Human Face Recognition Using Convolutional Neural Networks

Răzvan-Daniel Albu
[*] Department of Electronics, University of Oradea,
Faculty of Electrical Engineering and Information Technology,
Postal address 410087, Oradea, Bihor, Romania, 1, Universității Street,
E-Mail: ralbu@uoradea.ro

Abstract *- In this paper, I present a novel hybrid face recognition approach based on a convolutional neural architecture, designed to robustly detect highly variable face patterns. The convolutional network extracts successively larger features in a hierarchical set of layers. With the weights of the trained neural networks there are created kernel windows used for feature extraction in a 3-stage algorithm. I present experimental results illustrating the efficiency of the proposed approach. I use a database of 796 images of 159 individuals from Reims University which contains quite a high degree of variability in expression, pose, and facial details.*

Keywords*: Face recognition, convolutional neural networks, hybrid systems, pattern recognition, image classification, machine learning.*

I. INTRODUCTION

Over the last ten years, face recognition has become a popular area of research in computer vision and one of the most successful applications of image analysis and understanding. Face detection and recognition has a wide range of possible applications, like security access control, model-based video coding, content based video indexing, or advanced human and computer interaction. Numerous approaches for face detection and recognition have been proposed in the last decade.

Most face detection methods are based on local facial feature detection and classification using statistical and geometric models of the human face. Low level analysis first deals with the segmentation of visual features using image properties such as edges, intensity, color, motion, or generalized measures.

Other approaches are based on template matching where several correlation templates are used to detect local sub-features, considered as rigid in appearance or deformable.

In this paper, I propose a novel image-based approach that is designed to precisely detect face patterns of variable size and appearance.

II. TRAINING METHODOLOGY

MyNeuron class is implemented based on the following model.

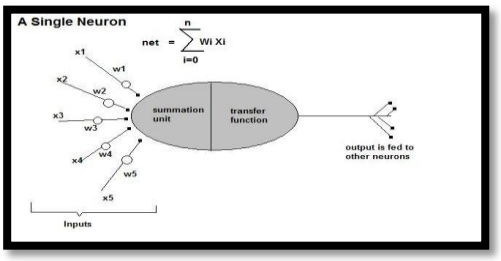


Figure 1 A single neuron

For transfer function I used the sigmoid function:

$$S=\frac{1}{1+e^{-net}}; \qquad net=\sum_{i=0}^{n} Wi * X i \qquad (1)$$

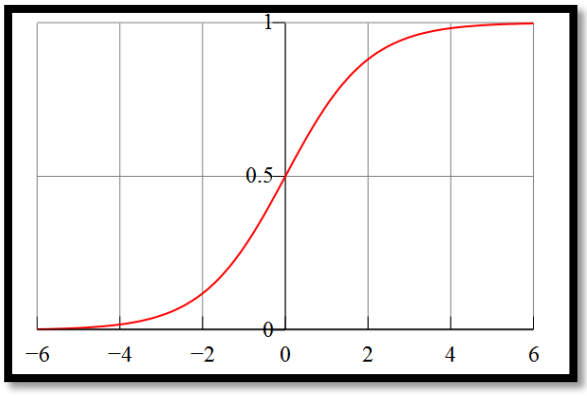


Figure 2 Standard logistic sigmoid function

From each picture in database there are extracted a variable number of 5x5 windows called units from random positions. The number of units extracted in each stage is also variable in my software. Those windows are used for feature extraction in each stage of the algorithm, being inputs to the neural network.

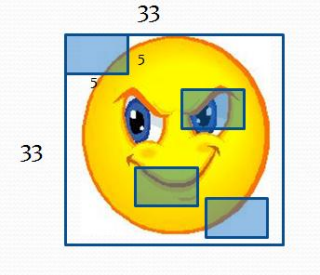


Figure 3 Units extraction from random positions

For neural networks training I used the backpropagation algorithm. As the algorithm's name implies, the errors (and therefore the learning) propagate backwards from the output nodes to the inner nodes. So technically speaking, backpropagation is used to calculate the gradient of the error of the network with respect to the network's modifiable weights. Backpropagation is the generalization of the Widrow-Hoff learning rule to multiple-layer networks and nonlinear differentiable transfer functions. Input vectors and the corresponding target vectors are used to train a network until it can approximate a function, associate input vectors with specific output vectors, or classify input vectors in an appropriate way as defined by you.

