International Journal of Advanced Research in Engineering and Applied Sciences
ISSN: 2278-6252

EXTRACTION AND ANALYSIS OF LINEAR FEATURES IN IMAGE PROCESSING

Navid Samimi*

Hedayat Bahadori*

Shabnam Azari*

Abstract: *Quantizing to pictures is the best method to introduce of features. Many variety models and equations are introduced for features of a picture that each model can consider unequal number of feature for a picture. We will represent some mathematical model to earn features of picture. At first texture description at a picture will be represented and then a method statistical matrix is called Co-occurrence matrix that is about using texture description classification algorithm will be represented. At the end, we will introduce feature extract algorithm of pictures.*

Key words*: feature extract, Texture description, Co-Occurrence Matrix.*

*Sama Technical and Vocational Training College, Islamic Azad University, Omidiyeh Branch, Omidiyeh, Iran

1. INTRODUCTION

In pattern recognition and in image processing, feature extraction is a special form of dimensionality reduction. When the input data to an algorithm is too large to be processed and it is suspected to be notoriously redundant (e.g. the same measurement in both feet and meters) then the input data will be transformed into a reduced representation set of features (also named features vector). Transforming the input data into the set of features is called feature extraction. If the features extracted are carefully chosen it is expected that the features set will extract the relevant information from the input data in order to perform the desired task using this reduced representation instead of the full size input.

Feature extraction involves simplifying the amount of resources required to describe a large set of data accurately. When performing analysis of complex data one of the major problems stems from the number of variables involved. Analysis with a large number of variables generally requires a large amount of memory and computation power or classification algorithm which over fits the training sample and generalizes poorly to new samples. Feature extraction is a general term for methods of constructing combinations of the variables to get around these problems while still describing the data with sufficient accuracy.

Best results are achieved when an expert constructs a set of application-dependent features. Nevertheless, if no such expert knowledge is available general dimensionality reduction techniques may help. These include:

- Principal component analysis
- Semi-definite embedding
- Multifactor dimensionality reduction
- Multi-linear subspace learning
- Nonlinear dimensionality reduction
- Is map
- Kernel PCA
- Multi-linear PCA
- Latent semantic analysis
- Partial least squares

- Independent component analysis
- Auto encoder

2- TEXTURE DESCRIPTION METHOD

There exist a number of classification algorithms. Among the most widely used are parametric statistical classifiers derived from the Bayesian decision theory, nonparametric k-nearest neighbor classifier, and various neural networks such as multilayer perceptions. See [1] for an introduction to statistical pattern recognition methods for an introduction to neural network methods.

Given a texture description method, the performance of the method is often demonstrated using a texture classification experiment, which typically comprises of following steps (please note that not all steps may always be needed and the order of the steps may vary):

a) Selection of image data: the image data and textures may be artificial or natural, possibly obtained in a real world application. Textures are probably the most widely used image data in texture analysis literature. Other well known data sets are VisTex and MeasTex textures. An important part of the selection of image data is the availability and quality of the ground truth associated with the images: do we really know that each image indeed represents the texture category it is supposed to represent according to the ground truth?

b) Partitioning of the image data into sub images: image data are often limited in terms of the number of original source images available, hence in order to increase the amount of data the images are divided into sub images, either overlapped or disjoint, of a particular window size.

c) Preprocessing of the (sub) images: the (sub) images may have different gray scale properties. In texture analysis the goal is to discriminate (sub) images based on texture, not on first or second order gray scale properties. Therefore (sub) images are often preprocessed to have uniform gray scale distribution, or equal first and second order statistics, by histogram equalization, for example.

d) Partitioning of the (sub) images data into training and testing sets. In order to obtain an unbiased estimate of the performance of the texture classification procedure, training and testing sets should be independent. Different approaches can be used, including N-fold (the collection of (sub)images is divided into N disjoint sets, of which

