A Comparison of Texture Features Based on SVM and SOM

Chih-Ming Chen[1], Chien-Chang Chen[2] and Chaur-Chin Chen[1]
[1]*Department of Computer Science, National Tsing Hua University, Hsinchu 300, Taiwan*
[2]*Department of Information Management, Hsuan Chang University, Hsinchu 300, Taiwan*
Email: cchen@cs.nthu.edu.tw

Abstract

Experimental results of texture features derived from Gabor and other four wavelet transforms classified and clustered based on Support Vector Machine (SVMs) and Self-Organizing Maps (SOMs) are reported in this paper. A comparison of SVM and SOM in texture classification is illustrated. The results show that these texture sets with appropriate classifiers perform reasonably well.

1. Introduction

Over the past few decades a considerable number of studies have been made on texture analysis [5]. Recent multimedia applications tend to use texture extractors derived from wavelet transforms [4][11]. What seems to be lacking, however, is a comparison of texture features by Support Vector Machine (SVM) and Self-Organizing Map (SOM). Besides, the texture features by SVM and SOM have performed from a slightly different viewpoint: the training data with or without normalization may affect the results of performance.

Texture features are extracted by simulating the perceptual properties such as orientation, coarseness, fineness and regularity. The recently used texture extractors, Gabor and four wavelet transforms [1][2][3], are reviewed in Section 2. SVM and SOM are summarized in Section 3. Section 4 shows experimental results of texture features by SVM and SOM. Section 5 gives the conclusion.

2. Review of recent texture extractors

2.1. Features derived from Gabor

The Gabor transform proposed in 1946 by Gabor is popular method for simulating human visual system. For texture analysis, the 2-d symmetrical Gabor filter can be introduced as follows [1] [4]

$$g(x,y)=\exp\left(-\frac{1}{2}\left[\frac{x^2}{\sigma_x^2}+\frac{y^2}{\sigma_y^2}\right]\right)\times\cos\left[2\pi\mu_0(x\cos\theta+y\sin\theta)\right] \quad (1)$$

where μ_0 is the frequency of a sinusoidal plane and σ_x and σ_y are corresponding to the variance of the Gaussian distribution. The Fourier transform of (1) can be computed as

$$G(u,v)=\pi\sigma_x\sigma_y\left\{\exp\left(-\frac{1}{2}\left[\frac{(u-\mu_0\cos\theta)^2}{\sigma_u^2}+\frac{(v-\mu_0\sin\theta)^2}{\sigma_v^2}\right]\right)+\exp\left(-\frac{1}{2}\left[\frac{(u+\mu_0\cos\theta)^2}{\sigma_u^2}+\frac{(v+\mu_0\sin\theta)^2}{\sigma_v^2}\right]\right)\right\} \quad (2)$$

where $\sigma_u=\frac{1}{2\pi\sigma_x}$ and $\sigma_v=\frac{1}{2\pi\sigma_y}$.

To derive the Gabor features, we define a set of Gabor filters by setting the same σ_x, σ_y and μ_0 with four different θs [4] to obtain 12 Gabor subbands by using (σ, μ_0) = (0.5426, 0.75), (1.6279, 0.25) and (4.8836, 0.0833) with $\sigma=\sigma_x=\sigma_y$ and the four directions θ = 0º, 45º, 90º, 135º. Consequently, 24 texture features are defined as the mean and standard deviation from each of 12 Gabor subbands.

2.2. Features derived from wavelet transforms

The wavelet transforms have been widely studied in image processing since 1988 [2][3]. Wavelets include various bases, we aim at the four commonly used bases, Haar, Daubechies' four (Daub4), 5/3, and 9/7 wavelet transforms [2]. *For easy implementations, a wavelet basis W is represented as a matrix form* [4] [5]. Let *f*(*x*, *y*) be an input image and let $\otimes$ be a matrix multiplication. Then, the output *Y* of wavelet transform on *f* can be written as

$$Y\leftarrow W\otimes f\otimes W^t. \quad (3)$$

The procedure of a 3-scale wavelet transform is described as Figure 1 [5]. The L and H indicate the low and high pass filters of a wavelet transform. Then, the four subbands of the first level, LL_1, LH_1, HL_1 and HH_1, are obtained from the row and column convolutions with low and high pass filters. The procedure is to repeat sequentially the row and column convolutions on the subband composed of coefficients of lowest-frequency twice. Thus, the input image is decomposed into ten subbands of different spatial frequency.

