $I(\mathbf{x})$ the gray-level intensity at point \mathbf{x} .

The idea introduced by (Ahonen et al., 2004) is to consider a face as a composition of micro-patterns. They create features in Bag-of-Visual-Word style, by computing histograms of

LBP-patterns from image regions. Since a single histogram does not encapsulate any spatial information, the face images are divided into a set of grid regions and histograms of the LBP-patterns are computed locally for each region. The histograms are then concatenated into a single global feature vector, i.e. describing a whole face. Using feature vectors as described above (Ahonen et al., 2004) applied a *nearest neighbor* classifier using χ^2 -distance to implement face identification.

LBP Versions

The *original* version of the LBP-operator simply uses the 8 neighboring pixel values as neighborhood, thus it computes very fast.

In the *extended* version neighbor samples are instead uniformly distributed on a circle around the center pixel and the intensity values are approximated with bi-linear interpolation using the intensities of the 4 nearest pixels. This operator is parameterized by the circle radius and the number of neighbor samples.

Another improvement to the LBP-operator is the *Uniform*-LBP, introduced by (Ojala et al., 2002) and described in (Pietikäinen et al., 2011). If assuming that flat-surfaces, edges and corners characterize a texture well, a descriptor based on LBP-patterns with at most 2 binary transitions would be enough. Patterns with more transition would correspond to high-frequency, noisy image features. Actually, as stated in (Ahonen et al., 2004), it was shown in experiments with texture images that uniform patterns account for about 90 % of all patterns when using a (8,1) neighborhood and for around 70 % in case of a (16,2) neighborhood. The Uniform LBP-operator is often denoted $\text{LBP}_{P,R}^{u2}$, where $u2$ means uniform with at most 2 binary transitions.

For an 8-neighborhood, a Uniform LBP-histogram is created by mapping the 256 LBP-patterns to 58 uniform LBP-bins and a single non-uniform bin.

Properties of the LBP-Operator

A convenient property of the LBP-operator is that the LBP-codes are invariant to monotonic gray-scale transformations. This is due to its definition and that the relative intensities stays the same after a monotonic gray-scale transformation is applied. This for example reduces

the benefit from pre-processing steps such as histogram normalization and explains the LBP-operator’s robustness to illuminations changes in general. The spatial histograms strategy introduce a certain amount of robustness to translation and rotation.

Related Works: LBP for Face Verification

The main inspiration to investigate LBP-based features’ plausibility for real-time face verification is motivated by (Chen et al., 2013) and their face verification algorithm with LBP-histogram features extracted from grid regions around 27 dense facial landmark points in multiple image scales and then concatenated to a single high-dimensional descriptor. With a high-dimensional feature of 100,000 dimensions they report good results on the LFW-benchmark. The dimensionality is reduced with Principal Component Analysis (see Section 3.5.1) before applying the Joint-Bayesian classifier (Chen et al., 2012). As a further improvement they make a sparse approximation, *Rotated Sparse Regression*, of the dimensionality reduction projection matrix in order to save space and computation. They show that the sparse approximation performs well compared with other feature selection methods such as backward greedy and structured sparsity. The question we want to answer is whether a similar but somewhat reduced approach could give sufficient accuracy while being light-weight enough for use in real-time video applications.

In (Chen et al., 2012) a hierarchical classification approach is taken, by using a linear Support Vector Machine to create an ensemble of individual Joint Bayesian classifiers trained on separate types of LBP-features (and other local features).

An alternative to use subspace methods is to use boosting in order to do feature selection, as is done in (Wang et al., 2009) with their boosted multi-task learning framework. They train subject specific verifiers in a one-vs-all fashion on a high-dimensional pool of LBP-features of different types and use early termination to only compute the LBP-codes that were selected by their boosted multi-task framework. It is not clear, however, how to apply this method to the pair-wise face verification formulation. But

we note the possible efficiency improvements of early termination in the computation of LBP-codes and that multi-scale features can be encoded by using LBP-operators of different radii.

3.3.2 Histogram of Oriented Gradients

Histogram of Oriented Gradients (HOG) is a local feature type that has been successful, for example in pedestrian detection (Dalal and Triggs, 2005).

HOG features are usually computed by scanning a block region with a specified step size over a gamma and color normalized input image, usually with overlapping blocks. Each block typically contains 4 cells for each of which a histogram of gradients is computed by weighted voting. The gradient directions are often described with 9 (or 18 if mirrored) bins. A contrast normalization scheme is applied to the blocks and finally block features are concatenated to a single feature vector describing the image content.

The authors of (Sivic et al., 2009) used a HOG feature with 9×9 grid of overlapping blocks and 6 bins as features in subject-specific Multiple-Kernel-Learning Support Vector Machines classifiers to discriminate face tracks appearing in TV-series episodes.

3.3.3 Framework for Extracting Local Features for Face Verification

Inspired by (Chen et al., 2013) we present a feature extraction framework for extracting local features around landmark-centered image regions. The question we want to answer is whether a similar but somewhat condensed approach could give sufficient accuracy while being light-weight enough for use in real-time video applications.

Landmark-Centered Feature Extraction

Facial landmarks are detected with the Flandmark library, see Section 3.1.2, and are first used for the face alignment. Square image regions are then extracted from the aligned and histogram equalized face image around 7 landmark points Figure 3.3 that have been transformed into the aligned frame. The size of the extracted regions is specified relative to the width of the aligned face image or with fixed size in pixels. In case parts of the regions are

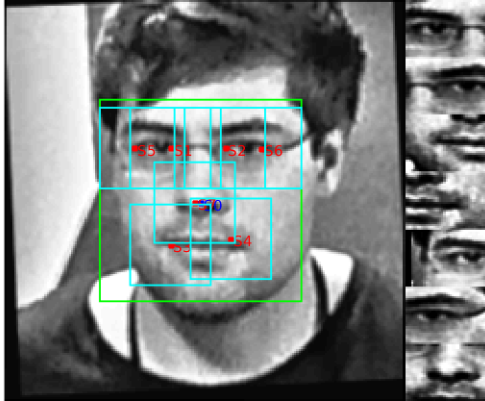


Figure 3.3: Feature regions (cyan) located around 7 facial landmarks (red) outlined by their respective bounding boxes.

located outside the bounding box of the aligned face we apply padding.

The infrastructure allows for local features to be extracted from these image regions. For example this allows us to extract Uniform LBP histogram features from a 4×4 grid with or without grid-cell overlap (inspired by the HOG feature we give the option to do this for LBP as well) and with or without histogram equalization applied to the local image region. This results in 7 image region descriptors, in the example with dimension 944 for a single $\text{LBP}_{8,R}^{u,2}$ operator, and we can create a face descriptor either by using the local descriptors individually or to concatenate them into a single holistic descriptor. The concatenated feature would be of dimension 6608 in the example case.

A concatenated representation enforces spatial constraints and might be more discriminative. We hypothesize that using the local descriptors to train independent classifiers is possibly more robust to occlusions and changes in facial expressions.

Multiple Scales

To capture information on multiple scales we have implemented two strategies. The first strategy is to extract features from at multiple resolutions by creating image pyramids for the local image regions. The second strategy is specific to the LBP features and relies on the use of multiple LBP-operators with different radii.

If we for instance use 2 scales we end up with

14 local descriptors and in addition to the options mentioned before when it comes to representation we can choose to concatenate features on a per scale basis and create independent descriptors for each scale.

When using 2 scales for LBP features with 8-neighborhood and 4×4 grids around the 14 landmarks locations, then the feature dimension becomes 13216. This is a fairly modest dimension in comparison with the 100,000 dimensional raw feature of (Chen et al., 2013) and is because we want to investigate the performance characteristics of a condensed model but also because of limited computational resources and memory bottlenecks (Singular Value Decompositions of large matrices) in the computational framework.

Dimensionality Reduction

The framework relies on subspace methods for feature selection and compression of the relatively high-dimensional descriptors. Subspace projections are done with Principal Component Analysis Section 3.5.2 or Linear Discriminant Analysis Section 3.5.2 and a normalization scheme similar to (Azizpour et al., 2014).

Holistic Grid-Region Feature Extraction

For reference the framework can also extract features in holistic grid regions directly from the aligned face image. This allows for implementations of feature extraction in the style of (Ahonen et al., 2004) or (Sivic et al., 2009).

3.4 Convolutional Neural Network-based Deep Face Representations

3.4.1 Motivation

A recent development in visual recognition is to use deep representations extracted from *Convolutional Neural Networks* (CNN) as generic object descriptors (Sermanet et al., 2014) (Razavian et al., 2014b) (Chatfield et al., 2014). In the following text we use the term *convnet* when referring to Convolutional Neural Networks.

The convnets used for visual recognition purposes are often trained on huge data-sets for generic object recognition, for example the subset of ImageNet data-set used in the *ImageNet Large-Scale Visual Recognition Challenge*

(ILSVRC)-competition with roughly 1.2 million training images spanning 1000 categories (Krizhevsky et al., 2012).

In this project we have investigated if the generic object recognition task can be transferred to face recognition tasks. Our approach is inspired by (Karayev et al., 2013) who transfer-learned a generic convnet to recognize image styles according to Flickr-tags. In particular, we fine-tune convnets with architecture similar to the AlexNet (Krizhevsky et al., 2012), and use the activations of the last hidden layer as face descriptors. We used the pre-trained weights of the *BVLC-Reference-Caffenet* convnet that is available via the *Caffe*-framework (Jia et al., 2014).

The idea is that the filters learned by such convnets are of a generic nature. That the filters, as a result of the versatile training set with a vast number of object categories, capture generic concepts (high and mid-level features composed of lower-level features such as edges and textures) and consequently are able to characterize other types of objects, in our case faces.

From a data perspective it makes sense to transfer learn from a task where a lot of labeled data is available and then fine-tune to a more specific task using a limited amount of labeled data and as a result also less computational resources.

The training and feature extraction process are more closely described in Section 3.4.4 while the following sections give some technical background.

3.4.2 Background: Convolutional Neural Networks

The biologically inspired *Convolutional Neural Network* is an efficient supervised statistical machine learning method. It organizes and learns a feature extraction hierarchy that is well suited for visual recognition tasks. As a special type of *Feed-Forward Neural Network* it can be trained with the *Back-Propagation Algorithm*.

In accordance with one of the definitions of *Deep Learning* given in (Deng and Yu, 2014):

Machine learning that attempt to learn in multiple levels, corresponding to different levels of abstraction.

It typically uses artificial neural networks. The levels in these learned statistical models correspond to distinct levels of concepts, where higher-level concepts are defined from lower-level ones, and the same lower-level concepts can help to define many higher-level concepts.

CNNs are considered to be deep learning methods.

A CNN, such as the classic *LeNet5* (Lecun et al., 1998) used for handwritten character recognition, is roughly described a hierarchical convolutional feature extraction with a Multi-Layer Perceptron (MLP) classifier stacked on top.

The feature extraction part of the *LeNet5* CNN consist of alternating convolutional and sub-sampling (pooling) layers, where each layer extracts features from the previous one and in that way describe increasingly more global concepts. The core idea is to model *local receptive fields* as linear filters, where each receptive field responds to a small region of the input image. By applying convolution (evaluating the filter at different locations of the input) the filter weights can be shared throughout the image. This effectively reduces the number of parameters representing the network and allows for shared computations. The input is usually convolved with several filters and each convolution results in a *feature map*, which can be seen as an *activation* in response to the *visual stimuli* presented by the input.

More recent CNN architectures, such as AlexNet (Krizhevsky et al., 2012), use *Rectified Linear Units* (ReLU) as activation functions rather than sigmoidal functions and include layers for local contrast normalization *Local Response Normalization* (LRN) and *Dropout*, i.e. completely inhibiting neurons with a specified probability.

The classification part can contain several *fully connected layers* of *neurons*. For recognition tasks the final layer, the *output layer*, is often a *Softmax* (i.e. *Logistic Regression*) layer that has as many *neurons* as class labels and where the neurons represent the probabilities of each class respectively. The intermediate neuron layers are often called *hidden layers* and

their neurons *hidden units*. The first hidden layer is connected to the feature maps of the last convolutional (or pooling) layer.

3.4.3 Related Work

Deep representations seem to gain success now in an ever-growing list of tasks, including: speech recognition, natural language modeling, traffic sign recognition and face verification. For a handful of tasks the algorithms are competitive to (or even surpass) the human performance. In the case of *face verification* evaluated on the popular *Labeled Faces in the Wild* (LFW) benchmark (Huang et al., 2007), several top performing contributions have recently surpassed the estimated human performance. These include among others the convnet-based approaches (Zhou et al., 2015) (Sun et al., 2014a) (Sun et al., 2015) but also the Gaussian process-based (Lu and Tang, 2014).

When it comes to large-scale generic visual object recognition, the success story of the convnets seem to really have taken off when the generic object recognition competition *ImageNet Large-Scale Visual Recognition Challenge 2012* (ILSVRC-12) was won by the AlexNet convnet (Krizhevsky et al., 2012). In this work we experiment with network architectures that are all related to the AlexNet.

The availability of high-performing GPU-accelerated frameworks for feature extraction based on convnets, such as *OverFeat* (Sermanet et al., 2014) (the winner of ImageNet ILSVRC-13) and *Caffe* (Jia et al., 2014), has spawned a lot of research. For example (Razavian et al., 2014b), who investigate if the generic descriptors *OverFeat* in combination with linear Support Vector Machines (SVM) are suitable for a variety of more specific visual recognition tasks. They conclude:

Deep learning with CNN has to be considered as the primary candidate in essentially any visual recognition task.

A thorough comparative study is presented in (Chatfield et al., 2014), comparing the state-of-the-art among local feature representations and with the deep representations produced by CNNs pre-trained for generic object detection,

with or without fine-tuning. They evaluate performance on several large object recognition benchmark data-sets while using the same learning algorithm (linear SVM) for both feature types. They study convnet architectures of three different sizes, all implemented with Caffe and with designs related to OverFeat and AlexNet (Krizhevsky et al., 2012). They show that limiting the convnets to use grayscale intensity image input only gives a moderate accuracy penalty, a drop of about 3%. Similarly, they show that representations from convnet architectures incorporating non-linear dimensionality reduction can give compact representations without giving too much negative impact on the performance. They conclude that deep representations outperforms local features (SIFT features encoded in Improved Fisher Vectors (IFV)) by a large-margin and with the extra convenience of more compact representations.

Research from the last couple of months, (Azizpour et al., 2014), investigate how deep representations transfer from generic visual tasks to more specific. In the process the authors achieve state-of-the-art performances on 16 visual recognition tasks.

In the face verification algorithm *DeepID* (Sun et al., 2014b) (which is a top performer on the LFW benchmark) a powerful discriminative face descriptor is learned by training multiple convnets, each for a certain patch of the input image, on hard face identification problems including 10000 identities. The learning process has later been refined to supervise with both identification and verification signals (Sun et al., 2014a) (Sun et al., 2014c) and recently to also include supervision in the early stages of the convnets (Sun et al., 2015), this improved the results substantially. *DeepID*, together with two other convnet approaches to face verification *DeepFace* (Taigman et al., 2014) and *Face++* (Zhou et al., 2015), serves as motivation to perform fine-tuning for a face identification problem even though the goal is to produce a face descriptor for face verification.

3.4.4 Fine-Tuning for Face Representations

The authors of (Sun et al., 2014b) (Sun et al., 2014b) emphasize the large number of identities (10000 subjects) as a key to the successful performance of their face descriptor. In the case of this project however, it is not feasible to train a large convnet from scratch as we only have small and medium-sized face recognition datasets at our disposal and because the time and computational resources of this project are limited.

Instead we choose to fine-tune a pre-trained convnet, initialized with the weights of *BVLC-Reference-Caffenet* (Jia et al., 2014) which was trained for generic object recognition with 1000 image categories, for face identification and we restrict the feature extraction model to only include a single convnet.

The *fine-tuning* process optimizes the convnet weights for classification using the new data-set. For obvious reasons the layers initialized with pre-trained weights need to have the same dimensions as in the original network architecture. Conversely, the *output layer* dimensionality need to be adjusted to match the new number of object categories. This is fundamental as the Softmax-Loss is computed from the output layer activations and the labels and it is the derivatives of the *Softmax-Loss* w.r.t. the parameters (weights) that governs the *Stochastic Gradient Descent* of the back-propagation algorithm. We adjust the learning rates and step size according to (Karayev et al., 2013).

Our fine-tuning setups are all combinations of the following modifications to the *BVLC-Reference-Caffenet* convnet:

- **Replace output layer** with a new fully connected layer with as many neurons as object classes in the training data-set. The neuron and bias unit weights are randomly initialized.
- **Adjust learning rates** to learn the new layers faster than the pre-trained layers. To achieve this the learning rate of the layers initialized with pre-training is set lower than for the new layers, so that the pre-training is not forgotten.
- **Decrease the step-size of optimization**

algorithms Since we initialize with pre-training the training process should be closer to a local minimum and it should be reasonable to decrease the step-size of the gradient-based optimization algorithms.

- **Update mean-subtraction** to subtract the mean of the new training data as opposed to the old.
- **Extend with dimensionality reduction layer** As a way to achieve more compact descriptors, a fully-connected layer with reduced number of hidden units can be inserted before the output layer.

Once the convnet is trained, our aim is to use it as a feature extractor for faces. This can be done by performing the forward pass through the convnet with a given input image and then extract the activations in the last hidden layer as face descriptor. The face descriptor is then used as feature in separate classification algorithms performing the actual face verification, the classification algorithms can be something as simple as nearest neighbor.

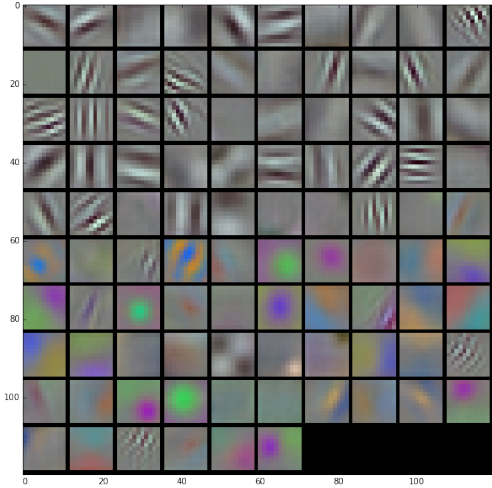
Figure 3.4 illustrate the learned filters 3.4a and filter activations 3.4b 3.4c 3.4d for a few selected layers from the resulting forward pass when a convnet is presented with a face image.