N-1 serve as training data in turn and the Nth set is used for testing), leave-one-out (each (sub)image is classified one by one so that other (sub)images serve as the training data) and holdout (the data is, preferably randomly, divided into separate training and testing sets, this can be repeated for a number of iterations for a more reliable estimate of performance).

e) Selection of the classification algorithm. In addition to classification algorithm this may involve other selections such as metrics or similarity measures. Selection of classification algorithm can have great impact in the final performance of the texture classification procedure - no classifier can survive with poor features, but good features can be wasted with poor classifier design.

f) Definition of the performance criterion: two basic alternatives are available, analysis of feature values and class assignments, of which the latter is used much more often. In the former the similarity of feature values between training and testing sets, or the separation of class clusters provided by the feature values, provides the basis for the quantitative performance analysis. In the case of class assignments the items in the testing set are classified, and the proportion of correctly (classification accuracy) or erroneously (classification error) classified items is used as performance criterion.

It is obvious that the final outcome of a texture classification experiment depends on numerous factors, both in terms of the possible built-in parameters in the texture description algorithm and the various choices in the experimental setup. Results of texture classification experiments have always been suspect to dependence on individual choices in image acquisition, preprocessing, sampling etc., since no performance characterization has been established in the texture analysis literature. Haralick criticized this questionable status quo from the perspective of computer vision, which applies to texture analysis as well: "This is an awful state of affairs for the engineers whose job is to design and build image analysis or machine vision systems" [2]. Therefore, all experimental results should be considered to be applicable only to the reported setup. Fortunately, there is some recent work aimed at improving the situation with standardized test benches, for example the Tex framework for benchmarking texture classification algorithms [3]. Additionally, an increasing number of researchers are making the imagery and algorithms used in their work publicly available in the web.

3- GREY LEVEL CO-OCCURRENCE MATRIX (GLCM)

The GLCM was introduced by Haralick *et al*. [1]. It is a second order statistical method which is reported to be able to characterize textures as an overall or average spatial relationship between grey tones in an image. Its development was inspired by the conjectured that second order probabilities were sufficient for human discrimination of texture.

In general, GLCM could be computed as follows. First, an original texture image *D* is re-quantized into an image *G* with reduced number of grey level, N_g. A typical value of N_g is 16 or 32. Then, GLCM is computed from *G* by scanning the intensity of each pixel and its neighbor, defined by displacement *d* and angle ø. A displacement, *d* could take a value of 1,2,3,…*n* whereas an angle, ø is limited 0°, 45°, 90° and 135°.

The GLCM *P(i,j|d,ø)* is a second order joint probability density function *P* of grey level pairs in the image for each element in co-occurrence matrix by dividing each element with N_g. Finally, scalar secondary features are extracted from this co-occurrence matrix. In this paper, 8 of most commonly used GLCM secondary features as defined were employed as defined in Eq. 1 to Eq.8. All these features were employed as inputs to the neural network classifier.

Energy: $\sum_{i,j} P(i,j)^2$ (1)

Entropy: $-\sum_{i,j} P(i,j)\log P(i,j)$ (2)

Homogeneity: $\sum_{i,j} \frac{1}{1+(i-j)^2} P(i,j)$ (3)

Inertia: $\sum_{i,j} (i-j)^2 P(i,j)$ (4)

Correlation: $-\sum_{i,j} \frac{(i-\mu)(j-\mu)}{\sigma^2} P(i,j)$ (5)

Shade: $\sum_{i,j} (i+j-2\mu)^3 P(i,j)$ (6)

Prominence: $\sum_{i,j} (i+j-2\mu)^4 P(i,j)$ (7)

Variance: $\sum_{i,j} (i-\mu)^2 P(i,j)$ (8)

Where $\mu = \mu_x = \mu_y = \sum_i i \sum_j P(i,j) = \sum_j j \sum_i P(i,j)$

And $\sigma = \Sigma_i (i-\mu_x)^2 \Sigma_j P(i,j) = \Sigma_j (j-\mu_y)^2 \Sigma_i P(i,j)$

To visualize the mechanism of GLCM, it is best described by example. To make the example simple, consider a re-quantized image of four intensities as illustrated in Fig. 1(a). Let's assumed that the displacement *d* and angle, ø is 1 and 0°, respectively. The co-occurrence matrix element P(1,2) is computed by counting all pixels in an image which intensity value of 1 and its next neighboring pixel in a same row (d = 1 and ø = 0°) of intensity 2. In this example, there are 2 of such cases, thus P(1,2) = 2 as shown in Fig.1(b). Fig. 1(c) shows the GLCM in the form of probability estimates.

