Biometric Images Attacks Detecting Using Convolutional Neural Networks

Kyrylo Chernov [1[0000-0003-2417-3793]], Kateryna Kuznetsova [1[0000-0002-5605-9293]], Inna Oleshko [2[0000-0002-8021-0467]], Mykhaylo Bagmut [1[0000-0002-4160-4316]], Alexandr Fedorchuk [3[0000-0003-4732-8305]] and Dmytro Kryzhanovskyi [2[0000-0002-3065-4035]]

[1] V. N. Karazin Kharkiv National University, Kharkiv, Ukraine
kirillfilippsky@gmail.com, sapsanmiha@gmail.com
[2] Kharkiv National University of Radio Electronics, Kharkiv, Ukraine
inna.oleshko@nure.ua, dmytro.kryzhanovskyi@nure.ua
[3] Kherson State University, Kherson, Ukraine
15961980@ukr.net

Abstract. Nowadays biometric user authentication systems have become widespread. These systems are implemented not only in enterprises, controlled-access facilities, but also on smartphones of ordinary users and in online applications. There are special attacks designed to impersonate another person by submitting fake biometric data to the authentication system. The composition, purpose and main components of biometric face recognition systems are considered in the article, the essence of some known types of attacks, existing methods of counteracting counterfeit attacks (spoofing attacks) are analyzed and a new method for detecting them is proposed. The method is based on the use of an artificial convolutional neural network which was trained using a Replay-Attack Database from Idiap Research Institute. The results of modeling attacks on training, validation and test data sets are presented. The obtained results show high efficiency of the proposed method: the probability that an attack will be detected is 94.98%. This indicates the prospect of further studies of artificial neural networks for detecting attacks of counterfeit biometric images.

Keywords: biometric images, spoofing attacks, convolutional neural networks, authentication

1 Introduction

Biometrics refers to the science of defining a personality basing on analysis of its behavior or physical or chemical characteristics [1]. The main biometric methods are fingerprinting, face and voice recognition. The biometric system recognizes personality by performing a number of operations. At the global level, a system using biometric data consists of four parts [1]:

 CybHyg-2019: International Workshop on Cyber Hygiene, Kyiv, Ukraine, November 30, 2019.

1. Scanner is a physical device used to collect input data to a biometric system. Depending on the application, the functionality of the device may be different, for example, fingerprint images can be obtained with optical sensors, while a face sample can be obtained with a camera.
2. Features extraction algorithm – important features contained in the input data are removed using the extraction algorithm.
3. Data storage – a database contains templates of a similar type of input data, for instance, a fingerprint database consists of information about the fingerprints of several people. The data is not stored open for security reasons, but instead of that it is stored in a different special format.
4. Decision-making algorithm (Decision algorithm) – the removed features are compared with those stored in the database to obtain similarity evaluation. Depending on this assessment, a decision regarding the authentication (identity) of the user is made.

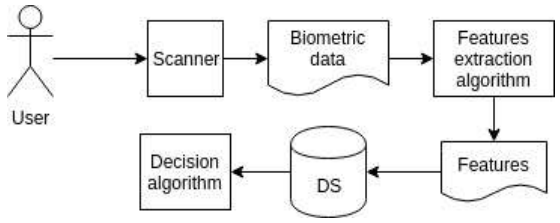

Fig. 1. The components of the biometric system of authentication (identification) of users

Depending on the application, biometric systems are divided into two types: to authenticate or identify a person in the system. The identification system compares the input data with each image that lies in the database, and then makes its decision about the identity of the person. The authentication system compares the input data only with an image that matches certain unique identifier in the database.

Since there are too many features that can be removed from a person's biometric data, for a feature to be considered useful, it must meet certain conditions. There are several of the following conditions given in [2]:

1. Universality (versatility) – means that everyone must have this feature.
2. Uniqueness – means that there must be sufficient difference among the characteristics of different people.
3. Constancy (permanency) – the feature must remain unchanged during the time when the prototype of the user's biometric attribute is stored in the database.
4. Measurability – the feature must be possible to remove and/or to process.

One of the biometric systems that meet these conditions is the face recognition system. In this system the biometric data scanner is a typical or specially designed camera for this task. In the article we will keep in mind that the scanner will use a regular smartphone camera or webcam for biometric data capture in the system.

The objective of this work is to develop a method for detecting a spoofing attack on a biometric face recognition system. The proposed method is based on the use of

modern elements of deep learning convolutional neural networks, which is used to detect attempts to fake biometric images (for example, by replacing biometric data when scanning the face with appropriate photo or video images).