Architectures

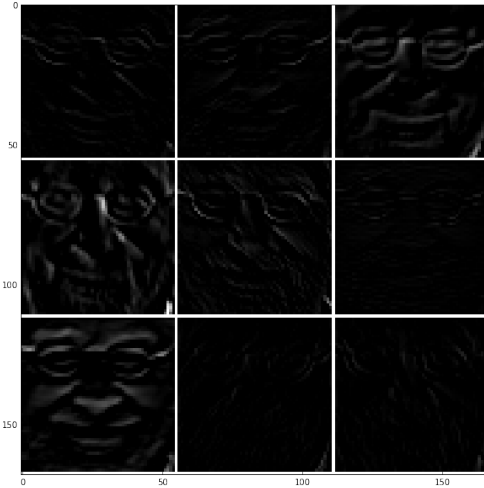
The three convnet architectures researched in this project is:

1. **Fine-tuned AlexNet** An AlexNet-architecture pre-trained on ImageNet and fine-tuned for faces. The last hidden layer activations of dimension 4096 is used as feature.
2. **Fine-tuned AlexNet-1024** An AlexNet-architecture fine-tuned for faces with an additional hidden layer for dimensionality reduction. The activations of this new layer is used as a 1024-dimensional feature.
3. **Fine-tuned AlexNet-128** As 2, but we try to create a really compact feature by reducing the dimension of the additional hidden layer to 128.

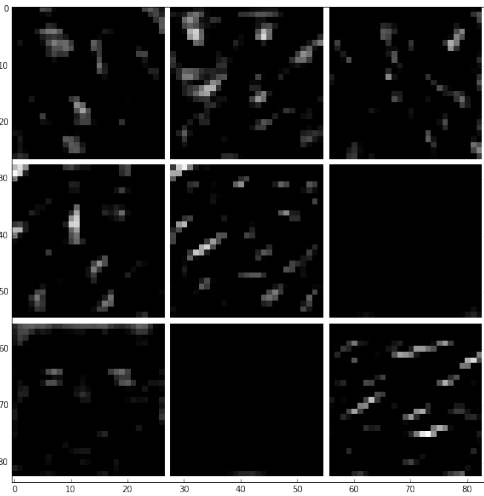
We fine-tune convnets with data both with and without face alignment pre-processing, so 6 architectures in total.



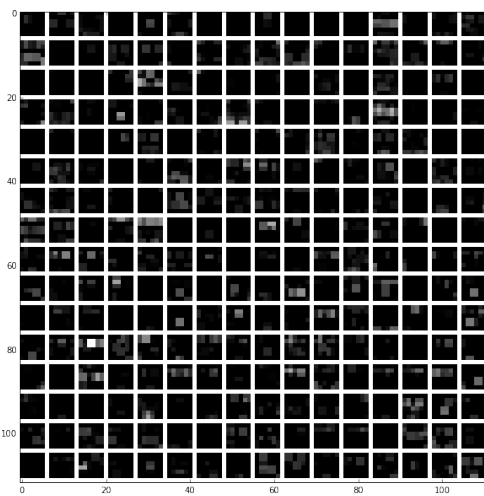
(a) The 96 filter kernels of the first convolutional layer. The learned filters are similar to Gabor filters and behaves similar to how the human visual receptors are believed to work.



(b) Responses of the 9 first filters in 3.4a when applied to a face image. The shown filter outputs capture mostly edge patterns with different frequency and orientation.



(c) The 9 first activations from the filters of the second convolutional layer. These filter responses contain more abstraction and corresponds to patterns in the earlier layer activations that are discriminative for identities.



(d) All 256 activations of the final convolutional layer after max-pooling. The activations are sparse and localized. For a local group of pixels the activations possibly signals how much of a specific trait is present in the input image.

Figure 3.4: We visualize what happens during a forward-pass of a convnet by showing learned filter weights and some example activations from a couple layers of a *Fine-tuned AlexNet-1024*-convnet (Section 3.4.4). The visual stimuli to generate the responses is a face image of Steven Spielberg in case it feels familiar.

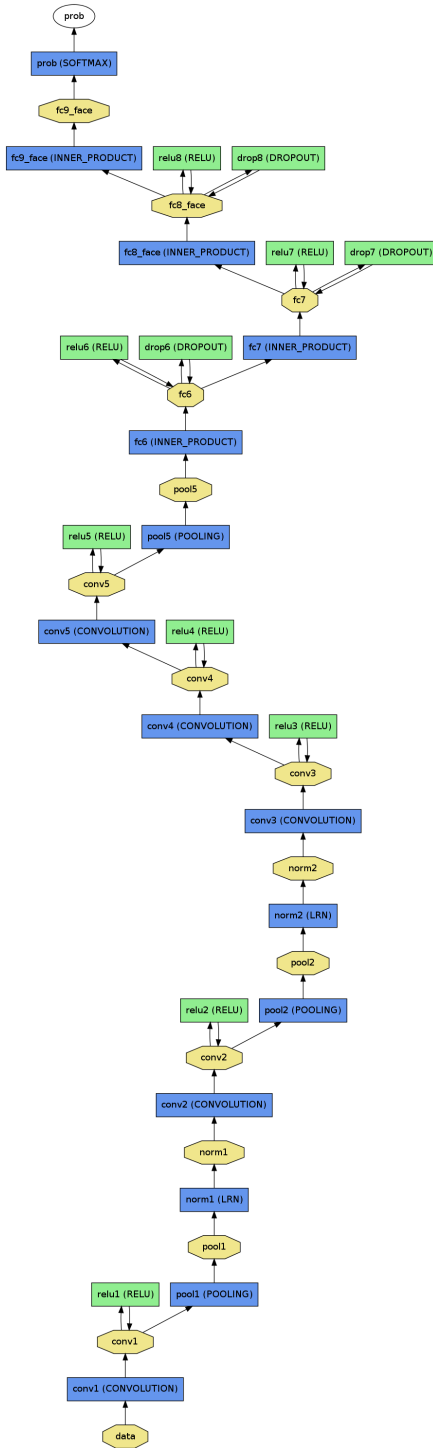


Figure 3.5: Convnet architecture 2, Fine-tuned AlexNet-1024.

3.5 Linear Dimensionality Reduction Techniques

There are several reasons why dimensionality reduction may be a good idea and why it is often included in visual recognition pipelines. One important reason is that more compact object descriptors reduce the computational complexity. This is vital in face verification, matching problems or content retrieval in general because it leads to faster processing.

Another desirable effect of low dimensionality is that the data storage and network bandwidth requirements are reduced. This is important when it comes to building a database of object descriptors, embedding object descriptors as meta-data or sending object descriptors over the network. For some learning algorithms it may be necessary to employ dimensionality reduction as pre-processing to reduce the algorithm's training time and memory footprint. In some cases lower dimensions fits learning algorithms better and is a way to counter the *curse of dimensionality*.

Dimensionality reduction techniques can be interpreted as a kind of feature selection, selecting dimensions according to some criteria, e.g. maximizing variance as in Principal Component Analysis or finding the most discriminative dimensions as in Linear Discriminant Analysis. Discarding less important dimensions can be interpreted as a way to reduce noise or remove redundant information.

3.5.1 Principal Component Analysis

Principal Component Analysis (PCA) is a unsupervised statistical analysis method that finds an orthogonal basis that maximize total variance. The directions in the data responsible for the most variance are called *principal components* and can be found by diagonalizing the covariance matrix of the data (Lindgren, 2006).

Given a $m \times n$ data matrix X we can estimate the covariance matrix with the sample covariance

$$\Sigma_X = \frac{1}{n-1} \sum_{i=1}^n (x_i - \bar{x})(x_i - \bar{x})^T, \quad (3.20)$$

where \bar{x} is the mean and then diagonalize Σ_X

$$\Sigma_X = P\Lambda P^T \Leftrightarrow \Lambda = P^T \Sigma_X P \quad (3.21)$$

to find the principal components P and the diagonal matrix Λ with the eigenvalues λ on the diagonal. A related way to find the directions P is to instead perform a Singular Value Decomposition (SVD) on the centered data matrix P , because the *singular values* are the square root of the eigenvalues of the covariance matrix. As the SVD computations became a major memory bottle-neck in our training framework, we alleviated this problem by replacing the default truncated SVD algorithm (Matlab’s svds) with *Limited Memory Block Krylov Subspace Optimization for Computing Dominant Singular Value Decompositions* (LMSVD) (Liu et al., 2013).

The principal component, or eigenvector, corresponding to the largest eigenvalue is the direction with maximum total variance. The second eigenvector is the direction of maximum variance in the vector space orthogonal to the first eigenvector and so on.

The principal components are a basis for the data, meaning the data points can be expressed as linear combinations of the principal components. Dimensionality reduction can be performed by truncating this basis, i.e. discarding the principal components contributing the least variance and by keeping only the k first. The coordinates in this reduced space are given by projecting the data points onto the k first principal components and these coordinates are a compact representation of the data points.

As in the classic *Eigenface* algorithm (Turk and Pentland., 1991) it is possible to perform face recognition by applying PCA directly to intensity images of faces. However, this works well only under very controlled circumstances as it is sensitive to occlusions, illumination and pose changes.

A major assumption in PCA is that all variance is useful. But if we for example apply PCA to intensity images of faces, non-discriminative factors such as large pose and illumination changes or occlusions may contribute the most variance and often the 2-3 first principal components can be identified as sharp directed illumination.

We apply the normalization scheme described by (Azizpour et al., 2014), normalizing data points to unit length according to l_2 -norm before subtracting the mean and re-normalizing

after the subspace projections and feature variance normalization. The details can be found in 2.

Data: X , n data points of dimension m .

Result: X' , n projected data points of dimension m' .

1. $x_j \leftarrow \frac{x_j}{\|x_j\|_2}, \forall x_j \in X = (x_1 \cdots x_n)$
2. $X \leftarrow X - E[X]$
3. $[U, S, V] \leftarrow \text{svds}(X, m')$
4. $X' \leftarrow U^T X$
5. *Whitening (feature variance normalization)*
 $x_i \leftarrow \frac{x_i}{\sigma_i}, \forall x_i \in X = (x_1; \cdots; x_{m'})$
6. *Re-normalization*
 $x'_j \leftarrow \frac{x_j}{\|x'_j\|_2}, \forall x'_j \in X = (x'_1 \cdots x'_n)$

Algorithm 2: PCA as done in this project.

3.5.2 Linear Discriminant Analysis

Linear Discriminant Analysis (LDA) is a supervised linear dimensionality reduction technique. Just like the PCA-based Eigenface algorithm it has been applied directly on intensity images to find a subspace suitable for face identification Fisherfaces (Belhumeur et al., 1997). In contrast to PCA it makes use of class labels to find the most discriminative directions. Intuitively, it finds the directions in the training data that separates the classes the most, instead of the directions that separates all training data the most like in PCA. The key idea is to minimize within-class variance and maximize the between-class variance.

Given a $m \times n$ data matrix X and classes $C_i, i \in [1, K]$, we compute the $m \times m$ total *within-class scatter* matrix by

$$S_W = \sum_{i=1}^K S_i = \sum_{i=1}^K \sum_{\{j: x_j \in C_i\}} (x_j - \bar{x}_i)(x_j - \bar{x}_i)^T, \quad (3.22)$$

i.e. centering the data in a per class manner by subtracting the individual class means \bar{x}_i and computing the scatter. This describes the movement of data points $x_k \in C_c$ away from

its class mean \bar{x}_c . A scatter matrix is a proportional to the covariance matrix.

The $m \times m$ *between-class scatter* matrix measures how the class means scatter around the centre of mass and is computed as

$$S_B = \sum_{i=1}^K n_i (\bar{x}_i - \bar{x})(\bar{x}_i - \bar{x})^T, \quad (3.23)$$

where n_i is the number of samples per class and where mean of the class means $\bar{x} = \frac{1}{n} \sum \bar{x}_i$ (Alpaydin, 2010) or the weighted mean of the class means $\bar{x} = \frac{1}{n} \sum n_i \bar{x}_i$ (Mian and Pears, 2012).

The aim is to find a subspace projection $\hat{x} = W^T x$ by maximizing *Fisher's Criterion*

$$J(W) = \max_W \frac{\det(W^T S_B W)}{\det(W^T S_W W)}, \quad (3.24)$$

i.e. maximizing between-class scatter and minimizing within-class scatter. The determinants are the products of the eigenvalues and can be interpreted as the square of the scattering volumes (Mian and Pears, 2012).

Differentiating w.r.t. W and setting to zero leads to the *Generalized Eigenvalue Problem*

$$S_W^{-1} S_B W - J W = 0 \quad (3.25)$$

where the eigenvectors of $S_W^{-1} S_B$ corresponding to the largest eigenvalues are the directions we want to project the data onto.

The number of direction that can be found with LDA is limited to $K - 1$. This is because of the rank of the S_B matrix that has been formed from K outer products. Also, S_W need to be non-singular and the initial feature dimension might be too large for matrix inversion. For these reasons it is common to perform a PCA before applying the LDA.

3.6 Classification

With the chosen pair-wise face verification formulation Section 1.2 and given a pair of extracted face descriptors x_1 and x_2 , one per face image, the purpose of the classifier is to give a confidence, or similarity measure if you like, how likely the face descriptors derive from the same identity. The classification is performed by thresholding the confidences to reach a binary decisions: match or non-match. Then

later, an appropriate threshold (w.r.t the target application) can be selected to make a reasonable trade-off between the two error types, false positives (non-matches classified as matches) and false negatives (matches classified as non-matches).

3.6.1 Nearest Neighbor

A minimalistic approach is to simply take the distance between the two feature vectors according to some metric, for example the l_2 -norm that gives the *Euclidean Distance*, as dissimilarity measure. We just negate it to get a similarity measure

$$d(x_1, x_2) = -\|x_1 - x_2\|_2. \quad (3.26)$$

The authors of (Zhou et al., 2015) claim that their simple norm-based verification scheme work as well as or better than more sophisticated classification schemes, such as the Joint Bayesian Classifier, when the amount of training data is sufficiently large. Their convnet-based, deep feature representation is learned from massive amount of training data, about 5 million face images. This makes the nearest neighbor classification interesting in its simplicity. It also serves as a baseline classifier and it is a good indicator if a specific feature space does a good job in separating the classes.

3.6.2 Joint Bayesian Classifier

A more sophisticated take on the classification is the Joint Bayesian Classifier (Chen et al., 2012). It is the classifier of choice in several state-of-the-art face verification algorithms (Chen et al., 2013) (Sun et al., 2014b) (Sun et al., 2014a) (Sun et al., 2014c) (Sun et al., 2015). Comparisons presented in (Sun et al., 2014b) shows the Joint Bayesian performs better than verification based on a Siamese Convolutional Neural Network.

Model

A face $x \in N(0, \Sigma_X)$, represented in some arbitrary feature space, is assumed to be a vector of stochastic variables drawn from a zero-mean multi-variate Gaussian distribution. A face is then modeled as a sum of two independent latent zero-mean multi-variate Gaussian variables

$$x = \mu + \epsilon, \quad (3.27)$$

where $\mu \in N(0, S_\mu)$ represents the person identity and $\epsilon \in N(0, S_\epsilon)$ the sample-specific variation within the subject distribution. Here S_μ and S_ϵ are two unknown covariance matrices which the algorithm eventually will estimate.

The idea is to find a probabilistic measure of the similarity of a pair of faces x_1 and x_2 by modeling the joint distribution $\{x_1, x_2\}$ and then formulate the log-likelihood ratio

$$r(x_1, x_2) = \log \frac{P(x_1, x_2|H_I)}{P(x_1, x_2|H_E)}, \quad (3.28)$$

for the conditional probabilities given by each of the two hypotheses H_I and H_E . This is a binary Bayesian decision problem with hypotheses H_I , *intra-personal*, that assumes that x_1 and x_2 are of the same identity and H_E , *extra-personal*, that stipulates x_1 and x_2 belong to different identities.

