Investigation of the design and use of artificial neural networks in the CBIR

Dragan Markovic, Milan Milosavljevic, Ranko Popovic

Abstract - **Artificial neural networks (ANNs) are used for content based image retrieval (CBIR). Training of a neural network requires that the user specifies the network structure and sets the learning parameters. In this study, the optimum design of ANN's for retrieval of images is investigated. We use test image datasets in a series of experiments that evaluate the effects on network performance (measured in terms of the Mean Square Error and number of iteration (time required for training)) of different choices of network size and structure, network parameters, training samples size. We use a test image database of 1000 images including 10 classes. Each class has 100 images. The backpropagation algorithm, also called the generalized delta rule, was used for neural network training. An activation function was hyperbolic tangent. Experiments show that with used category of images optimal number of neurons in hidden layer is half of number of images in used training set. We intend to design small CBIR system for educational purposes and potentially for mobility environment.**

Keywords - **Artificial neural networks, backpropagation algorithm, color histogram, content-based image retrieval.**

I. DATA USED AND METHODOLOGY

Content-based image retrieval, uses the visual contents of an image such as *color*, *shape*, *texture*, and *spatial layout* to represent and index the image [1]. A visual content descriptor can be either global or local. A global descriptor uses the visual features of the whole image, whereas a local descriptor uses the visual features of *regions* or *objects* to describe the image content. To obtain the local visual descriptors, an image is often divided into parts first. The simplest way of dividing an image is to use a *partition*, which cuts the image into tiles of equal size and shape. A simple partition does not generate perceptually meaningful regions but is a way of representing the global features of the image at a finer resolution. Color is the most extensively used visual content for image retrieval [2], [3], [4], [6], [8], [9]. Its three-dimensional values make its discrimination potentiality superior to the single dimensional gray values of images. Each pixel of the image can be represented as a point in a 3D color space. Commonly used color space for image retrieval include *RGB*, *Munsell*, *CIE L*a*b**, *CIE L*u*v**, *HSV* (or *HSL*, *HSB*), and *opponent color* space.

D. Markovic, FIM, University Singidunum, Danijelova 32, Belgrade, Serbia; (e-mail: dsamarkovic@singidunum.ac.yu).

M. Milosavljevic, FIM, University Singidunum, Danijelova 32, Belgrade, Serbia; (e-mail: mmilosavljevic@singidunum.ac.yu).

R. Popovic, FIM, University Singidunum, Danijelova 32, Belgrade, Serbia; (e-mail: rpopovic@singidunum.ac.yu).

There is no agreement on which is the best. RGB space is a widely used color space for image display. It is composed of three color components *red*, *green*, and *blue*. These components are called "*additive primaries*" since a color in RGB space is produced by adding them together.

The color histogram serves as an effective representation of the color content of an image if the color pattern is unique compared with the rest of the data set. The color histogram is easy to compute and effective in characterizing both the global and local distribution of colors in an image. In addition, it is robust to translation and rotation about the view axis and changes only slowly with the scale, occlusion and viewing angle. Since any pixel in the image can be described by three components in a certain color space (for instance, red, green, and blue components in RGB space, or hue, saturation, and value in HSV space), a *histogram*, i.e., the distribution of the number of pixels for each quantized bin, can be defined for each component. Clearly, the more bins a color histogram contains, the more discrimination power it has. However, a histogram with a large number of bins will not only increase the computational cost, but will also be inappropriate for building efficient indexes for image databases.

The human visual system can perceive thousands of colors and about 50 gray levels (luminance levels) [13]. Existing retrieval systems use about 200 colors, which allow the use of four to seven luminance levels. However this is not enough, because a “color group“ contains too large a luminance variation. Luminance is one component in some color spaces, for example Lab and Luv. Some systems [10], [11, [14] use a luminance histogram as one of the features.

