ISSN 1009 - 3095 *Journal of Zhejiang University(SCIENCE)* V.1, No.4, P.398 - 401, Oct. - Dec., 2000
http://www.chinainfo.gov.cn/periodical; http://www.zju.edu.cn;
http://lib.zju.edu.cn/English; http://www.zjupress.com

TRAINING A NEURAL NETWORK FOR MOMENT BASED IMAGE EDGE DETECTION

WANG Hong-yu (王洪玉), LI Hong-dong (李宏东), YE Xiu-qing(叶秀清), GU Wei-kang(顾伟康)
(*Department of Information and Electronics, Zhejiang University, Hangzhou, 310027, China*)
Received Apr.6, 1999; revision accepted Jan.8, 2000

Abstract: Edge detection is a crucial step to computer vision. Currently, there is not a single edge detector that has both efficiency and reliability. Traditional differential filter-based algorithms have the advantage of theoretical strictness , but require excessive post-processing. This paper introduces a neural network edge detector that takes advantage of moments features. It functions as a neural pattern classifier that directly estimates the posterior probability from the training data set. Two subsystems can be distinguished and different kinds of learning rules are used. For the end-user, it works as a black box that directly transforms raw images into the edge maps so no complicated postprocessing is required. Tests on both simulated and real images showed the proposed neural network edge detector is superior to traditional operators.

Key words: neural network, edge detection, image processing
Document code: A **CLC number:** TP391.41

INTRODUCTION

Edge detection is a crucial step towards the ultimate goal of computer vision, and is an intensively researched subject. Edges, the sharp changes in the intensity profile of an image, can be detected by a family of traditional differential filter-based edge detectors, most of which follow Canny's approach of first smoothing then filtering (Canny, 1986), or are based on Marr's Laplace of Gaussian(LOG) (Marr et al., 1980). These filter-based operators are derived from the theory of signal estimation, which make them theoretically strict. But to overcome the inevitable noise corruption within the image, too many extra processing steps, such as adaptive threshold and non-maximum suppression, are introduced to these operators, which degrade their practicability.

Artificial neural network (ANN), as a new kind of computing structure, has many applications in science and technology, especially in artificial intelligence computer vision (Kosko, 1991).

This paper introduces a multi-layer neural network edge detector that functions as a pattern classifier using spatial moments as its intermediate feature vector. We regard edge detection as a form of statistical pattern classification. Only two classes of pixels, edge or non-edge, need to be discriminated in this case. Instead of estimating the conditional probability and solving a Bayesian formula, the neural edge detector can directly estimate the probability by training. It collects the input image data and outputs the edge map directly with some adjustment of parameters. Tests on both simulated and real images with various level of noise showed good results, indicating the multi-layer neural network approach introduced in this paper is simple but powerful for general purpose edge detection.

This paper's two dimensional spatial moments (in short: moment) of order $(p + q)$ of the image function $f(x, y)$ are defined in terms of the Riemann integral as

$$m_{pq} = \iint_{x,y \in s} x^p y^q (x, y) \mathrm{d}x \mathrm{d}y \quad p, q = 0, 1, 2, \cdots \tag{1}$$

So the first six m_n, moments that are written as $\{m_{00}, m_{01}, m_{10}, m_{11}, m_{02}, m_{20}\}$.

Due to the special properties of the integral, moments are insensible to noise. Many moment-based edge detectors have been developed, some robust results were reported (Machuca, 1981; Lyvers, 1988; Lyvers 1989). Simple mathematical derivation showed that moments can provide unique, compact, and robust representation of

the image function if the function is piecewise continuous with bound support. For example, a step edge within a small image window, as shown in Fig. 1, can be calculated completely by the first six moments as

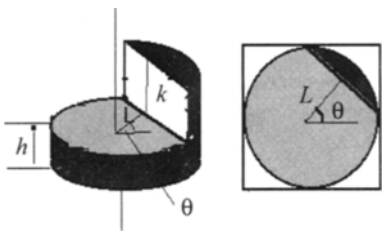


Fig. 1 Ideal step edge model

$$
\begin{aligned}
l &= (4m'_{20} - m'_{00})/3m'_{10} \\
\theta &= a\tan(m_{01}/m_{10}) \\
h &= [2m'_{00} - k(\pi - 2\sin^{-1} - \\
&\quad (2l\sqrt{1-l^2})]/2\pi \qquad (2) \\
k &= 3m'_{10}/2\sqrt{(1-l^2)^3}
\end{aligned}
$$

This is why we choose the moments as the statistical features in our neural edge classifier.

