Combining Deep Learning Technologies with Multi-Level Gabor Features for Facial Recognition in Biometric Automated Systems

Catalin-Mircea DUMITRESCU*, Ioan DUMITRACHE

University POLITEHNICA of Bucharest, 313 Splaiul Independenței, Bucharest, Romania
dumitrescu.catalin.m@gmail.com (**Corresponding author*), ioan.dumitrache@acse.pub.ro

Abstract: Face recognition is one of the most important abilities that humans use in their daily lives. It represents a natural, robust and non-intrusive manner for identifying individuals. However, it is also a very challenging problem in the field of computer vision and pattern recognition. A good face recognition algorithm should be able to automatically detect and recognize a face in an image, regardless of lightning, expression, illumination and pose. In this paper, we present a novel approach for the face model representation and matching issues in face recognition. Our approach is based on multi-level Gabor features and Deep Learning techniques. In the experiments presented in this paper, ORL, Caltech, Yale and Yale B databases were used in order to obtain the face recognition rate. The results show that the new face recognition algorithm outperforms the conventional methods such as global Gabor face recognition based on PCA in terms of recognition rate.

Keywords: Face Recognition, Multi-level Gabor features, Diabolo Networks, Auto-encoders, ORL (Olivetti Research Laboratory).

1. Introduction

A biometric marker (trait) can be defined as an anatomic attribute (*face*, *fingerprint*, *DNA*, *iris* etc.) or behavioural characteristic (*voice*, *signature* etc.), which can be used alone, or in combination with other traits, in order to identify / recognize a person, a prerequisite in many applications such as control, defence, banking and so on.

The decrease in the production costs of biometric sensors, combined with the advances registered by *Information and Communication Technology* (*ICT*) and mobile devices, stimulated the research and development of *Automated Biometric Recognition Systems* (*ABRS*).

Thus, according to Acuity Market Intelligence (2014) the *ABRS* market in mobile context will grow from 1.6 billion $, in 2014, to 34.6 billion $ in 2020, each year, about 800 billion transactions requiring different levels of biometric authentication.

A ***Facial Recognition System*** (***FRS***) is a computer application capable of identifying an individual from a digital image or video frame.

Face recognition represents a non-intrusive and user-friendly biometric security technique (it is the least invasive method); thus, it has a variety of potential applications in public security, law enforcement, e-commerce, access control, information security and video surveillance.

Facial biometrics have been integrated with consumer electronics in order to facilitate authentication, replacing the usage of passwords. Using data from cameras, a computer / mobile phone can be unlocked when the user stands in front of it.

FRS solutions can be used to authenticate the right of electronic access to *Collaborative Decision Support Systems* (*CDSSs*). Under the influence of the new technologies, *CDSSs* have evolved from few people located in a "decision room" to an unlimited number of participants over a distributed architecture (Filip, Zamfirescu & Ciurea, 2017). Thus, the identification of the participants has become an important topic.

Face recognition combined with *Artificial Intelligence* (*AI*) is also set to change the face of the *Advertising Sector*. The retailer Tesco intends to install hi-tech *OptimEyes* (Amscreen, 2018) screens in order to deliver targeted ads to customers after identifying the age range and gender of the waiting customers.

The biggest problem of *automatic facial recognition systems* is that, the recognition rate is heavily affected by changes in illumination, pose and facial expressions. So, it is difficult to design a system sufficiently simple to handle all these problems.

The most popular global facial feature extraction methods are: *Principal Component Analysis* (*PCA*), *Linear Discriminant Analysis* (*LDA*), *Independent Component Analysis* (*ICA*), *Gabor*

https://doi.org/10.24846/v28i2y201910

filters combined with *Local binary pattern analysis* (Lei et al., 2011), *Gabor Wavelet* (Zhou & Wei, 2006) (Vinay & Shreyas, 2006).

Zhou & Wei (2006) presented a new facial feature extraction algorithm based on *Gabor wavelets* and *AdaBoost*. Vinay & Shreyas (2006) described the usage of *Gabor wavelets* for efficient face representation.

Li, Gao & Wang (2017) presented a face recognition algorithm based on ***auto-encoder*** networks with dropout. Using a 4-layer network with 784-800-800-40 neuron structure, the system achieved a 97.5% recognition rate on the **ORL** (Samaria & Harter, 1994) dataset, when 90% of the data was used for training and 10% for testing.

In many computer vision and image processing applications, the algorithms based on features constructed from responses of *Gabor filters*, also known as ***Gabor features***, are among the top performers. Some top biometric systems based on ***Gabor features*** are: Daugman's (Daugman J. G., 1993) iris recognition system, Jain's (Jain, Chen & Demirkus, 2007) fingerprint matching algorithm, face recognition systems (Messer et al., 2004).

