UNIVERSITY OF TARTU
FACULTY OF SCIENCE AND TECHNOLOGY
Institute of Computer Science
Computer Science

Zepp Uibo

Comparison of Face Recognition Neural Networks

Bachelor's thesis (6 ECST)

Supervisor: Tambet Matiisen

Tartu 2016

Comparison of Face Recognition Neural Networks

Abstract:

The goal of this work was to compare three face recognition neural networks that had been recently published. All of those networks had shown good results on a benchmark containing mostly higher quality images of celebrities. The interest lies in finding whether these networks are able to perform as well on a different dataset of lower quality archive images. A new benchmark dataset was created on images from the National Archives of Estonia. Then the accuracy of determining whether two face images belong to the same person or not was measured on the new dataset. The network with the strongest reproducible result showed a strong results on the new benchmark, an accuracy of 91.18%. A suggestion is made by the author of using the same network for further work on the images from the National Archives dataset.

Keywords: artificial neural networks, machine learning, face recognition algorithms

CERCS: P175

Näotuvastuseks treenitud tehisnärvivõrkude võrdlemine

Lühikokkuvõte:

Selles töös uuriti kolme hiljuti avaldatud näotuvastuseks treenitud tehisnärvivõrku. Kõik need võrgud on seni näidanud häid tulemusi kõrge kvaliteediga piltide identifitseerimist kontrollivates testides. Huvi tekitas küsimus, kas need võrgud on võimelised samaväärseid tulemusi saavutama madalama kvaliteediga arhiivipiltide peal. Loodi uus testandmestik Eesti Rahvusarhiivi piltidest ja võrreldi, kui täpsed on võrgud tuvastama, kas kaks nägu kuuluvad samale või erinevatele inimestele. Parim korratava tulemusega närvivõrk saavutas uute andmete peal täpsuse 91.18%. Töö autor soovitab sama närvivõrguga Eesti Rahvusarhiivi andmete peal tööd jätkata.

Võtmesõnad: tehisneurovõrgud, masinõpe, näotuvastusalgoritmid

CERCS: P175

Table of Contents

Introduction

The National Archives of Estonia has over 500 000 digital photos in its databases. All of them have textual descriptions and some are associated with the people on those pictures.. Having automated means to recognise the faces on the images and connect them with people would allow for better search capabilities and enrich the data in the archives.

Convolutional neural networks have been the most successful approach to the face recognition problem in the last decade. Lately, several pre-trained networks have been published and made open source. All those networks were trained on a large number of face images gathered from online resources like the Internet Movie Database and Google search results. The goal of the current thesis is to find out which of these networks could be the best one to use with the faces in the FOTIS database of the National Archives. The challenge is not trivial, because of the difference in the data the networks were trained on and the data we want to use them for. The networks were trained on mostly higher quality colorful "red carpet" photos of celebrities. The photos in the archives are of lower quality, are grayscale and contain mostly faces of Estonian politicians from previous decades, where the haircuts, facial expressions, and the age and gender distributions are different.

In this thesis three different neural networks are compared. First, the accuracies of the networks are calculated on a common benchmark dataset, to validate our setup against the known results published on the same dataset. Then, similar benchmark datasets are generated for the FOTIS database and the accuracies of the networks calculated using the same methodology.

In Chapter 1, the theoretical background of this work is discussed, describing artificial neural networks and face recognition in general. Also, convolutional neural networks are described in more detail. In Chapter 2, the benchmark datasets are described. In Chapter 3, the architecture and training methodology of the three networks is described. In Chapter 4, the methodology and setup of the benchmark tests is laid out. In Chapter 5, the results are presented and discussed. Chapter 6 contains the conclusions and suggestions for further work by the author.

1. Theoretical Background

1.1 Terminology/Definitions Used

Artificial neural network (ANN) is an information processing model inspired by biological neural systems. An ANN is made up of *artificial neurons*, that are usually organized in layers. A schematic of a simple neural network is presented in Figure 1.

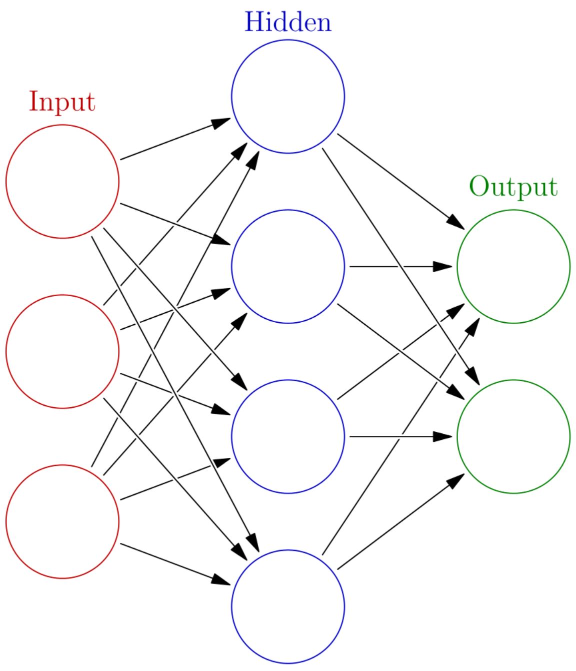


Figure 1. An image of a three-layer neural network with three input nodes, four hidden nodes and two output nodes. Image from [1]

Similarly to a biological neuron, an artificial neuron takes its input from several other neurons (corresponding to synaptic input in a biological neuron) and generates a single output (propagated by an axon in a biological neuron). The inputs are weighted, which means that different inputs influence the output with different strength. A graphical presentation of an artificial neuron is shown in Figure 2.