2 Face recognition system

Deep learning methods that have developed significantly in recent years are used in our face recognition system. Their application allows achieving high accuracy [3].

In the general case, the process of face recognition using deep learning methods can be described as follows:

1. User takes a photo of his face in a real-time mode.
2. Next, the module that finds the person's face in the image works. The found face is sent to a module that processes unique features.
3. The unique features of a mathematical image along with a unique user ID are compared with those already stored in the database.
4. The similarity of these two mathematical images is verified. If the difference between them does not exceed a specified value, then the user is allowed to access some resource or other type of service, that is, the authentication was successful. If the difference between the images exceeds the specified value, the user is informed that the authentication attempt has failed and the access to resource is failed.

A trained deep convolutional neural network is used as a module that finds the face in the image. The algorithm which is used to find the object is described in [4]. Moreover, another trained convolutional neural network which is currently state-of-the-art within the scope of this task is used as an algorithm for extracting important features [5].

The described system does not work in real time mode, that is, the input of algorithms is submitted with only one image of the user. Therefore, such a system is vulnerable to special attacks on face recognition algorithms. Mainly, such as substitution or spoofing attacks [6].

The general scheme of such attacks implementation is shown in Figure 2. If Figure 1 shows the correct biometric system, an attempt to spoof the attack will look like a substitution of real data of the user to the image of his data (in our case, this is the face).

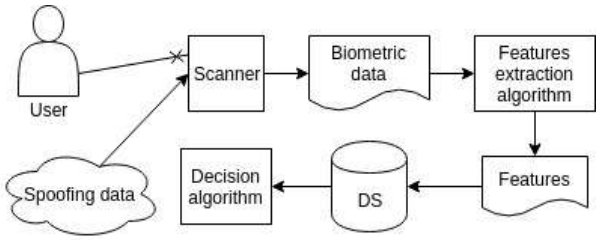

Fig. 2. General scheme of the attack on the face recognition system

Let's analyze known attacks on face recognition systems that are built using deep learning methods.

3 Attacks on the face recognition systems

Currently, there are two types of attacks on the face recognition systems:

- Adversarial attacks [7, 8].
- Spoofing attacks [9].

To begin with, adversarial attack information will be outlined and analyzed. This type of attack is aimed specifically at convolutional neural networks.

3.1 The essence of Adversarial attacks and methods of counteraction

In 2014, a team of Google and NYU researchers found out [8] that attacking convolutional neural networks is an extremely simple, carefully constructed push-in. Figure 3 shows an example of such an attack.

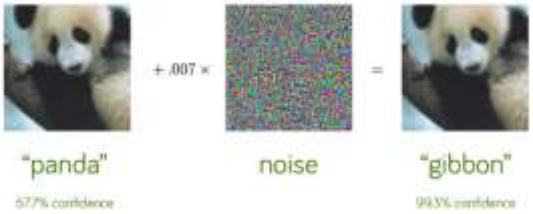

Fig. 3. An example of an adversarial attack

We start with an image of a panda that is correctly recognized by deep learning methods (neural network) as a "panda" with a confidence of 57.7%. If carefully constructed noise is added to this, the neural network itself will assume that it is a Gibbon image with 99.3% confidence. This is obviously an optical illusion, but only for the neural network. For human vision, nothing has changed, and it is safe to say that both images look like pandas – in fact, one cannot even say that some noise was added to the original image to build an example on the right. But adding ordinary noise is not the only type of attack. Another example is already aimed at bypassing of biometric facial recognition system via creating special glasses. It is showed in [9] that it is possible to deceive face recognition software by creating Adversarial goggles avoiding face recognition. An example of such goggles is shown in Figure 4.

The most interesting thing about this type of attack is that it is completely impossible to counteract it. There are only a few ways that can just reduce the likelihood of an attack of this kind [8]. One of the simplest and most violate ways is Adversarial training. Its main point is to create a series of Adversarial examples for its own neural network and then teach the model not to be mistaken with such examples. This improves the overall situation, but sufficient resistance from this attack is not achieved.

In fact, an attacker may be able to make a new attack in the same way but with the noise picked up for a new neural network. Such a struggle can go on indefinitely.