Gabor features extract local pieces of information which are then combined to recognize an object or a region of interest.

In this paper, a new face recognition algorithm that combines ***Deep Learning*** techniques with ***multi-level Gabor features*** is proposed.

First, for each face image the ***Gabor features*** are extracted using a ***multi-level*** approach. Then, the user's **face model** is generated using ***Deep Learning*** techniques. In the recognition phase, the classifier compares the feature vectors of a sample image with the facial models that are learned during training, and selects the model with the maximum likelihood value.

The rest of paper is organized as follows: a short review of the *Gabor wavelet* filter is described in **Section 2**. In **Section 3**, the *Neural Networks* used for subject modelling are presented. The proposed algorithm is described in **Section 4**. In **Section 5**, experimental results are shown. Finally, the conclusions are presented in **Section 6**.

2. Gabor Wavelet Filters

First introduced in the 40s by the Hungarian-born engineer Denis Gabor (Gabor, 1946), the *Gabor functions* or *atoms* have been used in many different fields such: audio signal processing, image compression, edge detection, filter design, object recognition.

The ***Gabor Wavelet Filter*** represents a band-pass linear filter whose impulse response is defined by a harmonic function multiplied by a *Gaussian* function. Basically, it consists of a group of wavelets; each wavelet captures the energy at a specific frequency and in a specific direction. In image processing, ***Gabor Wavelet filters*** are used for features extraction in texture-based image analysis. This means that it basically analyses whether there is any specific frequency content in the image in a specific direction in a localized region around the point or region of analysis. These *Gabor wavelets* provide a complete image representation (Lee, 1996).

Marĉelja (1980) and Daugman (1985) advanced the idea that simple cells in the visual cortex of mammalian brains can be modelled using ***Gabor functions***. Thus, image analysis using ***Gabor filters*** is thought to be similar to perception in the human visual system. By presenting optimal localization properties in both spatial and frequency domain and thus, are well suited for texture segmentation, edge detection and image representation problems (Weldon, Higgins, & Dunn, 1996).

A *bi-dimensional* ***Gabor filter,*** $G_{f,\theta}(x,y)$ can be viewed as a *Gaussian kernel function* modulated by a sinusoidal plane wave of a particular frequency and orientation as follows:

$$G_{f,\theta}(x,y) = e^{-\left[\frac{x_{\theta_n}^2}{\sigma_x^2}+\frac{y_{\theta_n}^2}{\sigma_y^2}\right]} e^{i\left(2\pi f x_{\theta_n}+\varphi\right)} \quad (1)$$

where, $x_{\theta_n} = x\cos\theta_n + y\sin\theta_n$, $y_{\theta_n} = y\cos\theta_n - x\sin\theta_n$, f represents the central frequency of the sinusoidal wave, φ the phase, θ_n the orientation of the wave plane and σ_x / σ_y represent the standard deviations of the *Gaussian* envelope along the *X / Y* axes.

The filter has two components (**Figure 1**), a real and an imaginary component, representing the orthogonal directions:

$$RE_{f,\theta}(x\ddot{y}y) = e^{-\left[\frac{x_{\theta_n}^2}{\sigma_x^2}+\frac{y_{\theta_n}^2}{\sigma_y^2}\right]}\cos\left(2\pi f x_{\theta_n} + \varphi\right), \quad (2)$$

$$IM_{f,\theta}(x\ddot{y}y) = e^{-\left[\frac{x_{\theta_n}^2}{\sigma_x^2}+\frac{y_{\theta_n}^2}{\sigma_y^2}\right]}\sin\left(2\pi f x_{\theta_n} + \varphi\right) \quad (3)$$

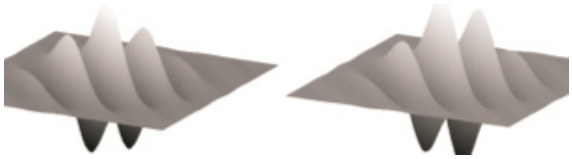

Figure 1. Example of Gabor filter (real and imaginary part) © (Štruc & Pavešić, 2010)

The *2D filters* defined by relation (1), represent a group of wavelets and can optimally capture both local orientation and frequency information from an image. By using $G_{f,\theta}(x,y)$, the image is filtered at various orientations, frequencies and standard deviations. Thus, in order to design a ***Gabor filter***, the phase, the orientations, the frequencies and the standard deviations must be defined.

The phase is set to $\varphi = \frac{\pi}{2}$ and the angle θ_n is defined by:

$$\theta_n = \frac{\pi}{p}(n-1), \quad (4)$$

where, p denotes the number of orientations and $n \in \{1...p\}$.

