# Fingerprint Detection using CNN

K. Abhijith Saralaya, T Prasanna Lakshmi, Sudeepthi Devasari, R Rajini, Srigiri Shree Puruhoothi

Department of Computer Science Engineering

Gokaraju Rangaraju Institute of Engineering and Technology

**Abstract.**
For biometric recognition, fingerprints are often used. Spoofing attacks based on a synthetic fingerprint, on the other hand, are frequent. We present in this research a method for detecting fingerprints that employ guided filtering and hybrid image analysis. When examining a denoised image, the problem of neglecting the contribution of sharp features is addressed in this study. The method described in this project uses the increased sharp features as well as the denoised features from the hybrid images to get better outcomes**.**

**Keywords:** Hybrid image analysis, Guided filtering, Convolutional Neural Networks.

## 1. INTRODUCTION:

Biometric techniques are used to recognize people based on a physical or behavioral trait. Biometric information differs from one another. As a result, biometric technology is widely employed for personal identity verification. Furthermore, biometric technology has been successfully employed for over a decade. Many biometric traits, ranging from fingerprints to faces, Iris, veins, and blood flow, have been utilized in the security field in the past few years. The fingerprint has become the most extensively used of these features nowadays due to its uniqueness and ease of capture.

Many applications exist for the fingerprint-based system, including fingerprint login, fingerprint transaction, and second-generation authenticity validation. However, because of their extensive use, counterfeit assaults on fingerprint authentication systems are becoming more common, and the devices are subject to attacks. The generalized fraud technique uses latex, gelatin, silicone, play-both, and other materials to create imitation copies and duplicate actual user fingerprint identification. Many fingerprint recognition algorithms have been developed to discern whether a captured fingerprint image is authentic or not to tackle spoofing attempts.

The design should meet the following needs:

- We propose an approach that uses the sharp as well as the denoised characteristics from hybrid images to remove unnecessary noise.
- To make the most use of the foreground image, our suggested model uses a simple image cropping technique (where image cropping is conducted).

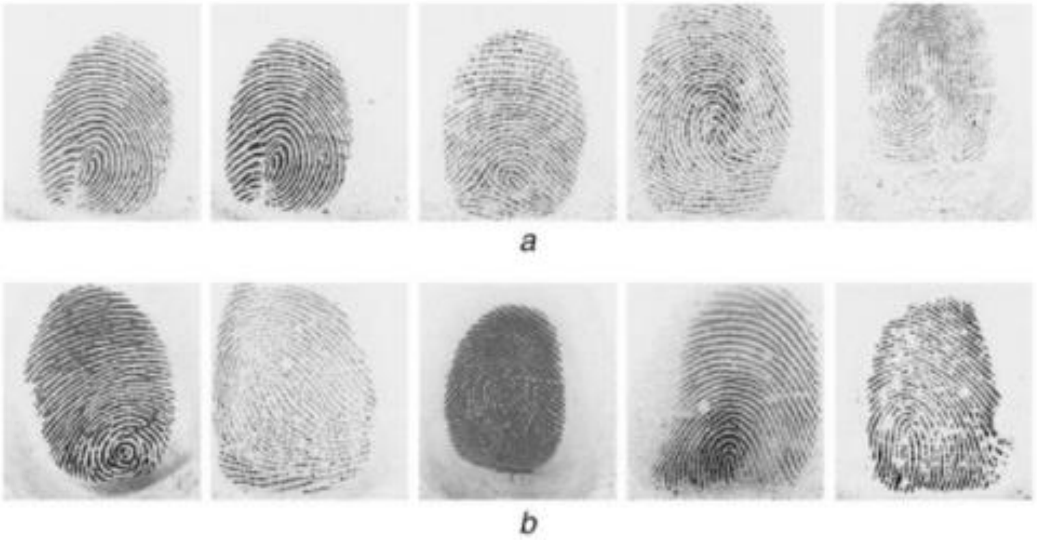


**Fig. 1.** a) Real fingerprint images

b) Fake fingerprint images

**Existing System:**

There are numerous ways in the existing system, such as employing several texture operators for feature concatenation of the denoised image, using multi-feature fusion, and another based on wavelet analysis and LBP. All of these methods have their own set of flaws that lead to erroneous outcomes.

**Drawbacks:**

• Image denoising causes blurred features and the loss of sharp information, which might affect accuracy.

• Multi-feature fusion has a larger dimensionality, resulting in higher verification and recognition time complexity.

• Wavelet analysis produced residual images that had redundant information that was not relevant.