1	1	2	3	4
1	2	3	4	1
2	2	3	3	3
4	3	3	2	1
3	3	3	4	4

Figure 1(a). Image matrix

	1	2	3	4
1	1	2	0	0
2	1	1	3	0
3	0	1	5	3
4	1	0	1	1

Figure 1(b). Co-occurrence matrix

	1	2	3	4
1	0.0625	0.125	0	0
2	0.0625	0.0625	0.1875	0
3	0	0.0625	0.3125	0.1875
4	0.0625	0	0.0625	0.0625

Figure 1(c). Actual GLCM values.

Curvelets as proposed by [4], constitute a relatively new family of non-separable wavelet bases that are designed to effectively represent seismic data with reflectors that generally tend to lie on piece-wise smooth curves. This property makes Curvelets suitable to represent events in seismic whether these are located in shot records or time slices. For these types of signals, Curvelets obtain nearly optimal sparseness, because of (i) the rapid decay for the reconstruction error as a function of the largest coefficients; (ii) the ability to concentrate the signal's energy in a limited number of coefficients; (iii) the ability to map noise and signal to different areas in the Curvelet domain. So how do Curvelets obtain such a high non-linear approximation rate? Without being all inclusive, the answer to this question lies in the fact that Curvelets are

- Multi-scale, i.e. they live in different dyadic corona (see for more detail [3] or the other contributions of the fist author to the proceedings of this conference) in the FK-domain.
- Multi-directional, i.e. they live on wedges within these coronas.
- Anisotropic, i.e. they obey the following scaling law width length2.
- Directional selective with .
- Local both in (x, t)) and KF.
- Almost orthogonal, they are tight frames with a moderate redundancy.

Curvelets live in a wedges of the 2-D Fourier plane and become more directional selective and anisotropic for the higher frequencies. They are localized in both the space (or (x, t)) and spatial KF-domains and have, as consequence of their partitioning, the tendency to align themselves with curves/wave fronts. As such they can be more flexible then a representation yielded by high-resolution Radon because they are local and able to follow any piece-wise smooth curve [5].

