A NONLINEAR FEATURE EXTRACTOR FOR TEXTURE SEGMENTATION

Fok Hing Chi Tivive[*]*, Member, IEEE, and Abdesselam Bouzerdoum*[†]*, Senior Member, IEEE*

School of Electrical, Computer and Telecommunications Engineering,
University of Wollongong
Northfields Avenue, Wollongong, NSW 2522, AUSTRALIA
[*]tivive@uow.edu.au, [†]a.bouzerdoum@uow.edu.au

ABSTRACT

This article presents a feed-forward network architecture that can be used as a nonlinear feature extractor for texture segmentation. It comprises two layers of feature extraction units; each layer is arranged into several planes, called feature maps. The features extracted from the second layer are used as the final texture features. The feature maps are characterised by a set of masks (or weights), which are shared among all the units of a single feature map. Combining the nonlinear feature extractor with a classifier, we have developed a texture segmentation system that does not rely on pre-defined filters for feature extraction; the weights of the feature maps are found during a supervised learning stage. Tested on the Brodatz texture images, the proposed texture segmentation system achieves better classification accuracy than some of the most popular texture segmentation approaches.

Index Terms— Image texture analysis, pattern recognition, neural network architecture, nonlinear filters.

1. INTRODUCTION

Texture analysis including texture classification and segmentation is an important area of research that has a wide variety of machine vision applications, e.g., ground cover type classification of satellite imagery, industrial and biomedical surface inspection, and content-based image retrieval. In the last two decades, a number of texture analysis techniques have been reported, which can be grouped into four different categories: statistical, structural, transform-based and model-based approaches (see, e.g., [1, 2] for a comprehensive review). In recent years, Gabor and wavelet frame decompositions have become popular analysis tools. They have been used as feature extraction techniques in combination with a classification technique, such as Bayer classifier, support vector machines, nearest-neighbour classifier and neural networks, among others, to perform texture segmentation [3, 4]. These texture segmentation approaches share a common framework that integrates different processing stages, as shown in Fig. 1. The texture image is first converted into several transformed images using wavelet or Gabor filters. Then a feature extractor operates on these latter images to produce a set of feature images that will form the basis for the classification. These "raw" feature images are then passed to the feature conditioning stage, where a smoothing filter and a nonlinear transformation function are used. At the classification stage, the conditioned feature images are arranged into feature vectors as inputs for the classifier. When Gabor or wavelet is used for feature extraction in the image transform stage, there is a problem of selecting the appropriate filters from a bank of pre-defined filters. Most often, these filters are manually selected based on other existing studies. Therefore, other researchers have circumvented this problem by adapting a set of convolutional kernels as texture filters [3]. For example, Lin and Shou [5] proposed a feature extraction method based on cellular neural networks (CNNs), where several templates of CNNs are selected by the genetic algorithm. They showed that these adapted templates can be used for texture classification with promising classification performances.

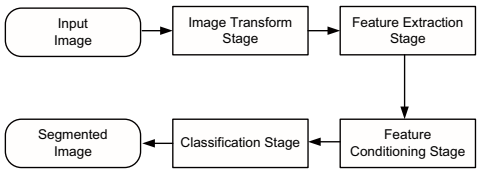


Fig. 1. A common texture classification-segmentation scheme.

In this paper, we propose a network architecture which is used as a nonlinear feature extractor for texture segmentation. The novelty of this approach is that the synaptic weights of the computing element or neuron behave as an adaptive convolutional kernel to extract features from the input image. This alleviates the use of pre-defined filters for feature extraction. The next section gives a description of the network architecture, its connection scheme and neuron model. Section 3 presents the training process of the network and its integration into the texture classification-segmentation scheme. The experimental results and discussion is given in Section 4. Finally, Section 5 presents some concluding remarks.