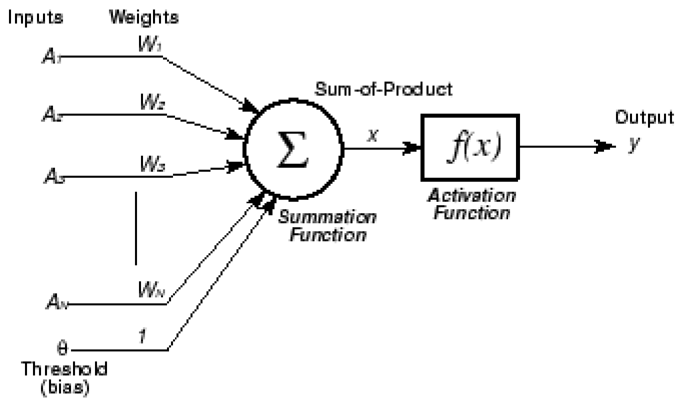


Figure 2. An artificial neuron. Inputs $A_1...A_n$ are multiplied by their corresponding weights $W_1...W_n$, the products are summed and a bias value Θ is added. The result is passed to the activation function, which generates the output of the neuron. Image from [2].

A layer in a network is usually made up of neurons using the same activation function. The choice of an activation function is very important for the performance of the neural network. A common type of an activation function is *sigmoid function* – a function that has a large slope when the input is close to 0 and has slope approaching 0 when the values approach $-\infty$ or $+\infty$. Some examples of activation functions are pictured in Figure 3.

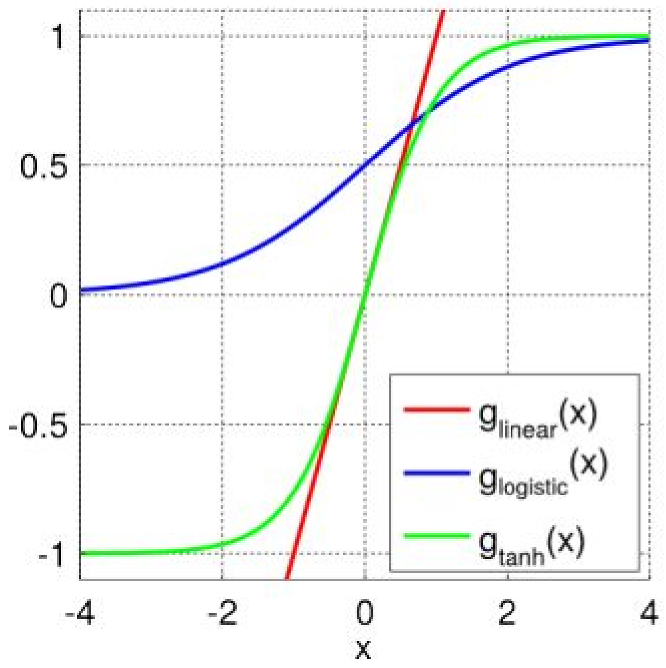


Figure 3. Common artificial neuron activation functions. Notice how $g_{logistic}$ and g_{tanh} change their values quickly near x = 0, and flatten off at larger values of x. Image modified from [3].

The neurons and their connections form a directed graph. A *feedforward neural network* is an ANN where the nodes in that graph are connected in a non-cyclical way. By comparison, a *recurrent neural network* has cycles. All of the networks studied in this thesis are feedforward networks.

K-fold cross validation is a statistical method of validating the *generalisability* of a model - that is, how well the model performs when presented with new data. For this method the dataset that is being studied is divided into k equal-sized sets. In each *validation run*, one of those sets is left out as a *validation fold*. The model is trained on the remaining k-1 folds, and it's accuracy tested on the validation fold. This process is repeated k times, using each fold as a validation fold once. The results can be used to calculate average and a standard deviation.

1.2 Detection-Alignment-Recognition Pipeline

As noted by Huang et al., face recognition can be thought of as part of a Detection-Alignment-Recognition pipeline [4]. Each phase in the pipeline generates the input for the next phase, as pictured in Figure 4.

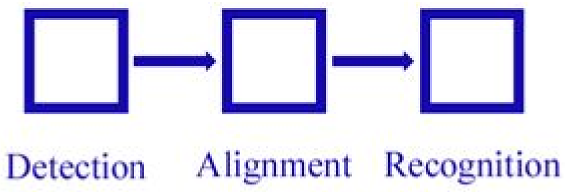


Figure 4. Detection-alignment-recognition pipeline. Each phase generates an input for the next phase. Image from [5]

In detection phase, the positions of faces in an image are located. Usually, the largest face found is considered to belong to the person identified by that image. The rectangular *bounding box* of the largest face found is then cropped from the image. In the alignment phase, the cropped image is transformed to a *canonical pose*, which generally means rotating and translating the image so that the eyes would be level and the nose approximately in the centre of the image. Finally, the recognition phase involves matching the aligned image to either a known identity or comparing two images to guess whether they belong to the same person.

For all images in the datasets used in this study, the detection part was already done and the alignment algorithms were applied optionally; the main interest was comparing the recognition phase.

1.3 Convolutional Neural Networks

A convolutional neural network is a feedforward artificial neural network inspired by the structure of cat's visual cortex, which was studied by Hubel and Wiesel in the 1960s[6].
Seminal work was done in the field of convolutional networks in the 1990s by Lecun et al.[7], after which many of the architectural features of these types of networks have remained similar:

- *Local receptive fields* – each neuron in a layer takes its input from a small area of the previous layer. This allows for smaller features of an image, like edges and corners, to be recognized in the first layers of the network, before combining those into features of higher abstraction in the later layers.
- *Shared weights* – a convolutional layer is a three-dimensional structure organized in planes. All neurons in a plane share the same set of weights. Because the weights are shared, all neurons in a plane perform the same operation on the input from the previous layer, and the output of the neurons form a mapping of features in the previous layer. This aspect is exemplified in Figure 5, where all of arrows of the same color represent the same weight.
- *Max-pooling* or *average-pooling* layers – neurons in a pooling layer take their input values from small (usually 2x2) non-overlapping areas from the previous layer and find the maximum or average of the values. This lessens the dimensions of the layer and provides invariance towards the position of the features in the image [8].