In this paper the real part of the ***Gabor*** representation is used for feature extraction. Thus, the ***2D Gabor filter bank*** is defined by equations (2) and (4).

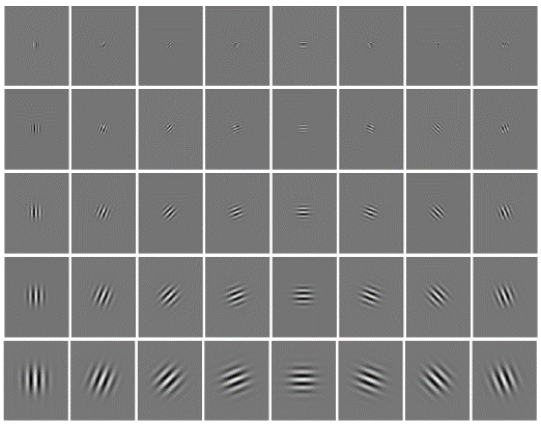

Figure 2. Real part of the used Gabor filters © (Štruc & Pavešić, 2010)

The most important step in designing ***Gabor filters*** is the selection of the filter parameters. This research uses a *40* channels (**Figure 2**) filter bank consisting of eight orientations $\theta \in \left\{0, \frac{\pi}{8}, \frac{2\pi}{8}, \frac{3\pi}{8}, \frac{4\pi}{8}, \frac{5\pi}{8}, \frac{6\pi}{8}, \frac{7\pi}{8}\right\}$, five spatial frequencies $f \in \left\{\frac{1}{4}, \frac{1}{4\sqrt{2}}, \frac{1}{8}, \frac{1}{8\sqrt{2}}, \frac{1}{16}\right\}$, and variance values $\sigma_x = \sqrt{2}/f$, $\sigma_y = \sqrt{2}/f$.

The ***Gabor feature representation*** of a *grey-scale* image $I(x,y) \in R^{m \times n}$, where $m \times n$ represents the image size in pixels, is obtained by convolving the input face image with the created ***Gabor filters*** (**Figure 3**).

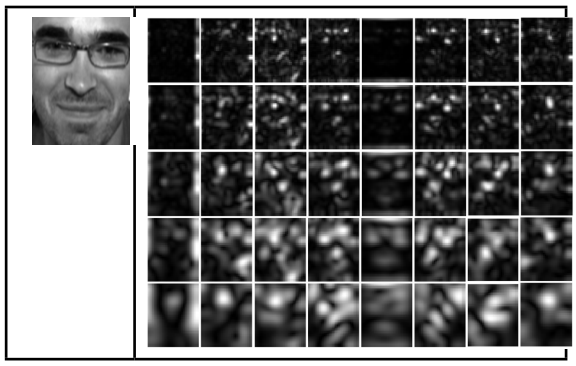

Figure 3. Gabor filter output

3. Deep Learning Using Auto-encoders

Normally, the neural networks used for classification are trained to map an input vector to an output that represents the known classification classes. An alternative approach is to use ***auto-associative*** neural networks, also called ***auto-encoders*** or ***Diabolo networks*** in order to learn a model of each class.

An ***auto-encoder***, ***auto-associator*** or ***Diabolo network*** (Bengio, 2009) is an artificial neural network used for unsupervised learning of efficient codings (Liou, Huang, & Yang, 2008) (Liou et al., 2014). Typically, the aim of an ***auto-encoder*** is to generate (learn) a compressed representation (also known as encoding) for a dataset, with the scope to obtain a dimensionality reduction. This initial concept has evolved, and ***auto-encoders*** have become widely used for learning generative models of data.

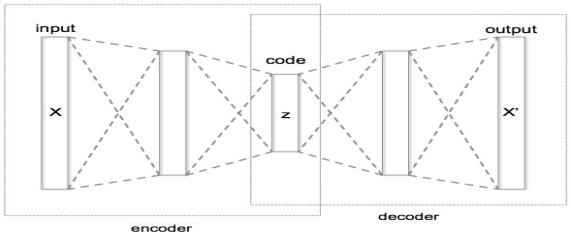

Figure 4. Schematic structure of an autoencoder with 3 fully-connected hidden layers

Figure 4 illustrates the basic architecture of an ***auto-encoder***. From the architectural point of view, the simplest form of an ***auto-encoder*** is a feedforward, non-recurrent neural network that is very similar to the multilayer perceptron (***MLP***). It consists of an input layer, an output layer and one or more hidden layers connecting them. The main difference between a ***MLP*** and a ***Diabolo network*** consists in the number of nodes contained in the output layer. In the case of ***Diabolo networks*** the output layer has the same number of nodes as the input layer.

