Detection of Face Morphing Attacks by Deep Learning

Clemens Seibold[1], Wojciech Samek[1], Anna Hilsmann[1] and Peter Eisert[1,2]

[1] Fraunhofer HHI, Einsteinufer 37, 10587 Berlin, Germany
[2] Humboldt University Berlin, Unter den Linden 6, 10099 Berlin, Germany

Abstract. Identification by biometric features has become more popular in the last decade. High quality video and fingerprint sensors have become less expensive and are nowadays standard components in many mobile devices. Thus, many devices can be unlocked via fingerprint or face verification. The state of the art accuracy of biometric facial recognition systems prompted even systems that need high security standards like border control at airports to rely on biometric systems. While most biometric facial recognition systems perform quite accurate under a controlled environment, they can easily be tricked by morphing attacks. The concept of a morphing attack is to create one synthetic face image that contains characteristics of two different individuals and to use this image on a document or as reference image in a database. Using this image for authentication, a biometric facial recognition system accepts both individuals. In this paper, we propose a morphing attack detection approach based on convolutional neural networks. We present an automatic morphing pipeline to generate morphing attacks, train neural networks based on this data and analyze their accuracy. The accuracy of different well-known network architectures are compared and the advantage of using pretrained networks compared to networks learned from scratch is studied.

Keywords: automatic face morphing - face image forgery detection - convolutional neural networks - morphing attack

1 Introduction

Biometric facial recognition systems are nowadays present in many areas of daily life. They are used to identify people, find pictures of the same person in your digital image collection, to make suggestions for tagging people in social media or for verification tasks like unlocking a phone or a computer. Apart from these consumer market applications, biometric facial recognition systems found also their way into sovereign tasks like automatic border control at airports. In particular, for these tasks the verification system has to be reliable and secure.

Even though biometric facial recognition systems achieve false rejection rates below 1% at a false acceptance rate of 0.1% in a controlled environment [1], they can easily be tricked by a specific attack, known as morphing attack [2]. The concept of this attack is to create a synthetic face image that is, in the eye of the recognition system, similar to the faces of two different individuals. Thus, both can use the same synthetic face image for authentication. This attack is usually performed by creating a face image that contains characteristics of both faces, for example by face morphing. Since such an attack has a drastic impact on the authenticity of its underlying system, its automatic detection is of outmost importance.

The detection of attacks by image manipulation has been studied for various tasks. Several publications deal with the detection of resampling artifacts [3, 4], the use of specific filters, e.g. median

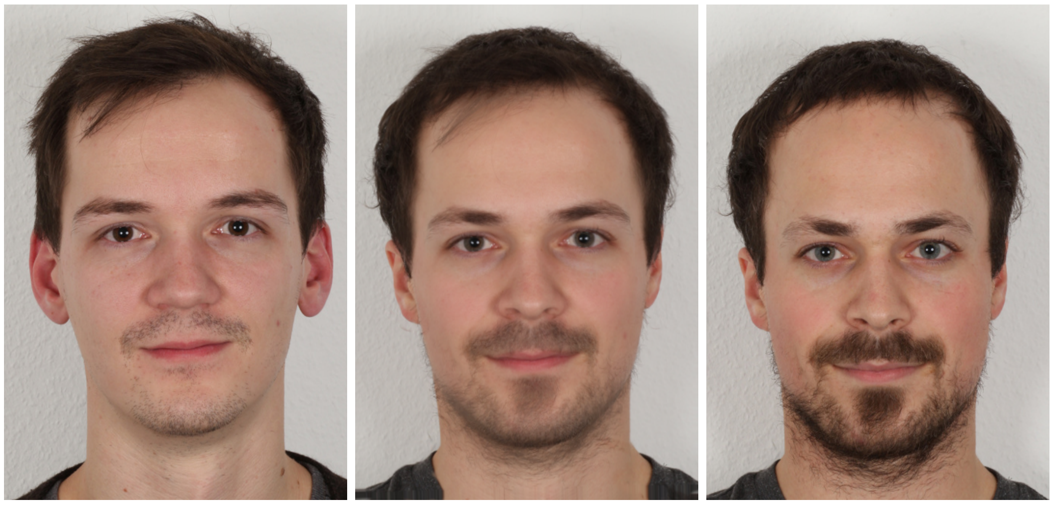


Fig. 1. Morphing attack (left, right: original images, center: morphing attack)

filters [5], or JPEG-double compression [6, 7]. Beside these analyses based on signal theory, several authors proposed image forgery detection methods based on semantic image content by analyzing reflections and shadows [8, 9]. Recently, the detection and analysis of forged face images with the purpose of tricking a facial recognition system became interesting to many researchers, which were certainly influenced by the publication of a manual for morphing attacks from Ferrara et al. [2]. Makrushin et al. [10] proposed an approach for automatic generation of facial morphs and their detection based on the distribution of Benford features extracted from quantized DCT coefficients of JPEG-compressed morphs. Raghavendra et al. [11] presented a morphing detection method based on binary statistical images features and evaluated its accuracy on manually created morphed face images.