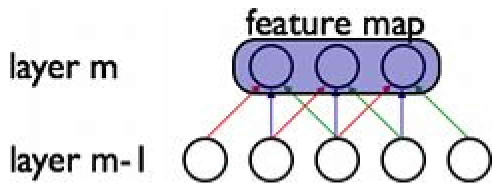


Figure 5. An example of a *convolutional filter*. Arrows of the same color represent equal weights. Image from [9].

2. The Benchmark Data

2.1 The LFW Dataset

Created by Huang et al. in 2007, Labeled Faces in the Wild (LFW) is a database of face images commonly used for reporting the performance of face recognition algorithms. It contains 13233 RGB images of 5749 different individuals (an average of ~2.3 images per person). The image dimensions are 250x250. The creators of the database also defined pairs of images and divided them into sets meant for training face recognition algorithms and reporting their performance [10]. While all of the networks compared in this paper were trained on different, larger databases, the LFW was still used for performance reporting; its wide use makes it well suited for comparing different techniques.

Other versions of the dataset are also available. One of them contains the same images as the original dataset, but the images are modified using an automatic image alignment technique called *deep funneling* [11]. Examples of both original and deep funneled images are presented in Figure 6.

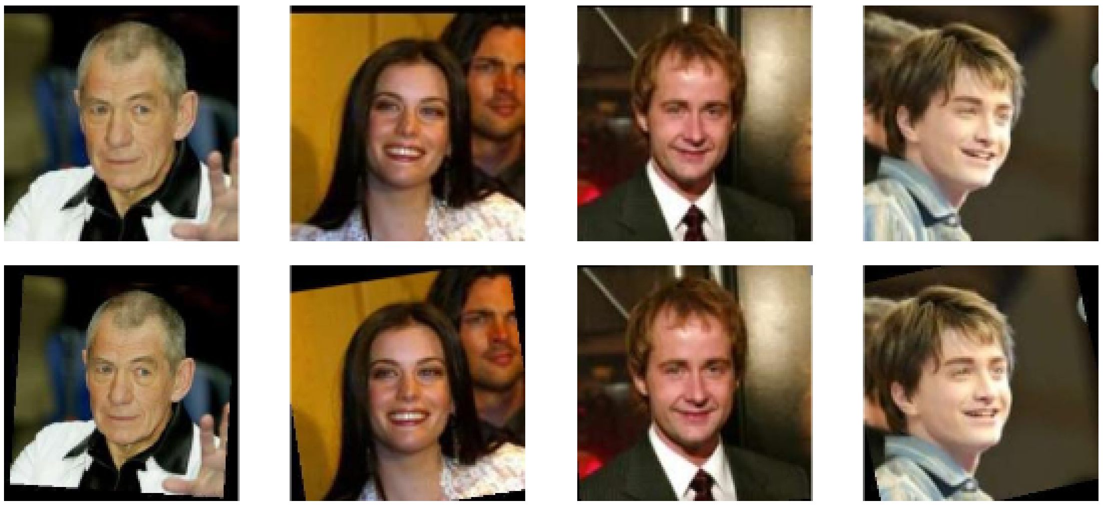


Figure 6. In the top row are the images from the original LFW dataset [10] and on the bottom row the deep funneled versions of the same images[11]. Notice how the funneling has rotated the images.

2.1 The FOTIS Dataset.

The dataset that was used for comparing the performance of the network is gathered from the photo database of The National Archives of Estonia (FOTIS)[1]. It contains 2781 labeled and human-reviewed pictures of 184 individuals (an average of ~15.1 images per person), mostly pictures of Estonian politicians from the 1990s. There are at least 2 images of each person, compared to LFW, where there are many people of whom there is only one image. The images

[1] http://www.ra.ee/fotis/

are black-and-white and of varying dimensions. Examples of some of the images are presented in Figure 7.

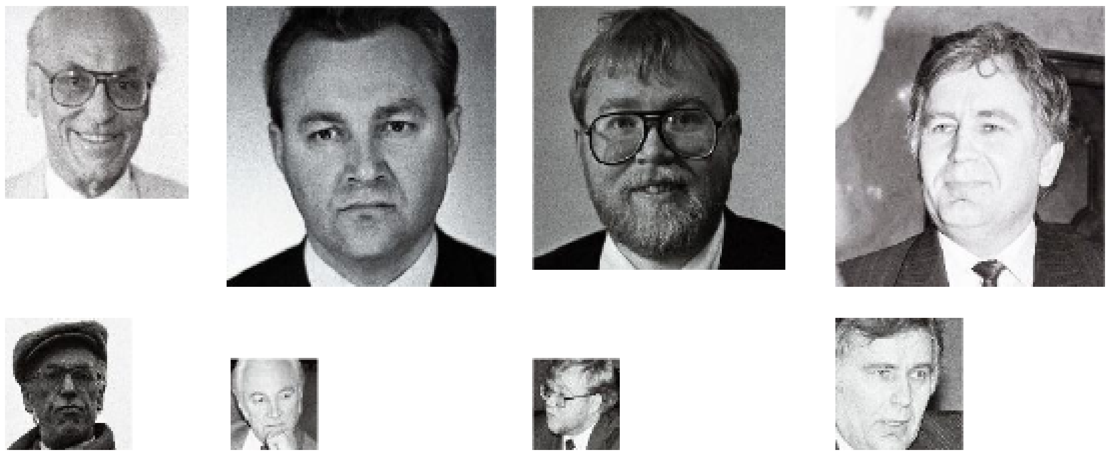


Figure 7. Example images from the FOTIS dataset [2].