The training of an ***auto-encoder*** is accomplished by using one of the many backpropagation algorithms (conjugate gradient method, steepest descent, etc.), with the purpose that the network reconstructs its own inputs.

An ***auto-encoder*** always consists of two parts, an ***encoder*** and a ***decoder***. The ***encoder*** and ***decoder*** can have multiple layers, but for simplicity each of them has only one layer. Being given an input $x \in \mathbb{R}^D$, the encoder will map it onto another vector $z \in \mathbb{R}^P$ as follows:

$$z = \sigma_1(Wx + b), \tag{5}$$

where: σ_1 is an element-wise activation function (transfer function) for the encoder, $W \in \mathbb{R}^{D \times P}$ is a weight matrix and $b \in \mathbb{R}^P$ is a bias vector.

By using the decoder, the encoded representation z is mapped onto the reconstruction vector x', an estimate of the original input vector x.

$$x' = \sigma_2(W'z + b'), \tag{6}$$

where: σ_2 is the transfer function for the decoder, $W' \in \mathbb{R}^{D \times P}$ is a weight matrix and $b' \in \mathbb{R}^D$ is a bias vector.

Because an ***auto-encoder*** is designed to replicate the input data to the outputs, the biggest risk is that it will learn the identity function, the trivial case. In order to prevent this situation, different constraints can be placed on the network. The simplest one consists in limiting the number of hidden neurons: e.g. if the input vector has 100 elements, then the hidden layer has 50 elements. With this constraint the encoder will compress the 100-element vector into a 50-element vector and the decompression phase will expand the 50-element vector into a 100-element vector that is ideally close to the original input. This simple **auto-encoder** often ends up learning a low-dimensional representation of the inputs very similar to ***PCA*** (in the case that the input data is correlated).

Another type of constraint, more often used, is the "**sparsity constraint**" or "**sparsity parameter**". In this case the hidden layer has a large number of neurons (compared with the number of inputs), but for each input only a small number of them will be activated (it produces an activation value closer to "1") during the training of the network. Such encoders are called "**sparse auto-encoder**" (Olshausen & Field, 1997) and they produce a sparse representation of the inputs. The "**sparsity parameter**" is a small value close to zero (e.g. *0,05*) and it represents the average activation value, for any hidden node, over all the training samples.

4. The Proposed Algorithm

The algorithm proposed in this paper (presented in **Figure 5**) consists of the following stages: multi-level feature vectors extraction using ***Gabor filters***, input data model generation using **auto-encoders** and unknown subject's identification. The algorithm is designed to use g*rey*-scale images.

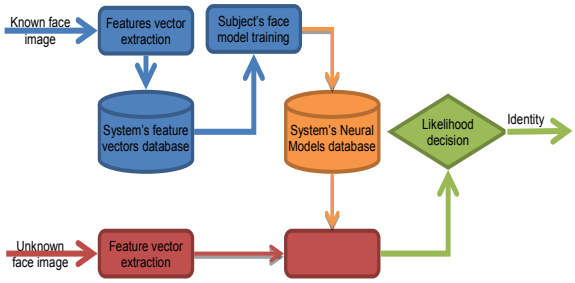

Figure 5. Block diagram of the proposed face recognition procedure

The "*Feature Vector Extraction*" phase generates the feature vector from each known subject image using ***Gabor filters*** in a *multi-level* approach (**Figure 6**).

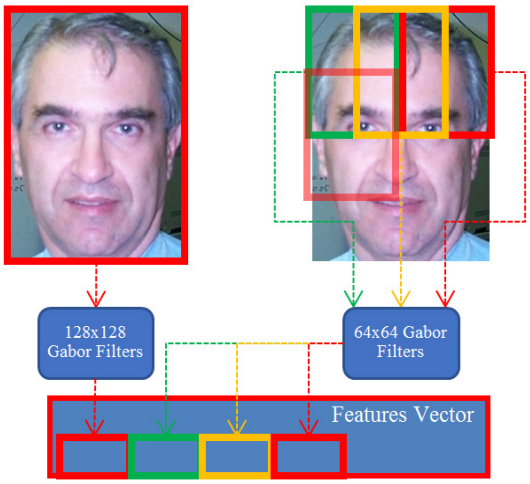

Figure 6. Feature Vector Extraction

First, the image is converted to grey-scale, if necessary, by converting the ***RGB*** (*red – green – blue*) colour space to ***HSV*** (*hue – saturation – value*) colour space and the keeping only the ***V***; and afterwards, is resized to *128x128* pixels.