Finally, the mean and standard deviation of wavelet coefficients in each subband are computed as features,

0-7695-2521-0/06/$20.00 (c) 2006 IEEE

so there are twenty features derived from a 3-scale wavelet transform. The matrices corresponding to the four wavelet transforms are given as follows.

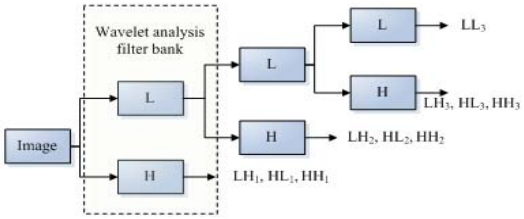


Figure 1. A 3-scale wavelet transform.

2.2.1. Haar Wavelet Transform: The Haar transform is the simplest type of the wavelet transforms. In brief, the concept of Haar wavelet can be explained by moving average and moving difference whose matrix form is given below.

$$W_{\text{Haar}} = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 & & \cdots & & 0 \\ 0 & 0 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & & \cdots & 0 \\ \vdots & & & & \ddots & & \vdots \\ 0 & & & & 0 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 & & \cdots & & 0 \\ 0 & 0 & -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & & \cdots & 0 \\ \vdots & & & & \ddots & & \vdots \\ 0 & & & & 0 & -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix}. \quad (4)$$

2.2.2. Daub4 Wavelet Transform: As in the Haar transform, the Daub4 transform is expressed as

$$W_{\text{Daub4}} = \begin{bmatrix} c_0 & c_1 & c_2 & c_3 & 0 & & \cdots & & & 0 \\ 0 & 0 & c_0 & c_1 & c_2 & c_3 & & \cdots & & 0 \\ & & & & & & \ddots & & & \\ c_2 & c_3 & 0 & & & \cdots & & 0 & c_0 & c_1 \\ c_3 & -c_2 & c_1 & -c_0 & 0 & & \cdots & & & 0 \\ 0 & 0 & c_3 & -c_2 & c_1 & -c_0 & & \cdots & & 0 \\ & & & & & & \ddots & & & \\ c_1 & -c_0 & & & & \cdots & & & c_3 & -c_2 \end{bmatrix} \quad (5)$$

where $c_0 = \frac{1+\sqrt{3}}{4\sqrt{2}}, c_1 = \frac{3+\sqrt{3}}{4\sqrt{2}}, c_2 = \frac{3-\sqrt{3}}{4\sqrt{2}}$, and $c_3 = \frac{1-\sqrt{3}}{4\sqrt{2}}$.

2.2.3. 5/3 Wavelet Transform: The 5/3 wavelet transform is a reversible transform for lossless image data compression in JPEG2000 [11] whose matrix form is given in (6).

2.2.4. 9/7 Wavelet Transform: Besides reversible 5/3 wavelet transform, a floating bi-orthogonal 9/7 wavelet transform, is presented to compress images more efficiently. The forward 9/7 wavelet transform is given in (7). Based on (3), the texture features can be derived by using the 9/7 wavelet matrix.

$$W_{53} = \begin{bmatrix} \frac{3}{4} & \frac{1}{4} & -\frac{1}{8} & 0 & \cdots & \cdots & & 0 & -\frac{1}{8} & \frac{1}{4} \\ -\frac{1}{8} & \frac{1}{4} & \frac{3}{4} & \frac{1}{4} & -\frac{1}{8} & 0 & & & & 0 \\ \vdots & & & & & \ddots & & & & \vdots \\ \vdots & & & & & & \ddots & & & \vdots \\ -\frac{1}{8} & 0 & & \cdots & & 0 & -\frac{1}{8} & \frac{1}{4} & \frac{3}{4} & \frac{1}{4} \\ -\frac{1}{2} & 1 & -\frac{1}{2} & 0 & & & \cdots & & & 0 \\ \vdots & 0 & -\frac{1}{2} & 1 & -\frac{1}{2} & 0 & \cdots & & & 0 \\ \vdots & & & & & \ddots & & & & \vdots \\ \vdots & & & & & & \ddots & & & \\ -\frac{1}{2} & 0 & & \cdots & \cdots & & & 0 & -\frac{1}{2} & 1 \end{bmatrix} \quad (6)$$