[2] http://www.ra.ee/fotis/

3. Comparison of the Networks

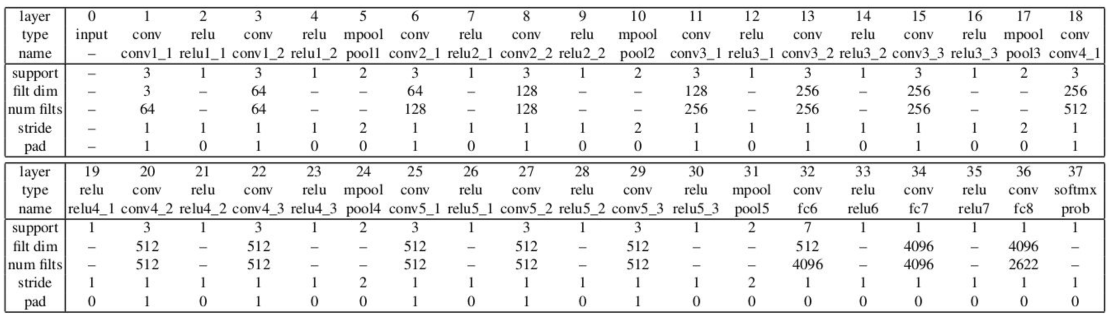
The three networks that were compared are:

- VGG – trained by the researchers of Visual Geometry Group[3] at the University of Oxford
- CASIA – trained by Matiisen and Tampuu at the Institute of Computer Science, University of Tartu[4]. The original version of the network was trained by Dong Yi et al. at the Institute of Automation of the Chinese Academy of Sciences (CASIA)
- Openface – trained by Brandon Amos at Carnegie Mellon University[5]

3.1 Network Descriptions

VGG

The VGG network has 38 layers. A detailed structure is presented in Figure 8. The input is a 224x224 color image. The network was trained as a classifier for recognising 2622 different people and the 2622 outputs in the last fully connected layer fc8 correspond to the classes (different people) [12].



layer	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
type	input	conv	relu	conv	relu	mpool	conv	relu	conv	relu	mpool	conv	relu	conv	relu	conv	relu	mpool	conv
name	–	conv1_1	relu1_1	conv1_2	relu1_2	pool1	conv2_1	relu2_1	conv2_2	relu2_2	pool2	conv3_1	relu3_1	conv3_2	relu3_2	conv3_3	relu3_3	pool3	conv4_1
support	–	3	1	3	1	2	3	1	3	1	2	3	1	3	1	3	1	2	3
filt dim	–	3	–	64	–	–	64	–	128	–	–	128	–	256	–	256	–	–	256
num filts	–	64	–	64	–	–	128	–	128	–	–	256	–	256	–	256	–	–	512
stride	–	1	1	1	1	2	1	1	1	1	2	1	1	1	1	1	1	2	1
pad	–	1	0	1	0	0	1	0	1	0	0	1	0	1	0	1	0	0	1

layer	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37
type	relu	conv	relu	conv	relu	mpool	conv	relu	conv	relu	conv	relu	mpool	conv	relu	conv	relu	conv	softmx
name	relu4_1	conv4_2	relu4_2	conv4_3	relu4_3	pool4	conv5_1	relu5_1	conv5_2	relu5_2	conv5_3	relu5_3	pool5	fc6	relu6	fc7	relu7	fc8	prob
support	1	3	1	3	1	2	3	1	3	1	3	1	2	7	1	1	1	1	1
filt dim	–	512	–	512	–	–	512	–	512	–	512	–	–	512	–	4096	–	4096	–
num filts	–	512	–	512	–	–	512	–	512	–	512	–	–	4096	–	4096	–	2622	–
stride	1	1	1	1	1	2	1	1	1	1	1	1	2	1	1	1	1	1	1
pad	0	1	0	1	0	0	1	0	1	0	1	0	0	0	0	0	0	0	0

Figure 8. The VGG network architecture [13].

The network was trained on a set of 982 803 images, which the researchers gathered using the following process [12]:

1. a list of celebrities was extracted from the Internet Movie Database (IMDb),
2. more images of them were collected using Google Image Search,
3. pictures of all people already present in the Labeled Faces in the Wild and Youtube Faces dataset were removed.
4. Two steps of both manual filtering and automatic filtering were done to improve cluster purity and remove duplicates.

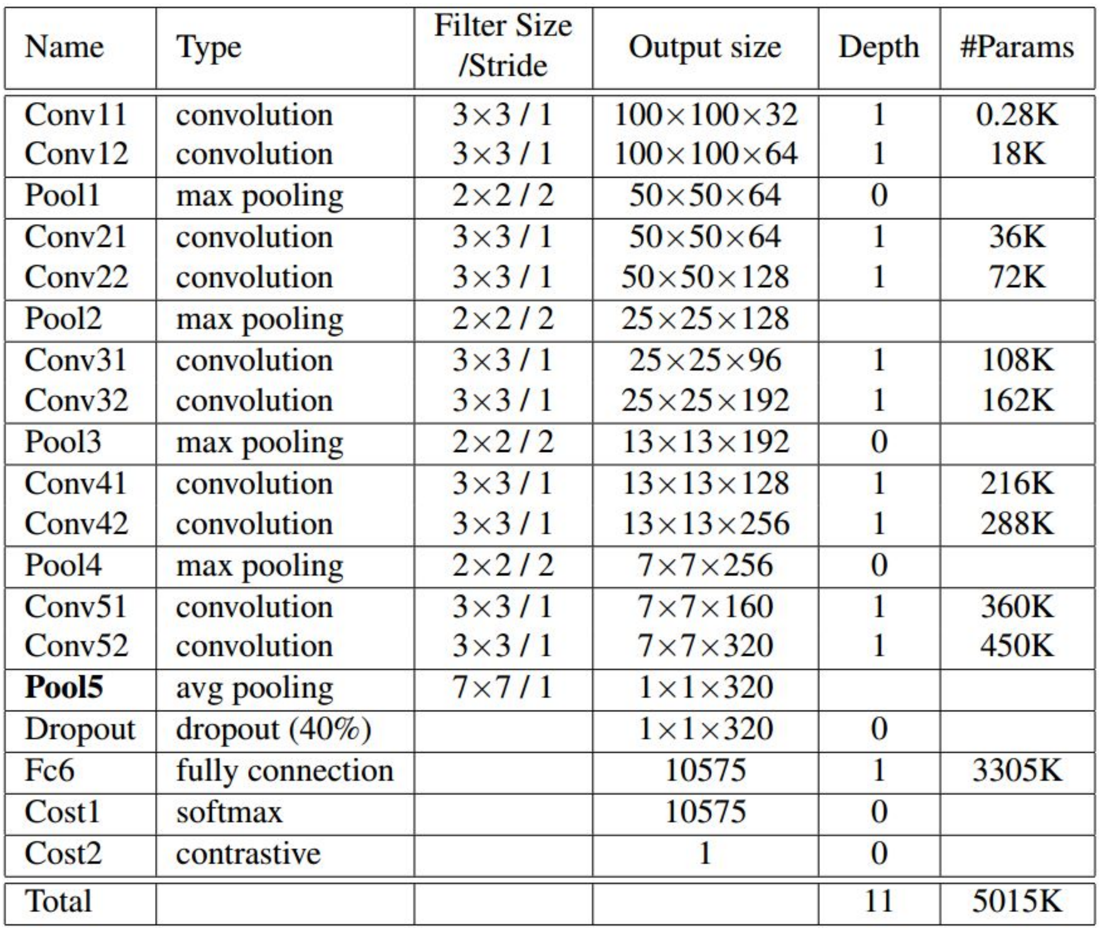
[3] http://www.robots.ox.ac.uk/~vgg/
[4] http://neuro.cs.ut.ee/
[5] https://cmusatyalab.github.io/openface/