From the resulted image the first part of the feature vector is extracted by applying the first set of ***Gabor filters***. A total of *40* outputs are obtained, each output with the size of *128x128* pixels. In order to overcome the dimensionality issue, the outputs are down-sampled by a factor of *6x6,* superimposing over the image a rectangular sampling grid and retaining only the values located under the sampling grid's nodes. And finally, the down-sampled outputs are normalized in order to have zero mean and unit variance.

For the second part of the feature vector extraction, the image is divided into blocks of *64x64* pixels with an overlap between the blocks of *32* pixels. Each image block is then convoluted with the second ***Gabor filter*** bank; the resulted output is down-sampled with a factor of *6x6* and normalized. The **final feature vector** consists of ***53640*** elements, calculated as follows:

$$40 \times \left\lfloor \frac{128}{6} \right\rfloor \times \left\lfloor \frac{128}{6} \right\rfloor + 9 \times 40 \times \left\lfloor \frac{64}{6} \right\rfloor \times \left\lfloor \frac{64}{6} \right\rfloor, \quad (7)$$

Both filter banks have the same orientations and frequencies. The difference between them is given by the size of the used filters: while the first bank uses filters of *128x128*, the second filter bank has a size of *64x64*.

By applying a two-stage feature extraction technique, the resulted feature vector contains two types of features: *full-face* features and *local* features. The first ***Gabor filter*** bank generates a *full-face* presentation of the subject, while the second ***Gabor filter*** bank is focused on extracting relevant information around major facial features (such as eyes, mouth, nose).

The second step of this algorithm consists in "*Data Model generation*" with the help of a stacked neural network architecture combining a ***Diabolo network*** and **a *SoftMax layer*** (**Figure 7**). In the first step the ***auto-encoder*** network is trained using the feature vectors extracted in the previous step. The ***hidden layer*** of the ***auto-encoder*** will map the input data to a reduced subspace that represents the compressed version of the input. For training the ***auto-encoder***, the selected algorithm is the scaled conjugate gradient descent (***SCG***) backpropagation.

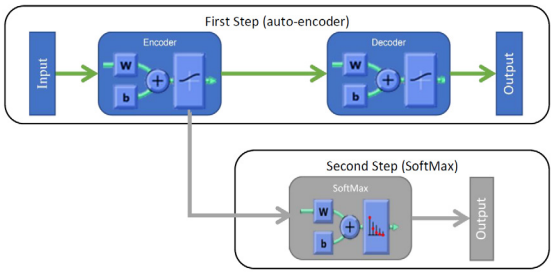

Figure 7. Stacked Architecture

The ***encoder*** of the trained network is used to generate the feature vectors for the final step of the algorithm. In this stage a ***SoftMax*** network is trained to classify the compressed feature vectors into different subject classes. The number of the output classes of the ***SoftMax*** layer is equal to the number of the known subjects.

5. Experimental Results

All experiments were performed using one colour face database **Caltech** 101 (Weber, 1999) and three grey-scale face databases **Yale Face** (Bellhumer, Hespanha & Kriegman, 1997), **Extended Yale B** (Georghiades, Belhumeur, & Kriegman, 2001) and **ORL** (Samaria & Harter, 1994). The Caltech database contained *450* frontal face colour images (*896x592* pixels) of *26* unique subjects with different lighting, expressions and backgrounds (**Figure 8**). The face regions from the images were extracted using a ***Viola-Jones*** (Viola & Jones, 2001) face detector. Some images

had very poor lighting conditions (were too dark or too bright) and were excluded. After the face segmentation, more than *200* faces of *26* unique individuals remained.

Figure 8. Sample images from **Caltech** database

The **Yale Face** database consisted of *11* different *grey-scale* pictures of 15 unique individuals. The images were acquired with several configurations and facial expressions: normal, centre-light, with glasses, happy, left-light, no glasses, right-light, sad, sleepy, surprised, and wink (**Figure 9**).