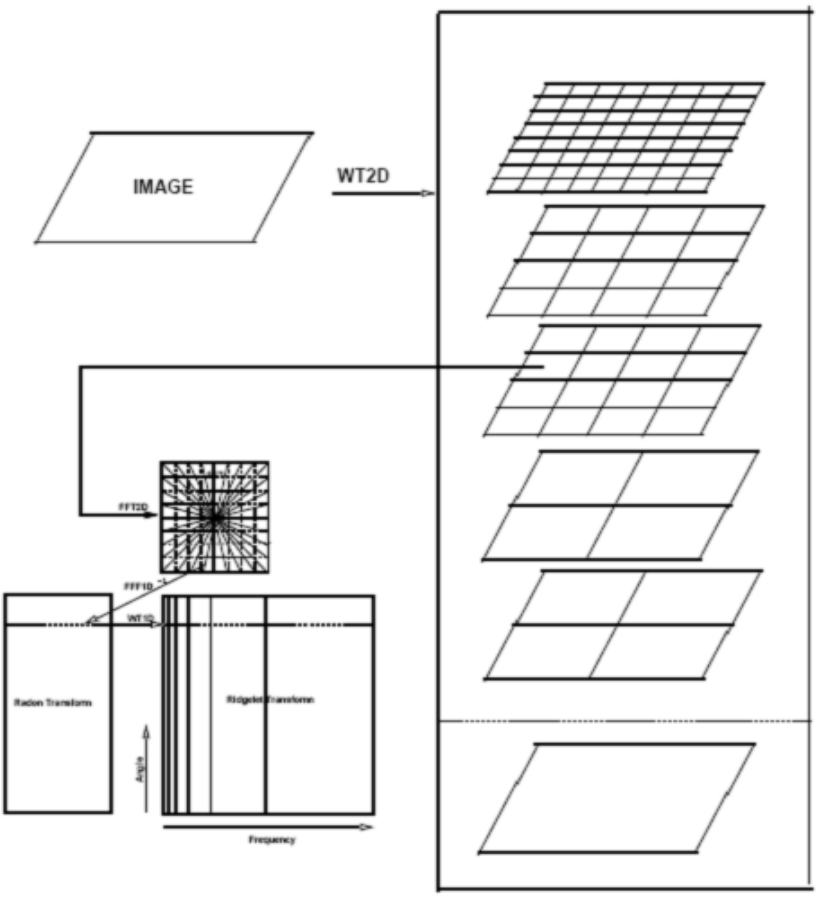

Figure 2: calculation of Curvelet transform

4- SUPPORT VECTOR MACHINE

A Support Vector Machine (SVM) is a discriminative classifier formally defined by a separating hyper plane. In other words, given labeled training data (*supervised learning*), the algorithm outputs an optimal hyper plane which categorizes new examples.

In which sense is the hyper plane obtained optimal? The following simple problem:

For a linearly separable set of 2D-points which belong to one of two classes, find a separating straight line.

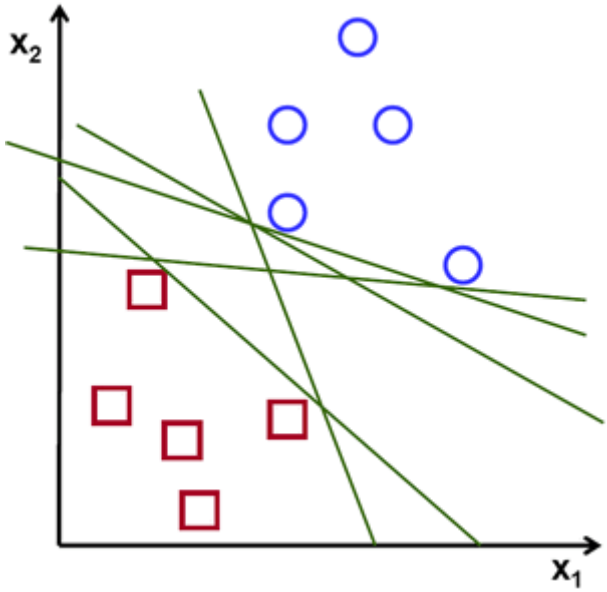

Figure 3: find a separating straight line by svm algorithm

In the above picture you can see that there exist multiple lines that offer a solution to the problem. Is any of them better than the others? We can intuitively define a criterion to estimate the worth of the lines:

A line is bad if it passes too close to the points because it will be noise sensitive and it will not generalize correctly. Therefore, our goal should be to find the line passing as far as possible from all points.

Then, the operation of the SVM algorithm is based on finding the hyper plane that gives the largest minimum distance to the training examples. Twice, this distance receives the important name of margin within SVM's theory. Therefore, the optimal separating hyper plane *maximizes* the margin of the training data [6].