III. THE PROPOSED ALGORITM

The algorithm has 3 stages, in each one there are extracted a variable number of feature maps from each picture in database. Each picture has a 33x33 size. In the first stage from each picture I obtain a variable number of feature maps. Each feature map is the result of the convolutional product of the kernel window and a slicing window that moves over all the surface of the image. For each stage I can set some parameters in my SRF software:

- Feature maps from each picture. This number is equal with the neurons in hidden layer of the neural network created in that stage.
- Unit's extraction step. This value is the step with witch the slicing window is moved over the image.
- Units extracted. Thin value specifies how many 5x5 windows are extracted randomly from each picture and applied as inputs for the neural network.
- The number of epochs for this stage. One epoch is finished when the neural network is trained with all the images.

Figure 4 Training parameters of SRF application

In this example in first stage I obtain 4 feature maps from each picture.

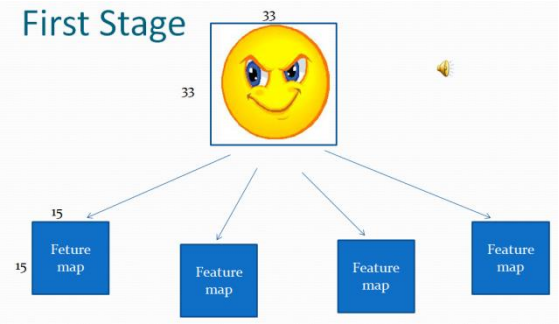
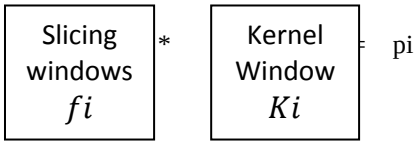


Figure 5 First stage

The neural network of this stage has 3 layers, input layer has 25 neurons, because it is made by convolutional product of 5x5 windows, the hidden layer has 4 neurons because we extract 4 feature maps and the output layer has 25 neurons also.

The slicing windows and the kernel windows have the same size 5x5 pixels.



$$Pi = \sum fi * Ki, \quad (2)$$

Each kernel is a 5x5 window and its pixels are the weights of one neuron in hidden layer. Because we have 4 neurons in hidden layer we have 4 kernel windows in the first stage. Each neuron in hidden layer has 25 connections with the input layer. All the neural networks are fully connected. I put the weights of each neuron from hidden layer into 5x5 kernel windows, when the training is completed. The training is completed when the neural networks is trained with each unit from each picture for the given number of epochs.

With the 4 kernels I obtain 4 feature maps by convolutional product.

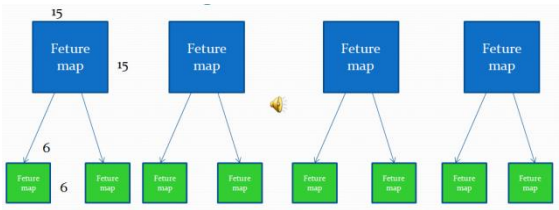


Figure 6 Second stage

The second stage do the same thing but the inputs now are the feature maps extracted in the first stage not the images in database. In this example for each feature map of stage one I obtain 2 feature maps of stage 2. The size of feature maps decreases from 15x15 in the first stage to 6x6 in the second stage because of the convolutional product and the movement of the slicing windows with a step of two.

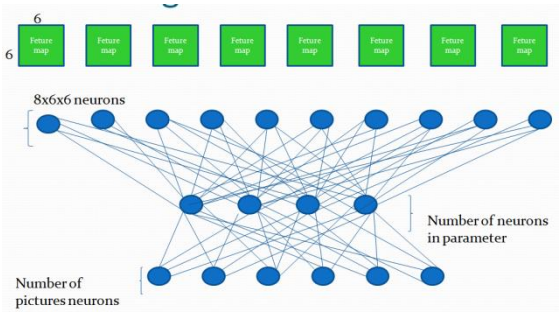


Figure 7 Third stage

The kernel windows remains at the same 5x5 size. The neural network use in the second stage in this example is a 25-2-25 one, fully connected.