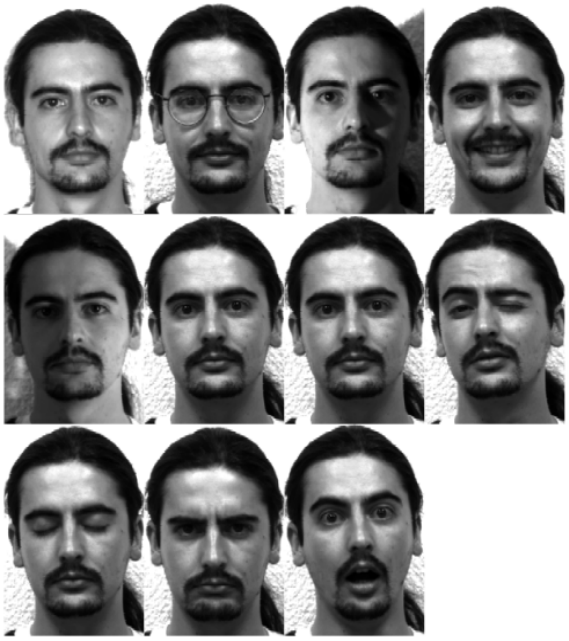

Figure 9. Sample images from **Yale** database

The extended **Yale Face Database B** contained *16128 grey-scale* images of *28* human subjects under *9* poses and *64* illumination conditions, resulting *576* viewing conditions for every subject. The images were aligned, cropped and resized. The nine poses recorded were as follows: pose 0 was the frontal pose; poses 1, 2, 3, 4, and 5 were about 12 degrees from the camera optical axis, while poses 6, 7, and 8 were about 24 degrees. In this paper, only a subset of the database was used, consisting of 63 pictures per subject under the following conditions: nine poses (0 to 8), five azimuth positions (0, +0.5, -0.5, +1, -1 degrees) and three elevation positions (0, +10, -10 degrees) (see **Figure 10**).

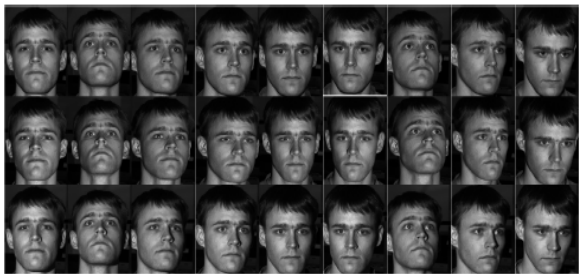

Figure 10. Sample images from **Yale B** database

The **ORL** (Olivetti Research Laboratory) face database contained a set of *grey-scale* images of faces taken between April 1992 and April 1994. There were *10* different images of *40* distinct subjects. **Figure 11** illustrates some of the images from the database. The images were taken at different times, varying slightly, with different facial expressions (open / closed eyes, smiling / non-smiling) and facial details (glasses/ no-glasses). All the images were taken against a dark homogeneous background and the subjects were in up-right, frontal position.

Figure 11. Sample images from **ORL** database

By running the experiments with different configuration parameters and measuring the error rates, the optimal neural network configuration parameters were determined. The corresponding values are as follows:

- **L2 Weight Regularization 0.04**
- **Sparsity Regularization 1.6**
- **Sparsity Proportion 0.1**
- **Hidden Layer Size 200**

In the experiments, the ***encoder*** used the positive saturating linear transfer function, given by:

$$f(z)=\begin{cases}0 & ,z\le 0\\ z & ,0<z<1\\ 1 & ,z\ge 1\end{cases}, \quad (8)$$

While the ***decoder*** transfer function was set to pure linear:

$$f(z)=z, \quad (9)$$

For the testing of this algorithm, all the available subjects were used, and the databases were divided in two: the first part, containing *4* pictures per each subject, was used to generate the face model; the second part consisting of the remaining pictures for each person was used as test data. With the given setup, the effectiveness of the proposed algorithm was measured by using several standard error and recognition rates, as follows:

- the first rate is the rank-one recognition rate (***ROR***) – it means that the nearest neighbour is an image of the same person;
- the false acceptance rate (***FAR***) indicates the percentage of accepted non-authorized users;
- the false rejection rate (***FRR***) indicates the percentage of incorrectly rejected authorized users;
- the minimal half total error rate (***mHTER***):

$$\boldsymbol{mHTER}=min\left(\frac{FAR+FRR}{2}\right), \quad (10)$$

During the recognition experiments that are performed on the test data, a ***similarity matrix*** (**Figure 12**) is produced from the scores generated by the ***SoftMax layer***. For M test images and N subjects in the database, the corresponding ***similarity matrix*** has a size of $N\times M$.

	Subject 1	Subject 2	.	.	.	Subject N
Image 1 (Subject 1)	1	0.44	.	.	.	0.3
Image 2 (Subject 1)	0.9	0.2	.	.	.	0.1
.	.	.	.	.	.	.
.	.	.	.	.	.	.
.	.	.	.	.	.	.
Image M (Subject N)	0.3	0.2	.	.	.	1

Figure 12. Visualization of a similarity matrix (size $N\times M$)

The ***FAR*** and ***FRR*** values are calculated by imposing different decision threshold values (Δ) on the ***similarity matrix*** (e.g. image i corresponding to subject j will be counted as impostor if $Score_{i,x}>\Delta$ where $1\le x\le N$ and $x\ne j$).