**Proposed system:**

The current approach proposes a fine-grained feature fusion structure in which ROI (region of interest) extraction is used to execute image pre-processing (reshaping, resizing, pixel and color conversion), and then guided filtering is used to create a denoised image. (A guided filter is a type of edge-preserving smoothing filter that can also be used to remove noise or texture while maintaining crisp edges.).

We use CNN to improve fingerprint detection results by describing the increased sharp features as well as the denoised features from the hybrid images.

**Advantages:**

• Experiments show that our suggested method outperforms the current system in terms of accuracy.

•CNN can identify relevant qualities without the need for human interaction, resulting in higher accuracy.

## 2. LITERATURE REVIEW:

**MenottiD.ChiachiaG.PintoA. et al.: 'Deep representations for iris, face, and fingerprint spoofing detection', *IEEE Trans. Inf. Forensics Sec.*, 2014, 10, (4), pp. 864–879.**

- The first method involves learning appropriate convolutional network topologies for each domain, while the second method uses backpropagation to learn the network's weights.
- In this method, we use a combination of the two learning methods to develop deep representations for nine biometric faking standards, each of which contains real and fake samples of a certain biometric modality and attack type.
- This technique not only improves understanding of how different methodologies interact but also results in systems that outperform the best-known outcomes in eight of the nine performance metrics.
- The findings strongly suggest that spoofing detection systems based on convolutional networks can be resistant to known attacks and easily adapted to image-based techniques that have yet to be discovered.

**DubeyR.K.GohJ.ThingV.L.L.: 'Fingerprint liveness detection from a single image using low-level features and shape analysis', *IEEE Trans. Inf. Forensics Sec.*, 2016, 11, (7), pp. 1461–1475.**

- Fingerprint liveness detection (FLD), a proposed anti-fraud technique, has been investigated to ensure that authorized users' fingerprint data is not used fraudulently.

- Unlike, traditional techniques, the deep convolutional neural network (DCNN) can automatically learn deep semantic detail using a supervised learning algorithm without the need for any expert background knowledge.

- However, one flaw in most CNN models is that fixed-scale images are required in the input layer. Although cropping or scaling approaches can be employed to tackle the scale problem by transforming a picture of any scale into a fixed scale, they may result in the loss of critical textural detail and a decrease of picture quality, lowering the classifier model's generalization capability.

**ZhangY.FangS.XieY. et al.: ‘ Fake fingerprint detection based on wavelet analysis and local binary pattern’. Biometric Recognition, Shenyang, People's Republic of China, 2014, pp. 191–198.**

- This study uses a unique software-based liveness recognition approach based on the uniform local binary pattern (ULBP) in a spatial pyramid to distinguish fingerprint liveness.
- Each fingerprint must first be processed.
- This study then uses three-layer spatial pyramids of fingerprints to solve image orientation and size invariance.
- A consistent local binary pattern is then used to extract the features from provided fingerprints, resulting in texture features for three layers of spatial pyramids.

## 3. METHODOLOGY

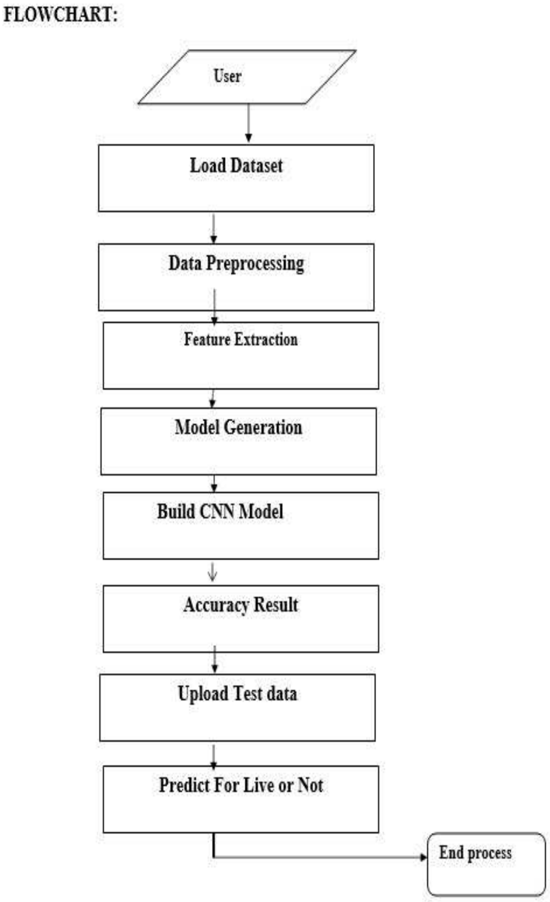


**Fig. 2.** Flowchart

**Dataset collection:**

We took the standard dataset from the 2013 Liveness Detection Competition and used it in our project. The process of uploading the dataset begins here.