Taking advantage of the linear form of Equation 3.27 and the independency assumption between the hidden variables μ and ϵ the covariance of the joint distribution can be expressed as

$$\text{cov}(x_i, x_j) = \text{cov}(\mu_i, \mu_j) + \text{cov}(\epsilon_i, \epsilon_j), \quad (3.29)$$

$$i, j \in \{1, 2\}.$$

Assuming hypothesis H_I we see that the identity variables, μ_1 and μ_2 , are dependent while the in-class variation variables, ϵ_1 and ϵ_2 , are independent. This leads to a covariance matrix for the conditional distribution $P(x_1, x_2|H_I)$ on the form

$$\Sigma_{H_I} = \begin{pmatrix} S_\mu + S_\epsilon & S_\mu \\ S_\mu & S_\mu + S_\epsilon \end{pmatrix}. \quad (3.30)$$

If we instead assume hypothesis H_E the identity variables, μ_1 and μ_2 , are independent as well as μ_1 and μ_2 and the covariance matrix for $P(x_1, x_2|H_E)$ is on the following form

$$\Sigma_{H_E} = \begin{pmatrix} S_\mu + S_\epsilon & 0 \\ 0 & S_\mu + S_\epsilon \end{pmatrix}. \quad (3.31)$$

Log-likelihood Ratio Closed Form Expression

The log-likelihood ratio has a closed-form expression

$$r(x_1, x_2) = x_1^T A x_1 + x_2^T A x_2 - 2x_1^T G x_2, \quad (3.32)$$

where

$$A = (S_\mu + S_\epsilon)^{-1} - (F + G),$$

$$\begin{pmatrix} F + G & G \\ G & F + G \end{pmatrix} = \begin{pmatrix} S_\mu + S_\epsilon & S_\mu \\ S_\mu & S_\mu + S_\epsilon \end{pmatrix}^{-1}. \quad (3.33)$$

Proof. Consider the multi-variate Gaussian

$$P(x_1, x_2) = \frac{1}{\alpha} e^{-\frac{1}{2}(x-\bar{x})^T \Sigma^{-1}(x-\bar{x})}. \quad (3.34)$$

The log-probabilities in the similarity measure can be expressed in terms of the exponents and a constant term

$$r(x_1, x_2) = \log P(x_1, x_2|H_I) - \log P(x_1, x_2|H_E)$$

$$= -\frac{1}{2}(\mathbf{x} - \bar{\mathbf{x}})^T \Sigma_{H_I}^{-1}(\mathbf{x} - \bar{\mathbf{x}}) +$$

$$\frac{1}{2}(\mathbf{x} - \bar{\mathbf{x}})^T \Sigma_{H_E}^{-1}(\mathbf{x} - \bar{\mathbf{x}}) + C. \quad (3.35)$$

The linear scaling and the translation does not affect the relative ordering of the similarity measure and $\mathbf{x} = [x_1; x_2]$ is zero-mean, we can write

$$r(x_1, x_2) = -\mathbf{x}^T \Sigma_{H_I}^{-1} \mathbf{x} + \mathbf{x}^T \Sigma_{H_E}^{-1} \mathbf{x}$$

$$= -[x_1; x_2]^T \Sigma_{H_I}^{-1} [x_1; x_2] +$$

$$[x_1; x_2]^T \Sigma_{H_E}^{-1} [x_1; x_2]. \quad (3.36)$$

From the *Matrix Inversion Lemma* (Lindgren et al., 2013) for inversion of a block-partitioned matrix and because of the block-structure of Σ_{H_I} and Σ_{H_E} we conclude that their inverses are on the form

$$\Sigma_{H_I}^{-1} = \begin{pmatrix} \hat{A} & \hat{B} \\ \hat{B} & \hat{A} \end{pmatrix}, \Sigma_{H_E}^{-1} = \begin{pmatrix} \hat{C} & 0 \\ 0 & \hat{C} \end{pmatrix}. \quad (3.37)$$

This allows us to continue the computations and we end up with

$$r(x_1, x_2) = x_1^T (\hat{C} - \hat{A}) x_1 + x_2^T (\hat{C} - \hat{A}) x_2$$

$$- 2x_1^T \hat{B} x_2. \quad (3.38)$$

with a change of notations we get the equivalent

closed form Equation 3.32 where

$$\begin{aligned}
A &= \hat{C} - \hat{A} \\
&= (S_\mu + S_\epsilon)^{-1} - \\
&\quad ((S_\mu + S_\epsilon)^{-1} - S_\mu(S_\mu + S_\epsilon)^{-1}S_\mu)^{-1} \\
G &= \hat{B}.
\end{aligned} \tag{3.39}$$

By assuming that Σ_I^{-1} has the following form

$$\Sigma_I^{-1} = \begin{pmatrix} S_\mu + S_\epsilon & S_\mu \\ S_\mu & S_\mu + S_\epsilon \end{pmatrix}^{-1} = \begin{pmatrix} F + G & G \\ G & F + G \end{pmatrix} \tag{3.40}$$

we can express the equation system $\Sigma_I \Sigma_I^{-1} = I$. On the diagonals we have equations

$$(S_\mu + S_\epsilon)(F + G) + S_\mu G = I \tag{3.41}$$

and on the off-diagonals

$$(S_\mu + S_\epsilon)G + S_\mu F + S_\mu G = 0. \tag{3.42}$$

Subtracting Equation 3.42 from Equation 3.41 gives

$$S_\epsilon F = I \Leftrightarrow F = S_\epsilon^{-1}. \tag{3.43}$$

Substitute F in Equation 3.42 and solve for G

$$G = -(2S_\mu + S_\epsilon)^{-1}S_\mu S_\epsilon^{-1}. \tag{3.44}$$

Now we have shown that the closed form exists and how to calculate A and G expressed in the parameters S_μ and S_ϵ . \square

Learning

The two covariance matrices S_μ and S_ϵ can be learned from the training data with the *Expectation-Maximization* algorithm presented by (Chen et al., 2012). For the sake of completeness we describe it here, see Algorithm 3.

The *E-Phase* is performed per subject and computes the expectation of the hidden variables, $\mathbf{h} = [\mu, \epsilon_1; \dots; \epsilon_m]$,

$$E(\mathbf{h}|\mathbf{x}) = \Sigma_h \mathbf{P}^T \Sigma_x^{-1} \mathbf{x}, \tag{3.45}$$

given an observation of m data points, stacked into a single column $\mathbf{x} = [x_1; \dots; x_m]$, all from a single subject.

The distribution of the latent variables is assumed to be $\mathbf{h} \sim N(0, \Sigma_h)$ with $\Sigma_h = \text{diag}(S_\mu, S_\epsilon, \dots, S_\epsilon)$. Equation 3.27 leads to the

Data: X .

Result: Estimated S_μ and S_ϵ .

Initialization:

$$\begin{aligned}
&S_\mu, S_\epsilon \leftarrow \\
&\text{cov}(\text{randomData}(\mu_X, \sigma_X))
\end{aligned}$$

repeat

E-Phase:

foreach *Subject with m observations \mathbf{x} .*

do

$$E(\mathbf{h}|\mathbf{x}) \leftarrow \Sigma_h \mathbf{P}^T \Sigma_x^{-1} \mathbf{x}$$

end

M-Phase:

$$S_\mu \leftarrow \text{cov}(\mu)$$

$$S_\epsilon \leftarrow \text{cov}(\epsilon)$$

until *Iterations done;*

Algorithm 3: EM-algorithm to learn covariance matrices S_μ and S_ϵ for latent variables μ and ϵ from data X .

following relationship between hidden variables and observations

$$\begin{aligned}
\mathbf{x} &= \mathbf{P}\mathbf{h}, \\
P &= \begin{pmatrix} I & I & 0 & \dots & 0 \\ I & 0 & I & & 0 \\ \vdots & \vdots & & \ddots & \vdots \\ I & 0 & 0 & \dots & I \end{pmatrix}. \tag{3.46}
\end{aligned}$$

This relationship in turn gives the distribution of \mathbf{x}

$$\Sigma_x = \begin{pmatrix} S_\mu + S_\epsilon & S_\mu & \dots & S_\mu \\ S_\mu & S_\mu + S_\epsilon & \ddots & S_\mu \\ \vdots & \ddots & \ddots & \vdots \\ S_\mu & S_\mu & \dots & S_\mu + S_\epsilon \end{pmatrix}. \tag{3.47}$$

For an image feature of dimension d , Σ_x is a huge $md \times md$ matrix and the algorithm requires it to be inverted. Because of the block-structure the inverse can be computed efficiently in block-wise manner. In our implementation of this algorithm all the computations are performed block-wise, when multiplying with \mathbf{P}^T which is a $(m+1)d \times md$ matrix and when multiplying with Σ_h .

In the *M-Phase* the parameters S_μ and S_ϵ are

updated from the expectations of \mathbf{h}

$$\begin{aligned} S_\mu &= \text{cov}(\mu) \\ S_\epsilon &= \text{cov}(\epsilon). \end{aligned} \quad (3.48)$$

The algorithm iteratively produce better estimations of S_μ and S_ϵ by alternating the E and the M-phase. When the parameters are learnt we can express the A and G matrices in the parameters and form the similarity measure in Equation 3.32.

Efficiency

The similarity measure Equation 3.32 can be evaluated efficiently, as the authors of (Chen et al., 2012) claim, by noting that the first two terms, $x_1Ax_1 = c_1$ and $x_2Ax_2 = c_2$, are image specific constants. Furthermore, they show the matrix G is negative definite and can be factorized, $G = -U^TU$. This means that the last term in Equation 3.32, $-2x_1Gx_2$ can be expressed as $-2x_1Gx_2 = 2(Ux_1)^T(Ux_2) = 2y_1^Ty_2$, where $y_1 = Ux_1$ and $y_2 = Ux_2$ are image specific vectors. With these pre-computations in place the similarity measure resolves to

$$r(x_1, x_2) = c_1 + c_2 + 2y_1^Ty_2, \quad (3.49)$$

which is basically a scalar product.

3.6.3 Standard Classifiers

To enable comparisons and verify results we investigate the performance of three *off-the-self* classifiers available in Matlab 2014b.

We define the two pair-wise learning setups compatible with those classifiers (Equation 3.50 and Equation 3.51) by concatenating the individual feature vectors of a pair x_1 and x_2 into a single feature vector x_{pair} and creating binary match or no-match labels. The x_{pair} descriptors and corresponding labels $\{-1, 1\}$ from many pairs were then used for the learning.

In the first setup we simply stack the feature vectors of the pair

$$x_{pair} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \quad (3.50)$$

and in the second setup we instead stack the sum and the difference of the feature vectors

$$x_{pair} = \begin{pmatrix} x_1 + x_2 \\ x_1 - x_2 \end{pmatrix}. \quad (3.51)$$

We generated pairs from data by first balancing the training data to contain an equal number of examples per subject and then generate an equal number (5000 for each class) of matching and non-matching descriptor pairs by uniformly sampling from the balanced training data.

Random Forests

A *Random Forest* (Breiman, 2001) is an ensemble of *decision trees* with a learning process specially designed to ensure variance between the ensemble members. For each decision tree in the ensemble a subset of the training data is selected by random sampling with replacement, this is often called *bagging* or *bootstrapaggregation*. Furthermore, the learning algorithm also performs *feature bagging*, i.e. at each candidate split in the learning process a subset of the features is selected. This efficiently reduce correlation between the ensemble members.

We train Random Forests of regression trees (Matlab’s BaggingTree) to give a confidence for the face verification problem.

Boosted Regression Trees

Boosting is another powerful ensemble learning algorithm. In general, boosting iteratively selects the best weak classifier from a set of weak candidate classifiers and then weights and adds it to the ensemble. In each iteration the importance of miss-classified data points is increased. In that sense the selected weak classifier, is the weak classifier that best adjust for the errors made by the ensemble so far.

The boosting algorithm we use to build an ensemble of regression trees producing a confidence score for the face verification problem is *Least Squares Boosting* (LSBoost option in Matlab’s fitensemble). As described in the documentation it *fits a new learner to the difference between the observed response and the aggregated prediction of all learners grown previously* and the ensemble is trained to *minimize mean-squared error*.

Support Vector Machines

Support Vector Machines (SVM) is a learning algorithm that finds the hyper-plane best separating the classes in some kernel space. Radial Basis Function (RBF) is a powerful ker-

nel, non-linearly mapping the data to its kernel space. With the problem formulation described above we had no success in getting the more commonly used Linear kernel SVM to converge even when changing the underlying optimization methods from Sequential Minimal Optimization to Quadratic Programming.

With the same setup as for the ensemble methods described above, we trained a binary SVM-RBF classifier (Matlab’s svmtrain with a Gaussian RBF-kernel and RBF sigma parameters set to 5 or 10) and used the signed distance to the hyper-plane as confidence.

3.7 Best-Shot Selection

With a face tracker such as the one we describe in Section 3.2.4 we obtain a sequence of face images for each subject. However, not all of the images in the obtained sequences are of high quality. There are three main problems: low-resolution face images for long-distance face detections, motion-blur and out-of-focus effects. Some of these images are even hard for humans to identify, see Figure 3.6a. In such cases there are little hope the algorithm will produce more than a sophisticated guess.

This motivates us to implement a form of *Best-Shot Selection*. We implement *Low-Resolution Pruning* where low-resolution images are filtered out and discarded.

To address the other two issues we implemented a *Low-Frequency Pruning* scheme. The high-frequency energy is computed for each of the images in the sequence.

A first idea was to sum up the high-frequency parts of the squared magnitude of the *Fourier transformed* image and summing up the high frequency content. But a more light-weight approach is to apply a Sobel filter to get a rough estimate the high-frequency content by computing the Root Mean Square (RMS) norm of the *Sobel* channel. Images with high-frequency content below the mean high-frequency content of the sequence minus some multiple of the standard deviation of the high-frequency content of the sequence can then be discarded.

3.8 Open-Set Face Identification

Combining the feature extraction, as in Sections 3.3.3 and 3.4.4, with a binary classifier,



(a) Example images from a captured face track sequence. There are images that suffers from low-resolution, motion-blur and out-of-focus effects. Some images are so low quality that the glasses are not visible.



(b) Three captured images (left column) and their Sobel channels (right column). The top one is up-sampled from lower resolution, the second out of focus and the bottom sharp. For the sharp image there is high-frequency content showing as increased intensity in the Sobel channel.

Figure 3.6: (a) The varying quality of the images of a face track sequence motivates best-shot selection. (b) Sharp images are assumed to, in general, have more intensity in the Sobel channel.

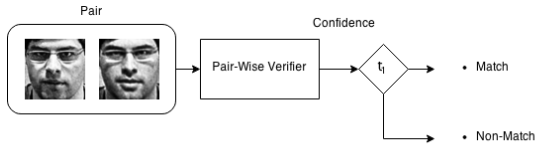


Figure 3.7: Pair-wise face verification.

see Section 3.6, we have a pair-wise face verification algorithm. A schematic is shown in Figure 3.7.

Typically, the feature extraction is incarnated in a still-image face recognition pipeline such as Figure 3.8. With the pair-wise face verification open-set face identification can be implemented and here we present two ways to do this based on pair-wise matching.

Figure 4.8 shows the first approach, where the face descriptor of the face to be recognized is matched in a pair-wise manner with all the face images belonging to the database with known identities. The face is recognized as the match-identity from the verification yielding the highest similarity score if the confidence is above a certain threshold otherwise it is decided that the identity is unknown.

Instead of only considering a single match-identity, the second approach Figure 3.9b consider the k match-identities with highest confidence and makes a majority vote. The scores of the winning identity is averaged and then thresholded to decide if the face belongs to the matched identity or an unknown identity.

3.9 Video-Based Face Matching: Closed Area Demo Application

To test the face verification algorithm’s applicability for the real-time video analysis applications described in Section 1.1.1 we have created a demo system keeping track of persons entering and leaving a closed area. For the scope of this thesis work we assume that the entry and the exit is unidirectional and that a face track can be created from only frontal-face face detections of each person. A schematic overview of the system setup is shown in Figure 3.10a.

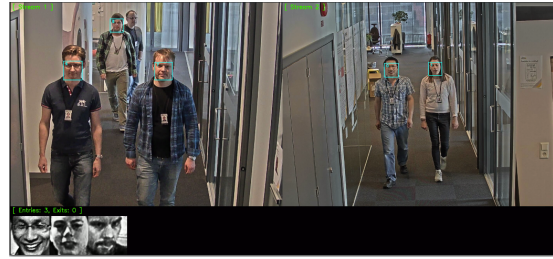


Figure 3.11: A screenshot of the closed area demo application. The bounding boxes (cyan) overlaid the two video streams indicating the currently open face tracks. The gallery (lower part) shows the dynamic database of persons currently inside the closed area.

System Description

The demo system is a multi-threaded program implemented with pthreads on-top of the OpenCV framework and, currently, runs in real-time on a single desktop computer. It analyses two video streams, one configured to be the entry to the closed area and the other setup to be the exit, and can stream video frames either live from two cameras or from recorded video files.

For each video frame requested by the application and as illustrated in Figure 3.10b, faces are detected with a previously trained face detector. From the face detections, facial landmarks are extracted and the face images are aligned according to similarity transform as presented in Section 3.1. Faces are then registered to a track managing module Section 3.2.4 which decides whether to open new tracks or append the face images to existing ones.

A track is closed if it remains inactive for a short time period. If the closed track is valid, meaning it has more than a certain number of registered face images, it is sent for feature extraction. This notion of valid or invalid face tracks is a way to temporally filter out sporadic false positives from mistakes made by the face detector.

For the demo application we use LBP-histogram features extracted around facial landmark points Section 3.3.3 that is reduced in dimension using LDA Section 3.5.2. The LBP-histogram features were selected because of its fast feature extraction and processed with

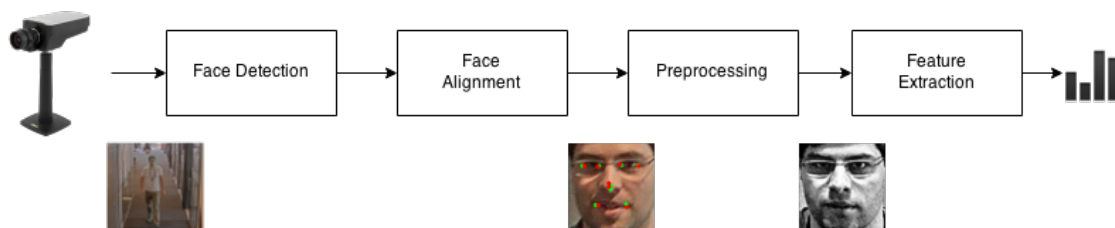
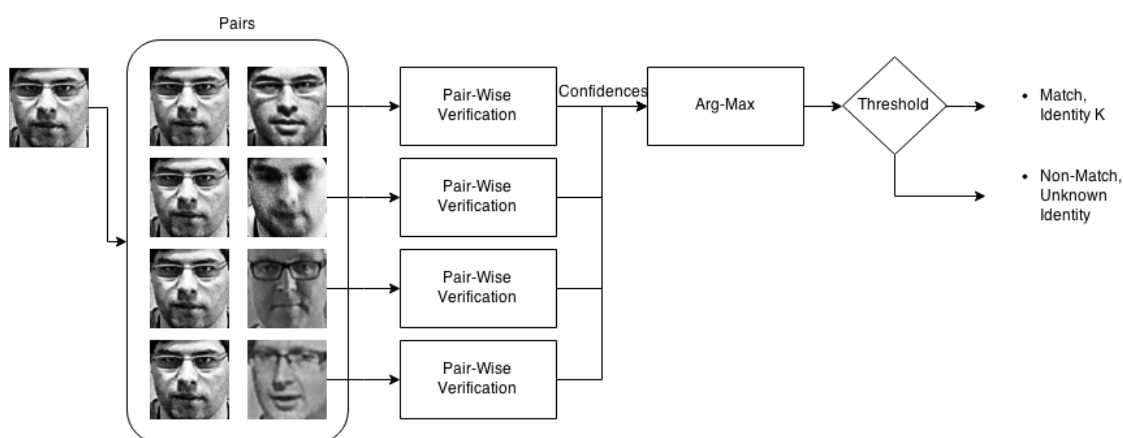
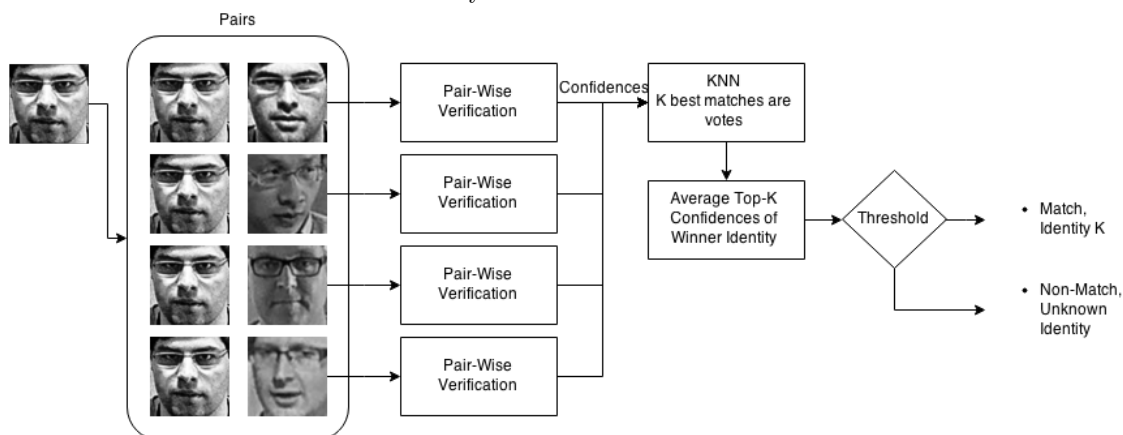


Figure 3.8: Still-image based face recognition feature extraction pipeline. A detected face is aligned and preprocessed before the features are extracted to create a face descriptor.