The color RGB image will be converted to the YUV (Y – lumince; U, V – hrominance) color space with some equtations. These equations assume that the red, green, and blue components have values between 0.0 and 1.0. Since this range is typical represented using 8-bit values between 0 and 255, they need to be scaled and processed as floating-point numbers:

$$Y = .299R + .587G + .114B$$
$$U = -.147R - .289G + .437B$$
$$V = .615R - .515G - .100B$$

The Y component will be quantized with a uniform quantizer to N different values. A normalized luminance histogram will be built from the quantized Y values of each pixel. In this histogram every index corresponds to one of these luminance values.

We are using a test image database of 1000 images including 10 classes, which can be download from the website: http://wang.ist.psu.edu/ jwang/test1.tar. Each class has 100 images.

Artificial neural networks (ANNs) are used in the image retrieval[12]. Backpropagation learning algorithm, also called the generalized delta rule, was an iterative gradient descent training procedure[13].

A specialized application developed in C# was used as software environment for performing of workflow.

II. Training of Multilayer Perceptron (MLP)

An artificial neural network (ANN) or commonly just neural network (NN) is an interconnected group of artificial neurons that uses a mathematical model or computational model for information processing based on a connectionist approach to computation. In most cases an ANN is an adaptive system that changes its structure based on external or internal information that flows through the network.

An artificial neuron is a device with many inputs and one output. The neuron has two modes of operation; the training mode and the using mode. In the training mode, the neuron can be trained to fire (or not), for particular input patterns. In the using mode, when a taught input pattern is detected at the input, its associated output becomes the current output. If the input pattern does not belong in the taught list of input patterns, the firing rule is used to determine whether to fire or not.

An important application of neural networks is pattern recognition. Pattern recognition can be implemented by using a feed-forward neural network that has been trained accordingly. During training, the network is trained to associate outputs with input patterns. When the network is used, it identifies the input pattern and tries to output the associated output pattern.

The commonest type of artificial neural network consists of three groups, or layers, of units: a layer of "**input**" units is connected to a layer of "**hidden**" units, which is connected to a layer of **"output**" units.

- The activity of the input units represents the raw information that is fed into the network.
- The activity of each hidden unit is determined by the activities of the input units and the weights on the connections between the input and the hidden units.
- The behavior of the output units depends on the activity of the hidden units and the weights between the hidden and output units.

This simple type of network is interesting because the hidden units are free to construct their own representations of the input. The weights between the input and hidden units determine when each hidden unit is active, and so by modifying these weights, a hidden unit can choose what it represents.

The MLP classifiers training process depends on a set of parameters. The value of these parameters can be varied to different extent (degree) depending on different conditions of experiment. There are the following parameters which were varied during the experiment:

- Number of training samples;
- Stopping criterion for the training process (error or number of iterations);
- Number of input nodes;
- Learning rate;
- Number of hidden layers;
- Number of hidden nodes;
- Type of an activation function

Training the network can be summarized as follows:

- Apply input to the network.
- Calculate the output.
- Compare the resulting output with the desired output for the given input. This is called the error.
- Modify the weights for all neurons using the error.
- Repeat the process until the error reaches an acceptable value (e.g. error < 1%), which means that the NN was trained successfully, or if we reach a maximum count of iterations, which means that the NN training was not successful.

III. Practical Aspects of Neural Network Implementation

We will present in this subsection some of the practical aspects related to the real implementation of neural networks. We will discuss the normalization of the data set, the splitting of the data set in training and test sets, the architecture of the neural network, the stopping criteria and several other practical conditions.

Let us assume that the original data set was already pre-processed and that problems with missing values and noise or outliers were already treated. In other words, we assume that our data set is a "clean" data set. The work on data preparation for neural network data analysis in [14] presents a study on several aspects regarding data pre-processing.

The first thing that usually one must do to this clean data set is to perform a data normalization, or standardization, to avoid that higher inputs assume a more important role in the learning process than small inputs. The usual normalization processes transform the data so that:

- every feature of the data is scaled in the interval [0; 1] or [-1; 1], or
- every feature is standardized to have zero mean and unitary standard deviation.