**Preprocessing**

- we start with the image reading process, which is followed by the image resize or cropping process. For the CNN operation, the photos were scaled to keep a constant aspect ratio of one with (128, 128) pixel size.
  Then conversion of an image into grey or black color. Next, we reshape the image into (-1,3) and Pixel conversion is done.

**Model Training, Testing, and Prediction:**

The CNN classification is done layer by layer.

**CNN Algorithm:**

A linear stack of layers was used to develop the Convolutional Neural Network (CNNs or ConvNets) for image categorization and recognition. Convolutional layers with kernel filters, max pooling, and fully linked layers were used to process training and testing pictures.

**CNN Architecture:**

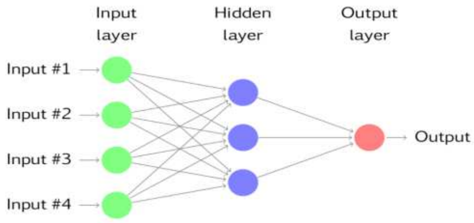


**Fig. 3.** CNN Architecture

1. The CNN is made up of three types of layers: convolutional layers, pooling layers, and fully-connected (FC) layers. When these layers are stacked, a CNN architecture is created.

2. Convolution Layer:
   This core layer retrieves the different features from an input image. This layer conducts the mathematical convolution between the input image and a filter of size MxM. In terms of the filter

size, sliding the filter across the input image produces the dot product between the filter and the sections of the input image. The Feature map, which contains information about the image's corners and edges, is the result. Following that, the feature map is passed on to succeeding layers, which use it to learn a variety of other features from the input image.

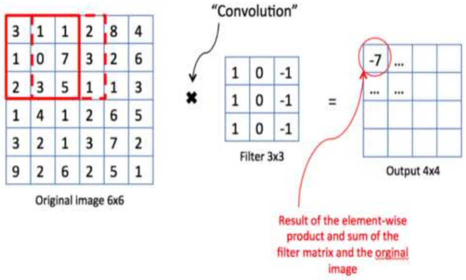


**Fig. 4.** CNN - Convolution Layer

3. Pooling Layer:
   Pooling refers to a little percentage of the input, thus we take a small portion of the input and try to average it out, referred to as average pooling, or take the maximum value, referred to as max pooling, so when we pool an image, we're calculating a total value based on all of the values present.

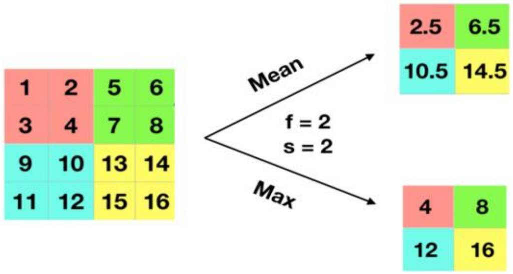


**Fig. 5.** CNN – Pooling layer

4. Fully Connected Layer:
   In this stage, the input images from the subsequent layers are flattened and sent to the FC layer. The flattened vector is then transmitted via a few more FC layers, where the mathematical functional operations are generally carried out. At this moment, the classification process begins.

**Activation Function:**

In a Neural Network, the activation function is a node that is either at the end or in the middle. They aid in determining whether or not a neuron will fire.

**Rectified Linear Unit (ReLU):**

The rectified linear activation function, or ReLU, is a linear function that outputs the input directly if it is positive but zeroes otherwise. It has become the default activation function for plenty of types of neural networks since it is faster to train and generally generates better performance.