In this work, we analyze the practical usability of deep neural networks (DNNs) for detection of morphed face images (morphing attacks). DNNs are state of the art image classification methods, since the DNN AlexNet [12] broke in 2012 the record error rate of 25.7% in the object recognition challenge (ILSVRC2012) and reduced it to 16.4%. This DNN architecture has five convolutional layers and ends with three fully connected layers. Pursuing the concept of "simple but deep", Simonya and Zisserman [13] showed in 2015 that a simple architecture concept (VGG19) with only 3x3 convolutional filters, but with a depth of 16 convolutional layers also performs as well as state of the art architectures with error rates for object classification of about 7%. Szegedy et al. [14] moved away from the concept of a simple architecture and introduced the GoogLeNet with a complex structure containing inception layers. An inception layer consists of several convolutional filters that operate in parallel and whose outputs are concatenated. On one hand, the networks needs, due to this structure, less parameters to describe even complex features and is thus less prone to overfitting, but on the other hand, the learned features are more difficult to interpret.

We focus on the accuracy analysis of the three architectures named above, since all of these architectures have successfully been used for the task of image classification and pretrained models are publicly available. Similar to the task of object classification, we do not want to detect low-level artifacts, e.g. resampling, median filtering or blurring artifacts like Bayar and Stamm did [15] using a different CNN architecture for image forgery detection. Instead, we want our DNNs to decide based on semantic features like unrealistic eyes forms, specular highlights or other semantic artifacts caused by the morphing process.

Since a huge amount of data is needed for the training of DNNs, we designed a fully automatic morphing algorithm that takes two 2D face images, aligns the facial features, such that they are at the same position, blends the whole aligned images and gets rid of ghosting artifacts due to different hairstyles, ear geometry etc. in a post-processing step that fits the blended face into a warped source image. Some components, like the face alignment, are exchangeable due to the pipeline structure of our morphing algorithm. Thus, we can use two different warping methods for the alignment and get more diversity in our training data. To overcome the problem of learning low-level features, like resampling artifacts, all images - original face images and morphing attacks - are preprocessed in several ways.

In section 2, we describe our fully automatic morphing algorithm with exchangeable components which is used to create a database of morphed images. The preprocessing steps, which are independent of the morphing process, and our database of original face images and morphing attacks are presented in section 3. Our morphing attack detection experiments are shown in section 4. Finally, the results are presented and discussed.

2 Automatized Morphing

In this section, we introduce our fully automatic morphing algorithm. We start by introducing the general steps of morphing algorithms and finally describe our specific algorithm.

2.1 Overview on a general morphing pipeline

To trick a facial recognition system to match two different persons with one synthetic reference image, the synthetic image has to contain characteristics from both faces. If the feature space of the recognition system is known, the characteristics can be directly combined in that space. Since commercial face recognition system are often a black box and an attacker would not rely on the presence of a specific system, we adapt the facial characteristics in image space. The usual process of face image morphing is divided into four to five steps:

1. Locate facial landmarks
2. Align images by image warping such that the detected facial landmarks are at nearly the same position
3. Blend the images, e.g. additive blending with a blending factor of 0.5
4. Cut out the inner part of the face from the blended image and put it in one warped original images with an optimal inconspicuous cutting path

Some morphing algorithms cut the inner part of the face and insert it into one original not warped face image. They apply an inverse morphing on the blended image and also need to handle the border between blended and original image.

