EDGE CHARACTERIZATION USING A MODEL–BASED NEURAL NETWORK

Hau-san Wong[†], *Terry Caelli*[‡] *and Ling Guan*[†]

[†]Department of Electrical Engineering, The University of Sydney, NSW 2006, Australia.
[‡]Center for Mapping, The Ohio State University , Columbus, Ohio 43212, USA.
E-mail: hswong@ee.usyd.edu.au, caelli@cfm.ohio-state.edu, ling@ee.usyd.edu.au

ABSTRACT

In this paper, we investigate the feasibility of characterizing significant image edges using a model-based neural network with modular architecture. Instead of employing traditional mathematical models for characterization, we ask human users to select what they regard as significant features on an image, and then incorporate these selected edges directly as training examples for the network. Unlike conventional edge detection schemes where decision thresholds have to be specified, the current NN-based edge characterization scheme implicitly represents these decision parameters in the form of network weights which are updated during the training process. Experiments have confirmed that the resulting network is capable of generalizing this previously acquired knowledge to identify important edges in images not included in the training set. Most importantly, the current approach is very robust against noise contaminations, such that no re-training of the network is required when it is applied to noisy images.

1. INTRODUCTION

In this paper, we investigate the feasibility of characterizing significant image features using *model-based* neural network through human-supplied training examples, instead of employing traditional mathematical models. As a first step, we consider the problem of the characterization of edges, which is usually regarded as significant image features by humans.

The primary aim of edge characterization is to locate and model those image pixels with significant change in intensities [1] . The process is usually divided into two stages: in the first stage, we define measures which characterize the dissimilarity of the current pixel with respect to the surrounding pixels. Typical examples of these include the Prewitt and the Sobel operators [1] . The second stage is a decision process on the resulting *edge magnitudes* which distinguishes the significant edges from the non-edges.

It is the proper selection of the decision criterion in the latter stage that the current work is addressing. In simple edge detection, a global threshold is usually interactively chosen to produce a final binary edge map. More sophisticated approaches, including the Canny [2] and the Shen-Castan [3] edge detectors, employ an expanded threshold set in the so-called hysteresis thresholding operation, where a minimum and maximum threshold is specified for edge linking. In addition, accurate location of edges usually requires some form of Laplacian of Gaussian (LoG) filtering [1] thus requiring the specification of filter width parameters . This, together with the previous thresholding parameters, gives rise to a large variety of possible parameter combinations, each of which will correspond to a very different final edge map.

In view of this, a logical choice for a proper representation of these parameters would be in the form of connection weights in a neural network [4] . We would implicitly specify the edge detection parameters by tracing out what we regard as significant edges in an image, and we will use them as training inputs for the network. Adopting a model-based network architecture [5, 6], it is expected that the network will be capable of generalizing this acquired knowledge to identify novel feature types similar to those in the training set.

Along this direction, we have developed a model-based neural network for edge characterization based on the decision based modular architecture proposed by Kung and Taur [6] . Conforming with the above formulation, the connection weights of the network encode both the edge-modelling parameters in the first high-pass filtering stage in edge detection, and the thresholding parameters in the second decision process. As a result, explicit specification of threshold parameters are avoided in favour of implicit parameter specification in the form of human-supplied training examples.

2. NETWORK ARCHITECTURE

The edge characterization NN is modelled after the decision-based modular architecture proposed by Kung and Taur [6] . This architecture is composed of a hierarchical structure consisting of clusters of neurons forming sub-networks . In the training stage, each sub-network will encode different aspects of the training set, and in the recognition stage an arbitration process is applied to the outputs of the various sub-networks to produce a final decision.

The motivation of our adoption of this architecture is due to our observation that, in edge detection, it would be more natural to adopt multiple sets of threshold parameters and apply the appropriate set of parameters depending on the local context, instead of just adopting a single parameter set across the whole image. As an example, we can consider the visibility of an edge with a certain strength under different background gray level values. Within a brightly lit background, the visibility of even relatively weak edges is enhanced compared with edges of similar strengths in dark backgrounds. As a result, we will possibly consider the former as significant but not the latter, and we have to adjust the thresholds accordingly.

The modular decision-based architecture thus constitutes a natural representation of the above adaptive decision process if we designate each sub-network to represent a different background illumination level, and each unit in the sub-network to represent different prototypes of edge-like features under the corresponding illumination level. The various hierarchies in the network are explained below.