$$f(u) = \max(0, u)$$

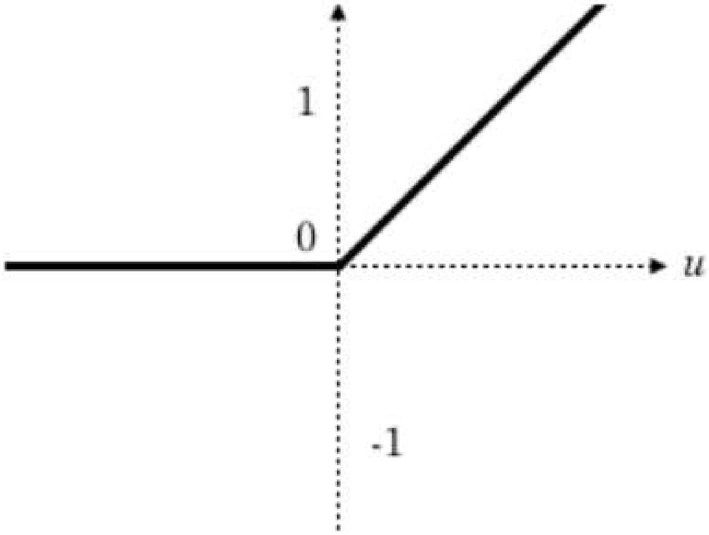


**Fig. 6.** ReLU

**Softmax:**

Integers and logits are converted to probabilities using the Softmax function. A Softmax generates a vector (let's call it v) containing the probability of each possible outcome. For all potential outcomes or classes, the probability in vector v sum to one.

Mathematically, Softmax is defined as,

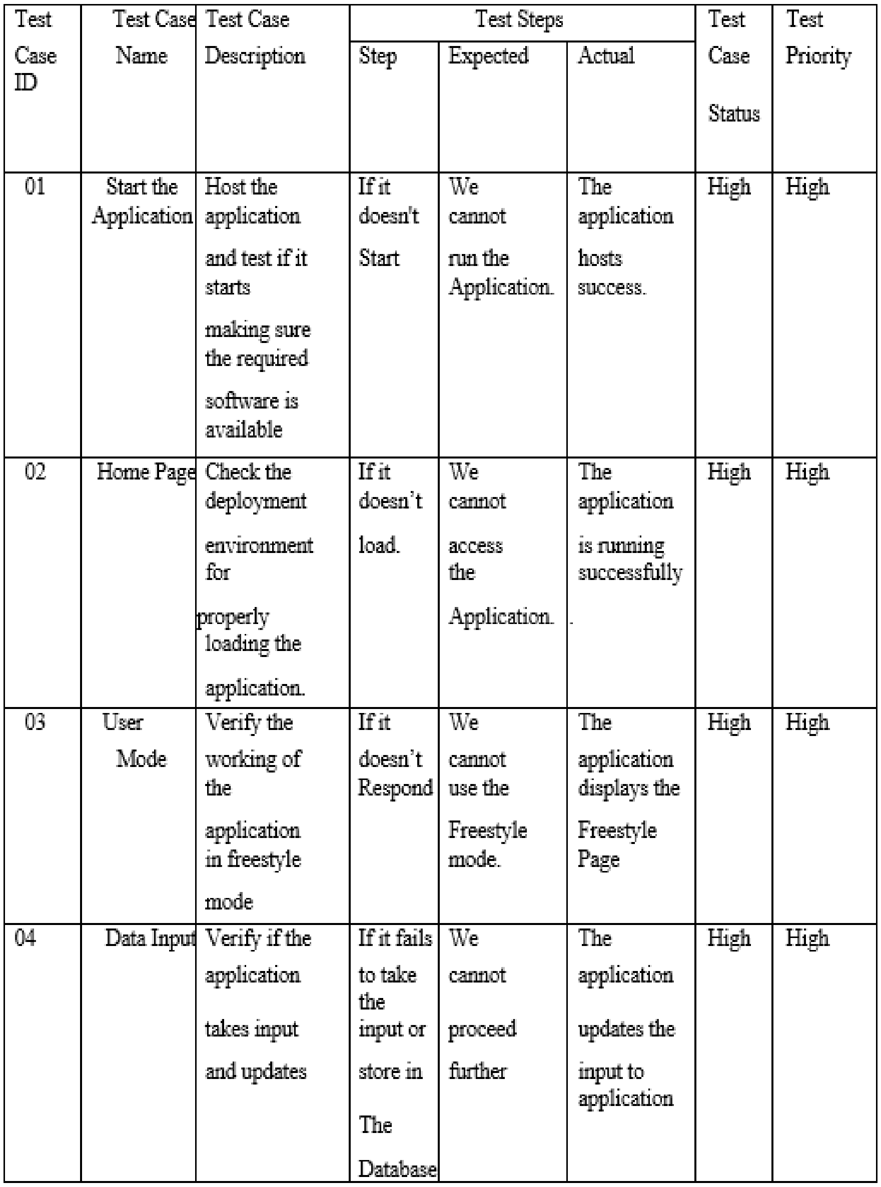
Softmax function,

$$\varphi(\eta) = \left(\frac{\exp(\eta_1)}{\sum_{i=1}^{k}\exp(\eta_i)}, \dots, \frac{\exp(\eta_k)}{\sum_{i=1}^{k}\exp(\eta_i)}\right)$$

**Web Deployment:**

We used the Tkinter library for the graphical user interface and pickle module to deploy the web applications.