2. DESCRIPTION OF THE NETWORK

The network that we have developed to extract texture features is a two-dimensional (2D) feed-forward neural network, which can have several hidden layers depending on the complexity of the texture segmentation problem. Herein, after some preliminary experiments on different sizes of network, the network has two hidden layers and one output layer. The input layer is a 2D array of input nodes of size 15×15, receiving small regions of the image as inputs. The first hidden layer (L1) has five planes of neurons and the second hidden layer (L2) has twice the number of planes. All the planes in L1- and L2-layers have the same size as the input layer, i.e., 225 neurons. These planes of neurons are commonly called *feature maps* [6]. Directly connecting all the neurons to those in the following layer will however generate a large number of trainable weights. Therefore, each neuron in a feature map is constrained to have a set of weights to connect locally to a small region of the previous feature map, and this set of weights is shared among all the neurons within the feature map, as shown in Fig. 2(a). The set of weights of the neuron can be considered as an adaptive convolutional kernel to generate the feature map in the next layer. Moreover, the size of the convolutional kernel is varied for each hidden layer so as to capture different textural information, i.e., a 9×9 for L1-layer and a 7×7 for L2-layer. Instead of fully connecting the feature maps between layers, a binary-connection scheme is employed, where each feature map branches out to two feature maps in the succeeding layer, forming a binary tree. This connection scheme allows the feature maps in L2-layer to extract different types of local features, and subsequently reduces the number of connections within the network. At the L2-layer, a 5×5 lowpass filter is used and a down-sampling operation is applied to reduce the size of all feature maps to 3×3. Then, all the nine output signals from each sub-sampled feature map are sent to the output neurons for classification. Figure 2(b) shows the schematic diagram of the network used as a nonlinear feature extractor. Such feature extractor has 2290 weights with 15 feature maps and five output neurons, where the outputs of the feature maps in the L2-layer are used as feature images.

Two types of neurons are used within the network. In L1- and L2-layers, the feature maps consist of *shunting inhibitory neurons*. These neurons have the capability of producing complex decision boundaries [7, 8]. The mathematical expression of this neural model is given by

$$z_j = \frac{g\Big(\sum_i w_{ji} I_i + b_j\Big)}{a_j + f\Big(\sum_i c_{ji} I_i + d_j\Big)}, \quad \text{for } i = 1, \ldots, N \qquad (1)$$

where z_j is the activity of the j^{th} neuron, I_i's are the inputs, a_j is the passive decay rate, w_{ji} and c_{ji} are the connection weights from the i^{th} neuron in the receptive field to the j^{th} neuron of the feature map in the subsequent layer, b_j and d_j are constant biases, and N is the size of the convolutional mask. The parameters f and g are the activation functions of the neuron: for L1-layer they are chosen as linear activation function, except that g is bounded below by zero, and for the L2-layer f is the hyperbolic tangent function. As mentioned before, the sets of weights w's and c's are the same for all the neurons within a feature map, and the biases and passive decay rate are also shared among the neurons. At the output layer, sigmoid neurons are used to receive features from the L2-layer as inputs. Each sigmoid neuron represents a texture class and the neuron with the maximum network response is assumed to be the correct class. The computation of the sigmoid neuron is to sum all the weighted input signals and pass the net signal to an activation function to yield a neural response, i.e.,

$$y = h\Big(\sum_v w_v z_v + b\Big), \quad \text{for } i = 1, \ldots, N = 90 \qquad (2)$$

where h is a linear function, w_v's are the connection weights, z_v's are the feature inputs, and b is the bias term.