CASIA

The CASIA network has five groups of two convolutional layers, the groups are separated by pooling layers. The first four pooling layers use max operator, the last pooling layer uses average pooling. ReLU neurons are used after each convolutional layer but the last, Conv52. The structure is presented in Figure 9, it does not include the ReLU neurons. The input to the network is a 100x100 grayscale image. The output feature vector has a length of 320 [14]. The original network trained by Yi et al. used both softmax and contrastive loss. In the training done in the University of Tartu only softmax loss was used.

During the tests done for this thesis, the 24 layers up to and including dropout layer were used, the dropout layer being the one from where the face features were extracted. Similarly to VGG, the last layers were used by [14] during training only to classify the training images.

The CASIA network was trained on the CASIA-Webface dataset, which contains 494 414 images [14].

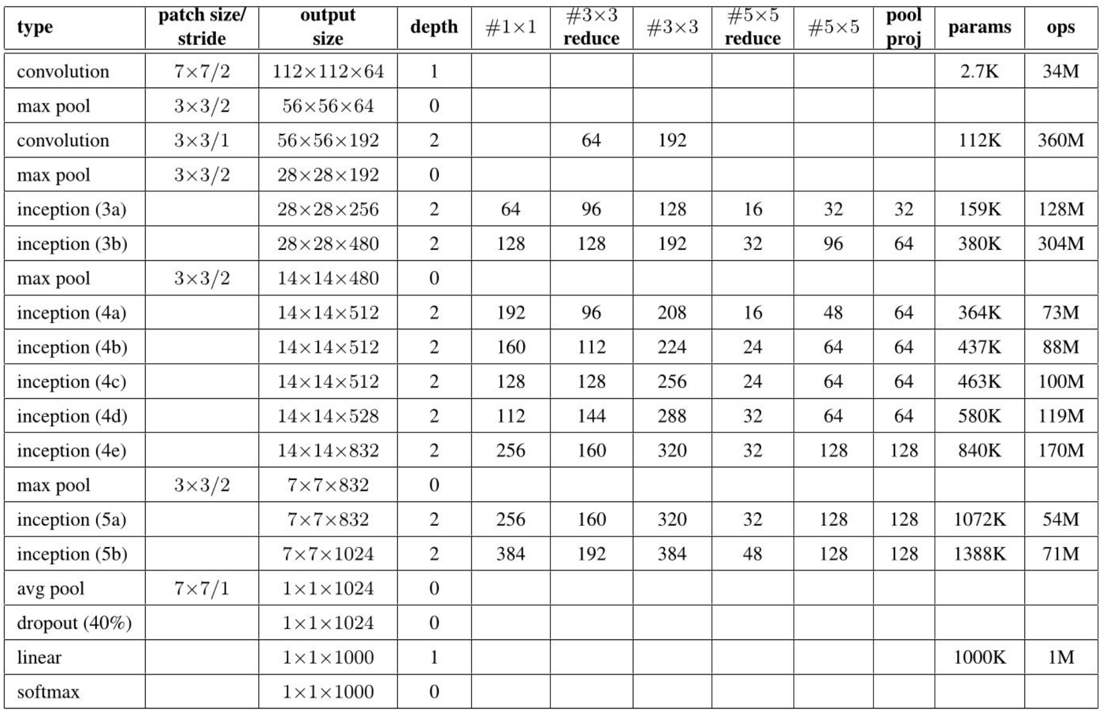


Name	Type	Filter Size /Stride	Output size	Depth	#Params
Conv11	convolution	3×3 / 1	100×100×32	1	0.28K
Conv12	convolution	3×3 / 1	100×100×64	1	18K
Pool1	max pooling	2×2 / 2	50×50×64	0	
Conv21	convolution	3×3 / 1	50×50×64	1	36K
Conv22	convolution	3×3 / 1	50×50×128	1	72K
Pool2	max pooling	2×2 / 2	25×25×128		
Conv31	convolution	3×3 / 1	25×25×96	1	108K
Conv32	convolution	3×3 / 1	25×25×192	1	162K
Pool3	max pooling	2×2 / 2	13×13×192	0	
Conv41	convolution	3×3 / 1	13×13×128	1	216K
Conv42	convolution	3×3 / 1	13×13×256	1	288K
Pool4	max pooling	2×2 / 2	7×7×256	0	
Conv51	convolution	3×3 / 1	7×7×160	1	360K
Conv52	convolution	3×3 / 1	7×7×320	1	450K
Pool5	avg pooling	7×7 / 1	1×1×320		
Dropout	dropout (40%)		1×1×320	0	
Fc6	fully connection		10575	1	3305K
Cost1	softmax		10575	0	
Cost2	contrastive		1	0	
Total				11	5015K

Figure 9. The CASIA network architecture [14].