Fig. 4. Example of glasses that can be used to trick the face recognition system

The second way is Defensive Distillation. Its main point is that it trains the second model, the upper layers of which are smoothed in the places that the attacker will usually try to use, complicating their detection, which can lead to incorrect categorization. The reason it works is that, unlike the first model, the second one is trained on the "soft" output performance of the primary model, rather than the "hard" (0 or 1) ones with real training data. This technique has been shown to be successful in protecting against simple variants of such attacks, but has become vulnerable to newer attacks. In particular, one can take the Carlin-Wagner attack as an example [10], which is a valid benchmark for evaluation of the neural network strength against competitive attacks.

Thus, as one can see, there are no explicit methods of defense against the attacks under consideration. Therefore, the task of solving this issue is still a method of research in the field of computer science or artificial intelligence [8].

3.2 The essence of Spoofing attacks and methods of counteraction

The situation is different with this type of attack. A spoofing attack is an attempt to grant someone else's privileges or access rights through a photo, video or other data. Some examples of such attacks include:

- Photo Attack: An attacker uses someone else's photo. The image is printed or displayed on a digital device.
- Video Attack: A more sophisticated way to fool a system that typically requires looping video of a person's face. This approach makes it possible to use facial expressions and facial movements to make them look more "natural" than the previous method.
- 3-D Mask Attack: During this type of attack, the mask is used as a forgery tool. This is more complicated attack than video playback. This allows you to use facial movements and to bypass some additional layers of protection, such as depth sensors.

There are many different approaches to counteracting attacks of this type. The most popular modern solutions include:

- Live Face Detection: A mechanism based on an analysis of how “live” is a test face. This is usually done by checking eye movements, such as blinking and face movement.
- Contextual information exploration: By exploring the surroundings of an image, we can try to detect whether a digital device or photo paper was in the area found.
- Texture analysis: In this method, small textured portions of the input image are probed to find patterns in fake and real images.
- User Interaction: By inviting the user to perform a specific action (turn their head left or right, smile or blink their eyes), the algorithm can detect whether the action was performed naturally.

Of course, one cannot ignore the special technical means specially designed to counteract such attacks. In practice, it was possible to offer Apple a list of deep-rendering and 3D-usage methods that allow for fake support with great precision [11]. However, this high-end equipment is exclusively the proprietary campaign used by large companies in secure regional programs.

During research and testing, it has been found that an extremely high level of real-time counterfeit detection can be achieved with a standard medium quality camera. We offer a new method of recognition spoofing attacks, which is based on the use of convolutional neural networks.

4 The proposed method and experimental research

As mentioned above, convolutional neural networks perform very well in computer vision tasks. Therefore choosing the approach to developing the method, it was decided to use them.

As is known from [12], in order to teach a neural network a specific task, it is necessary to have specific data. In our case, the training aims at recognizing the spoofing attack, that is, detecting a fake face in the system, provided that only one face image and no more additional information is fed to the decision algorithm.

As a result, the proposed method consists of using a convolutional neural network that receives biometric images of human faces at the input (128x128 pixels in our experiments) and outputs a binary solution: whether the input image is spoofed or not.

The convolutional neural network (CNN) was first proposed by Jan Lekun in 1988 and aims at effective pattern recognition [13, 14]. The main idea when constructing convolutional neural networks is to alternate subsampling layers or pooling layers. This is some approximation to the functioning of the visual cortex [15], in which so-called simple cells responding to straight lines at different angles and complex cells whose reaction is associated with the activation of a particular set of simple cells were discovered. This artificial neural network architecture was named because of the convolution operation, the essence of which is that each fragment of the image is multiplied by the convolution matrix (nucleus) element by element, and the result is summed up and written to a similar position of the original image.

Usually standard methods are used to train convolutional neural networks, most often the backpropagation method is the gradient calculation method used for updating

the weights of a multilayer perceptron [16, 17]. This is an iterative gradient algorithm that is used to minimize the error of the multilayer perceptron and obtain the desired output [16]. The basic idea of this method of training is to propagate error signals from the network outputs to its inputs, in the direction opposite to the direct propagation of signals in the normal mode [17]. To implement convolutional neural network training in the proposed method of detecting spoofing attacks, data belonging to Idiap Research Institute and contained in the Replay-Attack database were used [18].

The Replay-Attack database consists of 1,300 video clips of attempted photo and video attacks for 50 different people under different lighting conditions. Data from this database is already divided into 3 subgroups, including:

- Train data that will be used to train the anti-forgery classifier.
- Validation data used to find the optimal classification threshold.
- Test data to measure the effectiveness of the method developed.