2.1. Sub-Network G_r

For each image pixel, we consider the neighboring pixels in a 3×3 window with corresponding gray level values $\mathbf{x} = [x_1 \ \ldots \ x_9]^T$ and mean $\overline{x}$. We associate each sub-network $G_r, r = 1, \ldots, m$ with a prototype background gray-level value g_r, and assign all those windows with their average values $\overline{x}$ close to g_r to the sub-network G_r . This is in accordance with our previous assertion that the perception of edge-like features will in general vary with the local context, so that adaptive processing with respect to different illumination levels are required. Specifically, a particular window is assigned to the sub-network G_{r^*} if the following condition is satisfied

$$|\overline{x} - g_{r^*}| \quad < \quad |\overline{x} - g_r| \qquad r = 1, \ldots, m, r \neq r^* \qquad (1)$$

In this way, we partition the set of all 3 × 3 windows in the image into clusters, with all members in a single cluster exhibiting similar levels of background illumination. The operation of this sub-network assignment process is illustrated in Figure 1.

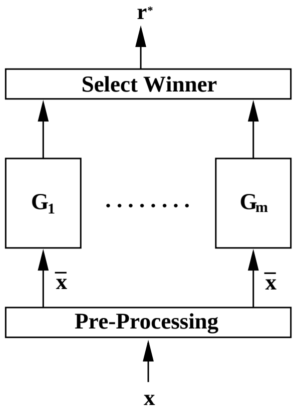

Figure 1: The architecture of the model-based neural network for edge characterization

2.2. Neuron N_{rs} in Sub-Network G_r

Each sub-network G_r contains n neurons $N_{rs}, s = 1, \ldots, n$, with each neuron encoding the various different edge prototypes which can exist under the general illumination level g_r. We associate each neuron with a weight vector $\mathbf{w}_{rs} = [w_{rs,1} \quad w_{rs,2}]^T \in \mathbf{R}^2$, which serves as a prototype characterizing the two dominant gray values in each 3×3 window. Defining the vector $\mathbf{m} = [m_1 \quad m_2]^T$ for each window , where $m_1(m_2)$ is the mean of all those pixels with gray values less(greater) than $\overline{x}$, we assign a certain window with corresponding vector $\mathbf{m}$ to the neuron N_{rs^*} if the following condition is satisfied

$$\|\mathbf{m} - \mathbf{w}_{rs^*}\| < \|\mathbf{m} - \mathbf{w}_{rs}\| \qquad s = 1, \ldots, n, s \neq s^* \qquad (2)$$

In this work, we have chosen $n = 2$ in order that one of the neurons will encode the prototype for weak edges and the other will encode the strong edges. This is determined according to the difference between the components $w_{rs,2} - w_{rs,1}$ of the weight vector. We will designate the vector with the greater(smaller) difference as the strong(weak) edge prototype . The weak edge prototype plays a similar role as the threshold parameter in conventional edge detection algorithms in specifying the lower limit of visibility for edges. The operation of the neuron assignment process is illustrated in Figure 2.

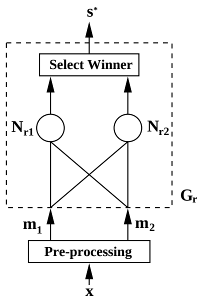

Figure 2: The architecture of a single subnetwork G_r

2.3. Binary edge configuration

Suppose that the vector $\mathbf{m}$ for the current window is assigned to neuron N_{rs} with weight vector $\mathbf{w}_{rs}$. We can now define a binary vector $\mathbf{b} = [b_1 \quad \ldots \quad b_9]^T$ with each component b_i determined according to the corresponding window component x_i as follows

$$b_i(x_i, \mathbf{w}_{rs}) = \begin{cases} 0 & \text{if } |x_i - w_{rs,1}| < |x_i - w_{rs,2}| \\ 1 & \text{if } |x_i - w_{rs,1}| \geq |x_i - w_{rs,2}| \end{cases} \qquad (3)$$

The binary vector $\mathbf{b}$ assumes a special form for valid edge configurations. Some of the possible valid edge configurations are shown in Figure 3.

0	0	0
0	0	0
1	1	1

0	0	1
0	0	1
0	0	1

0	1	1
0	0	1
0	0	0

Figure 3: Examples of valid edge configurations

3. NETWORK TRAINING

The training of the network proceeds in three stages: in the first pass, we determine the prototype illumination level $g_r, r = 1, \ldots, m$ for each sub-network G_r by competitive learning [4].

$$g_{r^*}(t+1) = g_{r^*}(t) + \alpha(t)(\overline{x} - g_{r^*}(t)) \tag{4}$$

where G_{r^*} is the winning sub-network corresponding to $\overline{x}$. In the second stage, we assign each window to its corresponding sub-network and then determine the weight vectors $\mathbf{w}_{rs}$, again using competitive learning.

$$\mathbf{w}_{rs^*}(t+1) = \mathbf{w}_{rs^*}(t) + \alpha(t)(\mathbf{m} - \mathbf{w}_{rs^*}(t)) \tag{5}$$

where N_{rs^*} is the winning neuron corresponding to $\mathbf{m}$. In the third stage, we assign each window to its corresponding neuron in the correct sub-network, extract the corresponding binary edge configuration pattern $\mathbf{b}$ as a function of the winning weight vector $\mathbf{w}_{rs}$, and insert these patterns into an edge configuration memory C.