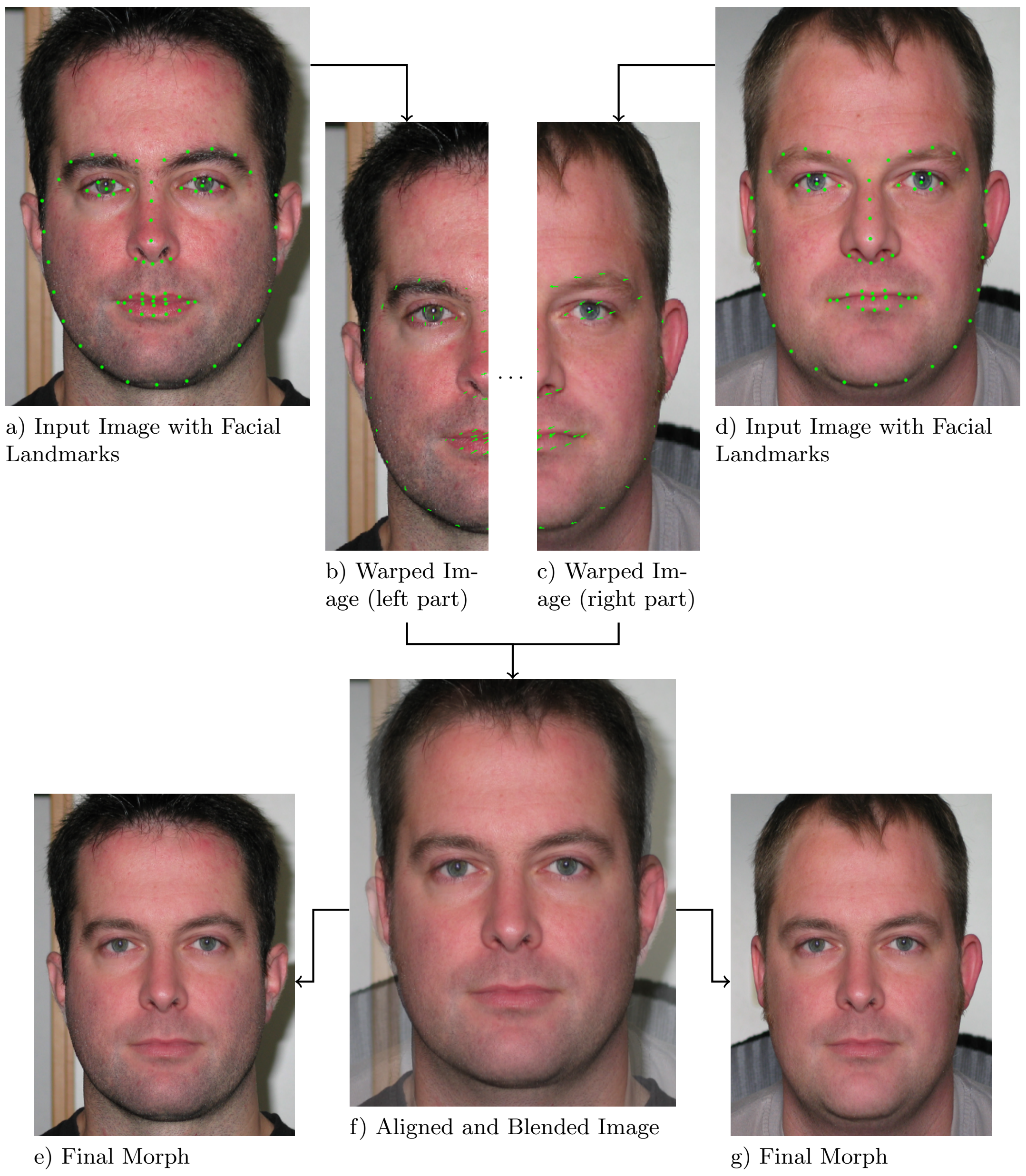

a) Input Image with Facial Landmarks
b) Warped Image (left part)
. . .
c) Warped Image (right part)
d) Input Image with Facial Landmarks
e) Final Morph
f) Aligned and Blended Image
g) Final Morph

Fig. 2. Morphing pipeline

2.2 Morphing pipeline implementation

Figure 2 illustrates our face image morphing pipeline. In the following we describe our implementation of the single steps in detail.

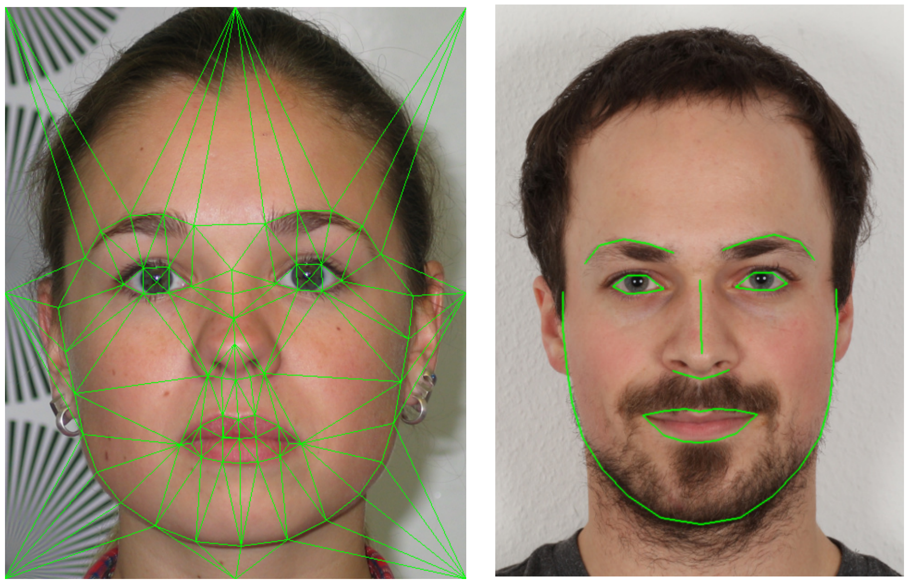
Landmark detection Sixty-eight facial landmarks are located using the dLib's [16] implementation of [17]. These landmarks are located around the lower half of the head's silhouette, around mouth, eyes, nose and eyebrows, see also figures 2a,d.