Persons who appear in one of the datasets (train, valid, or test) do not appear in any other dataset. After some experiments, it was decided to integrate this method immediately after the face image was found on the input. This is due to the fact that, in some examples, a spoofing attack, the face finding algorithm did not find the face in the glossy type photos.

The problem in our work is formulated as follows: a biometric image of the face is fed to the input of the spoofing attack recognition algorithm. At the output, the algorithm should give one of two possible binary values:

- 1 (or «yes») - if the input is classified as a spoofing attack,
- 0 (or «no») - if the input is classified as a real image.

An image argumentation algorithm was developed to extend our data, and the following types of argumentation were used:

- Adding a Gaussian noise with a value $scale = 0.05 * 255$, where scale is a standard deviation from the normal noise-generating distribution;
- Adding a Gaussian Blur with a value $sigma = 0.5$, where sigma is the standard deviation of a Gaussian filter;
- Rotations from -45 to 45 degrees;
- Zoom in or zoom out.

The next stage of the study was the justification of special metrics to measure the effectiveness of the developed method. Because as a result of detecting an attack, a binary decision is made (yes / no), we have the following set of possible situations (Table 1). In our case, the null hypothesis is defined as follows: at the entrance to the spoofing recognition algorithm, we consider that the input image is an attempt to spoof the attack, which is denoted as H_0. Then, the alternative hypothesis is defined as this: the input image that is fed to the input of the recognition algorithm is not an attempt to spoof the attack and is denoted as H_1. For each input image, a decision or conclusion is drawn that accepts or rejects the null hypothesis, i.e. whether it is an attempt to spoof the attack or not.

For all input data, the decision rule is: if the output of the algorithm is less than 0.5, then it is assumed that the image is not an attempt to spoof the attack, otherwise it is assumed that the image is a spoofing attack.

Table 1. Possible situations when fake attacks are detected

		The real value	
		H_0	H_1
Model predicted result	H_0	True positive	False negative
	H_1	False positive	True negative

Table 1 shows the following terms:

- True positive – the model correctly recognized the spoofing attack;
- False positive – the model recognized the spoofing attack as a real face image;
- True negative – the model correctly recognized the real face image;
- False negative – the model recognized the real face image as a spoofing attack.

False positives and False negatives are also called 1st and 2nd order errors. Depending on the system in which the algorithm is used, attention to one of the errors becomes greater. For example, if we take our situation into account, attention should be paid to a 1st-order error, since, when a model recognizes a spoofing attack for a real face, it can lead to critical consequences. Otherwise, for example, we have a system for recognizing spam emails on the e-mail box, then there will already be more attention to the error of the 2nd kind, since, when a non-spam email is recognized by the algorithm as spam, it can lead to that an important email may be transferred to the spam folder and not reach the end user.

After defining the basic concepts involved in evaluating the performance of a classification algorithm, let's determine what our model performance test is. Suppose we have trained our network on training data, and we need to know how the model would behave when dealing with data that was not in the training sample. To do this, there is or creates a special set of images that the algorithm has not yet seen. During testing, all results are stored in a special form, depending on Table 1. Thus, after testing throughout the test sample, we have four values that are the results of the model. Such values are the number of results True positive, True negative, False positive, False negative. But these are just basic metrics and, to better evaluate the performance of the model, use other custom metrics. In our work, such metrics are Precision and Recall:

$$Precision = \frac{True\ positives}{True\ positives + False\ positives}, \quad (1)$$

$$\mathrm{Re}call = \frac{True\ positives}{True\ positives + False\ \mathrm{negatives}}, \quad (2)$$

where:

- True positives - the total number of correctly recognized attacks;
- False positives - the total number of false unrecognized attacks;
- False negatives is the total number of mis-recognized attacks.

Thus, in formulas (1) and (2), Precision shows the proportion of detected attacks from all attacks that were in the data set, whereas Recall shows the proportion of correctly recognized attacks from all images that were recognized as attacks [19]. The choice of such metrics is due to the fact that they show how big the error of the 1st kind is, on which we focus our attention.

After defining the necessary metrics and data processing methods, the next step is to develop a neural network architecture. Figure 5 shows its architecture.