In the last stage there are no feature extractions, just learning with a huge neural network. Each pixel of each feature map from stage two is applied as input to this huge neural network fully connected. The input layer has 8x6x6=288 neurons, the hidden layer has a variable number of neurons given as parameter and the output layer has a number of neurons equal with the number of picture in database.

After training a number of epochs this huge neural network, I saved all the weights of all connections, not just those of hidden layer, in a text file. The training of this neural network needs most of the processing time.

In the recognition procedure, the index of the neuron from the output layer with the higher value is the index of the image recognized from the database. In the recognition procedure, the same algorithm is applied on the selected image, but all the neural networks are initialized with the weights saved in text files, and for each network is called just a forwardpropagate function.

IV. RESULTS

This paper presents a novel method for the recognition of human faces in 2-Dimensional digital images. To check the utility of my proposed algorithm experimental studies are carried out on the database images of Reims University. 796 face images from 159 individuals in different states from the database have been used to evaluate the performance of the proposed method. None of the 5 samples are identical to each other. They vary in position, rotation, scale and expression. In this database each person has changed his face expression in each of 5 samples.

The initial learning rate is 0.001, results in good coarse training quickly. For better performance I used a schedule of 0.0005 for two epochs, followed by 0.0002 for the next three, 0.0001 for the next three, 0.00005 for the next four, and 0.00001 thereafter. The learning rate is reduced by 79.4% of its value after every epoch.

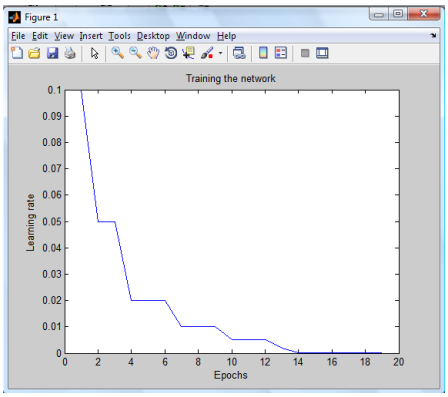


Figure 8 Learning rate decrementation by epochs

The same graph in logarithmic scaling is presented below:

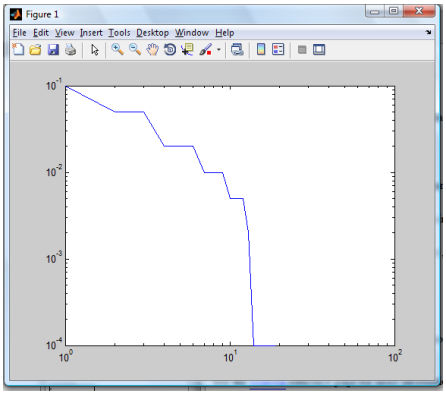


Figure 9 Learning rate graph with logarithmic scaling

V. CONCLUSIONS

Higher value of neurons in hidden layer may force the network to memorize. Lower value of neurons in hidden layer, would waste a great deal of training time in finding its optimal representation. More neurons require more computation, but they allow the network to solve more complicated problems. More layers require more computation, but their use might result in the

network solving complex problems more efficiently. One of the problems that occur during neural network training is called overfitting. The error on the training set is driven to a very small value, but when new data is presented to the network the error is large. The network has memorized the training examples, but it has not learned to generalize to new situations. One method for improving network generalization is to use a network that is just large enough to provide an adequate fit. The larger network you use, the more complex the functions the network can create.

VI. FUTURE RESEARCH

This approach has some restrictions I intend to eliminate.

- Images must have a square size.
- Image size must be a value like: 3k+9.
- The size of unit windows is constant: 5x5.

I also would like to implement a more efficient backpropagation algorithm for faster learning.

REFERENCES

[1] M. Yang, D. Kriegman, and N. Ahuja, "Detecting Faces in Images: A Survey," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 24, no. 1, pp. 34-58, Jan. 2002.

