TAGME! 2014 CONTEST: WRITEUP OF THE FEATURES AND ALGORITHMS USED FOR IMAGE CLASSIFICATION

Software Used : Weka, jFeatureLib, Apache Commons Collections

Feature Extraction Process:

The features are extracted from individual images and placed into a bucket for each image. The features are then written to an arff file, which is then used by the neural network classifier to perform the training and predictions. Altogether 582 features were extracted from the images.

Classifier Used : MultiLayer Perceptron with AdaBoost used in classifying. Random Forest used in cross validation.

Distance Measures: Euclidean distance for finding distance from centroids

References:

- Lux Mathias. Content Based Image Retrieval with LIRE. In proceedings of the 19th ACM International Conference on Multimedia, pp. 735-738, Scottsdale, Arizona, USA, 2011
- Steven J. Simske: A Statistical Combined Classifier and its Application to Region and Image Classification,Imaging Systems Laboratory, HP Laboratories Palo Alto
- Guillaume Thibault, Bernard Fertil, Claire Navarro, Sandrine Pereira, Pierre Cau, Nicolas Levy, Jean Sequeira, Jean-Luc Mari. Shape and Texture Indexes, Application to Cell Nuclei Classification », in International Journal of Pattern Recognition and Artificial Intelligence, vol. 27, n° 1, 2013.
- Data Mining: Concepts and Techniques –Jiawei Han and Micheline Kamber, Morgan Kaufmann Publishers

Training Algorithm:

Tunable Parameters : Learning Rate, Momentum, Number of Epochs, Hidden Layers

Input:

D, a data set consisting of the training tuples and their associated target values;

l, the learning rate;

network, a multilayer feed-forward network.

Output: A trained neural network.

Algorithm

Initialize all weights and biases in network;

while terminating condition is not satisfied { //number of epochs set as terminating condition

for each training tuple X in D {

// Propagate the inputs forward:

for each input layer unit j {

Oj = Ij; // output of an input unit is its actual input value

for each hidden or output layer unit j {

Ij = Σi (wij*Oi+qj); //compute the net input of unit j with respect to the previous layer, i

Oj = $1/(1+e^{-1}$ j ; } // compute the output of each unit j

// Backpropagate the errors:

for each unit j in the output layer

Err j = Oj(1-Oj)*(Tj -Oj); // compute the error

for each unit j in the hidden layers, from the last to the first hidden layer

Err j = Oj(1-Oj)Σk (Errk*wjk); // compute the error with respect to the next higher layer, k

for each weight wi j in network {

D(wij) = (l)Errj*Oi; // weight increment

wij = wij +D(wij) } // weight update

for each bias θj in network {

D(θj) = (l)Err j; // bias increment

θj = θj +D(θj); // bias update

}

}

AdaBoost was used in case the classifier turned out to be a weak classifier.

Features Extracted:

1. Haralick Features:
2. Gabor Filter Features
3. Luminance Features
4. Histogram Features
5. MPEG7 Layout Features
6. Reference Color Similarity Features
7. Centroid Distance Features
8. Mean Standard Skewness Features
9. Pyramid Histogram Of Gradients (PHOG) Features
10. Gray Level Size Zone Matrix Features

Rejected Features:

The images were divided into bands of equal size. For each band, the HSV components were calculated for each pixel. For each band, a vector was generated specifying whether atleast one pixel in that band exists in a particular range of values.

```
for (double x = 0.2; x <= 1; x += 0.2) {
	for (double y = 0.2; y <= 1; y += 0.2) {
		for (double z = 0.2; z <= 1; z += 0.2) {
			if ((hue >= x - 0.2 && hue <= x)
				&& (sat >= y - 0.2 && sat <= y)
				&& (bri >= z - 0.2 && bri <= z))
		vector[(int) (x * 5 - 1)][(int) (y * 5 - 1)][(int) (z * 5 - 1)] = 1;
		}
	}
}
```

The images were divided into 40 bands of length 6 pixels. Since the range of x,y,z varies from 0 to 1, the vector has a size of 125 values. The entire image has a vector of size 125*40 = 5000. It is a Boolean vector, so it can be used to classify with greater accuracy. However, it took too long to train the classifier with this feature and the accuracy provided did not increase dramatically. So it was rejected.

Interpretation of Validation Data results:

Accuracy using cross-validation was 91%. Accuracy on the validation data was 86.5%.

Converting the images to grayscale was not done for the purpose of finding the color features.

Also, SIFT keypoints matching was not done as the number of SIFT keypoints proved too large to store in memory. The accuracy could be improved by using SIFT and SURF descriptors.

Also, for the Neural Network, Dropout and Dropconnect can be used for increasing the accuracy.