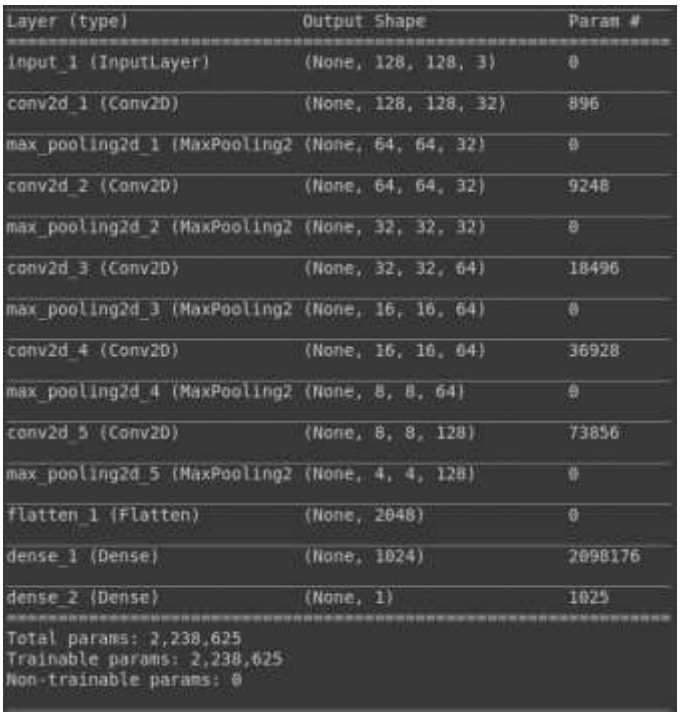

Fig. 5. Convolutional neural network architecture

From Figure 4 you can see that our neural network has 5 convolutional blocks consisting of a Convolutional (Conv2D) layer and a MaxPooling2D layer. The last digit in the Output Shape value indicates the number of filters in the convolutional layer, in all convolutional layers the filter size is 3x3, and the relu activation function [20] and the padding value equals same. Filter size in MaxPooling2D layer is 2x2. After the last convolutional block, there is a Flatten layer that changes the dimension of a 3-D matrix into a 1-D vector, followed by two Dense layers, the first size 1024 neurons with relu activation function, the second one with 1 neuron and sigmoidal activation function [20] to obtain the probability of a substitution attack.

The optimization of the neural network weights was performed using the Adam algorithm [21], which is given below.

The input to the algorithm:

- α : Learning speed (standard setting $\alpha = 0.001$);
- $\varepsilon = 10^{-8}$;
- $\beta_1, \beta_2 \in [0,1]$: Exponential decay rates for moment estimates (default settings $\beta_1 = 0.9, \beta_2 = 0.999$);
- $f(\theta)$: Stochastic function with parameter θ;
- θ_0 : Initial parameters vector.

Algorithm:

- $m_0 \leftarrow 0$ (Initialization of the first moment vector);
- $\nu_0 \leftarrow 0$ (Initialization of the second moment vector);
- $t \leftarrow 0$ (Initializing time step).

Until θ_t converges, we perform the following operations in a loop:

- $t \leftarrow t+1$;
- $g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$ (Take a gradient with respect to the stochastic function of each time step *t*);
- $m_t \leftarrow \beta_1 m_{t-1} + (1-\beta_1) g_t$ (Update the rejected value of the first moment);
- $\nu_t \leftarrow \beta_2 \nu_{t-1} + (1-\beta_2) g_t^{\,2}$ (Update the rejected value of the second moment);
- $m_t \leftarrow \dfrac{m_t}{1-\beta^t_1}$ (Calculate the rejected value of the first moment);
- $\nu_t \leftarrow \dfrac{\nu_t}{1-\beta^t_2}$ (Calculate the rejected value of the second moment);
- $\theta_t \leftarrow \theta_{t-1} - \alpha . \dfrac{m_t}{\sqrt{\nu_t} + \varepsilon}$ (Update parameters).

At the output of the algorithm we get θ_t (Result parameters).

The training of the neural network was held with the following parameters:

- learning_rate = 0.001;
- epochs = 25;
- batch_size = 32;
- loss: binary_crossentropy,

where:

- learning rate – a training step or rate at which weights change. If you choose too much of a learning step, the function can jump over the global minimum and nev-

er collapse, and vice versa, if set too small, the function can get stuck in the local minimum and never get out of it;

- epochs – number of training cycles, that is, how many times the model should go through all training data;
- batch size – the number of images that are fed into the model at the same time (to made the model geared towards generalizing the data and to speed up the learning process);
- loss - a special function that is used to optimize the neural network. Loss helps optimize neural network parameters. Our goal is to minimize loss to the neural network by optimizing its parameters (weights). Loss is calculated by using a special function, comparing the target (real) value and the predicted value with a neural network. Then we use Adam gradient descent method to update the neural network weights so that loss is minimized. So we train a neural network. Therefore, loss is a function of losses, it is in charge of what the neural network will study, in our case the task of binary classification, so the choice falls on binary_crossentropy loss [22]. It is defined as follows:

$$H_p(q) = -\frac{1}{N}\sum_{i=1}^{N} y_i \bullet \log(y_i) + (1-y_i) \bullet \log(1-y_i), \quad (3)$$

where:

- y_i is a real label (1 or 0);
- y_i is a label predicted by the neural network.