## Testing :



| Test Case ID | Test Case Name | Test Case Description | Test Steps |  |  | Test Case Status | Test Priority |
|---|---|---|---|---|---|---|---|
|  |  |  | Step | Expected | Actual |  |  |
| 01 | Start the Application | Host the application and test if it starts making sure the required software is available | If it doesn't Start | We cannot run the Application. | The application hosts success. | High | High |
| 02 | Home Page | Check the deployment environment for properly loading the application. | If it doesn't load. | We cannot access the Application. | The application is running successfully . | High | High |
| 03 | User Mode | Verify the working of the application in freestyle mode | If it doesn't Respond | We cannot use the Freestyle mode. | The application displays the Freestyle Page | High | High |
| 04 | Data Input | Verify if the application takes input and updates | If it fails to take the input or store in The Database | We cannot proceed further | The application updates the input to application | High | High |

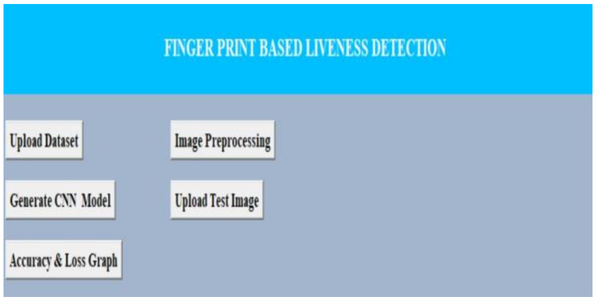


**Fig. 7.** Application user interface

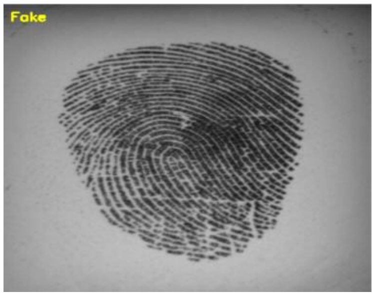


**Fig. 8.** Predicted as fake image

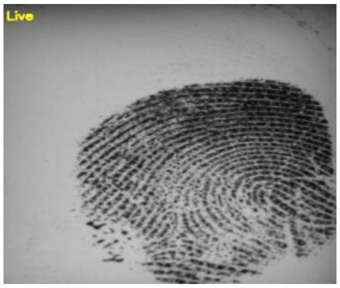


**Fig. 9.** Predicted as a real image

**Results:**

Our model, which employs CNNs, has an accuracy of 99.13 percent.

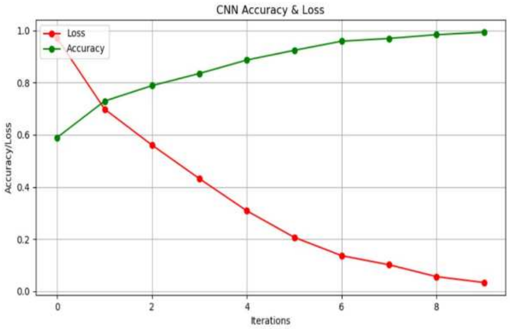


**Fig. 10.** Graph for output prediction

| Classifier | Accuracy |
|---|---|
| Support Vector Machine | 95.12% |
| CNN | 99% |

**Fig. 11.** Classifiers accuracy comparison

**Conclusion:**

In this paper, we used CNN to distinguish between fake and real fingerprints. Our proposed method has an accuracy of 99.13%, which is higher than the existing methods. In our experiment, the output is predicted by retaining the sharp features of the original images.