[2] E. Hjelma and B.K. Low, "Face Detection: A Survey," Computer Vision and Image Understanding, vol. 83, pp. 236-274, 2001.

[3] V. Govindaraju, "Locating Human Faces in Photographs," Int'l J. Computer Vision, vol. 19, no. 2, pp. 129-146, 1996.

[4] G. Yang and T.S. Huang, "Human Face Detection in Complex Background," Pattern Recognition, vol. 27, no. 1, pp. 53-63, 1994.

[5] C. Garcia and G. Tziritas, "Face Detection Using Quantized Skin Color Regions Merging and Wavelet Packet Analysis," IEEE Trans. Multimedia, vol. 1, no. 3, pp. 264-277, 1999.

[6] R. Fe´raud, O. Bernier, J. Viallet, and M. Collobert, "A Fast and Accurate Face Detector Based on Neural Networks," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 23, no. 1, pp. 42-53, Jan. 2002.

[7] B. Low and M. Ibrahim, "A Fast and Accurate Algorithm for Facial Feature Segmentation," Proc. Int'l Conf. Image Processing, 1997.

[8] C.C. Lin and W.C. Lin, "Extracting Facial Features by an Inhibitory Mechanism Based on Gradient Distributions," Pattern Recognition, vol. 29, pp. 2079-2101, 1996.

[9] B. Moghaddam and A. Pentland, "Probabilistic Visual Learning for Object Recognition," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 19, no. 7, pp. 696-710, July 1997.

[10] L. Wiskott, J. Fellous, N. Kruger, and C. V. der Malsburg, "Face Recognition by Elastic Bunch Graph Matching," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 19, no. 7, pp. 775-779,July 1997.

[11] C. Garcia, G. Simandiris, and G. Tziritas, "A Feature-Based Face Detector Using Wavelet Frames," Proc. Int'l Workshop Very Low Bit Coding, pp. 71-76, 2001.

[12] K. Yow and R. Cipolla, "Feature-Based Human Face Detection," Image and Vision Computing, vol. 15, no. 9, pp. 713-735, 1997.

[13] S. Jeng, H. Yao, C. Han, M. Chern, and Y. Liu, "Facial Feature Detection Using Geometrical Face Model: An Efficient Approach," Pattern Recognition, vol. 31, no. 3, pp. 273-282, 1998.

[14] D. Maio and D. Maltoni, "Real-Time Face Location on Gray-Scale Static Images," Pattern Recognition, vol. 33, pp. 1525-1539, 2000.

[15] H. Rowley, S. Baluja, and T. Kanade, "Rotation Invariant Neural Network-Based Face Detection," Proc. IEEE Conf. Computer Vision and Pattern Recognition, pp. 38-44, 1998.

[16] P. Viola and M. Jones, "Robust Real-Time Object Detection," Proc. ICCV Second Int'l Workshop Statistical and Computational Theories of Vision—Modelling, Learning, Computing, and Sampling, July 2001.

[17] Shultz, T.R. & Rivest, F. (2000) Knowledge-based Cascade-correlation, IEEE-INNS-ENNS International Joint Conference on Neural Network 2000, pp. V641-V646.

[18] Shultz, T.R. & Rivest, F. (2000) Knowledge-based Cascade-correlation: An Algorithm for Using Knowledge to Speed Learning. Proceedings of the Seventeenth International Conference on Machine Learning, pp. 871-878. San Francisco, CA: Morgan Kaufmann.

[19] Fahlman, S.E. & Lebiere, C. (1990) The Cascade-correlation Learning Architecture. In Advances in Neural Information Processing Systems **2**, pp. 524-532. Los Altos, CA: Morgan Kaufmann.

[20] CS-94-209, School of Computer Science, Carnegie Mellon University.

[21] Thivierge, J.P., Dandurand, F., & Shultz, T.R. (2004). Transferring domain rules in a constructive network: Introducing RBCC. IEEE International Joint Conference on Neural Networks 2004, pp. 1403-1409.

[22] Shultz, T. R., & Rivest, F. (2003). Knowledge-based cascade-correlation: Varying the size and shape of relevant prior knowledge.