That is, binary_crossentropy loss measures the effectiveness of a classification model whose output is a probability value between 0 and 1. Loss increases when the predicted probability deviates from the real value. The ideal model should have a value of loss function close to zero.

All software was programmed in Python using the following frameworks: tensorflow [23], opencv [24], imgaug [25].

5 The results of the experiment

After 20 epochs, the model began to show signs of retraining, so the end of training occurred on the 20th epoch. Loss of our network was 0.1796 and the overall accuracy on the validation data was 0.9485. Figure 6 shows how the training loss and validation loss of our model varied over the training period. You can also see in Figure 7 how the model's accuracy in training and validation data has changed.

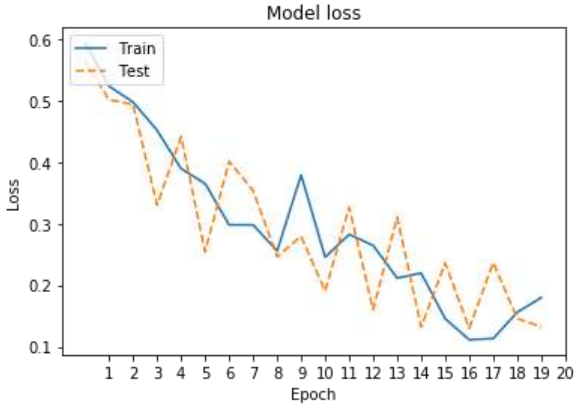

Fig. 6. Changes in tain_loss and test_loss during training.

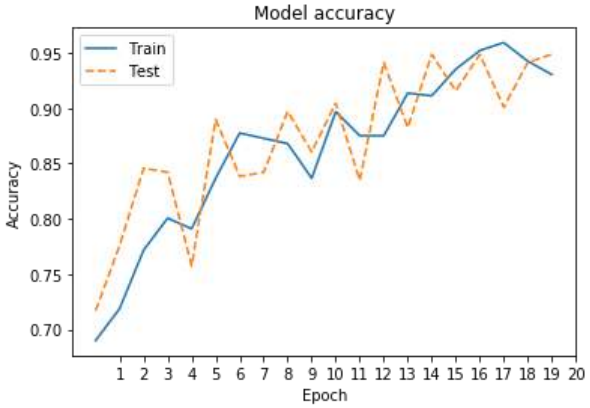

Fig. 7. Changing of model performance during training on validation and training data

You can see that in Figure 6, our loss function drops throughout the training. This means that the model was able to find the difference in the distribution of data in the real image of the human face and in the fake. Figure 7 confirms the above, we see how accuracy (the number of correctly predicted values) increases with each epoch. You can also see that the results of the test set are worse on both charts. This is because the training data is seen and memorized by the model and not by the test data. Therefore, the effectiveness of the test sample is less than the training sample.

The next step to improving the model's efficiency is finding the best decision threshold, which is 0.5 by default. The main criterion is the balance between false positive rate (FPR) and false negative rate (FNR) values [26]. Figure 8 shows how FPRs and FNRs change with the decision threshold being changed. The optimal threshold is usually the value where these two metrics have the smallest values relative to each other [27]. This is usually the point of intersection.

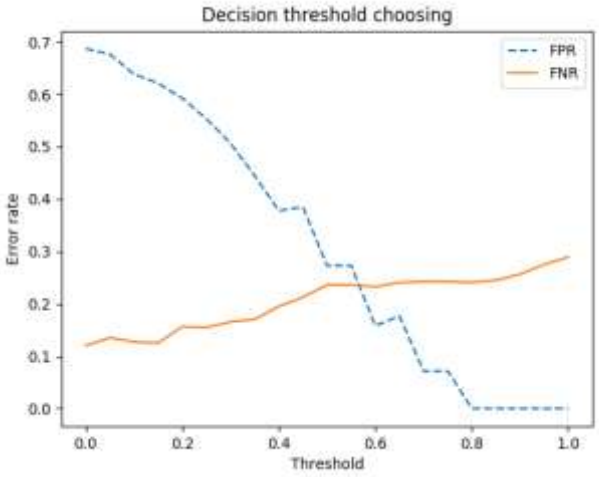

Fig. 8. Changing false positive rate and false negative rate from changing the decision threshold

As can be seen in Figure 8, the optimal decision threshold should be 0.59 in order to minimize the first-order error and the second-order error. From the figure you can see that further false negative rate does not increase much relative to how false positive rate decreases. The decision was made to set a decision threshold of 0.7 because it is more important for our system to reduce 1st order error.