Openface

The Openface network implementation by Brandon Amos et al. [15] is inspired by a network developed by Google researchers called the FaceNet [16]. There is ongoing development on the Openface implementation, in tests run for this thesis the latest version named nn4.small2 was used, which was released in January 2016 [15]. The structure of the FaceNet architecture is presented in Figure 10. The nn4.small2 version uses a similar structure to the FaceNet architecture, but with 4b, 4c and 4d layers removed and has smaller 5a and 5b layers. It contains a mix of regular convolutional layers, max pooling layers and inception layers. Inception layers are complex convolutional layers that contain parallel filters of different size, in-depth description of them is available in the original paper[17].



type	patch size/ stride	output size	depth	#1×1	#3×3 reduce	#3×3	#5×5 reduce	#5×5	pool proj	params	ops
convolution	7×7/2	112×112×64	1							2.7K	34M
max pool	3×3/2	56×56×64	0								
convolution	3×3/1	56×56×192	2		64	192				112K	360M
max pool	3×3/2	28×28×192	0								
inception (3a)		28×28×256	2	64	96	128	16	32	32	159K	128M
inception (3b)		28×28×480	2	128	128	192	32	96	64	380K	304M
max pool	3×3/2	14×14×480	0								
inception (4a)		14×14×512	2	192	96	208	16	48	64	364K	73M
inception (4b)		14×14×512	2	160	112	224	24	64	64	437K	88M
inception (4c)		14×14×512	2	128	128	256	24	64	64	463K	100M
inception (4d)		14×14×528	2	112	144	288	32	64	64	580K	119M
inception (4e)		14×14×832	2	256	160	320	32	128	128	840K	170M
max pool	3×3/2	7×7×832	0								
inception (5a)		7×7×832	2	256	160	320	32	128	128	1072K	54M
inception (5b)		7×7×1024	2	384	192	384	48	128	128	1388K	71M
avg pool	7×7/1	1×1×1024	0								
dropout (40%)		1×1×1024	0								
linear		1×1×1000	1							1000K	1M
softmax		1×1×1000	0								

Figure 10. The FaceNet network architecture [16].

The OpenFace network was trained on a combination of FaceScrub and CASIA-WebFace datasets [18]. The FaceScrub dataset contains 106 863 images [19] and the CASIA-WebFace dataset contains 494 414 images [14].

3.2 Image Processing and Alignment

For each of the networks, a different method was used by the groups that trained them for processing and aligning the image before passing it to the network.

The VGG team used a process called oversampling, whereby they cropped a part from each corner and from the middle of the image being processed. They took the same crops from horizontal mirror version of the image, passed the resulting 10 patches through the network (one by one), and averaged the resulting feature vectors. They also used 2-D alignment, which gave a slight improvement in accuracy if used at testing time [12].

For CASIA network, the images were only mirrored during training before being passed to the network.

For Openface, dlib[6] image processing library was used to align the images so that the eyes and nose would appear approximately the same place in the image[20].

[6] http://dlib.net/

4. Methodology

The experimental work done for this thesis included extracting features for each three of the networks and for both LFW and FOTIS datasets, producing the benchmark dataset for the FOTIS database, and analysing the classification accuracy of each dataset/network combination.

Some preliminary work for the LFW dataset with both CASIA and VGG networks was already done in the Face Kiosk project implemented by Tambet Matiisen [21]. The author of this paper added support for Openface network and FOTIS dataset, created scripts for pair generation and accuracy calculation and evaluated the results.

4.1 Experimental Setup

The tests were run on a Linux server with a Python programming language and Caffe installation. Caffe[7] is a deep learning network implemented in Python. The models of VGG and CASIA were implemented in Caffe. Openface is implemented in Torch deep learning toolkit, which is based on Lua programming language. For testing Openface, all of its requirements, including Torch needed to be installed. The exact requirements are available at Openface webpage[8].

4.2 Image processing

While in general, good face alignment is considered important for the recognition algorithms, the alignment algorithms were applied optionally during the tests in this study. Firstly, because the previous results and the preliminary tests showed great difference in sensitivity to the alignment between the different networks. Secondly, each of the teams that trained the networks used different algorithms for alignment. All of those were difficult to control for and the main interest of this thesis lied in the recognition part of the detection-alignment-recognition pipeline. As a compromise, to provide an experimental control for image alignment, a separate set of tests with each network were run on the deep funneled LFW dataset[11].

4.3 Producing the Benchmark for FOTIS Dataset

The methodology used for testing on the LFW dataset was also adapted for FOTIS. The methodology calculates accuracy of predicting if pair of faces are of the same person or not. For generating pairs from the FOTIS dataset, a Python script was written implementing the same algorithm that was used for LFW pair generation [10]. The description is outlined below.

[7] http://caffe.berkeleyvision.org/

[8] https://cmusatyalab.github.io/openface/setup/

To form matching pairs, a person was chosen at random from all of the people in the set. If there were two or more images of that person, a pair of those were chosen at random. If that pair was already present in the set of matched pairs, the selection process was started again, otherwise, the pair was added to the set.

To form mismatched pairs, two people were picked with uniform probability – people with different number of images had the same chance of being picked. A random image of each of those people was picked. If the resulting pair was already in the set of mismatched pairs, the process was started again, otherwise, the pair was added to the set.