4. RECOGNITION PHASE

In this stage, we consider all 3×3 windows in the image and assign each of them to their corresponding sub-network G_r and neuron N_{rs}. A particular pixel is considered a primary edge point if the component difference $m_2 - m_1$ of its associated vector $\mathbf{m}$ is greater that of the weak edge prototype vector . In the second stage, we locate all secondary edge points connected to the previous primary edge points. Both edge types are validated by checking the presence of their corresponding binary vector $\mathbf{b}$ in the edge configuration memory C.

5. EXPERIMENTAL RESULTS

We applied the NN-based edge characterization scheme to a number of images. One of them depicting an eagle is shown in Figure 4(a). We traced out some edges which we regard as significant in Figure 4(b). The detected edges using the current method are shown in Figure 4(c).

Comparing Figure 4(c) with Figure 4(b), we can immediately appreciate the generalization capability of the model-based NN: starting just from the training examples in Figure 4(b), the network is able to locate all the important edges as perceived by human beings, and the result is a valid caricature of the original image.

The performance of the model-based NN was validated further by comparing the result with that of a standard edge detector. We chose the Shen-Castan edge detector [3] for comparison. The associated parameters of this detector include the filter width parameter a and the hysteresis thresholds t_1, t_2. There are a large number of possible combinations between these parameters. We expect that, among those combinations, the result of the model-based NN will be close to those with their resulting edge profiles resembling faithful caricatures of the original images.

In Figure 4, we compared the result of the current scheme with the Shen-Castan edge detector under various hysteresis thresholds. The result of the model-based NN is shown in Figure 4(c) and the results for the Shen-Castan edge detector are shown in Figure 4(d) to 4(f). In general, lower values of t_1 and t_2 will reveal more details but at the same time cause more false positive detections (Figure 4(d)) . On the other hand, higher thresholds will lead to missed edges (Figure 4(f)). In our opinion, Figure 4(e), with $t_1 = 20$ and $t_2 = 25$, constitutes an adequate representation of the underlying edges of the image. Comparing with the current approach in Figure 4(c) , we can see that the edges detected by NN are similar to those under near optimal settings of the threshold parameters, but the important point is that the NN detector directly acquires the appropriate parameter settings through human-specified features without the need for trial and error.

More importantly, the current approach is very robust against noise contaminations. Figure 4(g) shows the result of applying the same network, without any re-training, to the eagle image corrupted with zero-mean Gaussian noise ($\sigma_n = 10$). We can observe that the result is very satisfactory. On the other hand, using the previous optimal thresholds, we can readily notice the effect of the noises for the Shen-Castan edge detection result (Figure 4(h)), and we have to re-adjust the thresholds to eliminate its effects (Figure 4(i)).

6. CONCLUSION

We have developed a model-based neural network for edge characterization which directly assimilates the essential characteristics of human-specified edge examples through learning. The specific architecture of the network divides the training examples into sub-classes which reflect the different preferences of human beings in regarding intensity discontinuities as significant under different illumination conditions. In addition, the current model-based NN implicitly represents the decision parameters in the form of network weights which are updated during the training process, thus requiring no explicit threshold parameter specifications as in conventional edge detection algorithms. In particular, no re-training of the network is required for applying the network to noisy images.

7. REFERENCES

[1] R. C. Gonzalez and R. Woods, *Digital Image Processing.* Reading, Massachusetts: Addison-Wesley, 1992.

[2] J. Canny, A computational approach to edge detection, *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 8, no. 6, pp. 679–698, 1986.

[3] J. J. Shen and S. S. Castan, An optimal linear operator for step edge detection, *CVGIP: Graphical Models and Image Processing*, vol. 54, no. 2, pp. 112–133, 1992.

[4] S. Haykin, *Neural Networks: A Comprehensive Foundation.* New York: Macmillan, 1994.

[5] T. Caelli, D. Squire, and T. Wild, Model-based neural network, *Neural Networks*, vol. 6, pp. 613–625, 1993.

[6] S. Y. Kung and J. S. Taur, Decision-based neural networks with signal/image classification applications, *IEEE Trans. Neural Networks*, vol. 6, no. 1, pp. 170–181, 1995.