NEURAL NETWORK FOR EDGE DETECTION

From the viewpoint of practicability, an edge detector that produces edge maps directly from the raw images so that no extra processing is needed, is an ideal approach to achieve image edge detection. The neural network proposed in this section is such an approach, and functions just like a pattern classifier, which collects the input features and outputs the decisions. Fig. 2 depicts the structure of this neural network. It is a multi – layer feed forward nonlinear network of three layers with 19 neurons in total. Each neuron is a nonlinear processing element that uses sigmoid function: $f(x) = 1/[1 + \exp(-x)]$ as its activity function. The output of a neuron can be written as:

$$y_i = f(\sum_j w_{ij}x_j) \qquad (3)$$

The first 6 moments $\{ m_{00}, m_{01}, m_{10}, m_{11}, m_{02}, m_{20} \}$ were chosen as the intermediate features for the proposed neural edge detector. Two subsystems of these neurons can be distinguished. One is used for feature extraction and the other is a decision maker. Six of the 19 neurons of the input layer are configured to compute the moment features. The other 12 + 1 neurons of the hidden and output layers serve as the classifiers that collate all the information to make a final decision. Data within a size 5 × 5 image window comprise the initial input vector of the network, and one output node represents the a posteriori probability of the central pixel. If it is nearer to 1.0 than to 0.0, then it is classified as an edge pixel.

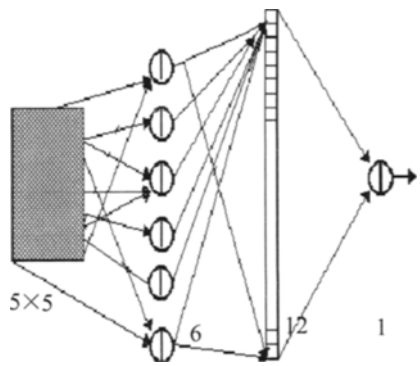


Fig. 2 Network Structure using in the paper

Application of the network throughout the whole image, yields a final edge map. In comparison with other traditional edge detectors, this one provides a more direct solution. Because all the computations and decision rules are coded in anatural way as connection weights, an appropriate learning rule should be developed.

TRAINING OF THE NEURAL EDGE DETECTOR

The training process is divided into two phases corresponding to the two subsystems mentioned above.

1. Training of the input layer

The task of the input layer is to extract the first 6 moment features from the underlying image window. Surely we can use a supervisor learning to train the first layer to compute the moments by using the following gradient descend method.

$$\Delta w_{ij} = -\eta \partial E/\partial w_{ij} \qquad (4)$$

where η is the learning rate. But for the sake of

simplicity, we configure these weights manually with

$$w_{pq} = \alpha \cdot x^p y^q , \quad (5)$$

where α is a small positive quantity.

Because Equations (1) and (3) are similar in the forms, we can compute (1) with a neural network structure.

2. Training of the hidden and output layers

The well-known back-propagation (BP) algorithm was applied because there are two layers to be trained. The teaching signals, or the expected a posteriori probabilities, were produced by a proper way (We will explain it in detail in the following discussion.). A BP rule is written as

$$\Delta w_{ij} = - \mu \sum \delta_{ij} \cdot x_i \quad (6)$$

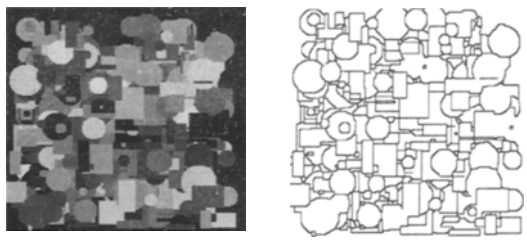
where x_i is the input , μ is the learning rate and δ_{ij} the error signal.

For the sake of good convergence, high quality or reasonable teaching signals are very important to the network, reasonable in the sense that the signals are: (a) sufficient, which means the more types of edges used for training, the more adaptable the network will be, and in the sense that they are; (b) typical, which means that the cases that occur in a training pair should be found easily in real images.

Three methods for generating the teaching signal (the expected edge map) are introduced in literature. The expected edge maps are produced by (1) Using edge maps delineated manually, (2) Using edge maps obtained from an optimum edge detector, and (3) Using synthetic images.

Training with method(1) yields an edge map very similar to a man-made map, but the teaching signals are not easy to obtain, as specific devices and personal experience are required. In training by method (2), such as by Canny's operator, the network's performance can not be superior to that of the first method, and is also not easy to implement. Method(3) proved to be efficient and easy to implement. Good results like those of Pinho et al., (1995) and Spreeuwers (1994) were obtained.