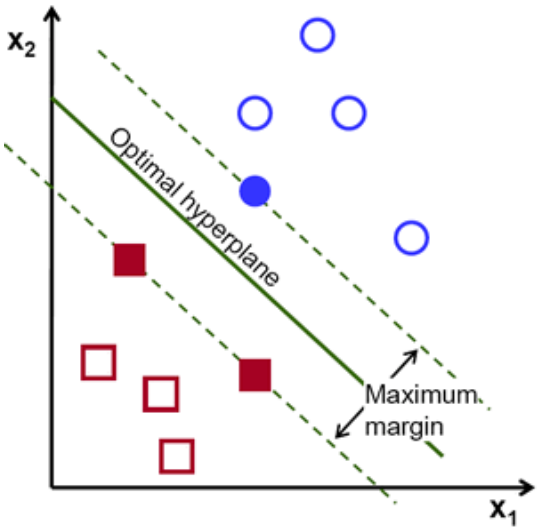

Figure 4: maximizing the margin of the training data

Let's introduce the notation used to define formally a hyper plane:

$$f(x) = \beta_0 + \beta^T x,$$

Where β is known as the *weight vector* and β_0 as the *bias*. The optimal hyper plane can be represented in an infinite number of different ways by scaling of β and β_0. As a matter of convention, among all the possible representations of the hyper plane, the one chosen is

$$|\beta_0 + \beta^T x| = 1$$

Where x symbolizes the training examples closest to the hyper plane. In general, the training examples that are closest to the hyper plane are called support vectors. This representation is known as the canonical hyper plane.

Now, we use the result of geometry that gives the distance between a point x and a hyper plane (β, β_0):

$$\text{distance} = \frac{|\beta_0 + \beta^T x|}{\|\beta\|}.$$

In particular, for the canonical hyper plane, the numerator is equal to one and the distance to the support vectors is

$$\text{distance}_{\text{support vectors}} = \frac{|\beta_0 + \beta^T x|}{\|\beta\|} = \frac{1}{\|\beta\|}.$$

Recall that the margin introduced in the previous section, here denoted as M, is twice the distance to the closest examples:

$$M = \frac{2}{\|\beta\|}$$

Finally, the problem of maximizing M is equivalent to the problem of minimizing a function $L(\beta)$ subject to some constraints. The constraints model the requirement for the hyper plane to classify correctly all the training examples x_i. Formally,

$$\min_{\beta,\beta_0} L(\beta) = \frac{1}{2}\|\beta\|^2 \text{ subject to } y_i(\beta^T x_i + \beta_0) \geq 1 \ \forall i,$$

Where y_i represents each of the labels of the training examples.

This is a problem of optimization that can be solved using Lagrange multipliers to obtain the weight vector β and the bias β_0 of the optimal hyper plane.

5. CONCLUSION

This study has covered most of the methods for feature extraction in image processing. We need to know how we see, in some form, where we can find information and how to process data. More importantly, we need an image, or some form of spatial data. This is to be stored in a computer and processed by our new techniques. As it consists of data points stored in a computer, this data is sampled or discrete. We need to know some of the bounds on the sampling process, on how the image is formed.

REFERENCES

[1] Kawakami Y., T. Abe, and T. Fukunaga, (1993), Muscle-fiber pennation angles are greater in hypertrophied than in normal muscles, J Appl Physiol, 74: p. 2740-2744.

[2] S. Chen and R. M. Haralick, “An automatic algorithm for text skew estimation in document images using recursive morphological transforms,” ICIP- 94, Austin, Texas, November, pp.139-143, 1994.

[3] D. Smith, Evidence for Neutrino Oscillations from LSND at the Los Alamos Meson Physics Facility, 8 July 1997, Non-Accelerator New Physics Workshop, Dubna, Russia, July 7-11, 1997.

[4] G. Ganesan, and C. Raghavendra Rao “Feature Selection using Fuzzy Decision Attributes”, International Journal of Information. Vol.9, No.3, May, pp. 381-384, 2006.

[5] R.C. Gonzalez, and R.E. Woods, “Digital Image Processing”, Prentice-Hall, New Jersey, pp.541, 2002.

[6] J. Weston, A. Gammerman, M. O. Stitson, V. Vapnik, V. Vovk, and C. Watkins. Density estimation using support vector machines. Technical report, Royal Holloway College, Report number CSD-TR-97-23, 1997.