Journal of Computer Science 2 (3): 257-260, 2006
ISSN 1549-3636
© 2006 Science Publications

Design of a Neural Networks Classifier for Face Detection

[1]F. Smach, [2]M. Atri, [3]J. Mitéran and [1]M. Abid
[1]CES Laboratory of ENIS University, Sfax 3000, Tunisia
[2] The EμE Laboratory of Science Monastir University, Monastir, 5000 Tunisia
[3]LE2I Laboratory of Université de Bourgogne Aile des Sciences de l'Ingénieur, France

Abstract: Face detection and recognition has many applications in a variety of fields such as security system, videoconferencing and identification. Face classification is currently implemented in software. A hardware implementation allows real-time processing, but has higher cost and time to-market. The objective of this work was to implement a classifier based on neural networks MLP (Multi-layer Perception) for face detection. The MLP was used to classify face and non-face patterns. The system described using C language on a P4 (2.4 Ghz) to extract weight values. Then a Hardware implementation achieved using VHDL based Methodology. We targeted Xilinx FPGA as the implementation support.

Key words: Classification, face detection, FPGA hardware description, MLP

INTRODUCTION

Human Face detection and recognition is an active area of research spanning several disciplines such as image processing, pattern recognition and computer vision. Face detection and recognition are preliminary steps to a wide range of applications such as personal identity verification, video-surveillance, liptrocking, facial expression extraction, gender classification, advanced human and computer interaction. Most methods are based on neural network approaches, feature extraction, Markov chain, skin color and others are based on template matching[1].

Pattern localization and classification is the step which is used to classify face and non-face patterns. Many systems dealing with object classification are based on ANN (Artificial Neural Networks). In this study we were interested by the design of a ANN algorithm in order to achieve image classification.

Classification for face detection: While numerous methods have been proposed to detect face in a single image of intensity or color images. A related and important problem is how to evaluate the performance of the proposed detection methods[1]. Many recent face detection papers compare the performance of several methods, usually in terms of detection and false alarm rates. It is also worth noticing that many metrics have been adopted to evaluate algorithms, such as learning time, execution time, the number of samples required in training and the ratio between detection rates and false alarms. In general, detectors can make two types of errors: *false negatives* in which faces are missed resulting in low detection rates and *false positives* in which an image is declared to be face.

$$\textit{False negative} = \frac{\text{Number..of.Missed.Falses}}{\text{Total.Number.of.Actual.Faces}}$$

$$\textit{False positive} = \frac{\text{Number..of.Incorrectly.Detected.Faces}}{\text{Total.Number.of.Actual.Faces}}$$

Face detection can be viewed as two-class recognition problem in which an image region is classified as being a “Face” or “nonFace”. Consequently, face detection is one of the few attempts to recognize from images a class of objects for which there is a great deal of within-class variability. Face detection also provide interesting challenges to the underlying pattern classification and learning techniques. The class of face and no face image are decidedly characterized by multimodal distribution function and effective decision boundaries are likely to be nonlinear in the image space.

Pattern localization and classification are CPU time intensive being normally implemented in software, however with lower performance than custom implementations. Custom implementation in hardware allows real-time processing, having higher cost and time-to-market than software implementation. Some workers[2-4] uses ANN for classification and the system is implemented in software, resulting in a poor performance (10 sec for localization and classification). A similar work is presented[5], aiming to object localization and classification and it was also implemented in software (10-15 frames/sec). An ANN MLP was implemented on DSPs, standard microprocessor and FPGA dedicated to image processing[6]. The proposed architecture is pipelined and results are given for a 256x256 image.

Corresponding Author: F. Smach, CES Laboratory of ENIS University, Sfax 3000, Tunisia

We are interested by the implementation of a ANN algorithm in order to provide image classification. The MLP *(Multi-layer Perception)* algorithm is used to classify face and non-face patterns before the recognition step.

Multi-layers perception: The MLP neural network[1] has feedforword architecture within input layer, a hidden layer and an output layer. The input layer of this network has N units for an N dimensional input vector. The input units are fully connected to the ***I*** hidden layer units, which are in turn, connected to the ***J*** output layers units, where ***J*** is the number of output classes.