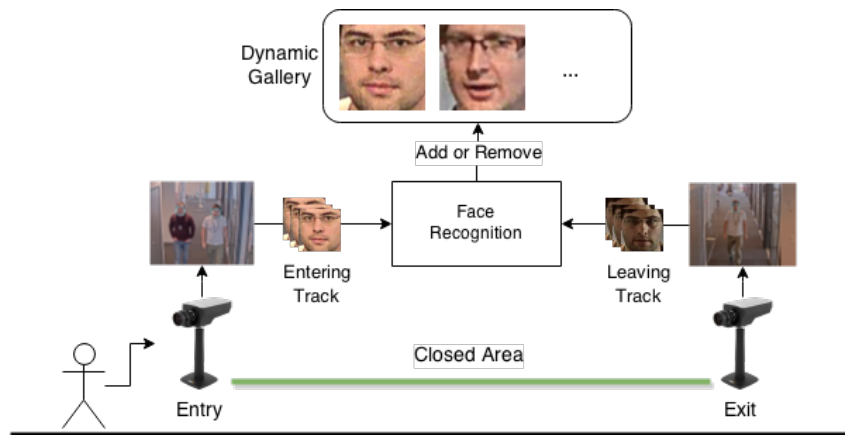


(a) Open-set face identification implemented with pair-wise face verification. The face descriptor of the face to be recognized (left column) is matched with the faces belonging to a database (right column). The match-identity from the verification yielding the highest similarity score is returned if the confidence is above some threshold otherwise the identity is unknown.

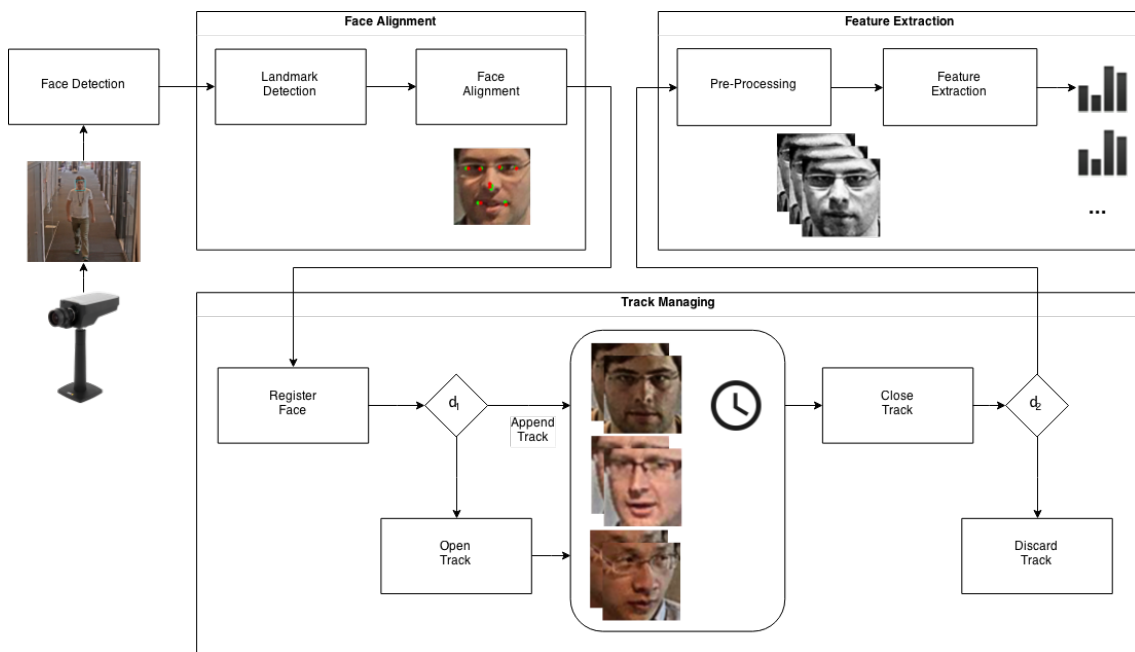


(b) Another way to implement open-set face identification with pair-wise face verification is to have a look at the k best match-identities, perform a majority vote and threshold the average score of the winning match-identity.

Figure 3.9: Two ways to implement open-set face identification with pair-wise face verification.



(a) Overview of the closed area application demo system setup. Persons are recorded while entering and leaving a closed area. Images from entry and exit is used to maintain information of who is currently inside the closed area, this is registered in the dynamic gallery.



(b) Schematic overview of the video-based face recognition pipeline used to implement the closed area demo. For each requested video frame, faces are detected and registered to the track managing. Track managing decides whether to open new tracks or append the face images to existing ones. A track is closed if it remains inactive for a short time period. If the closed track is valid, e.g. has enough images, it is sent to feature extraction. The track, now represented as a sequence of face descriptors, is compared to the dynamic database of persons currently inside the closed area and decisions are made whether to add or remove the track to the database or if nothing should be done.

Figure 3.10: Overview of the closed area demo application Figure 3.10a and the video-based face recognition pipeline we use to implement it Figure 3.10b.

LDA the relatively high-dimensional LBP features could be compressed to a 200-dimensional descriptors with reasonable accuracy.

The tracks are then represented as a sequence of face descriptors and is compared, using the Joint Bayesian pair-wise similarity described in Section 3.6.2, to the dynamic database of persons currently inside the closed area and decisions are made whether to add or remove the track to the database or if nothing should be done.

Chained Open-Set Face Identification Instances

The application can be viewed as two chained open-set face identification instances. At the entry, tracks are tested against the dynamic database and if the person is not recognized the person is added to the database. If the person is recognized we are left with uncertainty, either the algorithm has made a mistake, the person has a doppelgänger or the person has entered twice. This should probably trigger a log message but in the demo application we assume the person has already entered and we do nothing. At the exit there is a similar procedure, tracks are again tested against the database. If a person is recognized we can conclude that the recognized person is leaving the closed area and just remove them from the dynamic database. If the person on the other hand is not recognized we are left with a problem: algorithm mistake, doppelgänger or a person leaving twice.

From a design point of view we have until now modeled the problem as two chained open-set face identifications using the same threshold for both instances to decide if the face image is known or unknown. However, the closed area application gives additional structure: a person entering is probably not in the closed area database and a person exiting is more likely to be in the database. Individual thresholds can be set, such that the algorithm is biased to add persons to the database at the entry and biased to remove people from the database at the exit. This makes the closed area problem a little bit easier than applications such as *Multi-Camera Tracking* or more general *Person Re-Identification* where such structure is not available.

Chapter 4

Experimental Evaluation

4.1 Face Verification

4.1.1 Comparison of Features

In the early high-level prototyping we did feature evaluations to decide what features to continue to investigate. Some of the most interesting results are presented in Figure 4.1 in the form of *Receiver Operating Characteristics* (ROC) curves.

Training and evaluation is done on the relatively small-sized Pubfig data-set, training on Pubfig Eval and evaluating on Pubfig Dev. All the results are produced using the Joint Bayesian classifier.

These results are from early prototyping. Because of this the normalization scheme of (Azizpour et al., 2014) is not used, it was incorporated in the training algorithms at later time. Also, the region extraction, the Uniform LBP and HOG feature implementations are from the early stage high-level implementation. Those modules were later replaced by a low-level implementation. Therefore, rerunning these evaluations would most likely increase performance slightly for all feature types, so the reader is encouraged to consider the curves relative to each other. Figure 4.2 the results of the AlexNet-1024 features with a more recent implementation.

As can be seen in Figure 4.1 the AlexNet-1024 features learned from similarity transformed images (light blue) performs better than AlexNet-features learned from un-aligned images (green). On the other side of the spectrum, the single scale Uniform LBP features extracted from landmark regions without face alignment (blue) performs worse than with similarity transform alignment (orange). This im-

plicates that both convnet-features and local features benefit from the face alignment.

In order to make the local features more competitive with the convnet features it is necessary to use multiple scales. Using two scales we see that the HOG feature (yellow) performs slightly worse than the Uniform LBP (purple). We see that the Uniform LBP (purple) is the local feature (included in our experiments) that best competes with the convnet-features.

4.1.2 Comparison of Convnet Architectures

With the same data arrangements as in Section 4.1.1 we compare and evaluate the three different feature types extracted from the fine-tuned convnets, Section 3.4.4. We do this with and without the similarity transform alignment. The 6 different convnets that we compare were fine-tuned to learn features from the relatively modest Pubfig Eval data-set with its 140 subjects. Then, with each of the convnets we extracted features from the Pubfig Eval data-set and trained a Joint Bayesian classifier.

The results are presented in Figure 4.2. In contrast to the experiments done in the previous Section 4.1.1 these experiments is performed with an updated learning pipeline. We have improved feature normalization (Azizpour et al., 2014) and changing the PCA to use LMSVD (Liu et al., 2013)) enabled us to use more training data in the PCA. This explains the improved results in absolute terms compared to 4.1.

The Equal Error Rates (EER) in Table 4.2b suggests that the AlexNet-1024 architecture performs best, with its EER of 0.107. We also see that alignment generally improves the re-

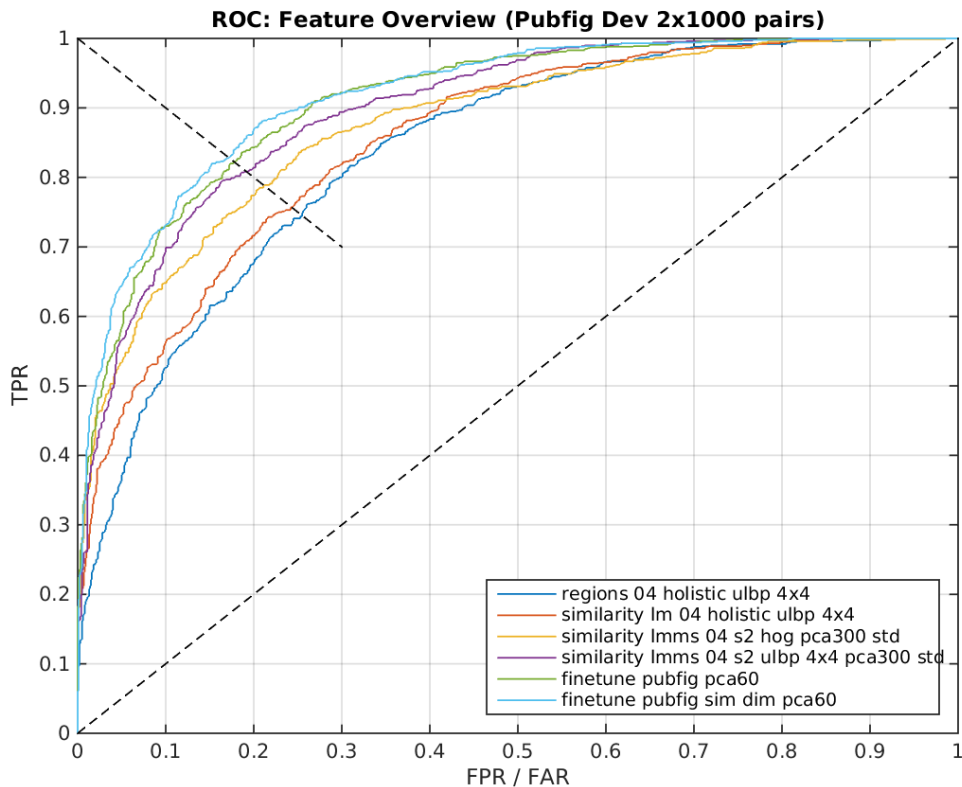


Figure 4.1: Here, an early prototyping feature evaluation is shown. Training and evaluation is done on the small-sized Pubfig data-set. The AlexNet-1024 features learned from similarity transformed images (light blue) performs slightly better than AlexNet-features learned from un-aligned images (green). On the other side of the spectrum we have the single scale Uniform LBP features extracted from landmark regions without face alignment (blue), with similarity transform alignment (orange) it performs slightly better. Results more competitive with the convnet features were obtained by using multiple scales and alignment for the landmark-based features. Using two scales we see that the HOG feature (yellow) performs slightly worse than the Uniform LBP (purple).

sults. Since we do not perform any averaging in these experiments (we perform averaging in the more formal evaluation on the LFW dataset in Section 4.1.5) it is a possibility that the 1000 matching and the 1000 non-matching pairs uniformly selected for that particular evaluation instance is not very representative for the entire data-set. For this reason it is important not to draw too large conclusions from these experiments, but they provide insights about the general picture. Instead of averaging we have lazily confirmed that re-running the experiments gives similar results.

The performance drop when aggressively reducing the feature dimension to 128 dimensions, using the AlexNet-128 convnet, but we note that the drop is not that large. Also, for all of the convnet features we note that we can get good performance with a compact descriptor, only 60-dimensions after performing PCA.

Based on these results the AlexNet-1024 architecture was selected for continued research with more time-demanding training on large-scale data-sets, Section 4.1.3.

4.1.3 Effects of Larger Training Set

In Figure 4.3 we show an illustrative example on the effect of increased data. Four face verification setups with AlexNet-1024 features and Joint Bayesian classifiers have been trained with increasing amount of data and then evaluated on 1000 matching and the 1000 non-matching pairs uniformly selected from Pubfig Dev. In these experiments we have incorporated a blacklist mechanism to prevent that any identity in the evaluation set is present in the training data.

The first two convnets were fine-tuned on 303 subjects from Facescrub data-set, not overlapping any subject in either Pubfig Dev or LFW and with more than 100 images per subject. This is approximately double the number of subjects compared with the training done in previous Section 4.1.2, but note that the number of images per subject is increased from 40 to 100. This resulted in 32362 images. In the first one the Joint Bayesian was only trained with data from the Pubfig Eval set and gives EER of 0.134. For the second setup we increased the training data for the Joint Bayesian to the 32362 images from Facescrub, this lowered the

EER to 0.124.

For the third setup we synthetically increased the amount of data to 325847 by first applying data augmentation and then selected 366 subjects with more than 400 synthetic examples for training. Both the convnet and the classifier used the increased amount of data. This gave an EER of 0.111.

The last convnet was fine-tuned with 394317 images of 478 subjects with more than 200 synthetic examples selected from the pool of augmented data from Facescrub and parts of MSRA-CFW. While the training images for the classifier remained 325847. The EER becomes 0.093.

Both the convnet feature extraction and the Joint Bayesian classifier benefit from increased amount of training data.

Assuming that the features compared in Section 4.1.1 scales in a similar manner, the performance for a certain amount of data with the AlexNet-1024 features, shown in Figure 4.3, can probably be used as an upper limit estimate of the performance of those other features.

4.1.4 Classification Algorithms

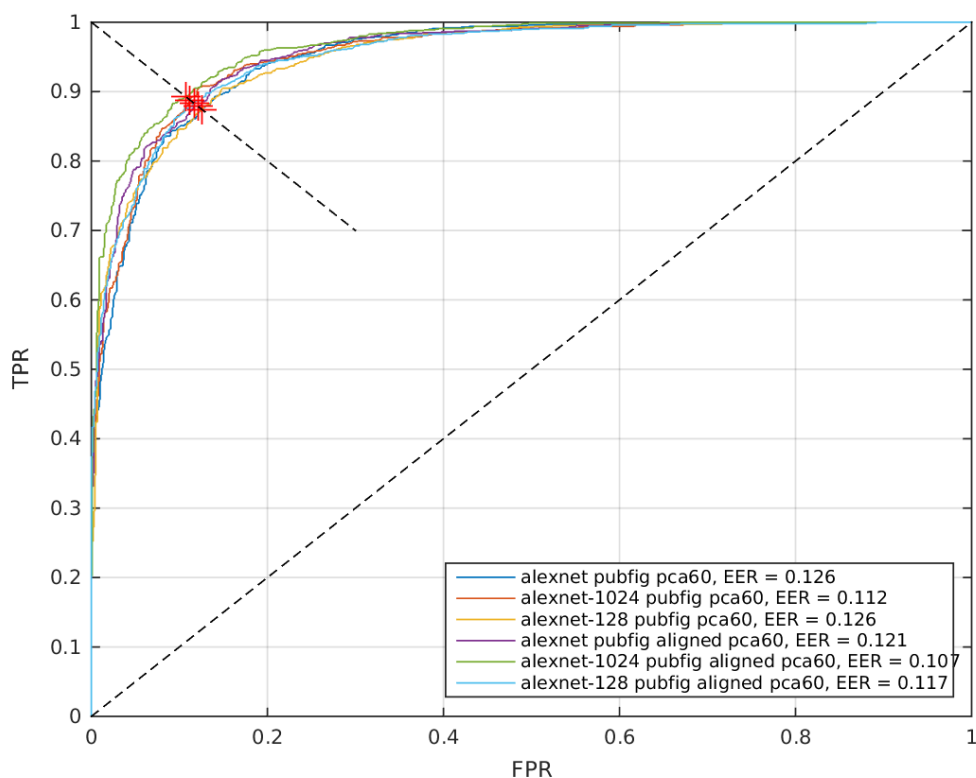
Joint Bayesian Implementation compared to Standard Classifier Algorithms

To verify the quality of the face verification based on our Joint Bayesian implementation Section 3.6.2, we compare it to face verification based on three off-the-self classifiers: Random Forests, Boosted Regression Trees and Support Vector Machine with Radial Basis Function kernel, see Section 3.6.3.