When the optimal decision threshold has been found, the final step of our experiment is to measure the performance of our model on new data. In our case, such data is the test data specified in paragraph 4 of this article. Thus, according to the metrics of paragraph 4, the results of the algorithm are shown in table 2.

Table 2. The results of the algorithm on all three databases.

	Training database	Validation database	Test database
Precision, %	0.9786	0.9523	0.9498
Recall, %	0.9642	0.9135	0.9204

The values in the table were calculated on an already trained model. Metrics were calculated using formulas (1) and (2). As you can see from these metrics, the probability that an attack will be detected is 94.98%. It should be noted that models with different parameters and architectures were tested. The table shows the results of the best model, the details of which were given in paragraph 4.

6 Conclusions

This paper examines existing attacks on face recognition systems. A method of counteracting Spoofing attack based on the use of artificial convolutional neural network was developed. The probability that an attack will be detected by our system is

94.98%. Out of the box, only adversarial attacks remain, and still remain unbeatable. It can be noted that if you use real-time recognition, adversarial noise-based attacks will lose their relevance. Further work should focus on the recognition of the special eyepieces that were mentioned in paragraph 3 of this article. In addition, the obtained research results can be useful for various methods to improve the reliability and security of biometric systems [28, 29]. These results can be used in other computer science applications [30-36].

References

1. A.K. Jain, P. Flynn, and A.A. Ross, “Introduction to biometrics”, in Handbook of Biometrics. 2008.
2. D. Maltoni, D. Maio, A.K. Jain, S. Prabhakar, Handbook of Fingerprint Recognition, 2003.
3. Deep Residual Learning for Image Recognition Kaiming He Xiangyu Zhang Shaoqing Ren Jian Sun. https://arxiv.org/pdf/1512.03385.pdf
4. YOLOv3: An Incremental Improvement Joseph Redmon, Ali Farhadi University of Washington. https://arxiv.org/abs/1804.02767
5. FaceNet: A Unified Embedding for Face Recognition and Clustering. https://arxiv.org/pdf/1503.03832.pdf
6. Lazarick, R. "Spoofs, subversion and suspicion: Terms and concepts." In Proc. NIST Int. Biometric Perform. Conf. (IBPC). 2012
7. Adversarial Examples: Attacks and Defenses for Deep Learning Xiaoyong Yuan, Pan He, Qile Zhu, Xiaolin Li∗ National Science Foundation Center for Big Learning, University of Florida. https://arxiv.org/pdf/1712.07107.pdf
8. Explaining and Harnessing Adversarial Examples. Ian J. Goodfellow, Jonathon Shlens & Christian Szegedy Google Inc., Mountain View, CA. https://arxiv.org/pdf/1412.6572.pdf
9. Accessorize to a Crime: Real and Stealthy Attacks on State-of-the-Art Face Recognition Mahmood Sharif Carnegie Mellon University Pittsburgh, https://www.cs.cmu.edu/~sbhagava/papers/face-rec-ccs16.pdf
10. Towards Evaluating the Robustness of Neural Networks Nicholas Carlini David Wagner University of California, Berkeley, https://arxiv.org/pdf/1608.04644.pdf
11. Face ID Security, Apple. https://www.apple.com/business/docs/site/FaceID Security_Guide.pdf
12. I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, 2016.
13. Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard and L. D. Jackel: Backpropagation Applied to Handwritten Zip Code Recognition, Neural Computation, 1(4): pp. 541-551, Winter 1989.
14. Convolutional Neural Networks (LeNet). DeepLearning 0.1. LISA Lab. http://deeplearning.net/tutorial/lenet.html
15. M. Masakazu, K. Mori, Y. Mitari and Y. Kaneda, “Subject Independent Facial Expression Recognition with Robust Face Detection Using a Convolutional Neural Network”, Neural Networks 16, No. 5-6 (June 2003): pp. 555–559. DOI:10.1016/s0893-6080(03)00115-1.
16. Rumelhart David E., Geoffrey E. Hinton, and Ronald J. Williams. “Learning Internal Representations by Error Propagation” (September 1, 1985). doi:10.21236/ada164453.
17. Le Cun, Yann, Yoshua Bengio, and Geoffrey Hinton. “Deep Learning.” Nature 521, No. 7553 (May 2015): 436–444. DOI:10.1038/nature14539.
18. Chingovska, Ivana and Anjos, Andre and Marcel, Sebastien, On the Effectiveness of Local Binary Patterns in Face Anti-spoofing, IEEE BIOSIG 2012.