A Multi-Layers Perception(MLP) is a particular of artificial neural network[7]. We will assume that we have access to a training dataset of *l* pairs (x_i, yi) where x_i is a vector containing the pattern, while y_i is the class of the corresponding pattern. In our case a 2-class task, y_i can be coded 1 and -1.

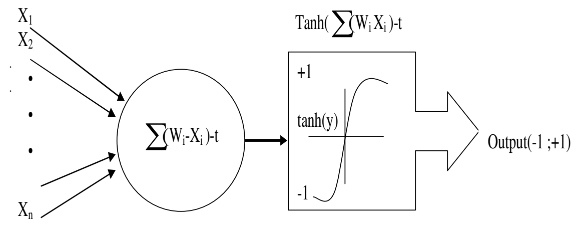
X_1
X_2
X_n
$\sum(W_i\text{-}X_i)\text{-}t$
$\text{Tanh}(\sum(W_i X_i)\text{-}t$
+1
tanh(y)
-1
Output(-1 ;+1)

Fig.1: The neuron of supervised training[8]

We considered a MLP (Multi-Layers Perceptron) with a 3 layers, the input layer is a vector constituted by n^2 units of neurons (n x n pixel input images). The hidden layer has n neurons and the output layer is a single neuron which is active to 1 if the face is presented and to otherwise.

The activity of a particular neuron ***j*** in the hidden layer is writing by: $S_j = \sum_{i \in \text{input}} w_{ji} x_i$, $x_i = f(s_j)$ (1), f a sigmoid function.

Where W_{1i} is the set of weights of neuron ***i,*** $b_1(i)$ is the threshold and x_i is an input of the neuron.

Similarly the output layer activity is:

$$S_j = \sum_{i \in \text{input}} w_{ji} x_i$$

In our system, the dimension of the retina is 15x15 pixels represent human faces and non face, the input vector is constituted by 225 neurons, the hidden layer has 15 neurons.

The examples were taken from the FERET database. The MLP was trained on 500 face and 200 non-face examples.

Training methodology: The MLP with the training algorithm of back propagation is universal mappers, which can in theory, approximate any continuous decision region arbitrarily well. Yet the convergence of back propagation algorithms is still an open problem. It is well known that the time cost of back propagation training often exhibits a remarkable variability. It has been demonstrated that, in most cases, rapid restart method can prominently suppress the heavy-tailed nature of training instances and improve efficiency of computation.

Multi-Layer Perception(MLP) with a back propagation learning algorithms was chosen for the proposed system because of its simplicity and its capability in supervised pattern matching. It has been successfully applied to many pattern classification problems[9]. Our problem has been considered to be suitable with the supervised rule since the pairs of input-output are available.

For training the network, we used the classical backpropagation algorithm. An example is picked from the training set, the output is computed. The error is calculated as the difference between the actual and the desired output. It is minimized by back-propagating it and by adjusting the weights.

Although back-propagation can be applied to network with any number of layers, it has been shown that one layer of hidden units suffices to approximate any function[8]. Therefore, in most application, a MLP NNs with a single layer of hidden units is used with a sigmoid activation function for the units $f(a) = \frac{1}{1+e^{-a}}$ (2), this function has the interesting property of having an easy to compute derivative

$f'(a) = f(a)\left[1 - f(a)\right]$ (3)

The MLP training is amount to: Repeatedly presented with sample inputs and desired targets, then the output and targets are compared and the error measured. At last, adjusts weights until correct output for every input.

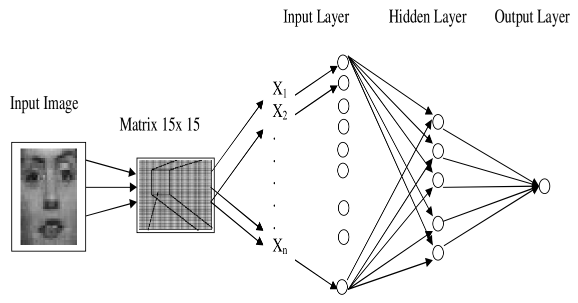

Fig.2: Architecture of proposed system

Hardware implementation: Our aim is to implement an efficient model of unconstrained face tracking and real time face detection in arbitrary images. Artificial Neural Networks (ANNs) have been proved to be an effective way to solve this problem[9], but due to long