The three off-the-self classifiers were trained on pairs of face descriptors belonging to 5000 matching and 5000 non-matching identities and were given labels in $\{-1, 1\}$ accordingly.

To create the balanced training data-set (5000 examples per class) pairs were generated by uniformly sampling from a version of the Pubfig Eval dataset balanced such that each subject had 40 images. The total of 10000 training pairs was judged a reasonable upper limit for the amount of training data considering the computational resources available to us.

The evaluation data was generated by uniformly sampling 1000 matching pairs and 1000



(a) Comparison of the features from finetuned convnets of the three different architecture and with or without similarity transform alignment and all using the Joint Bayesian.

Architecture	EER (Un-Aligned)	EER (Aligned)
AlexNet	0.126	0.121
AlexNet-1024	0.112	0.107
AlexNet-128	0.126	0.117

(b) Equal Error Rates (EER) of the features from finetuned convnets of the three different architecture and with or without similarity transform alignment. The AlexNet-1024 evaluated on aligned images gives the lowest EER, 0.107.

Figure 4.2: ROC curves Figure 4.2a and EER Table 4.2b for evaluation on Pubfig with features from finetuned convnets of the three different architectures Section 3.4.4 and with or without similarity transform alignment.

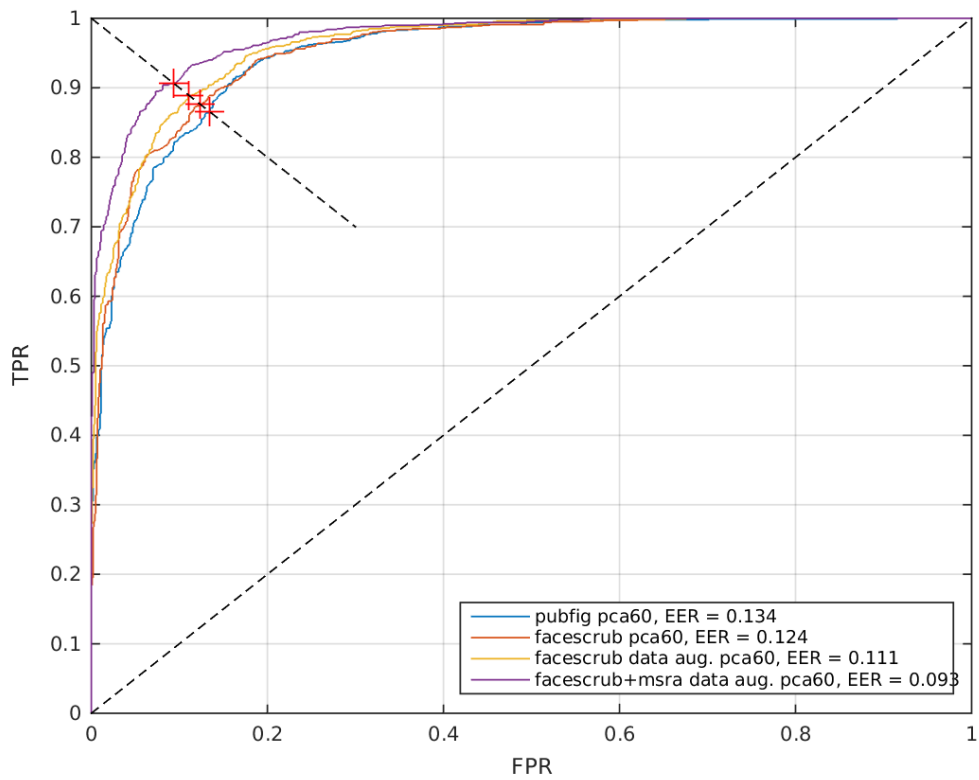


Figure 4.3: The effects of increasing the amount of training data available to the fine-tuned convnet features and the Joint Bayesian classifier.

non-matching pairs from the Pubfig Dev dataset balanced to have 40 images per subject.

All the experiments in Figure 4.4 were performed with the same feature type, i.e. the 1024-dimensional feature of AlexNet-1024, Section 3.4.4, trained on Pubfig Eval. For all the ROC curves shown in Figure 4.4 the feature dimension are compressed to 60 dimensions using PCA, except the one that says *nopca* for the Random Forests.

For the Random Forests and the SVM-RBFs the curves show evaluations for both the feature setups: Equation 3.50 and Equation 3.51. For the Boosted Regression Trees we only include ROC-curves for the better performing *stacked addition and subtraction*-feature. The legend in the figure says *stacked* for the ROC curves from experiments where the *stacked*-feature were used, otherwise we used the *stacked addition and subtraction*-feature.

The results, in Figure 4.4, show that the Joint Bayesian Classifier performs either insignificantly worse than or better than all the off-the-shelf classifiers, i.e. the true positive rate is generally higher for the same false positive rate. However, boosting using 1000 regression trees and Random Forests with 100 regression trees using the *stacked addition and subtraction*-feature gives performance comparable with the Joint Bayesian Classifier, especially when considering performance at the EER and for higher false positive rates. For the more interesting parts of the ROC space, i.e. where the false positive rate is low, the true positive rate is consequently higher for the Joint Bayesian. When it comes to SVM-RBF it performs notably worse than the other learning methods. Possibly searching for the best parameters with cross-validation or changing learning to another SVM-framework could increase the performance.

The off-the-shelf classifier ensembles with 100 and 1000 regression trees are starting to become large and we think the Joint Bayesian stands out as a good choice both when it comes to accuracy and prediction time efficiency.

Large-scale data effects on Joint Bayesian and l_2 -norm Nearest Neighbor

As we write in Section 3.6.1: *The authors of (Zhou et al., 2015) claim that their simple*

norm-based verification scheme work as well as or better than more sophisticated classification schemes, such as the Joint Bayesian Classifier, when the amount of training data is sufficiently large.

Therefore, we evaluate two different magnitudes of training data, ROC curves for an l_2 -norm nearest neighbor face verification and for Joint Bayesian face verification. The smaller training set in this example contained 303 identities. The larger training set contained 478 subjects and for the used data-set we also applied data augmentations.

From the curves in Figure 4.5 we conclude that with the amount of training data we use here the Joint Bayesian face verification still shows a clear advantage. For the l_2 -norm method there seems to exist a certain proportions of hard test images, explaining why the true positive rate for the l_2 -norm methods do not get close to 1 until very high false positive rates.

Possibly, we can observe a tendency that the difference between the methods becomes smaller for larger training sets. But that may be a too strong conclusion based on only this experiment.

4.1.5 LFW Evaluation

To formalize our face verification evaluation and to enable comparison with previous work we evaluate on the popular LFW benchmark Section 2.4. The relevant protocol in our case is *Unrestricted, Labeled Outside Data* since we train on other data-sets and make use of the labels. The 10-fold evaluation scheme requires the training data to have completely disjoint identities from the evaluation part of each fold. To make sure this is the case, we blacklist all subjects included in LFW and chose to only use other data-sets (outside data) for the training. Otherwise, the protocol specifies that 9 folds are allowed for training in each fold of the 10 folds.

We evaluate three face verification algorithms. The two first are multi-operator Uniform LBP features with two scales and AlexNet-1024 features, both trained on FaceScrub. The number of subjects in the training data were reduced to 303 since we blacklist all identities present in LFW. The third algo-

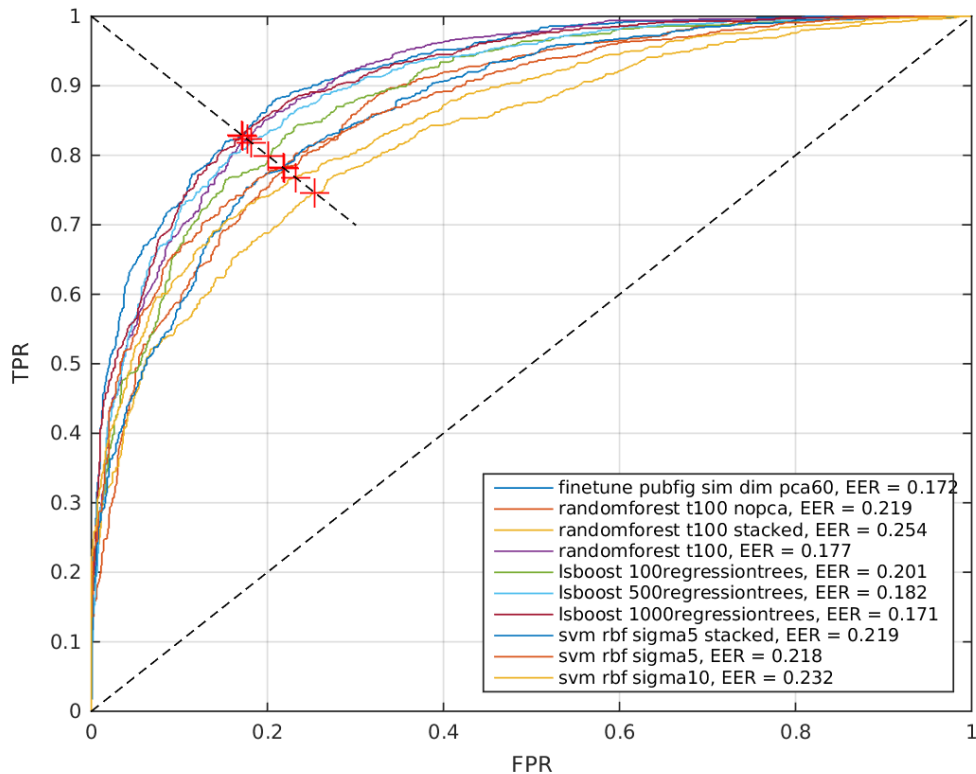


Figure 4.4: Comparisons of our Joint Bayesian implementation (blue) with face verification based on the three standard classifiers Random Forests, Boosted Regression Trees and Support Vector Machines. All training and evaluation was performed on the Pubfig data-set. The Random Forests ensembles, each with 100 regression trees, are trained on the *stacked addition and subtraction*-feature with or without PCA and on the *stacked*-feature with PCA. Results are also shown for Boosted Regression Tree ensembles trained on the *stacked addition and subtraction*-feature with PCA of dimension 60 and with increasing number of regression trees: 100, 500 and 1000. For the SVM-RBF classifiers, we show evaluations for both the *stacked addition and subtraction*-feature and the *stacked*-feature and different RBF sigmas, all with PCA dimension of 60.

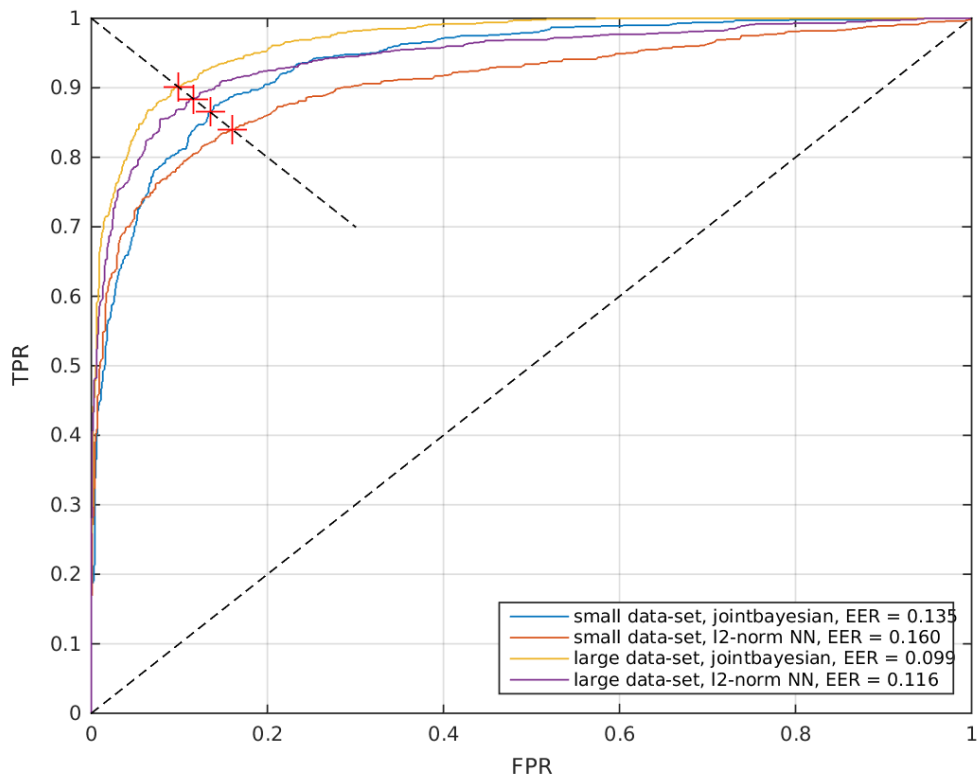


Figure 4.5: The Joint Bayesian compared to the simpler l_2 -norm distance face verification. Results for both algorithms are shown for a small (303 subjects) and a larger data-set (478 subjects).

rithm use AlexNet-1024 and a Joint Bayesian learned form a much larger data-set with data augmentations applied. It is the same convnet as evaluated in 4.1.3 trained on 478 identities from both FaceScrub and MSRA-CFW. For all three algorithms we use PCA-dimension of 200.

Figure 4.6 shows the LFW evaluations, where our algorithms are labeled as *Grundstrom CTAS*, and enables comparison to the other research efforts. The verification accuracy for the two smaller setups are 86.4% for the LBP feature and 85.3% for the convnet. However, corresponding evaluations on Pubfig Dev performed using the very same models show verification accuracies of 84.7% and 89.1% respectively, where we see the AlexNet-1024 perform better. Applying data augmentation to the FaceScrub data setup, the verification accuracy increase to 88.1% (not shown in figure).

With increased training data to 478 identities we score a verification accuracy of 91.37% with a AlexNet-1024. The data augmentations were not explored for the LBP-based algorithms because of limitations in the computational resources available.

4.1.6 Cross-Environment Scenarios with Realistic Data

Since the classifier training and feature learning have been done exclusively using frontal-face images originating from the web, it is interesting to see how face verification accuracy is effected when instead evaluating on data from surveillance video. This means the evaluation data is less consistent with the training data.

We arranged a couple of cross-environment evaluations using the Chokepoint data-set, described in Section 2.3. The way that we have processed the Chokepoint data-set, it is important to note that some face images are of significantly lower resolution compared to what have been used for the algorithm training, as low as 40×40 pixels. It would be interesting to rerun these tests with only higher resolution face images, that better match the resolutions the algorithms have been trained on.

The evaluation setup use sequences from the entry and the exit of the first portal, so the considered time-span is short. We aim to test how well subjects appearing at the entry are verified against subjects appearing at the exit. There-

fore, we create a cross-environment scenario by randomly generating pairs with the first image taken from the entry sequence and the second image from the exit.

We generated 1000 matching and 1000 non-matching pairs, from which we let the face verification algorithms, the same two 303-subject setups used in Section 4.1.5, produce confidences and finally ROC curves.

In Figure 4.7 we show the resulting ROC curves when considering three different angle offsets between the entry and exit cameras. For both algorithms we show results for centered 0° offset, 30° offset and a 45° offset.

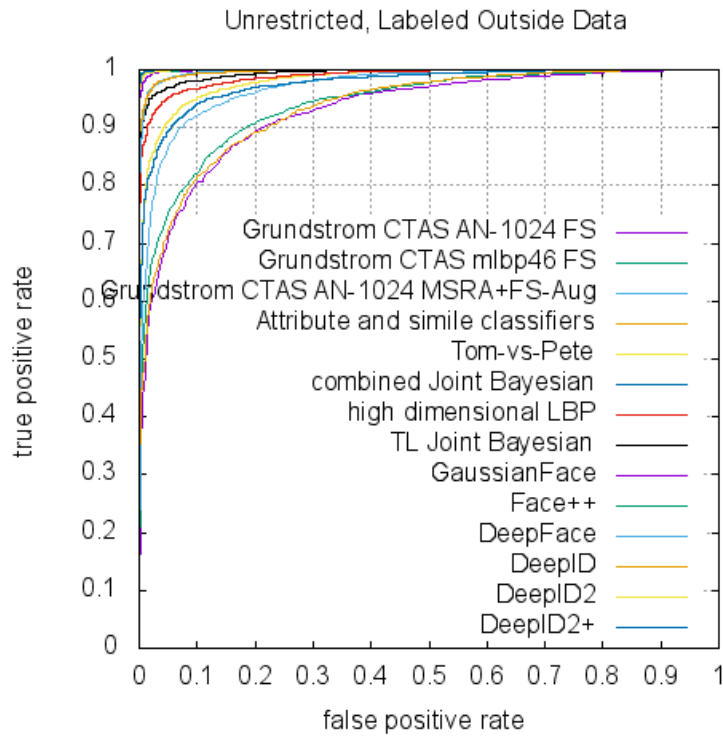
Interestingly, the performance drop is larger for the AlexNet-1024 features than for the multi-operator LBP feature when changing to the surveillance evaluation data-set and its low-resolution face images.

For both methods, the 30° offset evaluations show better results than the 0° offset evaluations. The camera angle for the data set sequences does not exactly correspond to the face pose of the captured subjects and all face images have been detected using a frontal-face detector.