In order to measure and prove the efficiency of the algorithm proposed in this paper, four different classical face recognition algorithms were trained and tested, using the same data set (Štruc & Pavešić, 2009) (Štruc & Pavešić, 2010):

- **Gabor - PCA** and the nearest neighbour classifier;
- **Gabor - LDA** and the nearest neighbour classifier;
- **Gabor - KFA (Kernel Fisher Analysis)** and the nearest neighbour classifier;
- **Gabor - KPCA (Kernel Principal Component Analysis)** and the nearest neighbour classifier.

The experimental results are presented in **Tables 1 - 4**.

Table 1. Recognition rate comparison

	Caltech	**ORL**	**Yale**	**Yale B**	**Mean**
Own algorithm	95.00%	97.14%	91.83%	98.20%	95.54%
PCA	84.58%	93.33%	81.93%	96.40%	89.06%
LDA	98.33%	98.10%	97.36%	99.10%	98.22%
KFA	98.33%	96.19%	96.31%	99.40%	97.56%
KPCA	82.92%	95.24%	84.57%	94.89%	89.41%

Table 2. mHTER comparison

	Caltech	ORL	Yale	Yale B	Mean
Own algorithm	0.89%	0.37%	2.43%	0.11%	0.95%
PCA	3.04%	1.80%	6.17%	1.66%	3.17%
LDA	0.44%	0.14%	0.92%	0.13%	0.41%
KFA	0.92%	1.97%	2.04%	0.34%	1.32%
KPCA	3.18%	1.09%	4.68%	1.34%	2.57%

Table 3. FAR at mHTER comparison

	Caltech	ORL	Yale	Yale B	Mean
Own algorithm	1.37%	0.75%	1.73%	0.23%	1.02%
PCA	3.16%	0.75%	2.99%	1.53%	2.11%
LDA	0.47%	0.27%	0.92%	0.26%	0.48%
KFA	1.42%	1.09%	1.62%	0.67%	1.20%
KPCA	3.87%	2.18%	3.77%	2.08%	2.98%

Table 4. FRR at mHTER comparison

	Caltech	ORL	Yale	Yale B	Mean
Own algorithm	0.42%	0.00%	3.13%	0.00%	0.89%
PCA	2.92%	2.86%	9.34%	1.80%	4.23%
LDA	0.42%	0.00%	0.92%	0.00%	0.34%
KFA	0.42%	2.86%	2.46%	0.00%	1.44%
KPCA	2.50%	0.00%	5.59%	0.60%	2.17%

6. Conclusion

In this paper, a ***novel hybrid*** algorithm for face identification / recognition was presented by combining ***Gabor-based*** features with ***Deep Learning*** methods. The two-stage feature extraction technique allows for both *full-face* features and *local* features to be used in the training and recognition stages.

Comparing the average recognition rate (***ROR*** - **Table 1**), the system analysed in the article outperforms two of the classical face recognition algorithms. In comparison with the other ones the difference is **less than 2%**, thus proving the effectiveness of the proposed algorithm.

By analysing the results presented in **Table 1**, it can be observed that the classical face recognition algorithms are heavily influenced by the different facial expressions. In the case of the classic approach, the recognition rate drops (in some cases) by as much as **12%**, from **95.24%** till **82.92%.** The new hybrid algorithm presented a degradation smaller than **7%.**

Being given a small number of training images per subject (4 images were used for training), the proposed system managed to achieve a recognition rate over **95%.** In the case of "***False Acceptance Rate***" (**Table 3**), the new method obtained a **1.02%** ***FAR***, making it an ideal candidate for face verification systems.

When compared to the ***auto-encoder*** architecture presented by Li et al. (2017), the proposed hybrid architecture offers, on the **ORL** dataset, almost the same recognition rate (**97.14% vs. 97.5%**) having a simpler structure (one layer vs. four layers), using a smaller number of training images and a higher number of test images.

During the experiments presented in this paper, the system configuration consisting of the image size, down-sample rate, and total number of neurons was determined by the trial and error methods. These experiments must be continued, in order to determine the optimal number of neurons contained in the network, and also to check if a correlation exists between the number of neurons and the number of subjects. Also, the system must be tested using different image sizes and multiple decomposition levels.

The experiments were carried out using MATLAB® and the PhD Toolbox (Štruc & Pavešić, 2010) on an PC with 48GB of RAM, Intel Core i7 8700K CPU and GeForce GTX 1080 GPU with CUDA capabilities.

REFERENCES

1. Acuity Market Intelligence. (2014). *The Global Biometrics and Mobility Report: The Convergence of Commerce and Privacy Market Analysis and Forecasts 2014-2020*.

2. Amscreen (2018). Amscreen to launch OptimEyes advertising platform. Available at: *http://www.amscreen.eu/amscreen-to-launch-optimeyes-advertising-platform-2/*

3. Bellhumer, P. N., Hespanha, J. P. & Kriegman, D. J. (1997). Eigenfaces vs. fisherfaces: Recognition using class specific linear projection, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *17*(7), 711-720.