Image alignment We implemented two different warping methods to align the images. One is based on a transformation of a triangle mesh and the other one uses the Beier-Neely morphing algorithm [18]. In both cases, the transformation is performed such that both faces are warped to an average geometry of both faces. The exact definition of the average geometry depends on the warping method, but both methods rely on average positions of the facial landmarks. These are calculated by averaging pairwise the location of the facial landmarks in the original images.

a. *Triangle warping* We first add eight additional control points to the detected 68 facial landmark locations in each image to be able to morph also the region outside of the convex hull of the detected points. They are located in the four corners and on the middle of the four lines of a bounding box around the head. These eight control points are also subject to the averaging of facial landmark positions as described above. Following, the average positions are used to create a triangle mesh using the Delaunay [19] triangulation, see also figure 3a. To create the piecewise affine warped version of the first source image, for each pixel in the target image, we calculate the barycentric coordinates for the triangle enclosing the point and use them to calculate the color by bilinear interpolation in the source image. This corresponds to rendering the triangle mesh using the common rendering pipeline with the average positions of the landmarks as coordinates, the landmarks positions of a source image as texture coordinates and the source image as texture map. The other image is warped accordingly.

b. *Field morphing* In contrast to the triangle warping, the field morphing [18] needs no corresponding feature points, but corresponding feature line segments. To get corresponding feature line segments, we use the estimated landmark positions and connect them according to a predefined pattern, see figure 3b. The basic idea of the field morphing is to move every pixel according to the movement of each line segment weighted by the distance to them. Assuming there is only one line segment Q in the source image with end points Q_s and Q_e, one corresponding line Q' in the destination image and a given position p_w in the warped image, the corresponding position p_s in the source image can be calculated as follows. First, the position of the point p_w is calculated relative to the line in the destination image. This is done by calculating the distance d from p_w to a line that is defined by extending the line segment Q' to an infinite length. The distance d is signed, such that it is positive, if the normal of the line segment Q' points towards p_w and otherwise negative. Given the closest point s on this line to p_w, we parameterize the position of this intersection point by p_w using the start and end point of the line segment:

$$u = \frac{s - Q'_s}{\|Q'_e - Q'_s\|} \tag{1}$$



a) Triangle mesh warping b) Feature lines the field morphing

Fig. 3. Initialization for Face Warping

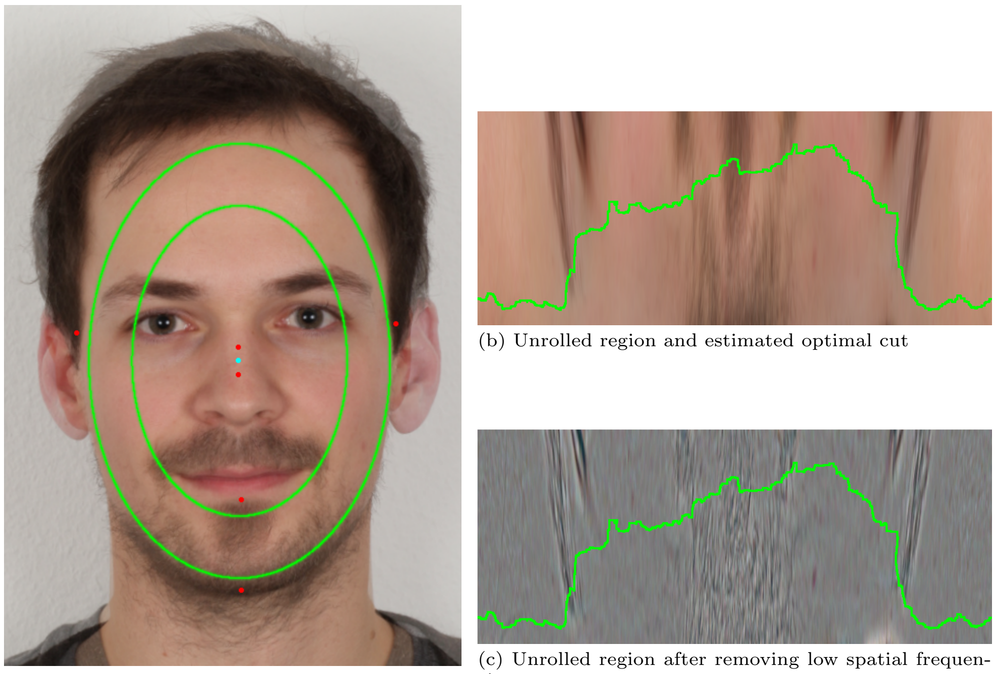
The point p_w can now be described relative to the line segment by u, d and the parameters of the line segment. The corresponding position in the source image is defined relative to the line in the source image as