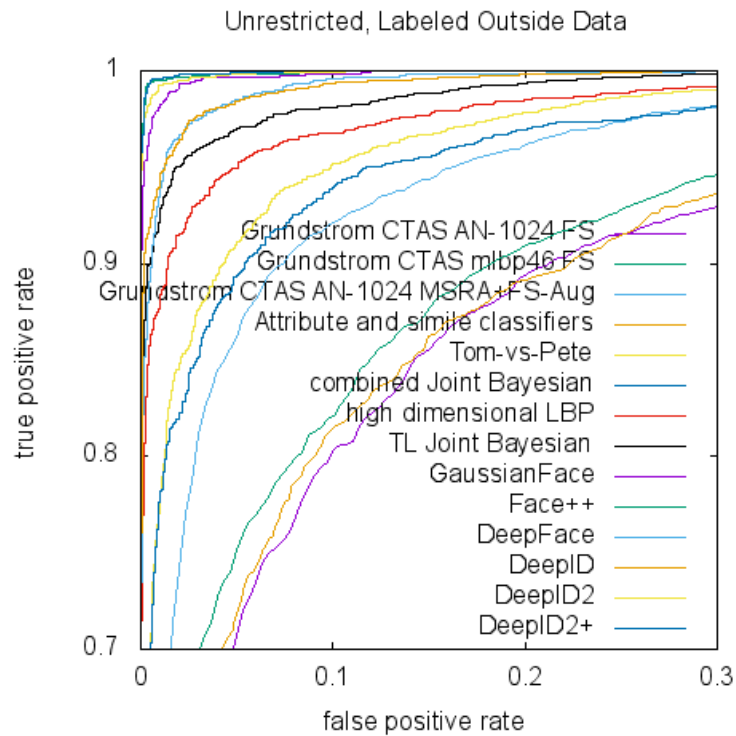
4.2 Open-Set Identification

The envisioned face recognition applications in Section 1.1.1 use real world data in relatively uncontrolled settings. The closed-area application and the multi-camera face tracking applications require the algorithm to decide if a face is known or unknown and in the former case what identity the face belongs to. This boils down to open-set face identification, where the algorithm suggests an identity and gives a confidence as for how likely the answer is true.

If the answer is accepted for any confidence, then you basically discard the possibility of unknown identities and you get closed-set identification. More reasonable is to find some threshold such that the false positive rate of the algorithm is below some application specific limit and then determine if the true positive rate is high enough. With ROC curves we show the true positive rates and false positive rates for all valid thresholds on the confidence.



(a) Selected ROC curves from the LFW *Unrestricted, Labeled Outside Data* protocol.



(b) Zoomed in ROC curves from the LFW *Unrestricted, Labeled Outside Data* protocol.

Figure 4.6: ROC curves from the LFW *Unrestricted, Labeled Outside Data* protocol. Our algorithms are the ones prefixed with *Grundstrom CTAS*.

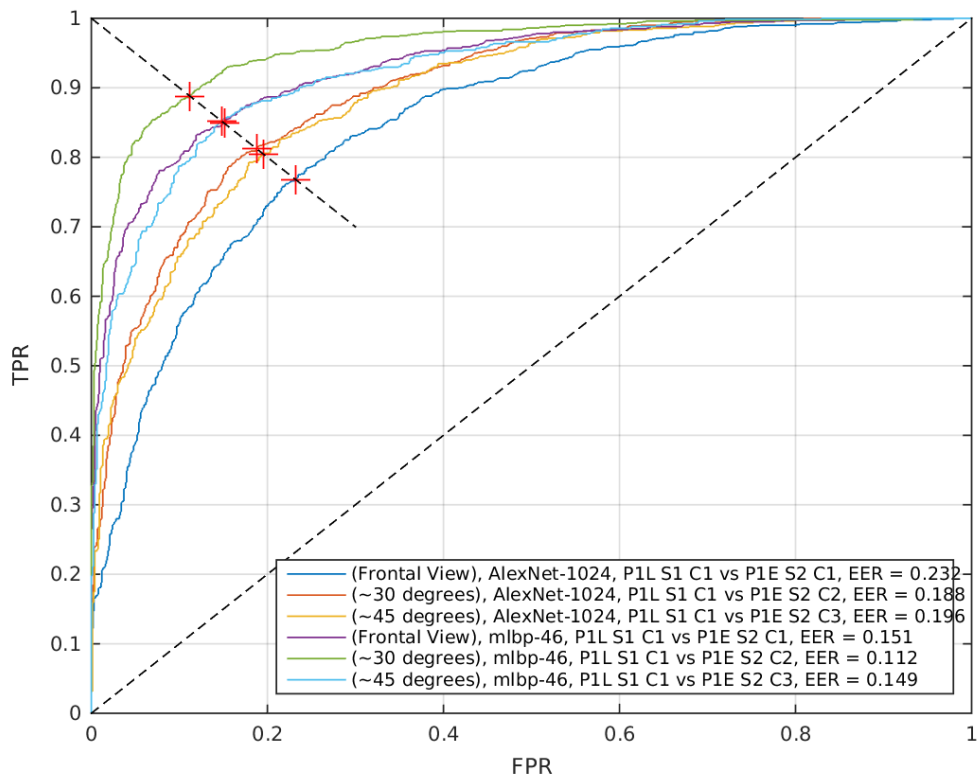


Figure 4.7: Cross-environment matching between an entry and an exit scene for three different camera angle offsets, centered 0° offset, 30° offset and a 45° offset.

4.2.1 Alarm Curves (Watch-list ROC)

Inspired by NIST’s *Face Recognition Vendor Test* (Grother and Ngan, 2013), we divide our data in a mate and a non-mate set with disjoint identities. The mate set contains the known identities and we partition it to create a gallery and a mate test set. Both of these partitions contain examples from all the mate identities. The gallery is the database of images for which the algorithm knows the identity. The purpose of the algorithm is to match test images from the mate test set and from the non-mate set with images in the database and return the suggested match-identity and a confidence score.

With the non-mate set we can estimate how likely the algorithm is to recognize a non-mate as one of the identities in the gallery of G identities. In this way we can define the *False Positive Identification Rate* (FPIR), as the ratio of false positives and the number of pursued non-mate searches. We also define the *True Positive Identification Rate* (TPIR), as the the number of mate-searches where the correct identity is found in the top R candidates and for which the confidence is above the threshold T divided by the total number of performed mate-searches. Plotting the TPIR against FPIR we get an *Alarm Curve* (or *Watch-list ROC*). Interestingly, the TPIR value for the highest FPIR value shows the performance for closed-set identification.

For $R \in \{1, 5, 10\}$ and $G = 30$ we get the results presented in Figure 4.8.

4.2.2 Cumulative Match Characteristics

For semi-automatic face recognition applications, that for instance present a list of the R top matches to the user, it is interesting to know the ratio of correct matches likely to be included above a certain rank R . A way to visualize this is to create a *Cumulative Match Characteristics* (CMC) curve, that shows the rate of searches correct containing a correct match within the top R candidates. CMC curves computed on the Pubfig Dev data-set is shown in Figure 4.9. We can see that the convnet trained on the larger data-set with data augmentation (yellow) gives highest top-1 accuracy and finds most faces in top-6. The top-1 accuracy is similar for the LBP-based algorithm (blue) and the

convnet (red) when trained on Facescrub without data augmentation.

We randomly select the face image to match with the remaining images, repeat this 1000 times and then averages.

4.2.3 Increasing Gallery Size

For completely automatic face recognition applications relying on open-set face identification it is often the Top-1 accuracy ($R = 1$) that counts, i.e. the accuracy of the best match returned by the open-set identification algorithms. In Section 4.2.1 we parameterized open-set identification based on the rank R . Here we investigate how the Top-1 performance vary with increasing gallery size G . For each gallery size $G \in \{5, 10, 20, 30, 40, 50\}$ the gallery subjects are randomly selected from the evaluation set, Pubfig Dev, and the remaining subjects are used as non-mates. For each gallery size this is repeated 10 times and Figure 4.10 shows the averages. Otherwise, this evaluation follows the same idea as described in Section 4.2.1.

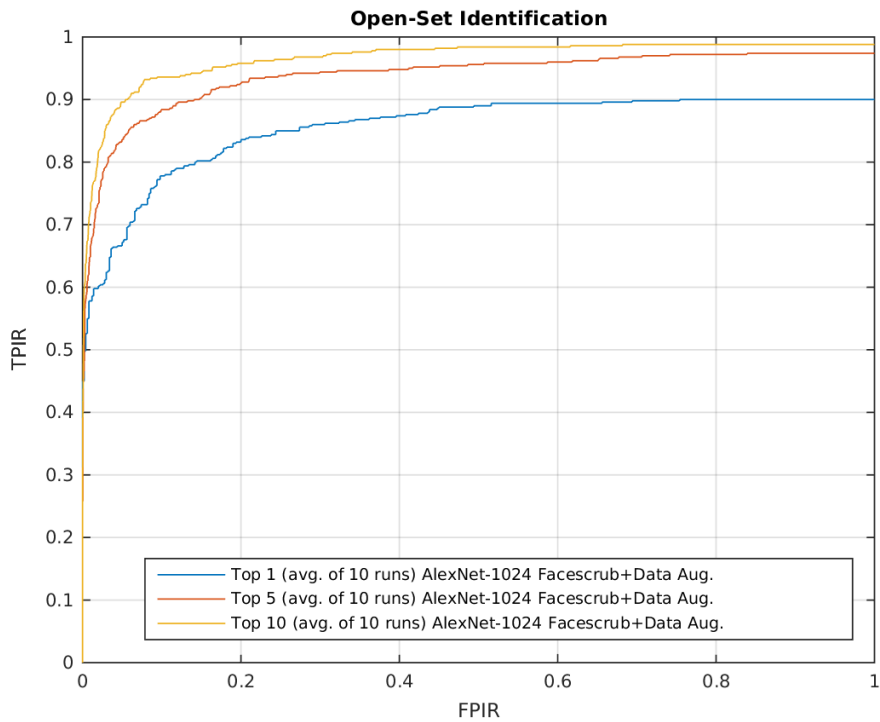
The results, Alarm Curves for a sequence of gallery sizes G , for two different algorithms is presented in Figure 4.10 and we can see how the problem get easier when the gallery size is low and how the problem becomes harder for increased number of gallery subjects. The figures also suggest that transforming the application setups to more controlled settings by giving additional structure, such that they become more similar to a closed-set identification problems, could help simplify the problem and give increased performance.

4.3 Timing Experiments

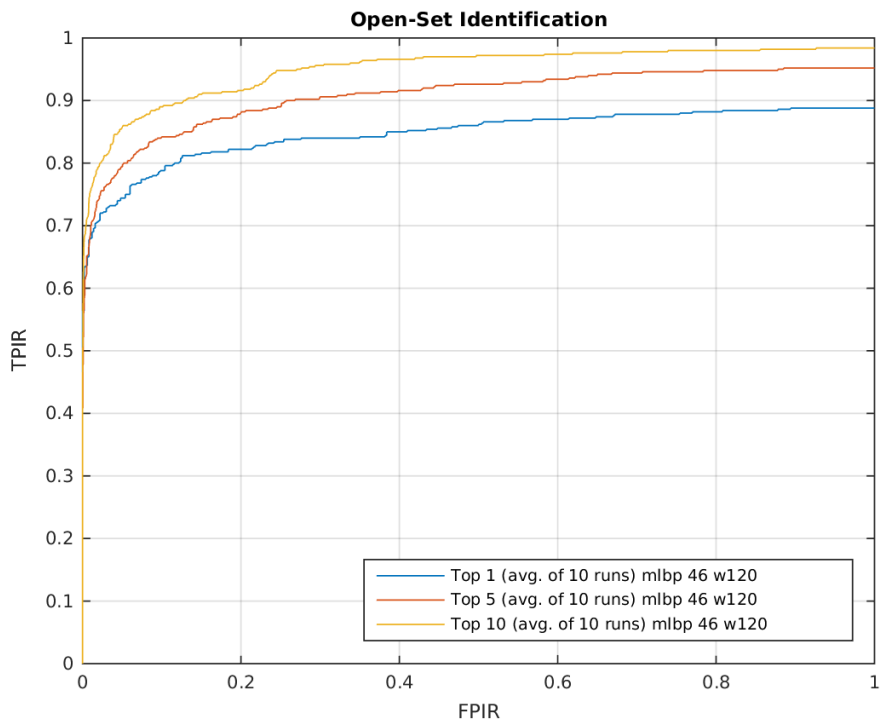
As a simple timing experiment we measure the feature extraction time averaged over 1000 images from the Pubfig data-set for both a landmark-based local feature and the AlexNet-1024 feature.

The multi-operator LBP feature with 13216 dimensions extracts in 12.2 ms on average, including face alignment 8.1 ms, pre-processing 0.2 ms and feature extraction 3.9 ms.

The AlexNet-1024 features extracts in 160.9 ms on average from aligned images using 4 CPU cores on a desktop computer. With



(a) Top 1, 5 and 10 alarm curves for the AlexNet-1024 feature trained on the combined parts of MSRA-CFW and Facescrub.



(b) Top 1,5 and 10 alarm curves for a multi-operator Uniform LBP feature.

Figure 4.8: Open-set identification evaluated with alarm curves for top 1, 5 and 10 and for deep (a) and local (b) features.

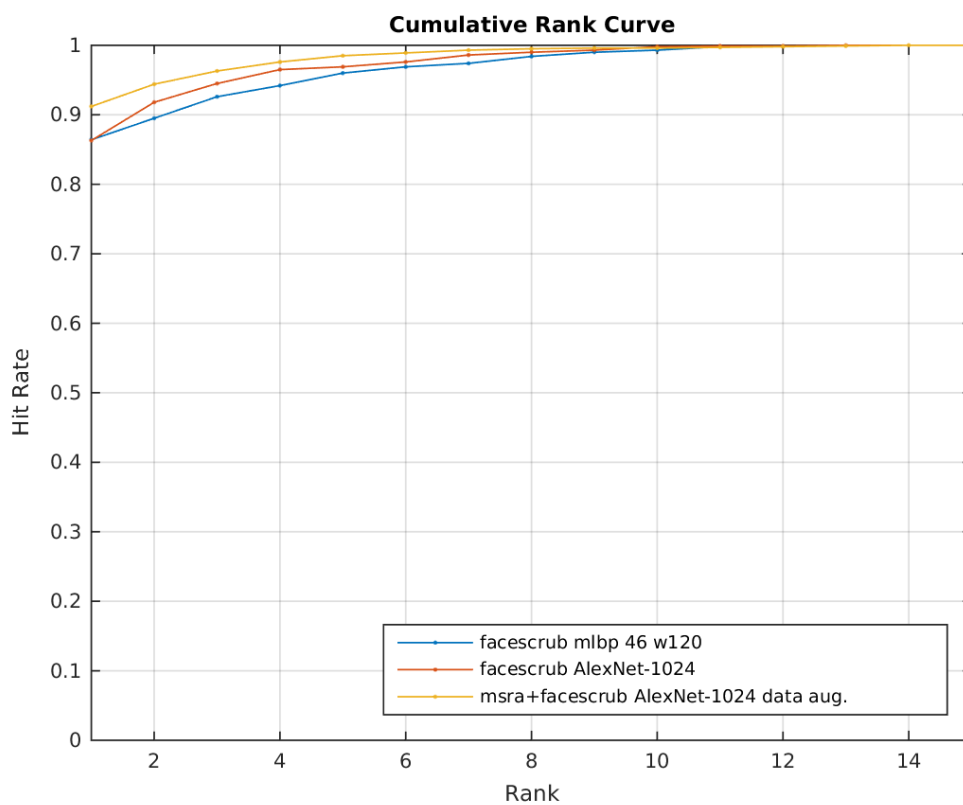
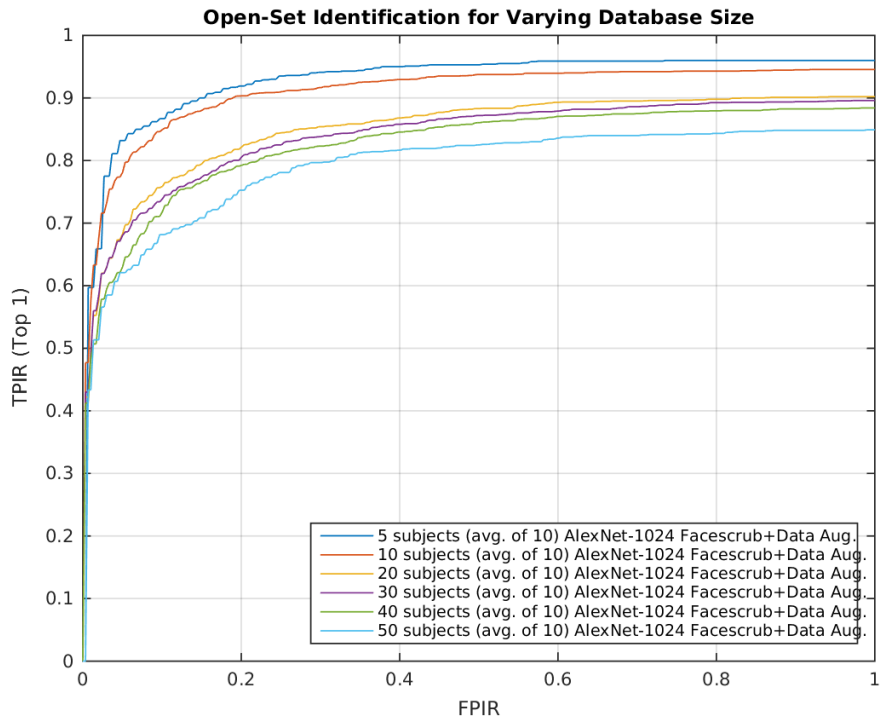
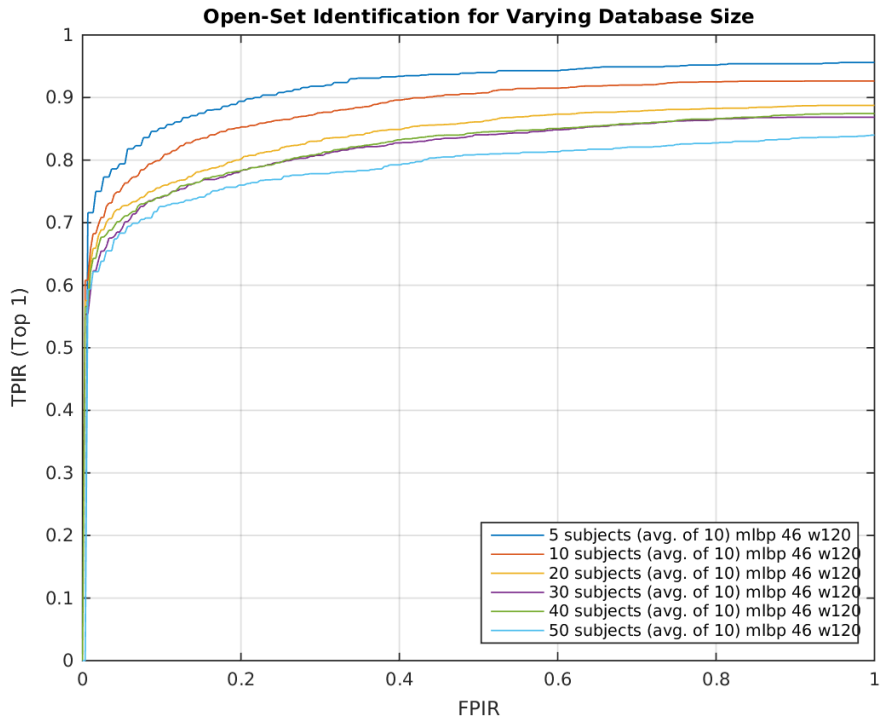


Figure 4.9: Cumulative Match Characteristics curves of two convnet-based features trained on different amount of data compared with multiple-operator Uniform LBP.