$$W_{97} = \begin{bmatrix} \alpha_0 & \alpha_1 & \alpha_2 & \alpha_3 & \alpha_4 & 0 & \cdots & \cdots & 0 & \alpha_{-4} & \alpha_{-3} & \alpha_{-2} & \alpha_{-1} \\ \alpha_{-2} & \alpha_1 & \alpha_0 & \alpha_1 & \alpha_2 & \alpha_3 & \alpha_4 & 0 & \cdots & \cdots & 0 & \alpha_{-4} & \alpha_{-3} \\ \vdots & & & \cdots & & & & & \ddots & \ddots & & & \vdots \\ \vdots & & & \cdots & & & & & & \ddots & \ddots & & \vdots \\ \alpha_2 & \alpha_3 & \alpha_4 & 0 & \cdots & \cdots & 0 & \alpha_{-4} & \alpha_{-3} & \alpha_{-2} & \alpha_{-1} & \alpha_0 & \alpha_1 \\ \beta_0 & \beta_1 & \beta_2 & \beta_3 & \beta_4 & 0 & \cdots & & & \cdots & 0 & \beta_{-2} & \beta_{-1} \\ \beta_{-2} & \beta_{-1} & \beta_0 & \beta_1 & \beta_2 & \beta_3 & \beta_4 & 0 & \cdots & & & \cdots & 0 \\ \vdots & & & & & & \ddots & & & & \ddots & & \vdots \\ \vdots & & & & & & & \ddots & & & & \ddots & \vdots \\ \beta_2 & \beta_3 & \beta_4 & 0 & \cdots & & & \cdots & 0 & \beta_{-2} & \beta_{-1} & \beta_0 & \beta_1 \end{bmatrix} \quad (7)$$

where the coefficients for the W_{97} are given in Table 1.

Table 1. The coefficients of 9/7 wavelet transform.

α_{-4}	0.0267487574108097 6	β_{-2}	0.09127176311424948
α_{-3}	-0.01686411844287495	β_{-1}	-0.05754352622849957
α_{-2}	-0.07822326652898785	β_0	-0.591271763114247 0
α_{-1}	0.2668641184428723	β_1	1.115087052456994
α_0	0.6029490182363579	β_2	-0.591271763114247 0
α_1	0.2668641184428723	β_3	-0.05754352622849957
α_2	-0.07822326652898785	β_4	0.09127176311424948
α_3	-0.01686411844287495		
α_4	0.0267487574108097 6		

2.3. Features derived from Gabor and wavelets

To implement a Gabor transform on an image X, we need to compute inverseFT(FT(X)×FT(G)), where × is a pointwise operation and FT represents Fourier transform. For other wavelet transforms, a matrix computation WXW^t is applied. For either Gabor or other four wavelet transforms, the mean and standard deviation of corresponding transformed coefficients in each subband are computed as features.

3. Texture classification

3.1. Support Vector Machine (SVM)

Support Vector Machine (SVM) investigated by Vapnik has recently been proposed as new machine learning system based on statistical learning theory [6]. SVM designs the classifier functions by constructing hyperplanes in a multidimensional space that separate different categories of the training data. The main idea is to build the hyperplanes as the decision boundaries by using the fitting kernel such as radial basis function (RBF), polynomial or linear classifiers [6] [5]. Then, the hyperplanes try to split the positive examples from the negative examples and maximize the distance of the marginal separation between classes.

In practice, there are various algorithms to train SVM. We adopt the Platt's SMO algorithm to train the input data in our experiments [7] [8]. For multi-class classification, the one-against-one approach is chosen to classify the test data.

3.2. Self-Organizing Map (SOM)

The Self-Organizing Map was developed by Kohonen in the early 1980s [9]. Based on the artificial neural networks, the weights of the neurons in the SOM are adjusted to fit the various input classes of patterns in the training data. In practice, the SOM will construct a topology map, preserving mapping from the high dimensional space onto map units with one or two dimensions. It is a useful tool for visualizing high dimensional data in one or two dimensional space. Moreover, the topology map can be easily adjusted to fit the particular patterns according to many external parameters of the SOM [5].