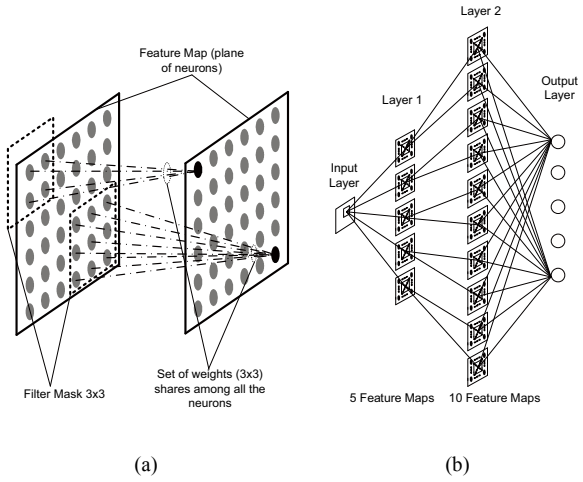
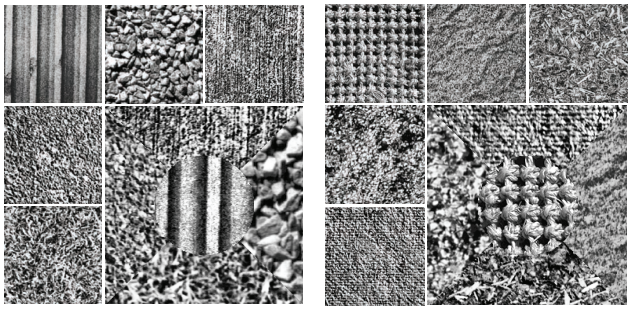


Fig. 2. The proposed neural model: (a) the local set of weights of the neuron (convolutional mask) in a feature map connecting to the previous feature map, and (b) a schematic diagram of the network used for feature extraction.

3. TRAINING AND TESTING PROCEDURES

To train and test the network, texture images from the Brodatz texture database have been used. This database is available from the website in [9], which has several texture mosaics with two or more textures. The texture images are natural textures with different density, roughness and regularity. Each texture mosaic has a separate set of texture images for training, as shown in Fig. 3(a). The network was trained on a training set using 2400 samples per texture image and tested on the



(a) (b)

Fig. 3. Brodatz texture images: (a) texture mosaic 11(e)-Nat-5m and (b) texture mosaics 11(d)-Nat-5v3. Each texture mosaic has a separate set of images used for training.

texture mosaic. Herein, the feature extractor was only trained on the set of texture images of Nat-5m. Before training, the weights of the network are initialized with random values taking from a uniform distribution on the interval $[-1/N, 1/N]$. The biases are initialized similarly with $N = 1$. According to (1), the sum of the denominator terms can become zero causing an error. Hence, to avoid this, a condition is imposed on the passive decay rate during the initialization and training process:

$$a_j + f\Big(\sum_i c_{ji} I_i + d_j\Big) \geq \varepsilon > 0. \tag{3}$$

Therefore, the passive decay rate is initialized in the range $(0, 1]$. The trainable weights of the network are adapted by a supervised training algorithm derived from Rprop, Quickprop and SuperSAB (see [10] for more details). Each weight update is performed after presenting all the training patterns, and the network was trained for 800 epochs.

In our texture classification-segmentation scheme, the image transform and feature extraction stages are replaced by the trained network where the feature maps in the L2-layer generate ten feature images. These feature images are then smoothed at the feature conditioning stage. In this experiment, seven different sizes of filter mask have been used in the feature conditioning stage. At the classification stage, there are two processing approaches to classify a texture pixel: pixel-wise or region-based. In pixel-wise each texture pixel is represented by a vector of ten feature coefficients taken from the respective feature images, and in region-based a certain number of neigbouring pixels are included to classify the central pixel. Hence, for the region-based processing, each texture pixel has $k \times k \times 10$ coefficients, where k is the size of the neighbourhood. To investigate the neighbourhood size with respect to the classification accuracy, three different sizes have been tested, ranging from 3×3 to 7×7. For the classification scheme, a *multilayer perceptron* (MLP) is trained to produce a segmented image as the output. Each output neuron produces an image of network responses that can be further post-processed with a Gaussian filter before applying the winner-take-all scheme to create the segmented image. The MLP used as classifier has eight hidden neurons and five output neurons. The activation function used in the hidden layer is a hyperbolic tangent function whereas the output activation function is simply a linear function.