(a) Alarm curves for increasing gallery size for the AlexNet-1024 feature trained on the combined parts of MSRA-CFW and Facescrub.



(b) Alarm curves for increasing gallery size for a multi-operator Uniform LBP feature.

Figure 4.10: Open-set identification evaluated for increasing gallery size with alarm curves for deep (a) and local (b) features.

an additional 8.1 ms for the alignment the total extraction time resolves to 169 ms.

4.4 Closed-Area Application

4.4.1 Memory Requirements

The dimension of the parameters required by the PCA and the Joint Bayesian models are listed in Table 4.1. In our demo application the matrices are all 32-bit floating point numbers. In this section we neglect the storage costs of the landmark detector and focus on the most demanding parts that differentiate the presented algorithms.

The raw feature, x_{raw} , as given by the feature extractor has the dimension $d_{raw} \times 1$. We apply a dimensionality reduction to d_{pca} dimensions to get the compressed feature x_{PCA} .

The last column of the table shows a typical scenario for a 2-scale local feature based on Uniform LBP $d_{raw} = 2 \times 7 \times 4 \times 4 \times 59 = 13216$ and $d_{pca} = 200$. We note that the projection matrix is huge, 2643200-dimensions, and account for most of the storage. This subspace projection results in a time consuming matrix-vector multiplication, performed once per face descriptor. The total number of parameters needed by this example model is 2736616 floating point values, summing up the last column in Table 4.1.

Table 4.1: Storage requirements for the PCA and the Joint Bayesian Model. The last column gives the storage requirements in absolute numbers for a 13216-dimensional local feature example.

Parameter	Dimension	$x_{raw} : 13216 \times 1,$ $x_{PCA} : 200 \times 1$
Means, M	$d_{raw} \times 1$	13216
Projection, R	$d_{pca} \times d_{raw}$	2643200
Stddevs., S	$d_{pca} \times 1$	200
Model, A	$d_{pca} \times d_{pca}$	40000
Model, U	$d_{pca} \times d_{pca}$	40000

Then we of course need to maintain a database of the p persons each represented with i images, who are currently inside the closed area, this requires $d_{pca} \times pi$ floating points. The current track with features for i images requires additionally $d_{raw} \times i$ floating points.

To put this in perspective the AlexNet-based feature extractors require at least the 60 mil-

lion parameters. That is substantial and far from optimal considering embedded deployment. Considering the an AlexNet-1024 example, the dimensions $d_{raw} = 1024$ and $d_{pca} = 200$ gives a projection matrix of 204800 floats.

For convnet features the heavy lifting, i.e. a lot of floating point multiplications, are performed during the convnet’s forward-pass and results in a compact feature. The local features is cheap to extract but requires a huge subspace projection with many floating point multiplications. For embedded environments these multiplications would ideally be implemented with fix point computations, which is faster.

4.4.2 Time-Complexity of Face Matching

Here we investigate the cost of matching a single face image against a database of size N using the Joint Bayesian Section 3.6.2. Note, that we do not include the complexity of the feature extraction in these calculations.

For each image we perform the following pre-computations using the same notation and dimensionality as in Table 4.1: a subspace projection $x_{PCA} = Rx_{raw}$, an image specific constant $c = x_{PCA}^T Ax_{PCA}$ and an intermediate feature vector $y = Ux_{PCA}$. The cost of these computations are $\mathcal{O}(d_{raw}d_{PCA})$, $\mathcal{O}((d_{PCA} + 1)d_{PCA})$ and $\mathcal{O}(d_{PCA}^2)$ respectively.

The cost of pair-wisely matching the face image with a single database image involves evaluating Equation 4.1 and is more or less a scalar product of the two intermediate feature vectors, $\mathcal{O}(d_{PCA})$.

In total, performing pre-computations and matching with N images, we get a cost of

$$\mathcal{O}(d_{raw}d_{PCA} + (2d_{PCA} + 1)d_{PCA} + Nd_{PCA}), \quad (4.1)$$

where the first term is governed by the dimensionality reduction, the second by the pre-computations and the last by the pair-wise matching with N images.

The cost of the comparisons scale linearly with the number of database images. As we tried to design a system for relatively large databases (up to 1000 subjects with m images each), we expect this cost to be dominant and prefer a low dimension d_{PCA} .

For the relatively high-dimensional local features Section 3.3.3, for example with $d_{raw} =$

13216, we note that dimensionality reduction term becomes dominant. For this reason we see a large room for improved efficiency in this area.

Chapter 5

Discussion

5.1 Analysis

We have selected and implemented a classifier, the Joint Bayesian, that shows good results for face verification, Section 4.1.5, that compares good with a set of standard classifiers, Section 4.1.4, and is relatively light weight. During our development it has shown to produce a versatile similarity measure that we used to implement face verification, face search, open-set identification and the closed area demo application.

As the results in Section 4.1.5 show we have reached relevant results with our face verification algorithms. The best results were achieved with the combination of the AlexNet-1024 features with the Joint Bayesian trained on 478 identities and using data augmentations, face verification accuracy of 91.37% Figure 4.6. This is currently slightly below the state-of-the-art methods but interesting since we use a fairly compact face descriptor of 200 dimensions and a limited amount of training data.

Comparison with Other Research Efforts

Recently convnet-based methods such as DeepID2+ (Sun et al., 2014c), DeepID3 (Sun et al., 2015), Face++ (Zhou et al., 2015) and Google’s FaceNet (Schroff et al., 2015) have just about knocked the ceiling in the LFW face verification benchmark. FaceNet published the current record verification accuracy 99.63% only three months ago and interestingly use a compact 128-bit representation. All of these methods are trained on data-sets that are a magnitude larger than the data-sets that we have used. In (Sun et al., 2014c) the authors note that their moderately sparse representation can be binarized while retaining most of the accuracy, this also yields a very compact descriptor.

Nevertheless, many of the other top-performing results use complex ensembles and high dimensional descriptors that are not directly applicable in more time critical practical applications, such as real-time video analysis. Since our focus is applications in live network video we have a slightly shifted goal, for this reason (and because of hardware limitations) we have mostly considered descriptors of relatively low dimension.

Our face verification performance with both the multi-scale Uniform LBP, with face verification accuracy of 86.4% on LFW, and the AlexNet-1024 comes closer to methods such as the *combined Joint Bayesian* (Chen et al., 2012) and *high dimensional LBP* (Chen et al., 2013) shown in Figure 4.6. This is not very surprising since the same classifier ((Chen et al., 2012)) is used and our local feature method is highly inspired by the *high dimensional LBP*. In contrast to our methods *combined Joint Bayesian* and *high dimensional LBP* reports performance with ensembles of several Joint Bayesian classifiers trained on different features and combined with a linear SVM.

It is also interesting to note that the *high dimensional LBP* (Chen et al., 2013) use a raw LBP feature of 100,000 dimensions, this makes the 13,216 dimensional raw LBP feature of our implementation seem rather modest. With such high dimensionality the training procedure gets cumbersome, memory bottlenecks need to be replaced and eventually out-of-core implementations will be needed. One bottleneck in our training framework is the SVD computations on such high dimensions, even when sub-sampling. Other research, such as (Chatfield et al., 2014) comparing deep representa-

tion with hand-crafted features for generic object recognition, indicate that local feature representation requires much higher dimensionality than deep convnet representations and they also show that local features are outperformed by deep representations when it comes to accuracy on generic object recognition but that both feature types benefit from data augmentations.

Dimensionality Reduction

Investigating linear dimensionality reductions for the local features representations we found that when using LDA instead of PCA, before learning the Joint Bayesian, we could reduce the number of dimensions more while retaining the same verification accuracy. Possibly this is because some of the most important dimensions according to the PCA encodes non-discriminant information. For this reason we chose to use LDA for the LBP features. According to (Chen et al., 2013), when increasing the target dimension in the LDA subspace projection the variance of the feature dimensions (eigenvalues of Equation 3.25) can become very un-balanced, the variance of new dimensions stagnates and then performance is not increased when using more dimensions. It may appear counter-intuitive to perform LDA before learning the Joint Bayesian, because both do much of the same thing, trying to separate classes assuming underlying Gaussian class-distributions.

For the convnet-based representations we did not observe the same effects and chose to use only PCA for the dimensionality reductions.

Convolutional Neural Network Architectures

From the comparisons of different convnet architectures, Section 4.1.2, we conclude that the AlexNet-1024 architecture performed slightly better than the other and this is the reason it was selected for training with the increased amounts of data and evaluation on LFW. Instead of reducing the dimension of the last hidden layer, like (Chatfield et al., 2014), we inserted an additional hidden layer with dimension reduced to 1024. Possibly, this extra layer allows for more non-linear transformation before the dimensionality reduction (similar to deep auto-encoders).

So far, we have trained convnets with holistic face images. Training instead with local

face image patches could give better robustness to occlusions and drastic illumination changes. Another way to improve the training, giving more scale-invariance to the feature and possibly increase accuracy is to perform scale-jittering in addition to the subcropping data augmentation during training (Simonyan and Zisserman, 2014b).

There already exist several newer convnet architectures that are top-performers of the more recent years' Image-Net Large Scale Recognition Challenge (ILSRC) competitions. OverFeat (Sermanet et al., 2014) for example was considered for use in this project, but its feature extraction was heavier than the CaffeNet's. The very deep nets (14 and 19 layers) of (Simonyan and Zisserman, 2014b) for ILSRC-14 and the 22-layer deep (Szegedy et al., 2014) might be too large for us to even fine-tune with the computational resources (RAM) given by a desktop computer.

Open-Set Identification

The Alarm curves in Figure 4.8 of the open-set identification evaluation shows the difference of open-set identification ($FPIR < 1$) and closed-set identification ($FPIR = 1$), the problem gets significantly easier if the test image can be safely assumed to be within the gallery set. In a similar way Figure 4.10 shows how the open-set identification gets harder when increasing the gallery size, while it at gallery size of 5 persons is fairly simple.

Real-Time Feasibility

There is no way to disregard the large number of parameters (60 million) of the convnets that we fine-tuned initialized with the weights of *CaffeNet*. The complexity of the models becomes obvious when timing the feature extractions, Section 4.3. The feature extraction for a single image takes about 169 ms using 4 CPU cores compared to 12.2 ms for a 13,216 dimensional LBP feature using a single CPU core. Here, we actually already tried to simplify the convnet feature extraction by only using the layer activation from the forward-pass of the centered subcrop. Instead of performing 10 forward-passes for different subcrops like in the original AlexNet paper (Krizhevsky et al., 2012), that used the center, the 4 corners and all their 5 mirror images.

Because we used pre-trained weights instead of training the convnets from scratch, we need to use layers that are compatible with the layers of the pre-trained weights. This limits the convnet architectures that we could explore. For example, we could not change the architecture to use gray-scale instead of the 3 RGB channels. For generic object detection, changing the architecture from color to gray-scale resulted in approximately 3% drop in accuracy (Chatfield et al., 2014). So, it is definitely a measure to consider for reducing the number of parameters. Another thing that we could not change is the input image resolution of 256×256 , this is a choice made considering generic objects, not faces as we consider here. In face verification algorithms, such as (Sun et al., 2014b), faces are represented as multiple image patches of size 31×31 . Reducing the input dimension saves a lot of parameters and increase processing speed. Such drastic changes of the architecture probably requires training from scratch and we do not have the time or the amounts of data needed to do that.

With the large number of parameters (60 million) and feature extraction times of 200 ms per image we cannot expect to use the AlexNet-1024 features for real-time video analysis in an embedded environment without serious hardware acceleration. But the results are interesting from another point of view. The Caffe framework is actually well renowned for being fast, especially when accelerated by high-end GPUs the feature extraction using the same net can be expected to take 1-4 ms (numbers taken from the web site of the Caffe framework). Several other convnet frameworks are also hardware accelerated and runs in real-time on cloud services. With this knowledge, we think that convnets and their impressive accuracy is the competition that embedded approaches will face from centralized solutions such as cloud services. Therefore, it is also a relevant technology for us to investigate here and because the convnets are well suited for hardware acceleration we should probably expect to see them also in embedded systems in the near future.

For us and at the current time, the local features Uniform LBP and HOG provide fast enough feature extraction to be deployed in

real-time on hardware with less resources. This is the main reason why we selected the Uniform LBP feature for the development of the demo application Section 3.9. We create a prototype implementation in a desktop computer environment and show that it can run live from two video streams (we have simply not tried adding more video streams).

Live-System Challenges

In Section 4.1.6 we saw that performance dropped for the AlexNet-1024 features when evaluating on low-resolution surveillance data, data that is significantly different that the data from which the features were learned. This indicate the need for more data, data that is similar to the application domain. The LBP-based features handled the evaluation data-set transition better. Perhaps the use of color information is a dis-advantage for the AlexNet-1024 in case of this evaluation set and compared to the LBP method.

A major weakness in the face recognition for the demo application turned out to be the interaction between the face detection and the landmark detection. The landmark detection (Flandmark) turned out to be more sensitive to initialization than we expected and the face detector more inconsistent with its returned face detection bounding boxes. This became even more complicated by the change of view in the training data compared to the video data from the live streams, as the training data is (for most part) sharp frontal face images of celebrities gathered from the web and the video stream data is captured from ceiling mounted cameras with motion blur and possibly out of focus effects. The change in angle is a challenge for the landmark detector and this currently degrade accuracy. Possibly, training the landmark detector with surveillance view data could help to resolve the issue.

5.2 Future Work

Improving Accuracy

The most important area of improvement is the data. Face verification algorithms like Face++ and DeepFace are trained on 4-5 million face images from data-sets with both large width (number of subject) and the depth (number of images per subjects). Actually, the authors

of Face++ (Zhou et al., 2015) observed that adding training data with too low width can actually degrade performance.

The data that is encountered in the demo application is very different from the current training data. Ideally, a large data-set of surveillance view face images would be used for training. Currently, no such data set is publicly available to the research community.

By using face alignment methods based on 3D methods, like (Hassner et al., 2014) or DeepFace (Taigman et al., 2014) described in Section 3.1.1, we think it is possible to gain significant accuracy. These methods are sometimes called *frontalization* as they effectively produce frontal-face images from un-aligned faces and greatly simplifies the unconstrained face recognition problem. Another interesting idea is to use 3D-keypoints as in the recent paper (Li et al., 2015) in which a near real-time registration-free matching algorithm for 3D-keypoint descriptors is presented.

Currently, the only way we use temporal information in our track-based face verification is when we use optical flow to create the face tracks. Certainly, we can exploit the temporal information available in the video stream further. Interesting work is done on action recognition (Simonyan and Zisserman, 2014a). Possibly temporal modeling with Hidden Markov Models or dynamic face representations are ways forward. There also exist completely video-based face recognition methods like *Affine Hull based Image Set Distance* (AHISD) (Cevikalp and Triggs, 2010) or manifold modeling based on the *Mutual Subspace Method* (MSM) (Yam, 1998) but according to (Wolf et al., 2011), introducing the *YouTube Faces DB* benchmark with video-to-video matching and comparing wide range of methods (including Constrained Mutual Subspace Method) using LBP features, other methods are more successful.

The convnets we fine-tuned for face representations were only supervised by identification labels. The (Sun et al., 2014a) and its successors supervise their convnet-based feature extractors with both identification and verification signals by training on image pairs. A further improvement (Sun et al., 2014c) added supervision also for the early layers, including the

convolutional layers, instead of just supervising from the output layer as we have done.

To counter problems with long distance shots that give low-resolution images and is quite common in surveillance video, it could be reasonable to train face verification models for multiple image resolutions and dynamically select which model to use at test-time.

The development of 3D face features using stereo-imaging, depth maps and sensor data are also directions where we expect progress. Face recognition is a strong biometric and can be a key component in the more general person re-identification problem. Even so, such a hard problem could benefit from multi-modal training data.

In the demo application we could gain some accuracy from improved consistency of the face detection regions because this improves the quality of the facial landmark detection and as a consequence also face alignment and feature extraction.

Increasing Efficiency

For convnet approaches, when considering efficiency in computational terms, a possible way forward is to use model compression (Bucila et al., 2006) (Ba and Caurana, 2013) (Hinton et al., 2015), i.e. learning a reduced model from a complex model. This allows for convnets light-weight enough to be deployed to be learned from more complex models, for example by doing regression on the SoftMax outputs. As mentioned earlier in the discussion we can improve efficiency by using gray-scale input and reduce the input dimensions. A lot of research is done on improving efficiency of convolutions, for example computing the convolutions in frequency domain accelerated by hardware (Mathieu et al., 2013).

In some cases benefits can be given by enforcing sparsity, for example the large PCA subspace projection matrix can be approximated with a sparse matrix like the authors of (Chen et al., 2013) do with their Rotated Sparse regression. The LBP histograms, if using relatively small regions for the histograms, are sparse. This should at the very least be exploited in computations, for example by skipping whole columns in the LDA-PCA subspace vector-matrix multiplication. Recent publica-

tions (Sun et al., 2014c) show that their relatively sparse convnet feature representation can be binarized without losing much accuracy. This enables extremely compact descriptors and could improve matching efficiency as well. A similar way to create compact binary descriptors is to use image retrieval methods such as Semantic Hashing, that learn sparse auto-encoders to create binary codes from feature vectors. Such compact binary descriptors are tractable to use as meta-information and send over the network.

Other kinds of features selections, such as AdaBoosting with multi-task learning (Wang et al., 2009), can be used to reduce the size of the *raw* local features by knowing beforehand what features are likely to be useful. However, no such works are presented among the state-of-the-art methods evaluated on LFW so it is unclear how they compare.