$$u \cdot \|Q_e - Q_s\| + Q_s + Q_n \cdot d, \tag{2}$$

where Q_n is the normal of the line segment Q. The multi-line case is straightforward and can be done for every pixel individually: For each line segment, the motion of a pixel is calculated as described above and weighted according to its distance to the line segment with the weight decreasing nonlinear with the distance to a line segment. These motions and weights are added up and finally the motion is divided by the sum of all weights at this pixel. For further details see [18].

Image blending Our blending is performed by an additive alpha blending with a blending factor of 0.5 for both warped images. More complex blending methods, like spatially invariant blending factors based of the spatial frequencies or selected features might be useful to retain facial characteristics, e.g. moles and freckles, and will be part of our future work.

Warped/blended image fusion The warped and blended images are already usable for tricking a biometric facial recognition system but would be rejected by every human due to the ghosting artifacts around the borders of the heads, see also figure 2f. To get rid of these artifacts, we use the blended image only for the inner part of the synthetic face image, the rest of the image is taken from one of the warped source image. As a consequence of combining these two images, we have to deal with the border between the blended part of the face and the background. In order to hide this border to the human eye and morphing detection systems, we calculate a subtly transition between foreground and background. Two ellipses define a transition zone between foreground and background. They are located such that the outer ellipse is barely on the chin and underneath the hairline and the inner ellipse surrounds all detected facial landmarks. Figure 4a shows the ellipses in one blended image. The cyan point denotes the center of the ellipse and is defined relative to the two red marked facial landmaks on the nose. The width is defined relative to the X-distance from the center to the position of the red marked facial landmarks next to the ears and the height relative to the Y-distance from the center to the red marked facial landmarks below the mouth and at the chin. The border or transition in this area is calculated separately for high and low spatial frequencies. The transition for low frequencies is calculated using Poisson Image Editing [20], while an optimal cutting path through the transition zone is searched for the high frequency details. For that purpose,the area defined by the two ellipses is unrolled from Cartesian to polar coordinates [21]. Thus, the column in the converted image defines the angle and the row defines the radius, see also figure 4a,b.


(a) Region for cutting edge estimation

(b) Unrolled region and estimated optimal cut

(c) Unrolled region after removing low spatial frequencies

Fig. 4. Estimation of cutting edge for the high spatial frequencies

On these images, we calculate an optimal path using the graph cut algorithm for image segmentation. The top row and the bottom row are defined as foreground and background, and the rest as unknown component. In contrast to the classical image segmentation, we are not looking for strong borders between segments, but for a path through the images that passes areas that are similar in both images to avoid artifacts along the cut. The cost function for two neighboring pixels $(x, y), (x + 1, y)$ or $(x, y), (x, y + 1)$ being not in the same class, in our case taken from different images, is thus defined as the weighted intensity difference between both images at these pixels:

$$C(x, y, x+1, y) = \sqrt{\frac{(I_b(x,y) - I_s(x,y))^2 + (I_b(x+1,y) - I_s(x+1,y))^2}{n_x(x,y)}} \tag{3}$$

$$C(x, y, x, y+1) = \sqrt{\frac{(I_b(x,y) - I_s(x,y))^2 + (I_b(x,y+1) - I_s(x,y+1))^2}{n_y(x,y)}}, \tag{4}$$

where $I_b(x, y)$ is the intensity at pixel (x, y) in the unrolled blended image after removing the low spatial frequencies, $I_s(x, y)$ in the corresponding unrolled background and

$$\begin{aligned} n_x(x,y) =&(I_b(x-1,y) - I_b(x+1,y))^2 + (I_b(x,y) - I_b(x+2,y))^2 \\ &+ (I_s(x-1,y) - I_s(x+1,y))^2 + (I_s(x,y) - I_s(x+2,y))^2 \end{aligned} \tag{5}$$

$$\begin{aligned} n_y(x,y) =&(I_b(x,y-1) - I_b(x,y+1))^2 + (I_b(x,y) - I_b(x,y+2))^2 \\ &+ (I_s(x,y-1) - I_s(x,y+1))^2 + (I_s(x,y) - I_s(x,y+2))^2 \end{aligned} \tag{6}$$

the normalization function. Put simply, cuts through regions that differ in both images are penalized. Figures 4b-c show the estimated cutting edge on an unrolled face image.