19. Ting K.M. Precision and Recall. In: Sammut C., Webb G.I. (eds) Encyclopedia of Machine Learning. Springer, Boston, MA, 2011.
20. C. Enyinna Nwankpa, W. Ijomah, Anthony Gachagan, and Stephen Marshall, Activation Functions: Comparison of Trends in Practice and Research for Deep Learning, 2018.
21. Diederik P. Kingma, Jimmy Ba. "Adam: A Method for Stochastic Optimization". Submitted on 22 Dec 2014 (v1), last revised 30 Jan 2017 (this version, v9). https://arxiv.org/abs/1412.6980.
22. On Loss Functions for Deep Neural Networks in Classification, Katarzyna Janocha, Wojciech Marian Czarnecki, Faculty of Mathematics and Computer Science, Jagiellonian University, Krakow, Poland, 2017
23. Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen et al, TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org
24. Bradski G., The OpenCV Library, Dr. Dobb's Journal of Software Tools, 2000.
25. Alexander B. Jung, Imgaug, https://github.com/aleju/imgaug,2018
26. Kimball A.W., "Errors of the Third Kind in Statistical Consulting", Journal of the American Statistical Association, Vol.52, No.278, (June 1957), pp. 133–142.
27. Mark H. Zweig' and Gregory Campbell, Receiver-Operating Characteristic (ROC) Plots: A Fundamental Evaluation Tool in Clinical Medicine, 1993.
28. S. Rassomakhin, A. Kuznetsov, V. Shlokin, I. Belozertsev and R. Serhiienko, "Mathematical Model for the Probabilistic Minutia Distribution in Biometric Fingerprint Images", 2018 IEEE Second International Conference on Data Stream Mining & Processing (DSMP), Lviv, Ukraine, 2018, pp. 514-518. DOI: 10.1109/DSMP.2018.8478496
29. A. Kuznetsov, A. Kiyan, A. Uvarova, R. Serhiienko and V. Smirnov, "New Code Based Fuzzy Extractor for Biometric Cryptography", 2018 International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T), Kharkiv, Ukraine, 2018, pp. 119-124. DOI: 10.1109/INFOCOMMST.2018.8632040
30. Bondarenko S., Liliya B., Oksana K., and Inna G. (2019). Modelling instruments in risk management. International Journal of Civil Engineering and Technology, 10(1), pp. 1561-1568.
31. Runovski K., and Schmeisser. On the convergence of Fourier means and interpolation means. Journal of Computational Analysis and Applications, 6(3), 2004, pp. 211-227.
32. Al-Azzeh J.S., Al Hadidi M., Odarchenko R., Gnatyuk S., Shevchuk Z., Hu Z. Analysis of self-similar traffic models in computer networks, International Review on Modelling and Simulations, № 10(5), pp. 328-336, 2017.
33. Chornei R., Hans Daduna, V. M., and Knopov P. Controlled Markov fields with finite state space on graphs. Stochastic Models, issue 21(4), 2005, pp. 847-874. DOI:10.1080/15326340500294520
34. R. Odarchenko, V. Gnatyuk, S. Gnatyuk, A. Abakumova, Security Key Indicators Assessment for Modern Cellular Networks, Proceedings of the 2018 IEEE First International Conference on System Analysis & Intelligent Computing (SAIC), Kyiv, Ukraine, October 8-12, 2018, pp. 1-7.
35. Molodetska K., Brodskiy Yu., Fedushko S. Model of Assessment of Information-Psychological Influence in Social Networking Services Based on Information Insurance. COAPSN-2020. CEUR. Vol 2616, 2020. p.187-198. http://ceur-ws.org/Vol-2616/paper16.pdf
36. Tkach, B. P., Urmancheva, L. B. Numerical-analytic method for finding solutions of systems with distributed parameters and integral condition. Nonlinear Oscillations, 12(1), 2009, pp. 113-122. DOI:10.1007/s11072-009-0064-6