For the face tracking a first improvement would be to use a motion model. Then the initial search positions for the interest points can be estimated and potentially the search becomes faster. In some cases a pure tracker solution, not solely relying on face detections, could capture a wider range of face poses and give longer and better tracks. Then alternating detection and tracking to save computational resources. An obvious optimization is to only compute the optical flow for a region of interest.

5.3 Conclusion

In this work we have set out to find a feature extraction and classifier combination leading to a face verification algorithm suitable for real-time video applications. The experimental evaluations of feature types and classifiers, Section 4.1.1 and 4.1.4, left us with two main alternatives for face verification: the landmark-based multi-operator LBP features and the deep representations extracted from an AlexNet-1024 convnet.

For those two algorithms we have considered the computational characteristics in terms of feature extraction times, Section 4.3, and the memory footprint in terms of the number of parameters, Section 4.4.1. It is clear that the local feature algorithm is more light-weight than an AlexNet-1024 algorithm with the presented setups, both in terms of feature extrac-

tion time and memory footprint. However, the obvious way to improve the accuracy for the local features requires increased dimensionality, either by increasing the number of landmark regions, scales or feature types. In Section 5.1 we discussed the high-dimensions necessary to achieve near state-of-the-art performance with local features. Using a more modest feature dimension, like the 13,216 that we have used, is a compromise solution. On the other hand, it is most likely that a convnet architecture with fewer parameters than the AlexNet, which was designed for generic object recognition, can produce good results for faces. Model compression, as suggested in Section 5.2, can be a way to achieve this. Currently, convnet methods run in real-time on GPU accelerated workstations or cloud services but we think it is only a matter of time before we see improved methods and sufficient hardware acceleration for deployment also in embedded environments and distributed systems.

Comparisons of a set of classifiers, 4.1.4, makes us confident that the Joint Bayesian classifier is a good choice even outside the LFW top-score list. With a compact feature descriptor we were able to use it live in the demo application and we expect matching to scale reasonably for increased database sizes, Section 4.4.2.

Evaluation on the LFW benchmark shows that the deep representations combined with the Joint Bayesian classifier and trained on a larger data-set (478 subjects) give high verification accuracy, 91.37%, see Section 4.1.5. With a data-set about half the size (303 subjects) we still achieve relevant results on LFW, 86.4% verification accuracy for the multi-operator LBP and 85.3% for the AlexNet-1024 feature. The experiments show that substantial gains in performance can be made by increasing the amount of training data.

We implemented open-set identification with pair-wise face verification. The alarm curves in Section 4.2.1 makes it clear that we face a much harder problem than closed-set face identification. We also observed how the open-set identification becomes harder when the size of the database increase, see Section 4.2.3. In the same way the closed area application problem becomes harder when the size of the dynamic database increase.

Altogether, we view the multiple-operator Uniform LBP extracted around landmark points as the best suited candidate for development in an embedded environment, at least in the immediate future. For this reason we implemented the prototype with the multiple-operator Uniform LBP. We also identified the subspace projection as the most critical part of the method because of the large matrix dimensions when the raw feature vectors are high dimensional, see Section 4.4.1. The dimensionality reduction is a vector-matrix multiplication and results in many expensive floating point multiplications, but it lend it self well to hardware accelerations and if using Uniform LBP histograms with small regions the high dimensional feature is sparse.

By developing a prototype system for the closed area application we show that the classifier and the rest of the video-based face recognition pipeline can handle (at least two) live video streams, when the prototype runs on a desktop computer, and maintain a dynamic gallery of the persons that are currently inside a closed area.

Acknowledgments

Thanks Axis Communications for showing great trust in me and giving the opportunity to perform this master's thesis together. I want to direct a big thanks to all the colleagues at Core Technologies Analytics and Systems for the innovative atmosphere and all the helpful discussions. Especially, I want to thank my supervisors Jiandan Chen, Martin Ljungqvist and Kalle Åström for their guidance, well directed feedback and endless patience.

Chapter 6

References

References

- [Ahonen et al.2004] Timo Ahonen, Abdenour Hadid, and Matti Pietikäinen. 2004. Face recognition with local binary patterns. *T. Pajdla and J. Matas (Eds.): ECCV 2004, LNCS 3021*, pp. 469–481.
- [Alpaydin2010] E. Alpaydin. 2010. *Introduction to Machine Learning, Second Edition*. MIT Press.
- [Apostoloff and Zisserman2007] Nicholas Apostoloff and Andrew Zisserman. 2007. Who are you? – real-time person identification. *Proceedings of the British Machine Vision Conference*.
- [Azizpour et al.2014] Hossein Azizpour, Ali Sharif Razavian, Josephine Sullivan, Atsuto Maki, and Stefan Carlsson. 2014. From generic to specific deep representations for visual recognition. *CoRR*, abs/1406.5774.
- [Ba and Caurana2013] Lei Jimmy Ba and Rich Caurana. 2013. Do deep nets really need to be deep? *CoRR*, abs/1312.6184.
- [Belhumeur et al.1997] P. Belhumeur, J. Hespanha, and D. Kriegman. 1997. Eigenfaces vs. fisherfaces: recognition using class specific linear projection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 711–720.
- [Bouguet2000] Jean-Yves Bouguet. 2000. Pyramidal implementation of the lucas kanade feature tracker description of the algorithm.
- [Breiman2001] Leo Breiman. 2001. Random forests. *Machine Learning*, 45(1):5–32.
- [Bucila et al.2006] Cristian Bucila, Rich Caruana, and Alexandru Niculescu-Mizil. 2006. Model compression. In *Proceedings of the Twelfth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Philadelphia, PA, USA, August 20-23, 2006*, pages 535–541.
- [Cao et al.2013] Xudong Cao, Yichen Wei, Fang Wen, and Jian Sun. 2013. Face alignment by explicit shape regression. *International Journal of Computer Vision*.
- [Cevikalp and Triggs2010] Hakan Cevikalp and Bill Triggs. 2010. Face recognition based on image sets. In *CVPR*, pages 2567–2573. IEEE Computer Society.
- [Chatfield et al.2014] Ken Chatfield, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2014. Return of the devil in the details: Delving deep into convolutional nets. *CoRR*, abs/1405.3531.
- [Chen et al.2012] Dong Chen, Xudong Cao, Liwei Wang, Fang Wen, and Jian Sun. 2012. Bayesian face revisited: A joint formulation. In *Proceedings of the 12th European Conference on Computer Vision - Volume Part III, ECCV’12*, pages 566–579, Berlin, Heidelberg. Springer-Verlag.
- [Chen et al.2013] Dong Chen, Xudong Cao, Fang Wen, and Jian Sun. 2013. Blessing of dimensionality: High-dimensional feature and its efficient compression for face verification. *Computer Vision and Pattern Recognition (CVPR)*.
- [Ciresan et al.2012] Dan Ciresan, Alessandro Giusti, Luca M. Gambardella, and Jürgen Schmidhuber. 2012. Deep neural networks segment neuronal membranes in electron microscopy images. In F. Pereira, C.J.C. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 2843–2851. Curran Associates, Inc.

- [Dalal and Triggs2005] N. Dalal and B. Triggs. 2005. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893 vol. 1, June.
- [Deng and Yu2014] Li Deng and Dong Yu. 2014. Deep learning: Methods and applications. Technical Report MSR-TR-2014-21, Microsoft Research, May.
- [Ekenel et al.2009] HazımKemal Ekenel, Lorant Szasz-Toth, and Rainer Stiefelhagen. 2009. Open-set face recognition-based visitor interface system. In Mario Fritz, Bernt Schiele, and JustusH. Piater, editors, *Computer Vision Systems*, volume 5815 of *Lecture Notes in Computer Science*, pages 43–52. Springer Berlin Heidelberg.
- [Everingham et al.2009] Mark Everingham, Josef Sivic, and Andrew Zisserman. 2009. Taking the bite out of automated naming of characters in tv video. *Department of Engineering Science, University of Oxford, Parks Road, Oxford OX1 3PJ, UK*.
- [Fischer et al.2011] Mika Fischer, Hazım Kemal Ekenel, and Rainer Stiefelhagen. 2011. Person re-identification in tv series using robust face recognition and user feedback. *Multimedia Tools Appl.*, 55(1):83–104.
- [Grother and Ngan2013] Patrick Grother and Mei Ngan. 2013. Face recognition vendor test (frvt) - performance of face identification algorithms - nist interagency report 8009.
- [Harris and Stephens1988] C. Harris and M. Stephens. 1988. A combined corner and edge detector. In *Proceedings of the 4th Alvey Vision Conference*, pages 147–151.
- [Hartley and Zisserman2004] R. I. Hartley and A. Zisserman. 2004. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition.
- [Hassner et al.2014] Tal Hassner, Shai Harel, Eran Paz, and Roei Enbar. 2014. Effective face frontalization in unconstrained images. *CoRR*, abs/1411.7964.
- [Hinton et al.2012] Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2012. Improving neural networks by preventing co-adaptation of feature detectors. *CoRR*, abs/1207.0580.
- [Hinton et al.2015] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. <http://arxiv.org/abs/1503.02531>.
- [Huang et al.2007] Gary B. Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller. 2007. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical Report 07-49, University of Massachusetts, Amherst, October.
- [Huang et al.2012] Gary B. Huang, Marwan A. Mattar, Honglak Lee, and Erik G. Learned-Miller. 2012. Learning to align from scratch. In *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States.*, pages 773–781.
- [Jafri and R.Arabnia2009] Rabia Jafri and Hamid R.Arabnia. 2009. A survey of face recognition techniques. *Journal of Information Processing Systems*, Vol.5, No.2.
- [Jia et al.2014] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. 2014. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*.
- [Karayev et al.2013] Sergey Karayev, Aaron Hertzmann, Holger Winnemoeller, Aseem Agarwala, and Trevor Darrell. 2013. Recognizing image style. *CoRR*, abs/1311.3715.
- [Kläser et al.2012] Alexander Kläser, Marcin Marszałek, Cordelia Schmid, and Andrew Zisserman. 2012. Human focused action localization in video. In Kiriakos N. Kutulakos, editor, *Trends and Topics in Computer Vision*, volume 6553 of *Lecture Notes in Computer Science*, pages 219–233. Springer Berlin Heidelberg.
- [Krizhevsky et al.2012] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. Imagenet classification with deep convolutional neural networks. In F. Pereira, C.J.C. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc.

- [Kumar et al.2009] N. Kumar, A. C. Berg, P. N. Belhumeur, and S. K. Nayar. 2009. Attribute and Simile Classifiers for Face Verification. In *IEEE International Conference on Computer Vision (ICCV)*, Oct.
- [Lecun et al.1998] Yann Lecun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, pages 2278–2324.
- [Li et al.2015] Huibin Li, Di Huang, Jean-Marie Morvan, Yunhong Wang, and Liming Chen. 2015. Towards 3d face recognition in the real: A registration-free approach using fine-grained matching of 3d keypoint descriptors. *International Journal of Computer Vision*, 113(2):128–142.
- [Lindgren et al.2013] Lindgren, Rootsén, and Sandsten. 2013. *Stationary Stochastic Processes for Scientists and Engineers*. CRC Press.
- [Lindgren2006] Finn Lindgren. 2006. Image modelling and estimation a statistical approach.
- [Liu et al.2013] Xin Liu, Zaiwen Wen, and Yin Zhang. 2013. Limited memory block krylov subspace optimization for computing dominant singular value decompositions. *SIAM Journal on Scientific Computing*, 35(3):A1641–A1668.
- [Lu and Tang2014] Chaochao Lu and Xiaoou Tang. 2014. Surpassing human-level face verification performance on LFW with gaussianface. *CoRR*, abs/1404.3840.
- [Lucas and Kanade1981] BruceD. Lucas and Takeo Kanade. 1981. An Iterative Image Registration Technique with an Application to Stereo Vision (DARPA). In *Proceedings of the 1981 DARPA Image Understanding Workshop*, pages 121–130.
- [Mathieu et al.2013] Michaël Mathieu, Mikael Henaff, and Yann LeCun. 2013. Fast training of convolutional networks through ffts. *CoRR*, abs/1312.5851.
- [Mian and Pears2012] Ajmal Mian and Nick Pears. 2012. Chapter 8 3d face recognition.
- [Ng and Winkler2014] H.W. Ng and S. Winkler. 2014. A data-driven approach to cleaning large face datasets. In *ICIP14*, pages 343–347.
- [Ojala et al.2002] Timo Ojala, Matti Pietikäinen, and Topi Mäenpää. 2002. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, VOL. 24, NO. 7*.
- [Pietikäinen et al.2011] Matti Pietikäinen, Abdenour Hadid, Guoying Zhao, and Timo Ahonen. 2011. *Computer Vision Using Local Binary Patterns*. Springer-Verlag London Limited.
- [Ramanan et al.2007] Deva Ramanan, Simon Baker, and Sham Kakade. 2007. Leveraging archival video for building face datasets. In *In Proc. of the IEEE 11th Int'l. Conf. on Computer Vision*.
- [Razavian et al.2014a] Ali Sharif Razavian, Hossein Azizpour, Atsuto Maki, Josephine Sullivan, Carl Henrik Ek, and Stefan Carlsson. 2014a. Persistent evidence of local image properties in generic convnets. *CoRR*, abs/1411.6509.
- [Razavian et al.2014b] Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. 2014b. CNN features off-the-shelf: an astounding baseline for recognition. *CoRR*, abs/1403.6382.
- [Schroff et al.2015] Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. Facenet: A unified embedding for face recognition and clustering. *CoRR*, abs/1503.03832.
- [Sermanet et al.2014] Pierre Sermanet, David Eigen, Xiang Zhang, Michael Mathieu, Rob Fergus, and Yann LeCun. 2014. Overfeat: Integrated recognition, localization and detection using convolutional networks. In *International Conference on Learning Representations (ICLR 2014)*. CBLS, April.
- [Simonyan and Zisserman2014a] Karen Simonyan and Andrew Zisserman. 2014a. Two-stream convolutional networks for action recognition in videos. *CoRR*, abs/1406.2199.
- [Simonyan and Zisserman2014b] Karen Simonyan and Andrew Zisserman. 2014b. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556.

- [Sivic et al.2009] J. Sivic, M. Everingham, and A. Zisserman. 2009. 'who are you?' - learning person specific classifiers from video. In *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR2009)*, pages 1145-1152.
- [Solem2012] Jan Erick Solem. 2012. *Programming computer vision with Python*. O'Reilly.
- [Sun et al.2014a] Yi Sun, Xiaogang Wang, and Xiaoou Tang. 2014a. *Deep Learning Face Representation by Joint Identification-Verification*. Ph.D. thesis, arXiv.
- [Sun et al.2014b] Yi Sun, Xiaogang Wang, and Xiaoou Tang. 2014b. Deep learning face representation from predicting 10,000 classes. In *Computer Vision and Pattern Recognition*, pages 1891-1898. IEEE.
- [Sun et al.2014c] Yi Sun, Xiaogang Wang, and Xiaoou Tang. 2014c. Deeply learned face representations are sparse, selective, and robust. *CoRR*, abs/1412.1265.
- [Sun et al.2015] Yi Sun, Ding Liang, Xiaogang Wang, and Xiaoou Tang. 2015. Deepid3: Face recognition with very deep neural networks. *CoRR*, abs/1502.00873.
- [Szegedy et al.2014] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2014. Going deeper with convolutions. *CoRR*, abs/1409.4842.
- [Taigman et al.2014] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. 2014. Deepface: Closing the gap to human-level performance in face verification. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.
- [Tomasi and Kanade1991] Carlo Tomasi and Takeo Kanade. 1991. Detection and tracking of point features.
- [Tomasi and Shi1994] C. Tomasi and J. Shi. 1994. Good features to track. In *9th IEEE Conference on Computer Vision and Pattern Recognition*, pages 593-600. Springer.
- [Turk and Pentland.1991] M. A. Turk and A. P. Pentland. 1991. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1):71-86.
- [Uřičář et al.2012] Michal Uřičář, Vojtěch Franc, and Václav Hlaváč. 2012. Detector of facial landmarks learned by the structured output SVM. In *VISAPP '12: Proceedings of the 7th International Conference on Computer Vision Theory and Applications*, pages 547-556, Portugal. SciTePress — Science and Technology Publications.
- [Wang et al.2009] Xiaogang Wang, Cha Zhang, and Zhengyou Zhang. 2009. Boosted multi-task learning for face verification with applications to web image and video search. *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*.
- [Wikimedia Foundation2015a] Inc Wikimedia Foundation. 2015a. Kanade-lucas-tomasi feature tracker. Accessed: 2015-04-09.
- [Wikimedia Foundation2015b] Inc Wikimedia Foundation. 2015b. Lucas-kanade method. Accessed: 2015-04-09.
- [Wikimedia Foundation2015c] Inc Wikimedia Foundation. 2015c. Optical flow. Accessed: 2015-04-09.
- [Wolf et al.2011] Lior Wolf, Tal Hassner, and Itay Maoz. 2011. Face recognition in unconstrained videos with matched background similarity. In *in Proc. IEEE Conf. Comput. Vision Pattern Recognition*.
- [Wong et al.2011] Yongkang Wong, Shaokang Chen, Sandra Mau, Conrad Sanderson, and Brian C. Lovell. 2011. Patch-based probabilistic image quality assessment for face selection and improved video-based face recognition. In *IEEE Biometrics Workshop, Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 81-88. IEEE, June.
- [Yam1998] 1998. *Face recognition using temporal image sequence*.
- [Zhang et al.2012] Xiao Zhang, Lei Zhang, Xin-Jing Wang, and Heung-Yeung Shum. 2012. Finding celebrities in billions of web images. *Multimedia, IEEE Transactions on*, 14(4):995-1007, Aug.
- [Zhou et al.2015] Erjin Zhou, Zhimin Cao, and Qi Yin. 2015. Naive-deep face recognition: Touching the limit of LFW benchmark or not? *CoRR*, abs/1501.04690.

[Zhu et al.2013] Zhenyao Zhu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. 2013. Deep learning identity-preserving face space. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, Dec.

Master's Theses in Mathematical Sciences 2015:E13

ISSN 1404-6342

LUTFMA-3273-2015

Mathematics

Centre for Mathematical Sciences

Lund University

Box 118, SE-221 00 Lund, Sweden

<http://www.maths.lth.se/>