The tests in this thesis were run with 2500 matching and 2500 mismatching pairs created from the FOTIS dataset. The number was chosen after calculating the standard deviation of the cross-validation test runs with different numbers of pairs, and choosing the number of pairs over which there was no noticeable decrease in the standard deviation. The result of those tests are presented in section 5.2

4.4 Calculating the Accuracy

The extracted face features in the VGG, CASIA, and Openface network are represented by vectors of n real numbers, n being equal to 4096, 320, or 128 respectively. These vectors can be thought of as points in an n-dimensional *Euclidean space.* A way of comparing 2 such vectors is calculating the *Euclidean distance* between them, which is calculated as follows: for $A = (a_1, a_2, ..., a_n)$ and $B = (b_1, b_2, ..., b_n)$,

$$distance(A,B) = \sqrt{(a_1 - b_1)^2 + (a_2 - b_2)^2 + ... + (a_n - b_n)^2}$$

(which is a generalization of the Pythagorean theorem).

For calculating the accuracy:

1. a list of face pairs was loaded, either the ones provided with the LFW dataset or the ones generated from FOTIS (as described in chapter 4.3.)
2. the Euclidean distances between the members of each of the pairs were calculated.
3. The resulting distances and known labels – whether the pair contained the images of the same person or not – were run through a ten-fold cross-validation process, where in each validation run, an optimal threshold distance was found on the training folds. This was found by iterating over 100 equidistant values between the minimum and maximum of the Euclidean distances calculated in the last step and calculating the binary classification accuracy on the training folds. The threshold that resulted in the maximal classification accuracy was recorded.
4. The optimal thresholds found on the training folds were used to calculate the classification accuracy on each corresponding validation fold and an average was taken of the results. These results are presented in section 5.1.

5. Results

5.1 Accuracy of the Networks

The classification accuracies of the different networks are presented in Table 1, with the original results added for comparison.

Table 1. Accuracies in tests run by the author and reported accuracies from the creators of the networks for comparison.

	Reported accuracy (LFW)	Tested accuracy (LFW)	Tested accuracy (LFW with deep funneling)	Tested accuracy (FOTIS)
VGG	97.27% [12]	97.13%	95.02%	91.18%
CASIA	96.13%[1] [14]	92.20%	91.93%	80.74%
Openface	92.9% [20]	86.88%[2]	58.68%	73.11%[2]

[1] this is the accuracy of the original CASIA network, which used both softmax and contrastive loss during training. The network that was tested used only sofmax loss.

[2] used the face alignment method suggested in Openface samples.

Only VGG network's result could be reproduced on the LFW dataset to a close margin. That network also showed the most promising accuracy of 92.12% on the FOTIS dataset.

For Openface, the discrepancy in the results is probably caused by an error in either the configuration of the network or the setup and configuration of the required software – because no paper has been published for the Openface, the information for the setup had to be gathered from sample scripts and separate articles published on the Openface web page, which increases the chance of human error. Also note, that for LFW and FOTIS runs of the Openface tests, the images were aligned before being passed to the network. A test without the alignment was also run, but showed a very unpromising accuracy of 57.50% on the LFW.

For CASIA, the difference in the results can probably be attributed to the missing contrastive loss function in training the network.

5.2 Standard deviation of test runs

The LFW dataset contains 13 233 pictures in total, from which 6000 test pairs were generated and published. But no reasoning for the selection of that number of pairs was given[10]. After both the pair generation and accuracy calculation scripts were created, it became simple to generate different number of pairs, and test how the variation in the cross validation runs changes if different numbers of pairs are used. Results of runs with 1000, 2000, 5000 and 10000 pairs of FOTIS images with VGG network are presented in Figure 11.

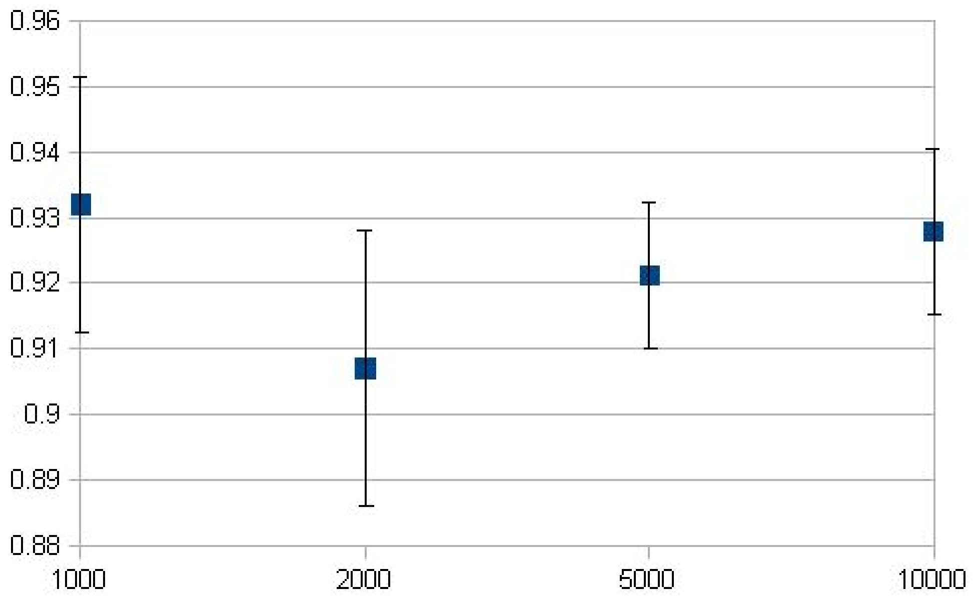


Figure 11. Average accuracies and standard deviations with different numbers of pairs.

The standard deviation dropped noticeably between the tests with 2000 and 5000 pairs, and increased slightly with 10 000 pairs. After these results the rest of the tests with FOTIS dataset were decided to be run with 5000 pairs, because no decrease in standard deviation was seen with higher number of pairs.