In this paper, a synthetic image and the corresponding edge map are used as a training pair. The synthetic image(Fig. 3) composed of random circles and rectangles is very similar to Pinho' s (Pinho et al., 1995). There are step changes (about 25 in 256 gray-levels) between adjacent regions. Linear trends and Gauss blurring were added into the raw synthetic image to assure the quality. Generation of this raw image simultaneously yielded the desired edge map.



(a) (b)

Fig.3 Synthetic training pair
(a) original model; (b) final model

EXPERIMENTAL RESULTS

The proposed neural network edge detector was tested on a PC computer using C + + language. The training set was composed of 10,000 data pairs randomly sampled from the synthetic image. Each data pair corresponded to the 5×5 data within the underlying image window and the expected signal. The initial weights of the hidden and output layers were set with small random noise $[-0.3, 0.3]$, and μ was set to 0.01. The small learning rate assures good convergence based on our experiments. After being trained by about 5000 epochs using only one synthetic image, the network converged finally. The average error was less than 0.05 for every pixel.

During the testing phase, the whole image was processed by this network pixel-by-pixel. The resulted edge map was obtained by applying a simple rule:

The underlying pixel is an edge point , if

output$(x, y) > 0.5$ && {[output$(x, y) >$ output$(x-1, y)$ && output$(x, y) >=$ output$(x+1, y)$] || [output$(x, y) >$ output$(x, y-1)$ && output$(x, y) >=$ output$(x, y+1)$]}.

Otherwise, it is non-edge.

Tests on a large number of images showed the acceptably practical value of this simple rule. No complicated postprocessing was required in order to produce the final edge map. Both simulated and real images were used to

evaluate the performance of our neural edge detector. We tested it on a simulated chessboard image with different level of additive Gaussian noise added in. Good results were obtained even when signal-to-noise-ratio (SNR) was less than 8 dB.

We also compared our neural network as shown in Fig. 4. They are 256 × 256 real images. We chose portrait images because the edges in human faces are always so smooth they are not easily extracted by conventional detectors. The left image was obtained by our neural network, the right image by LOG with $\Sigma = 2.0$. From these results, we can see that the proposed neural network works better than LOG. Edges extracted by the neural network are shown to be more exact in localization and orientation. We also tested this neural network detector on various classes of natural images including portraits, architectural forms, scenes, etc; and obtained satisfactory results (Fig. 5), testifying to the high quality standards of the proposed method.

Fig.4 Comparison between ANN and LOG on real images

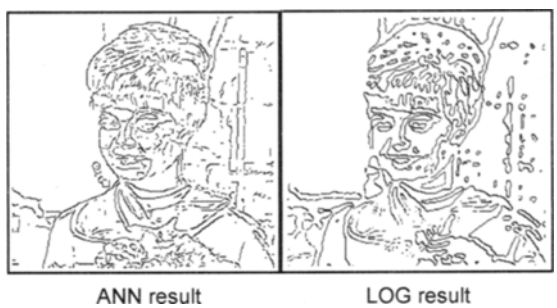


Fig.5 Other results of the ANN detector

CONCLUSIONS

An artificial neural network for edge detection that takes advantage of the good properties of spatial moments is introduced in this paper. It is based on the concept of pattern classifier that directly estimates the a posteriori probability from the training data. It transforms and transposes the raw images into the edge maps directly, so no complicated postprocessing is needed. Tests on both simulated and real images showed good performance of the neural edge detector, which make us believe this paper do provide an easy-yet-powerful approach to general purpose edge detection. Further research may include (a) generation of more reasonable training pairs; (b) incorporation of context information inherent in edge detection; (c) implementation of VLSI in this edge detector.

References

Canny, J., 1986. A computational approach to edge detection. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, **8** (1): 679 - 698.

Kosko, B., 1991. Neural Networks and Fuzzy Systems: A Dynamics System to Machine Intelligence. Printice-Hall, New York.

Lyvers, E., 1988. Precision edge contrast and orientation estimation, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, **10**: 927 - 937.

Lyvers, E., 1989. Subpixel measurements using a moment based edge operator, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, **11**(12): 63 - 72.

Machuca, R., 1981. Finding edges in noisy scenes, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, **3**: 103 - 111.

Marr, D., Hildreth, E., 1980. Theory of edge detection, *Proc. of Royal Society Landon*, 1980, **B**(207): 187 - 217.

Pinho, A.J., Almeida, L.B., 1995. Edge detection filters based on artificial neural networks, Pro. of ICIAP' 95, *IEEE Computer Society Press*, p.159 - 164.

Speeruses, L. J., 1994. Neural network edge detector, *SPIE*, **2451**: 204 - 215.