By separating high spatial frequencies and low spatial frequencies, we can obtain a border that provides a smooth transition and does not cut through high spatial frequency biometric characteristics. The estimation of the transition in the lower frequencies part allows a smoothly and subtly adjustment of different colors or different brightness on the skin due to both, differences in skin color and illumination. The estimation of a sharp border for the high frequencies provides a cutting path through the image along similar regions, thus visible borders or a cuts through individual biometric characteristics like freckles, moles or scares are avoided.

3 Database and preprocessing

In this section, we describe our face image database and the preprocessing steps that are applied to both types of images, original face images and morphing attacks, to increase the variety of the data and prevent the neural networks from learning low-level artifacts like gradient distributions. Finally, we introduce a modified cross-validation method. It considers the dependency between our original face images and morphing attacks, caused by the morphing process itself, since it fuses two original images to one forged synthetic image containing characteristics of both original images.

3.1 Database

Our face image database contains about 1250 face images of different individuals. The database can be divided into 9 different groups of images, with nearly each group having similar capture conditions regarding illumination and camera type and properties. One group contains face images captured under varying conditions and with different cameras. All images show a face in a frontal pose and the captured person has open eyes, the mouth closed and a neutral expression, as demanded for passport images. Morphing attacks are only created using images from the same group. In this way, we can acquire morphing attacks of high quality, since difficult alignments due to different focal length, distortions or other camera parameters are avoided.

3.2 Preprocessing

For more variety in the data, we artificially add two types of noise and two types of blur to both, the original images and the morphed images. The parameters for the noise and blur were chosen randomly for each image, but with an upper limit as described in the following table:

Kind of noise/blur	Upper limit
Salt and Pepper noise	1% of all pixels
Gaussian noise	Standard deviation up to 0.05
Gaussian blur	Standard deviation up to 0.5% of image width
Motion blur	Angle randomly, length up to 1.2% of image width

Beside adding more variety, this kind of preprocessing is also intended to prevent our networks from learning low-level artifacts caused by resampling or JPEG-compression. In summary, our dataset contains about 9000 fake images that are randomly processed by adding salt and pepper noise, Gaussian noise, motion blur, Gaussian Blur or none of them. The original face images are processed in the same way.

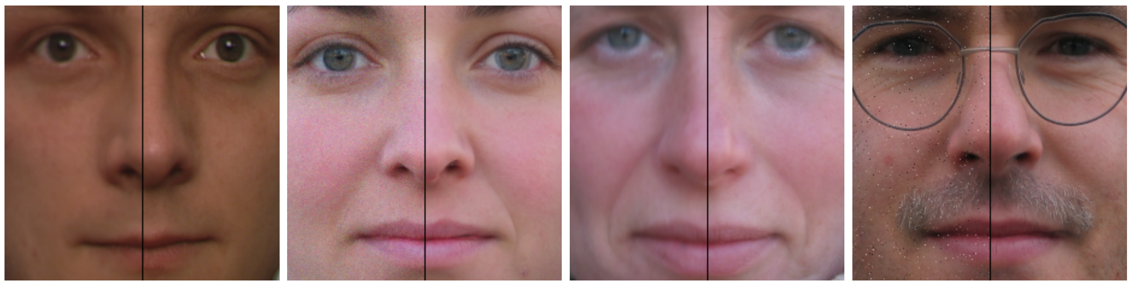


Fig. 5. Morphing attacks after (left half) and before (right half) preprocessing: Gaussian blur, color noise, motion blur, Salt-and-pepper noise (from left to right)

3.3 Test set selection

We use 4-fold cross-validation to evaluate the robustness of our trained neural networks. Since the samples in our data are not independent due to the fact that a morphing attack is performed by averaging two samples (original face images), the separation of the datasets for the 4-fold cross-validation has to consider this dependency. Hence, we separate the dataset of morphed and original face images slightly different to the classical 4-fold cross-validation. The test and training sets are created as follow:

1. We separate the individuals into four disjunctive sets $\mathbf{I}_{1,2,3,4}$ of the same size.
2. We create the test sets $\mathbf{T}_{1,2,3,4}$, with $\mathbf{T}_i$ containing all images showing individuals in $\mathbf{I}_i$ and morphing attacks based on individuals in $\mathbf{I}_i$.
3. A training set $\bar{\mathbf{T}}_\mathbf{i}$ contains all images that are not in $\mathbf{T}_i$.

By separating the images that way, some morphing attacks will appear in two test data sets, but we removed correlation between test and training data.

4 Experiments

In this section, we describe our experimental setup, in particular the preprocessing of the data to fit the requirements of the networks, e.g. the input size, outline the network architectures we use for the image forgery detection and present the accuracy of the networks for the task of forgery detection. For each network architecture, we analyze the morphing detection accuracy for the network trained from scratch and starting the training with pretrained models, which were trained for the task of object classification on the ILSVRC dataset.