In our experiments, we adopt the algorithm of Kohonen's SOM [9], which can be divided into several steps as follows [5]

Step 1: Each feature in the training data (cross patterns) is normalized to lie on the surface of unit sphere..

Step 2: Find the best matching neuron (winning node) and update the weight vectors of winning node and its neighborhood.

Step 3: Iterate Step 2 until a sufficiently accurate map is acquired. In the experiments, we assume T=1000 as a suitable number of passes through the training data.

The map is then calibrated by assigning the labels or classes to the neurons. Finally, the calibrated map serves as a classifier for input test vectors by using the nearest neighbor decision.

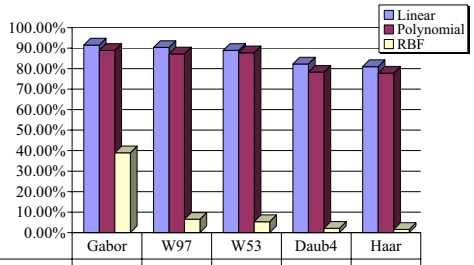
4. Experimental results

The 96 classes of visually selected homogeneous texture images of size 512 by 512 are taken from Brodatz images [5][11]. The database consists of 1536 texture images of size 128 by 128 which are collected from each 512 by 512 image with non-overlapping regions. The features derived from Gabor and four wavelet transforms. Among all images of the database, the 768 images are used as training for SVM and SOM, and the other 768 images are used to test the classifiers.

4.1. SVM-based classification using various kernels with/without normalization

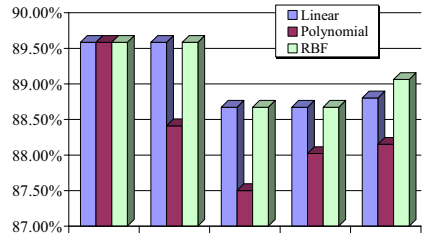
For SVM, we take three kernels: Linear, Polynomial and Radial Basis Function (RBF) [5][7], and evaluate the effects of normalized and non-normalized features.

Figures 2 and 3 show that the SVM using the RBF kernel is sensitive to the training data. A possible interpretation is that most features have extreme variance so that the distance between any two patterns is almost negligible. On the other hand, the texture features derived from Daub4 and Haar transforms are consistent for the SVM with linear kernel. Recall that the 5/3 and 9/7 use larger size of neighborhoods than Daub4 and Haar transforms to compute features, which might also introduce noise with non-local properties into texture features.



	Gabor	W97	W53	Daub4	Haar
Linear	91.54%	90.36%	88.93%	82.29%	80.99%
Polynomial	88.93%	87.24%	87.76%	78.26%	77.86%
RBF	38.93%	6.51%	5.34%	2.08%	1.56%

Figure 2. Results of SVM using various kernels without normalization.



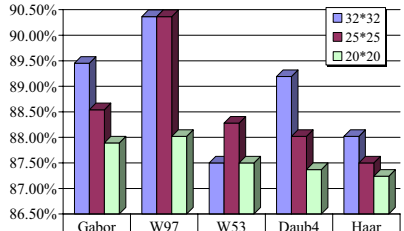
	Gabor	W97	W53	Daub4	Haar
Linear	89.58%	89.58%	88.67%	88.67%	88.80%
Polynomial	89.58%	88.41%	87.50%	88.02%	88.15%
RBF	89.58%	89.58%	88.67%	88.67%	89.06%

Figure 3. Results of SVM using various kernels with normalization.

4.2. SOM-based classification using different sizes of topology maps

For SOM classification, the three sizes (20×20, 25×25 and 32×32) of maps are selected to train and test the texture features. The SOM are trained with 1000 iterations, $\alpha(0)$=0.7 and neighborhood size 7×7.

The results in Figure 4 show that a map of larger size performs slightly better than others. However, the larger size of map takes more time to train input patterns. Besides, the larger size of map must affect the selection of neighborhood size.



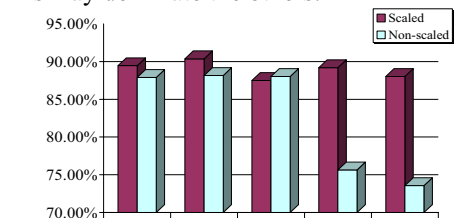
	Gabor	W97	W53	Daub4	Haar
32*32	89.45%	90.36%	87.50%	89.19%	88.02%
25*25	88.54%	90.36%	88.28%	88.02%	87.50%
20*20	87.89%	88.02%	87.50%	87.37%	87.24%

Figure 4. Results of the different size of SOM.