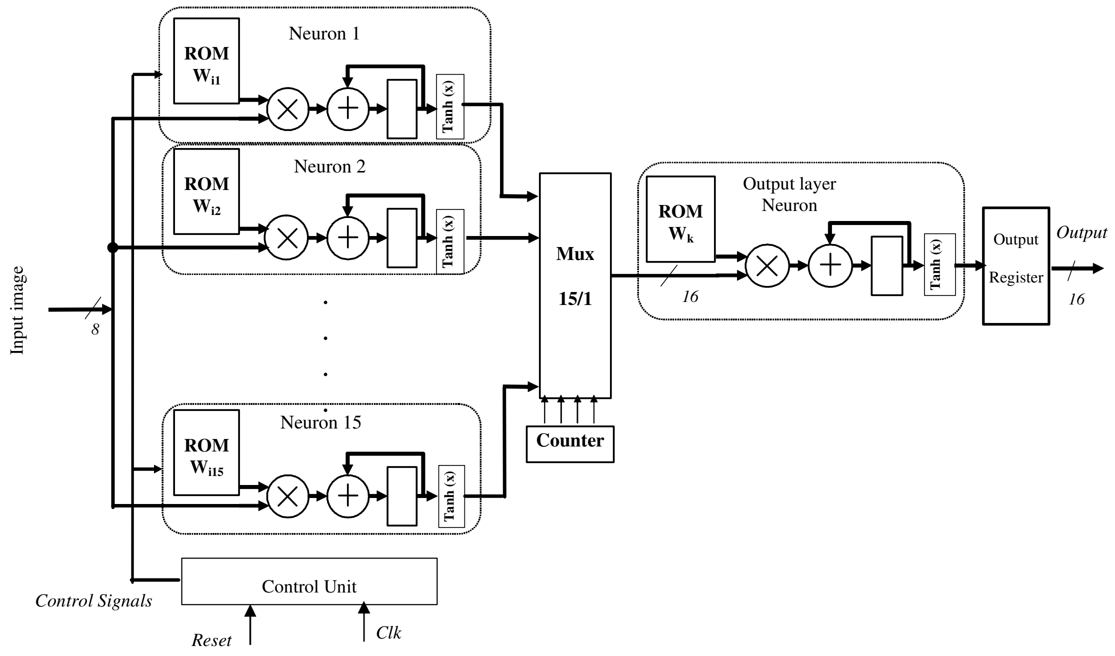

Fig.3: Block diagram

time process in software implementation. However, with today's design technology, we are given the chance to perform face detection at higher level, which involves the real time domain. We are able to shift the detection stage in hardware implementation, to achieve several advantages[10]. The papers[11, 12] discuss other methods implemented in a hardware implementation.

The MLP implemented is a three-layer perception(Fig. 2), one hidden layer and one output layer are used in this network system. The block diagram presented in Fig. 3 corresponds to the hardware implementation of our system. It is constituted of four units: input register bank, control unit, neuron and output register.

Computation of any activation neuron coefficient may be executed by employing a multiply and accumulate method where partial product are computed separately and subsequently added. Each neuron takes a vector input of N patterns. Each vector component is multiplied by a fixed weight value, which is determined at training time by software implementation. Weights are updated during the training process, but remain constant during the detection process. The result of the MAC operation is passed an input to a function activation and returns the value of the hyperbolic tangent (tanh) between 1 and -1.

This function implementation in hardware is very difficult in its known expression. In order to simplify function expression, it was linearized on several intervals $[C_i, C_{i+1}]$ and its value is evaluated using two constants (a_i and b_i) corresponding to this interval.

$F(x)= a_i x+ b_i \quad \text{for} \ x\in [C_i, C_{i+1}]$
$F(x)= 1 \quad \text{for} \ x > 3$

The W_{ij} values, calculated in the training step, were stored in a ROM in each neuron. Every W_{ij} coded with 8 bits represents a fixed-point weight. The operation of multiplication provide every time a result with 16 bits fraction.