4. EXPERIMENTS AND DISCUSSION

The texture segmentation system is evaluated on both texture mosaics Nat-5m and Nat-5v3, and its classification error rates are given in Tables 1 and 2 with/without post-filtering. Tested on the mosaic Nat-5m, the pixel-wise approach achieves a classification error rate of 25.4% with no feature conditioning and post-filtering, whereas for the region-based approach the error rate is 16.4% for a 7×7 neighbourhood. However, when the feature images are filtered at the feature conditioning stage before the classification, there is a significant decrease in error rate. Different Gaussian masks have been used and tested, and the mask of 15×15 achieves the lowest error rate of 9.9% for pixel-wise approach and 8.5% for region-based approach. In addition, post-filtering the network responses of the MLP decreases further the classification error rate of the texture segmentation system. Using a 5×5 input region and a post-processing stage at the output of the MLP, an error rate of 3.8% is achieved. Another experiment was performed where the same network has been used to do the feature extraction, but only the MLP classifier has been re-trained on another different training texture images, namely the texture mosaic 11(d) or Nat-5v3. From the table 2, it shows that the texture segmentation system has an error rate of 15.2% for pixel-wise and 8.7% for region-based after pre- and post-filtering. This results show that the proposed nonlinear feature extractor is capable of extracting discriminative features from the texture images even though it was trained on a different set of textures. Figure 5 shows the segmented images of the two texture mosaics. Analyzing the adaptive convolutional masks of the feature maps in the L2-layers, we found that the first-order Volterra kernels (i.e., linear part) behave as directional filters, see Fig. 4. In comparison with other existing texture segmentation techniques, our approach achieves the lowest error rates at the expense of having an additional filtering stage.

5. CONCLUSION

In this paper, we propose a network architecture as a nonlinear feature extractor for texture segmentation. The weights of the network are trained by a supervised training algorithm and are used as convolutional kernels. These convolutional kernels have the form of first-order directional linear filters that have been used to produce feature images for the classification. Using an MLP as a classifier, a texture segmen-

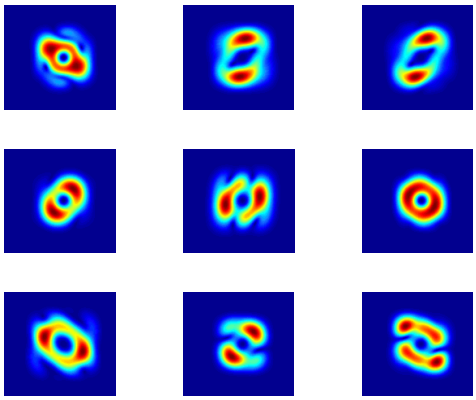


Fig. 4. The magnitude frequency spectrum of different convolutional masks of the network.

(a) (b)

Fig. 5. Samples of segmented image from the texture segmentation system: (a) segmented image of Nat-5m, and (b) segmented image of Nat-5v3.

tation system has been implemented and tested on Brodatz texture images. Based on some difficult texture mosaics, the proposed system achieves promising results and outperforms some of the best existing texture segmentation methods.

6. REFERENCES

[1] A. Materka and M. Strzelecki, "Texture analysis methods - a review," report COST B11, Institute of Electronics, Technical University of Lodz, 1998.

[2] M. Tuceryan and A. K. Jain, "Texture analysis," in *The handbook of pattern recognition and computer vision*, C.H. Chen, L. F. Pau, and P.S.P. Wang, Eds., pp. 207–248. World Scientific, Singapore, 2nd edition, 1998.

[3] A. K. Jain and K. Karu, "Learning texture discrimination masks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 2, pp. 195–205, 1996.

[4] B.-Y. Sun and D.-S. Huang, "Texture classification based on support vector machine and wavelet transform," in *Proc. of the 5th World Congress on Intelligent Control and Automation*, Hangzhou, P.R. China, 2004, pp. 1862–1864.