4.3. The effect of SOM-based classification with/without normalization

As for the SVM classification, the SOM with normalization and without normalization also affects the classification of Daub4 and Haar transforms.

Figure 5 shows the texture features of Daub4 and Haar transforms with normalized features are better than those without normalization, which indicates that some subsets of features from either Daub4 or Haar transforms may dominate the others.



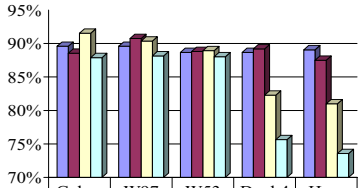
	Gabor	W97	W53	Daub4	Haar
Scaled	89.45%	90.36%	87.50%	89.19%	88.02%
Non-scaled	87.89%	88.15%	88.02%	75.65%	73.57%

Figure 5. Results of SOM of size 32×32 by using normalized and non-normalized data.

4.4. Comparison of texture classification based on SVM and SOM

This section takes a comparison of SVM and SOM with better classification rates from the above experimental results. The SOM is trained with 1000 iterations, map size 32×32, $\alpha(0)=0.7$ and neighborhood size 7×7, and the SVM selects the linear and RBF kernels corresponding to normalized and non-normalized data.

Figure 6 shows SVM of using Gabor transform without normalization performs better than others. SOM is more sensitive to the data with or without normalization than SVM. The normalized Daub4 and Haar features are better than non-normalization ones.



	Gabor	W97	W53	Daub4	Haar
SVM (scaled)	89.58%	89.58%	88.67%	88.67%	89.06%
SOM (scaled)	88.54%	90.76%	88.80%	89.19%	87.50%
SVM (non-scaled)	91.54%	90.36%	88.93%	82.29%	80.99%
SOM (non-scaled)	87.89%	88.15%	88.02%	75.65%	73.57%

Figure 6. A Comparison of SVM and SOM.

5. Conclusion

Experimental results show that SVM is more stable than SOM for evaluating texture features although both are good tools for studying classification and clustering. The computationally intensive Gabor features associated with SVM are good for classification which matches the previous studies in [11]. The use of normalization of texture features for classification must be carefully judged.

6. Acknowledgments

This work is supported by NSC 95-2213-E-007-035

7. References

[1] A.K. Jain and F. Farrokhnia, "Unsupervised Texture Segmentation Using Gabor Filters," Pattern Recognition, vol. 24, 1167-1186, 1991.

[2] M. Antonini, M. Barlaud, P. Mathieu, and I. Daubechies, "Image Coding Using Wavelet Transforms," IEEE Trans. on Image Processing, vol. 1, 205-220, 1992.

[3] I. Daubechies, "Orthonormal Bases of Compactly Supported Wavelets," Communications on Pure and Applied Mathematics, vol. 41, 909-996, 1988.

[4] C.C. Chen and C.C. Chen, "Filtering Methods for Texture Analysis," Pattern Recognition Letters, vol. 20, 783-790, 1999.

[5] C.M. Chen, *A Comparison of Texture Features Based on SVM and SOM*, Master Thesis, National Tsing Hua University, March 2006.

[6] V. Vapnik, Statistical learning theory, John Wiley & Sons, New York, 1998.

[7] J. Platt, Fast training of support vector machines using sequential minimal optimization. In B. Scholkopf, C. Burges and A. Smola, *Advances in kernel methods: support vector learning*, MIT Press, Cambridge, 1998.

[8] S. Keerthi, S. Shevade, C. Bhattacharyya, and K. Murthy, "Improvements to Platt's SMO algorithm for SVM classifier design," Neural Computation, vol. 13, 637–649, 2001.

[9] T. Kohonen, Self-Organizing Maps, 3rd Extended Edition, Springer, Berlin, 2001.

[10] J. Zhang and T. Tan, "Brief Review if Invariant Texture Analysis Methods," Pattern Recognition, vol. 35, 735-747, 2002.

[11] B.S. Manjunath, P. Salembier, and T. Sikora, Introduction to MPEG-7, John Wiley & Sons, 2002.