4.1 Outline on the used network architectures

We analyze three different deep convolutional network architectures, AlexNet, GoogLeNet and VGG19, which are all well known for object classification tasks. All three networks have nearly the same input image size that is 224x224 pixels for the VGG19 and GoogLeNet architecture and 227x227 pixels for the AlexNet. While the AlexNet was the first DNN that revolutionized the area of object classification and its design focuses on performance on two graphics cards, the architecture of VGG19 was designed to be simple but powerful and all of its convolutional layers have kernels of size 3x3 and exactly one predecessor and one successor layer. In contrast to the AlexNet and the VGG19 architecture, the GoogLeNet has a quite complex structure. Its key components are inception modules [14]. These modules consist of multiple convolution filters that process the same input and concatenate the results. Using this type of modules, the GoogLetNet achieves a similar accuracy for object classification tasks as the VGG19, but needs less than one-tenth of the parameters the VGG19 needs.

4.2 Preprocessing before feeding the network

To reduce unnecessary variety, that can be removed in a trivial preprocessing step, we rotate the image on the image plane, such that the eyes are at the same height. All networks were fed with a complete facial image that contain only the inner part of the face that is between the eyebrows and the mouth and between the outer parts of the eyes. Since neural networks need images of fixed size as input, the images are rescaled such that this region fits the input size of the network. In order to get more variety in the data, the region extracted from the rotated and scaled image is randomly moved for up to 2 pixels in any direction. This process does not harm the correctness of this data since the features in the face are not aligned, e.g. the position and size of the eyebrows, nose, pupils varies from human to human, moreover the detected borders of the region are subject to small inaccuracies.

4.3 Training

We trained all three network architectures from scratch as well as starting with pretrained models that were trained for the class of object classification on the ILSVRC dataset. The average color, that is recommended to subtract before feeding the network, was set for all experiments to the average color of the ILSVRC dataset, since the pretrained networks use this average color - except for the AlexNet, which used a average image, which has the same average color. The training samples for one epoch of the network were shuffled and some original images duplicated, such that the probability of selecting a morphing attack is equal to the probability of selecting an original face image. The networks that were trained from scratch were initialized randomly, except for the VGG19, which was initialized using the random initialization procedure of [22]. In both cases, pretrained and training from scratch, the training was terminated after the loss function was not improved for several epochs.

4.4 Results

The training of all networks converged and the average loss on images of the training set were below $5 \cdot 10^{-5}$ for all networks. The accuracy in terms of False Acceptance Rate (FAR) and False Rejection Rate (FRR) is shown in table 1. The FAR is defined as relative amount of morphing attacks classified as genuine images and the FRR as the relative amount of genuine images classified as morphing attacks. The FRR of VGG19 (pretrained) is only about a third of the FRR of the AlexNet (pretrained) and 2.1 percent points lower than the FRR of the GoogLeNet (pretrained). All network architectures performed better, when starting the training with pretrained networks instead of learning all weights from scratch. The FRR of the network architectures GoogLeNet and VGG19 outperformed the AlexNet in both cases, while the FAR is for all architectures nearly the same, but depends of the initialization of the weights.

	AlexNet		GoogLeNet		VGG19	
	FRR	FAR	FRR	FAR	FRR	FAR
From Scratch	16.2%	1.9%	10.0%	1.8%	10.9%	2.2%
Pretrained	11.4%	0.9%	5.6%	1.2%	3.5%	0.8%

Table 1. Accuracy of the trained networks in terms of False Acceptance Rate and False Rejection Rate

5 Conclusion

In this paper, we proposed a morphing attack detection method based on deep convolution neural networks. A fully automatic face image morphing pipeline with exchangeable components was

presented that was used to create training and test samples for the networks. Instead of detecting classical traces of tampering, e.g. caused by resampling or JPEG double compression, and their anti-forensic methods [23–25] we focused on semantic artifacts. To avoid learning a detector for these classical tampering traces, all images were preprocessed by scaling, rotating and cropping, before feeding them to the network. In addition, we added different kinds of noise and blur to the training and test data.
We trained three different convolutional neural network architectures from scratch and using already trained networks for the initialization of the weights. The FRR of our trained networks differ between 3.5% and 16.2% and the FAR between 0.8% and 2.2%. The VGG19 (pretrained) achieved for both rates the best result with a FRR of 3.5% and a FAR of 0.8%. The pretrained networks outperformed the networks trained from scratch for every architecture. This suggests that the features learned for object classification are also useful for detection of morphing attacks.

Acknowledgment

The work in this paper has been funded in part by the German Federal Ministry of Education and Research (BMBF) through the Research Program ANANAS under Contract No. FKZ: 16KIS0511.