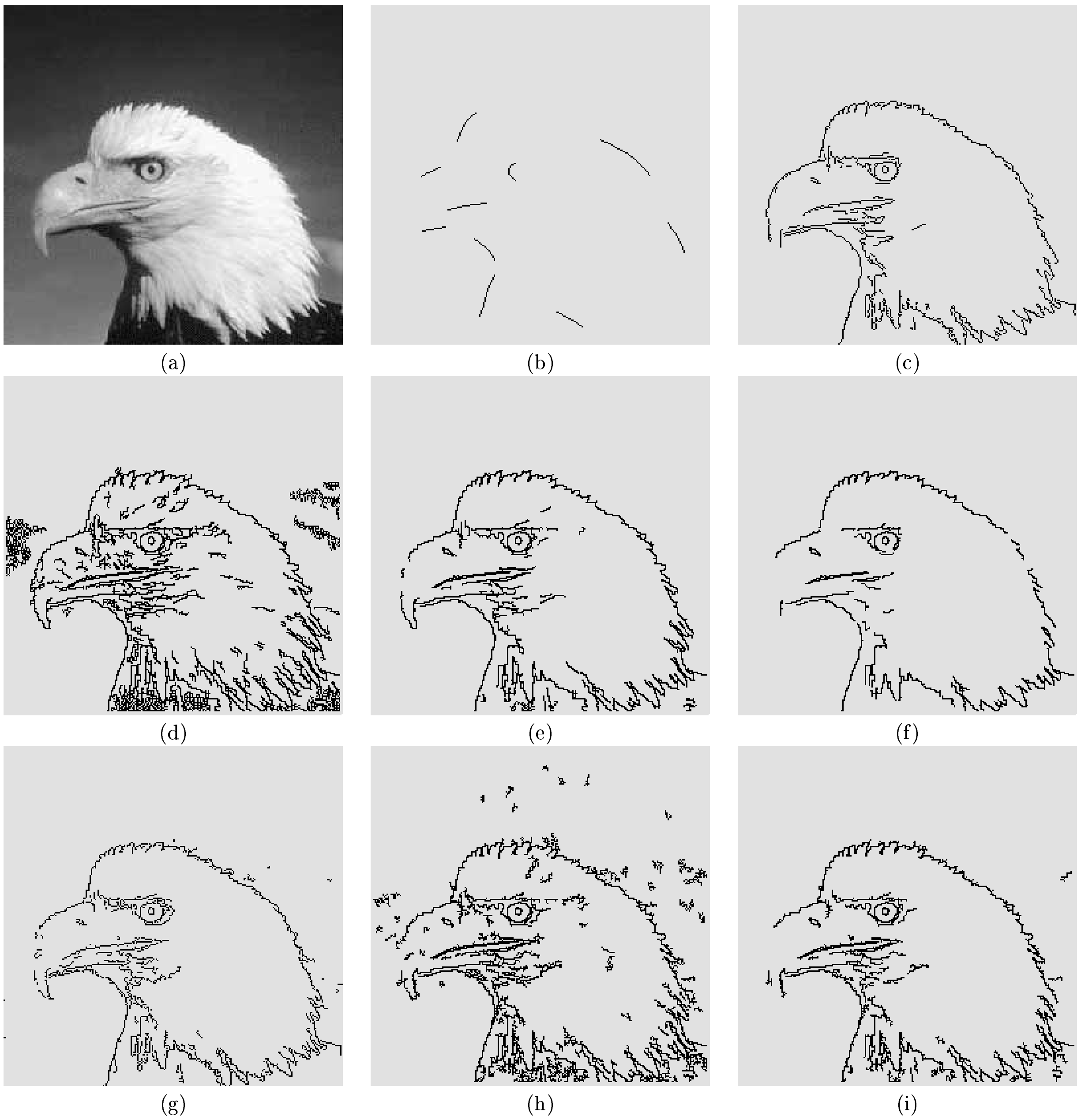

Figure 4: (a) Eagle image (b) Edges specified by humans (c) Detected edges using NN. (d)-(f) Detected edges using Shen-Castan edge detector with different hysteresis thresholds t_1, t_2 (filter parameters $a = 0.3$) (d) $t_1 = 10, t_2 = 15$. (e) $t_1 = 20, t_2 = 25$. (f) $t_1 = 40, t_2 = 45$. (g) Detected edges using NN under zero-mean Gaussian noise ($\sigma_n = 10$) (h)-(i) Detected edges using Shen-Castan edge detector under zero-mean Gaussian noise ($\sigma_n = 10$) (h) $t_1 = 20, t_2 = 25$. (i) $t_1 = 25, t_2 = 30$.