**Future scope:**

In the future work, we plan to perform further experiments in the two elements of feature selection and filter selection for fingerprint detection

## REFERENCES:

**1.** Menotti, D., Chiachia, G., Pinto, A., et al.: 'Deep representations for iris, face, and fingerprint spoofing detection', IEEE Trans. Inf. Forensics Sec., 2014, 10, (4), pp. 864–879
**2.** Dubey, R.K., Goh, J., Thing, V.L.L.: 'Fingerprint liveness detection from single image using low-level features and shape analysis', IEEE Trans. Inf. Forensics Sec., 2016, 11, (7), pp. 1461–1475
**3.** Zhang, Y., Fang, S., Xie, Y., et al.: 'Fake fingerprint detection based on wavelet analysis and local binary pattern'. Biometric Recognition, Shenyang, People's Republic of China, 2014, pp. 191–198
**4.** Nogueira, R.F., Lotufo, R.D.A., Machado, R.C.: 'Fingerprint liveness detection using convolutional neural networks', IEEE Trans. Inf. Forensics Sec., 2016, 11, (6), pp. 12061213
**5.** Schuckers, S., Abhyankar, A.: 'Detecting liveness in fingerprint scanners using wavelets: results of the test dataset'. Biometric Authentication, ECCV 2004 Int. Workshop, BioAW 2004, Prague, Czech Republic, 2004, vol. 3087, pp. 100–110
**6.** Schuckers, S.A.C., Parthasaradhi, S.T.V., Derakshani, R., et al.: 'Comparison of classification methods for time-series detection of perspiration as a liveness test in fingerprint devices', IEEE Trans. Syst. Man Cybern. C, 2005, 35, (3), pp. 335–343
**7.** Yambay, D., Ghiani, L., Denti, P., et al.: 'Livdet 2011—fingerprint liveness detection competition 2011'. Proc. 5th IAPR Int. Conf. Biometrics (ICB), New Delhi, India, March/April 2012, pp. 208–215
**8.** Ghiani, L., Yambay, D., Mura, V., et al.: 'Livdet 2013 fingerprint liveness detection competition 2013'. Proc. IAPR Int. Conf. Biometrics, Madrid, Spain, June 2013, pp. 1–6
**9.** Mura, V., Ghiani, L., Marcialis, G., et al.: 'Livdet 2015 fingerprint liveness detection competition 2015'. Proc. IEEE Int. Conf. Biometrics Theory, Applications and Systems, Arlington, VA, USA, September 2015, pp. 1–6
**10.** Kim, S., Park, B., Song, B.S., et al.: 'Deep belief network based statistical feature learning for fingerprint liveness detection', Pattern Recognit. Lett., 2016, 77, (C), pp. 58–65
**11.** Pala, F., Bhanu, B.: 'Deep triplet embedding representations for liveness detection', in Bhanu, B., Kumara (Ed.): 'Deep learning for biometrics', Advances in Computer Vision and Pattern Recognition (Springer, Cham, Switzerland, 2017), pp. 287–307
**12.** Yosinski, J., Clune, J., Bengio, Y., et al.: 'How transferable are featured in deep neural networks?', Neural Information Processing Systems, Eprint Arxiv, Montreal, Canada, 2014, vol. 27, pp. 3320–3328
**13.** Manjunath, B.S., Ma, W.Y.: 'Texture features for browsing and retrieval of image data', IEEE Trans. Pattern Anal. Mach. Intell., 1996, 18, (8), pp. 837– 842

**14.** Van Der Maaten, L., Hinton, G.E.: ‘Visualizing data using t-SNE’, J. Mach. Learn. Res., 2008, 9, (2605), pp. 2579–2605
**15.** Tan, G., Chen, H., Qi, J.: ‘A novel image matting method using sparse manual clicks’, Multimedia Tools Appl., 2016, 75, (17), pp. 10213–10225
**16.** Tan, G., Qi, J., Gao, C., et al.: ‘Saliency-based unsupervised image matting’, Int. J. Pattern Recognit. Artif. Intell., 2014, 28, (4), pp. 759–775
**17.** He, K., Sun, J., Tang, X.: ‘Guided image filtering’, IEEE Trans. Pattern Anal. Mach. Intell., 2013, 35, (6), pp. 1397–1409
**18.** Peng, F., Qin, L., Long, M.: ‘POSTER: non-intrusive face spoofing detection based on guided filtering and image quality analysis’. Security and Privacy in Communication Networks, Guangzhou, People's Republic of China, 2016, pp. 774–777
**19.** Zuiderveld, K.: ‘Contrast limited adaptive histogram equalization’, Graphics Gems (Academic Press, San Diego, CA, United States, 1994), pp. 474–485
**20.** Nosaka, R., Yasuhiro, O., Kazuhiro, F.: ‘Feature extraction based on co-occurrence of adjacent local binary patterns’. Pacific-Rim Symp. on Image and Video Technology, Gwangju, Republic of Korea, 2011, pp. 82–91
**21.** Hinton, G.: ‘Stochastic neighbor embedding’, Adv. Neural. Inf. Process. Syst., 2002, 15, (4), pp. 833–84