References

1. L.J. Spreeuwers, A.J. Hendrikse and K.J. Gerritsen, "Evaluation of automatic face recognition for automatic border control on actual data recorded of travellers at Schiphol Airport", in International Conference of Biometrics Special Interest Group, BIOSIG, pp. 1-6, (2012)
2. M. Ferrara, A. Franco, and D. Maltoni, "The magic passport" in Biometrics (IJCB), 2014 IEEE International Joint Conference on, 2014, pp. 17.
3. A.C. Popescu, H. Farid, "Exposing digital forgeries by detecting traces of re-sampling", IEEE Transactions on Signal Processing, vol. 53, no. 2, pp. 758-767, 2005.
4. M. Kirchner and T. Gloe, On Resampling Detection in Re-Compressed Images, in First IEEE International Workshop on Information Forensics and Security (WIFS '09), pp. 21-25, 2009.
5. M. Kirchner and J. Fridrich, "On Detection of Median Filtering in Images", Proc. SPIE, Electronic Imaging, Media Forensics and Security XII, vol. 7541, pp. 10 1-12, 2010
6. Estimation of Primary Quantization Matrix in Double Compressed JPEG Images, with J. Lukas, Proc. of DFRWS 2003, Cleveland, OH, USA, August 5-8 2003.
7. H. Farid, "Exposing Digital Forgeries from JPEG Ghosts", IEEE Transactions on Information Forensics and Security, 4(1), pp. 154-160, (2009)
8. M.K. Johnson and H. Farid, "Exposing Digital Forgeries Through Specular Highlights on the Eye", 9th International Workshop on Information Hiding, 2007
9. E. Kee, J. OBrien, and H. Farid, "Exposing Photo Manipulation from Shading and Shadows", ACM Transactions on Graphics, 33(5):165:1-165:21, 2014
10. A. Makrushin, T. Neubert and J. Dittmann, "Automatic Generation and Detection of Visually Faultless Facial Morphs", in proc. VISAPP, pp.39-50, (2017)
11. R. Raghavendra , Kiran Raja, Christoph Busch, "Detecting Morphed Face Images", IEEE Eighth International Conference on Biometrics: Theory, Applications, and Systems, 2016
12. A. Krizhevsky, I. Sutskever and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks" in Advances in Neural Information Processing Systems 25, 2012, pp. 1097-1105
13. K. Simonyan, A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition", arXiv technical report, 2014

14. C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, ”Going deeper with convolutions” in IEEE International Conference on Computer Vision and Pattern Recognition, pages 19, 2015
15. B. Bayar and M. C. Stamm, ”A Deep Learning Approach to Universal Image Manipulation Detection Using a New Convolutional Layer”, in Proceedings of the 4th ACM Workshop on Information Hiding and Multimedia Security, pp. 5-10 (2016)
16. Davis E. King. Dlib-ml: A Machine Learning Toolkit. Journal of Machine Learning Research 10, pp. 1755-1758, 2009
17. V. Kazemi and J. Sullivan, ”One Millisecond Face Alignment with an Ensemble of Regression Trees”, CVPR 2014
18. T. Beier and S. Neely, ”Feature-based image metamorphosis”, Proc. Computer Graphics 26 (2), pp. 3542, (1992)
19. B. Delaunay, ”Sur la sphere vide”, Izv. Akad. Nauk SSSR, Otdelenie Matematicheskii i Estestvennyka Nauk, vol. 7, 1934, pp. 793-800
20. P. Prez, M. Gangnet and A. Blake, ”Poisson image editing”. ACM Trans. Graph. 22, (3),pp. 313-318 (July 2003)
21. B. Prestele, D. Schneider, P. Eisert, ”System for the Automated Segmentation of Heads from Arbitrary Background”, Proc. IEEE International Conference on Image Processing (ICIP), Brussels, Belgium, pp. 3257-3260, Sep. 2011.
22. X. Glorot and Y. Bengio, ”Understanding the difficulty of training deep feedforward neural networks”, In Proc. AISTATS, volume 9, pp. 249256, 2010.
23. M. C. Stamm and K. J. R. Liu, ”Anti-forensics of digital image compression,” in IEEE Transactions on Information Forensics and Security, vol. 6, no. 3, pp. 1050-1065, Sept. 2011.
24. J. Yu , Y. Zhan, J. Yang, X. Kang A Multi-purpose Image Counter-anti-forensic Method Using Convolutional Neural Networks. In: Shi Y., Kim H., Perez-Gonzalez F., Liu F. (eds) Digital Forensics and Watermarking. IWDW 2016. Lecture Notes in Computer Science, vol 10082
25. M. Kirchner, R. Bohme, Hiding Traces of Resampling in Digital Images, IEEE Transactions on Information Forensics and Security, v.3 n.4, p.582-592, December 2008