[5] C.-T. Lin and Y.-W. Shou, "Texture classification and representation by cnn based feature extraction," in *2005 9th International Workshop on Cellular Neural Networks and Their Applications*, 2005, pp. 210–213.

Table 1. Classification error rates of the texture segmentation scheme based on texture mosaic 11(e)-Nat-5m.

Gaussian Mask					Post-processed by Gaussian Filter			
	Pixelwise	Region-based			Pixelwise	Region-based		
	1×1	3×3	5×5	7×7	1×1	3×3	5×5	7×7
None	25.4	21.3	19.9	16.4	5.7	4.8	4.7	3.8
3×3	21.2	17.3	13.5	15.2	5.3	4.6	4.1	4.8
5×5	17.7	13.5	11.2	12.7	4.8	3.9	4.2	4.2
7×7	14.2	14.1	9.5	13.2	4.1	5.3	3.8	5.3
9×9	13.2	12.5	14.2	11.5	4.5	4.3	7.1	4.5
11×11	10.4	12.6	14.2	11.2	3.9	5.5	7.6	5.3
13×13	10.6	11.7	8.1	12.3	5.2	6.4	5.0	5.9
15×15	9.9	8.5	9.4	13.7	5.7	5.4	5.1	7.3

Table 2. Classification error rates of the texture segmentation scheme based on texture mosaic 11(d)-Nat-5v3.

Gaussian Mask					Post-processed by Gaussian Filter			
	Pixelwise	Region-based			Pixelwise	Region-based		
	1×1	3×3	5×5	7×7	1×1	3×3	5×5	7×7
None	49.2	44.7	41.3	40.5	25.2	23.0	13.9	20.7
3×3	44.4	39.8	34.8	34.2	23.7	19.9	19.3	12.8
5×5	39.6	35.4	30.8	29.5	16.9	18.5	11.7	10.5
7×7	37.3	32.6	28.6	29.2	19.5	15.5	10.3	12.5
9×9	32.9	29.5	28.7	29.1	17.6	10.3	8.7	8.9
11×11	30.6	29.2	25.9	24.8	21.0	15.2	9.7	10.2
13×13	28.2	26.5	25.5	27.5	15.2	18.2	8.7	14.0
15×15	26.1	25.7	23.4	25.4	15.2	10.9	8.9	13.3

Table 3. Error rates of different texture classification approaches.

Texture classification approach	Classification error rate (%)	
	Texture (Nat-5m)	Texture (Nat-5v3)
Our system	16.4	25.2
Our system with post-filtering	3.8	8.7
Co-occurrence in [11]	35.7	51.1
diffusion-based in [12]	27.0	13.0
Wavelet - Daubechies 4(d) in [11]	21.8	23.4
Dyadic Gabor filter bank in [11]	25.2	24.6
QMF filter bank - f16b (d) in [11]	17.2	18.4

[6] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[7] A. Bouzerdoum, "A new class of high-order neural networks with nonlinear decision boundaries," in *Proc. of the Sixth International Conference on Neural Information Processing*, Perth, 1999, vol. 3, pp. 1004–1009.

[8] G. Arulampalam and A. Bouzerdoum, "A generalized feedforward neural network architecture for classification and regression," *Neural Networks*, vol. 16, no. 5-6, pp. 561–568, 2003.

[9] T. Randen, Personal homepage. Available: http://www.ux.uis.no/˜ tranden.

[10] F. H. C. Tivive and A. Bouzerdoum, "Efficient training algorithms for a class of shunting inhibitory convolutional neural networks," *IEEE Transactions on Neural Networks*, vol. 16, no. 3, pp. 541– 556, 2005.

[11] T. Randen and J. H. Husøy, "Filtering for texture classification: A comparative study," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 4, 1999.

[12] Xiaogang Dong and I. Pollak, "Multiscale segmentation with vector-valued nonlinear diffusions on arbitrary graphs," *IEEE Transactions on Image Processing*, vol. 15, no. 7, pp. 1993–2005, 2006.