6. Conclusions

The VGG network performed the best in the tests, both when trying to repeat the original results as well as on the FOTIS dataset. Not only was the accuracy on the LFW dataset the greatest, it was also closest to the results published by the authors of the network, which adds confidence that the experiment was set up correctly. The VGG network's accuracy of 91.18% on the FOTIS dataset is promising, considering the lower quality of the archive images; and it is not much lower than the accuracy published by Openface on a presumably easier LFW dataset. The author suggests to consider the VGG network as a starting point for further work on the FOTIS dataset, be it clustering the data or tuning the network to gain better accuracy. The Openface network should not be ruled out either - the project is still in development, and the latest published result showed a very large improvement in accuracy from 76.1% to 92.9%. The result on the CASIA network could also be possibly improved my using the contrastive loss metric during training.

An interesting observation is the difference in sensitivity to image alignment the different networks have. While the VGG team reported only a marginal degradation in performance if image alignment was not used, in the tests performed with Openface the difference was far from marginal - the accuracy dropped from 86.88% to 57.50%, if the alignment was skipped (a reminder - on a balanced dataset of equal number of matching and mismatching pairs, a random classifier would get a 50% accuracy on average).

All three of the networks used a different training data set, different method of face alignment and had different network architecture. Even combining only those three aspects results in 27 ways to train a network. That makes meta-analysis and controlling for different aspects of the experiments difficult. The author would like to see more studies with stricter experimental controls published, where other independent variables are left constant and only changes to the network architecture would be made. That could help to move the theoretical study of artificial neural network architectures forward.

References

1. File:Colored neural network.svg - Wikimedia Commons [Internet]. [cited 12 May 2016]. Available: https://commons.wikimedia.org/wiki/File:Colored_neural_network.svg

2. 2.3.3 Artificial Neuron with Continuous Characteristics [Internet]. [cited 11 May 2016]. Available: http://www.ece.utep.edu/research/webfuzzy/docs/kk-thesis/kk-thesis-html/node14.html

3. dustinstansbury. Derivation: Derivatives for Common Neural Network Activation Functions. In: The Clever Machine [Internet]. 9 Sep 2014 [cited 12 May 2016]. Available: https://theclevermachine.wordpress.com/2014/09/08/derivation-derivatives-for-common-neural-network-activation-functions/

4. Huang GB, Jain V, Learned-Miller E. Unsupervised Joint Alignment of Complex Images. 2007 IEEE 11th International Conference on Computer Vision. 2007. pp. 1–8. doi:10.1109/ICCV.2007.4408858

5. Vision Lab : Face Alignment [Internet]. [cited 12 May 2016]. Available: http://vis-www.cs.umass.edu/faceAlignment/

6. Hubel DH, Wiesel TN. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. J Physiol. 1962;160: 106–154. doi:10.1113/jphysiol.1962.sp006837

7. Lecun Y, Bottou L, Bengio Y, Haffner P. Gradient-based learning applied to document recognition. Proc IEEE. 1998;86: 2278–2324. doi:10.1109/5.726791

8. Lee H, Grosse R, Ranganath R, Ng AY. Convolutional Deep Belief Networks for Scalable Unsupervised Learning of Hierarchical Representations. Proceedings of the 26th Annual International Conference on Machine Learning. New York, NY, USA: ACM; 2009. pp. 609–616. doi:10.1145/1553374.1553453

9. Convolutional Neural Networks (LeNet) — DeepLearning 0.1 documentation [Internet]. [cited 3 May 2016]. Available: http://deeplearning.net/tutorial/lenet.html

10. Huang GB, Ramesh M, Berg T, Learned-Miller E. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. cs.brown.edu; 2007; Available: http://cs.brown.edu/courses/cs143/2011/proj4/papers/lfw.pdf

11. Huang GB, Mattar M, Lee H, Learned-Miller E. Learning to Align from Scratch. NIPS. 2012.

12. Parkhi OM, Andrea V, Andrew Z. Deep Face Recognition. Procedings of the British Machine Vision Conference 2015. 2015. doi:10.5244/c.29.41

13. Parkhi OM, Vedaldi A, Zisserman A. Deep face recognition. Proceedings of the British Machine Vision. 2015;1: 6.

14. Yi D, Lei Z, Liao S, Li SZ. Learning Face Representation from Scratch [Internet]. arXiv [cs.CV]. 2014. Available: http://arxiv.org/abs/1411.7923

15. Amos B, Ludwiczuk B, Harkes J, Pillai P, Elgazzar K, Satyanarayanan M. OpenFace: Face Recognition with Deep Neural Networks [Internet]. [cited 11 Jan 2016]. Available:

http://github.com/cmusatyalab/openface

16. Schroff F, Kalenichenko D, Philbin J. FaceNet: A Unified Embedding for Face Recognition and Clustering [Internet]. arXiv [cs.CV]. 2015. Available: http://arxiv.org/abs/1503.03832

17. Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, et al. Going Deeper with Convolutions [Internet]. 2014. Available: http://arxiv.org/abs/1409.4842

18. Models and Accuracies - OpenFace [Internet]. [cited 27 Apr 2016]. Available: http://cmusatyalab.github.io/openface/models-and-accuracies/

19. Winkler S. vintage - resources [Internet]. [cited 27 Apr 2016]. Available: http://vintage.winklerbros.net/facescrub.html

20. Amos B. OpenFace 0.2.0: Higher accuracy and halved execution time [Internet]. [cited 10 May 2016]. Available: http://bamos.github.io/2016/01/19/openface-0.2.0/

21. tambetm. tambetm/face_kiosk. In: GitHub [Internet]. [cited 8 May 2016]. Available: https://github.com/tambetm/face_kiosk

Appendice

The Linux shell scripts and Python scripts for face feature extraction, pair generation and accuracy calculation will be made available in the University of Tartu Institute of Computer Science Graduation Theses Registry at https://comserv.cs.ut.ee/ati_thesis/index.php?year=2016

There will be a README.txt available detailing the requirements of the setup and the scripts themselves.