4. Bengio, Y. (2009). Learning Deep Architectures for AI, *Foundations and Trends® in Machine Learning*, *2*(1), 1-127.

5. Daugman, J. G. (1985). Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters, *Journal of the Optical Society of America*, *2*(7), 1160-1169.

6. Daugman, J. G. (1993). High confidence visual recognition of persons by a test of statistical independence, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *15*(11), 1148-1161.

7. Filip, F. G., Zamfirescu, C. B. & Ciurea, C. (2017). *Computer-Supported Collaborative Decision-Making*. Springer International Publishing.

8. Gabor, D. (1946). Theory of communication, *Journal of the Institution of Electrical Engineers – Part III: Radio and Communication Engineering*, *93*(22), 429-457.

9. Georghiades, A. S., Belhumeur, P. N. & Kriegman, D. J. (2001). From Few to Many: Illumination Cone Models for Face Recognition under Variable Lighting and Pose, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *23*(6), 643-660.

10. Jain, A., Chen, Y. & Demirkus, M. (2007). Pores and ridges: Fingerprint matching using level 3 features, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *29*(1), 15-27.

11. Lee, T. S. (1996). Image representation using 2D Gabor wavelets, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *18*(10), 959-971.

12. Lei, Z., Liao, S., Pietikäinen, M. & Li, S. Z. (2011). Face recognition by exploring information jointly in space, scale and orientation, *IEEE Transactions on Image Processing*, *20*(1), 247 - 256.

13. Li, F., Gao, X. & Wang, L. (2017). Face Recognition Based on Deep Autoencoder Networks with Dropout. In *2017 2nd International Conference on Modelling, Simulation and Applied Mathematics Advances in Intelligent Systems Research, 32* (pp. 243-247).

14. Liou, C.-Y., Cheng, W.-C., Liou, J.-W. & Liou, D.-R. (2014). Autoencoder for words, *Neurocomputing*, *139*, 84-96.

15. Liou, C.-Y., Huang, J.-C. & Yang, W.-C. (2008). Modeling word perception using the Elman network, *Neurocomputing*, *71*(16-18), 3150-3157.

16. Marĉelja, S. (1980). Mathematical description of the responses of simple cortical cells, *Journal of the Optical Society of America, 70*(11), 1297–1300. DOI:10.1364/JOSA.70.001297

17. Messer, K., Kittle, J., Sadegh, M., Hamouz, M., & Kosti, A. (2004). Face authentication test on the BANCA database. In *Proceedings of the 17th International Conference on Pattern Recognition(ICPR)*, *4* (pp. 523-532). DOI: 10.1109/ICPR.2004.1333826

18. Olshausen, B. A. & Field, D. J. (1997). Sparse coding with an overcomplete basis set: A strategy employed by V1?, *Vision Research*, *37*(23), 3311-3325.

19. Samaria, F. S. & Harter, A. (1994). Parameterisation of a stochastic model for human face identification. In *Proceedings of 2nd IEEE Workshop on Applications of Computer Vision* (pp. 138-142). DOI:10.1109/ACV.1994.341300

20. Štruc, V. & Pavešić, N. (2009). Gabor-Based Kernel Partial-Least-Squares Discrimination Features for Face Recognition, *Informatica*, *20*(1), 115-138.

21. Štruc, V. & Pavešić, N. (2010). The Complete Gabor-Fisher Classifier for Robust Face Recognition. In *EURASIP Journal on Advances in Signal Processing*. DOI:10.1155/2010/847680

22. Vinay, K. B. & Shreyas, B. S. (2006). Face Recognition Using Gabor Wavelets. In *Fortieth Asilomar Conference on Signals, Systems and Computers* (pp. 593-597). IEEE, Pacific Grove, CA, USA.

23. Viola, P. & Jones, M. (2001). Rapid object detection using a boosted cascade of

simple features. In *Proceedings of 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *1* (pp. 511-518).

24. Weber, M. (1999). A frontal face dataset collected by Markus Weber at California Institute of Technology. Available at: *http://www.vision.caltech.edu/Image_Datasets/faces/faces.tar*

25. Weldon, T. P., Higgins, W. E. & Dunn, D. F. (1996). Gabor filter design for multiple texture segmentation, *Optical Engineering*, *35*(10), 2852-2863.

26. Zhou, M. & Wei, H. (2006). Face Verification Using Gabor Wavelets and AdaBoost. In *18th International Conference on Pattern Recognition*, *1* (pp. 404-407). IEEE, Hong Kong, China DOI:10.1109/ICPR.2006.536