The next step is to choose the architecture of the neural network. There is no rule specifying the number of hidden layers and the number of neurons in each hidden layer. We mentioned earlier that a neural network with one hidden layer can approximate arbitrarily close any decision boundary, so, for most problems, a two-layer neural network will be sufficient. A three-layer network can be considered if a data set is particularly hard to train.

Although there are some works suggesting formulas for determining the number of neurons in the hidden layer (examples are [15], [16], [18]), we still think that there is nothing like experimentation. Experiments should be performed with a range of values for the number of hidden neurons that must be chosen taking into account the complexity of the problem. There are also some techniques consisting on starting with a high complex neural network and then, during training, performing a pruning by eliminating those weights with small influence (very low value) [20] – [22].

As for the initial values for the weights and bias, we should use random small values. This is done to prevent the possibility that some of the initial output values could be in the saturation region of the activation function. As activation functions we use, in our experiments, the hyperbolic tangent in all neurons.

The program trains the network using bitmap images that are located in a folder. This folder must be in the following format:

- There must be one (input) folder that contains input images [*.*jpg*].
- Each image's name is the target (or output) value for the network (the bins of gray histogram of the image are the inputs).

The input layer consisted of number of neurons, corresponding to bins of used type of histogram for image from training set. The hidden layer had varying number of neurons in our experiment and the output layer had number of neurons equal to number of images in particular training set.

The backpropagation algorithm was used for neural network training. An activation function was hyperbolic tangent. A network structure was trained with the parameters listed in Table 1.

TABLE 1. SETTINGS OF NETWORK STRUCTURE AND LEARNING PARAMETERS

Parameters	**Choice**
Initial weight range	Random value [0 - 0.5]
Number of input nodes	varying according used image features
Number of hidden layers	1
Number of hidden nodes	varying
Number of output nodes	number of images in particular training set
Learning rate between input and hidden layers	0.2
Learning rate between hidden and output layers	0.2
Type of activation function	hyperbolic tangent

IV. IMPLEMENTATION

We implemented our CBIR system in C# using Windows-Forms. The database are images stored under folder and the trained network is stored as a binary serialization into a file. The network file is called *.net. Each time the system is loaded, the network is loaded and de-serialized into the main memory, and whenever the search takes place it is ready for comparisons.

Query formation by the user is done using the method of example query. Wherever the user wants to find the image he supplies the image, and the system should return similar pictures.

Figure 1. GUI for CBIR system

The system presents the top 2 results for the image query. Thus allowing the user to see some possible suggestions with appropriate confidence.

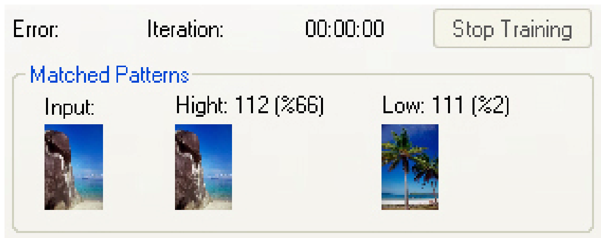

Figure 2. The system presents the top 2 results for the image query

V. EXPERIMENTS AND RESULTS:

We experimented with our system using the following techniques:

- Simple (Global) Histogram.
- Local (Areas based) Histogram.

Experimental platform is PC AMD Sempron 1.6 GHz with Windows XP.

The normalized grayscale histogram extracted from an image is a 256 dimensional vector that is contained in the histogram space S_H (represented by a unit hypercube), and used H as input vector X.