A multiplexing bloc (Mux+counter) was used to provide one neuron output at each clock cycle to the next stage. The output layer neuron achieve the same tasks as the other neurons before giving the output result.

RESULTS

This architecture was implemented using Matlab in a graphical environment allowing face detection in a database. It has been evaluated using the test data of 500 images containing faces, on this test set we obtained a good detection, if the input image presented face the answer of the output neuron is rate of 0.9. These results encouraged us to implement this architecture targeting a hardware device using a HDL based methodology.

In general, the process of designing a system will proceed from a behavioral to a physical representation, gaining implementation details along the way. High level synthesis converts a behavioral specification of a digital system into an equivalent RTL design that meets a set of stated performance constraint[13]. The designer describes his system with a high level specification at one of abstraction levels. This description with a HDL (Hardware Description Language) is synthesized using existent synthesis tool allowing passage to the next abstraction level until reaching either integration in ASIC or implementation in FPGA.

The system was implemented in VHDL and synthesized using Leonardo synthesis tool. Target technology was FPGA Xilinx operating at 52 Mhz. The used device was a vertex v1000bg560. It contains 12248 slices and was occupied at 99.67%.

CONCLUSION

Our experiments have shown that using MLP neural networks for face detection is a very promising approach. The model's robustness has been obtained with a back propagation learning algorithms and the *tanh* activation function. In our approach no pre-processing is needed since the normalization is incorporated directly in the weights of the input network.

Face classification are normally implemented in hardware allowing real-time processing. Classification is a step which must be complemented with feature extraction in order to demonstrate detection accuracy and performances.

REFERENCES

1. Ming-Husan, Y., D.J. Kriegman and N. Ahuja, 2002. Detecting Faces in Images: A Survey. IEEE Trans. Pattern Analysis and Machine Intelligence, 24: 1.
2. Rowley, H.A., S. Baluja and T. Kanade, 1998. Neural network-based face detection. IEEE Trans. on Pattern Analysis and Machine Intelligence, 20: 39-51.
3. Zhang, Z.Q., Zhu L., S.Z. Li, Z.H. Jiang, 2002. Real-time multi-view face detection. Proc. Fifth IEEE Intl. Conf. on Automatic Face and Gesture Recognition, pp: 142-147.
4. Feraund, R., O.J. Bernier, J. Viallet and M. Collobert, 2001. A fast and accurate face detector based on neural network. IEEE Trans. Pattern Analysis and Machine Intelligence, 23: 42-53.
5. Gavrila, D.M. and V. Philomin, 1999. Real-time object detection for smart vehicles. Intl. Conf. Computer Vision (ICCV99). Vol. 1. Corfu, Greece, 20-25 Sep.
6. Rolf, F.M., P.M. Engel, F.G. Moraes, L.Torres and M. Robert, 2001. System prototyping dedicated to neural network real-time image processing. ACM/SIGDA Ninth Intl. Symp. On Field Programmable Gate Arrays (FPGA 2001).
7. Haisheng, W. and J. Zelek, 2003. A multi-classifier based real-time face detection system. J. IEEE Trans. Robotics and Automation.
8. Fan, Y. and M. Paindavoine, 2003. Prefiltering for pattern recognition using wavelet transform and neural networks. Adav. Imaging and Electron Phys., Vol. 127.
9. Xiaoguang, L. and S. Areibi, 2004. A hardware/software co-design approach for face recognition. The 16th Intl. Conf. on Microelectronics, Tunisia.
10. Theocharis, T., G. Link, V. Narayanan and M.J. Irwin, 2004. Embedded hardware face detection. 17th Intl. Conf. on VLSI Design, Mumbai, India. Jan. 5-9.
11. Fan, Y. and M. Paindavoine, 2003. Implementation of an RBF neural network on embedded systems: Real-time face tracking and identity verification. IEEE Trans. on Neural Networks, 14: 5.
12. McCready, R., 2000. Real-time face detection on a configurable hardware system. Intl. Symp. on Field Programmable Gate Arrays. Montery, California, United States.
13. Gajski, D., N. Dutt and A. Wu, 1992. High-Level Synthesis: Introduction to Chip and System Design. Kluwer Academic Publishers, Boston.