TABLE 2. GLOBAL 256 GRAY-SCALE NORMALIZED HISTOGRAM USED AS IMAGE FEATURE (INPUT VECTOR SIZE = 256)

Training set (number of images)	Number of nodes in hidden layer	Mean-square error	Number of iteration	Training time	Confidence interval in image recognition (%)
30	20	0.8	1908	14”	62 – 93
30	15	0.8	2320	14”	77 – 95
30	10	0.8	3372	14”	71 – 92
50	25	0.8	2576	45”	78 – 95
100	50	0.8	3029	4’ 52”	77 – 97
200	100	0.8	3717	42’ 41”	–

TABLE 3. GLOBAL 256 GRAY-SCALE NORMALIZED HISTOGRAM USED AS IMAGE FEATURE (INPUT VECTOR = 256)

Training set (number of images)	Number of nodes in hidden layer	Mean-square error	Number of iteration	Training time	Confidence interval in image recognition (%)
40	20	0.8	2434	26”	65 – 93
40	20	0.5	2650	29”	66 – 96
40	20	0.1	5592	1’ 2”	85 – 97

In the second experiment images devided in four equal areas and calculated local 256 gray-scale histograms.

TABLE 4. FOUR AREAS 256 GRAY-SCALE NORMALIZED HISTOGRAM USED AS IMAGE FEATURE (INPUT VECTOR = 4 X 256)

Training set (number of images)	Number of nodes in hidden layer	Mean-square error	Number of iteration	Training time	Confidence interval in image recognition (%)
30	15	0.8	1446	36”	66 - 91
50	25	0.8	1780	2’ 16”	84 - 95
100	50	0.8	1695	10’ 43”	–
510	255	0.8	3717	8h 27’ 57”	–

VI. CONCLUSIONS:

It is known that histogram is quite primitive and mainly insufficient way for CBIR purposes. However, with certain image characteristics it may be useful, and works well.

Experiments show that with used category of images optimal number of neurons in hidden layer is half of number of images in used training set.

Despite used low level feature (gray-level histogram) as input vector trained artificial neural networks works perfectly on training set of images, but with growth of training set rapidly is increasing training time wich is inappropriate for practical solutions.

Generic CBIR will have to wait both for algorithmic advances in image understanding and advances in computer hardware, especially when constraints on real time performance are added.

REFERENCES

[1] Fuhui Long, Hongjiang Zhang, David D. Feng: Fundamentals of Content-based Image retrieval, in *Multimedia Information Retrieval and Management – Technological Fundamentals and Applications*, D. Feng, W.C. Siu, and H.J.Zhang. (ed.), Springer, 2002.

[2] J. D. Foley, A. van Dam, S. K. Feiner, and J. F. Hughes, *Computer graphics: principles and practice*, 2nd ed., Reading, Mass, Addison-Wesley, 1990.

[3] J. Huang, S.R. Kumar, M. Metra, W. J., Zhu, and R. Zabith, "Spatial color indexing and applications," *Int’l J. Computer Vision*, Vol.35, No.3, pp. 245-268, 1999.

[4] J. Huang, *et al.*, "Image indexing using color correlogram," *IEEE Int. Conf. on Computer Vision and Pattern Recognition*, pp. 762-768, Puerto Rico, June 1997.

[5] M. Ioka, "A method of defining the similarity of images on the basis of color information, "*Technical Report RT-0030*, IBM Tokyo Research Laboratory, Tokyo, Japan, Nov. 1989.

[6] K. Jain, *Fundamental of Digital Image Processing*, Englewood Cliffs, Prentice Hall, 1989.

[7] E. Mathias, "Comparing the influence of color spaces and metrics in content-based image retrieval, "*Proceedings of International Symposium on Computer Graphics, Image Processing, and Vision*, pp. 371 -378, 1998.

[8] G.Pass, and R. Zabith, "Comparing images using joint histograms," *Multimedia Systems*, Vol.7, pp.234-240, 1999.

[9] M. Stricker, and M. Orengo, "Similarity of color images," *SPIE Storage and Retrieval for Image and Video Databases III*, vol. 2185, pp.381-392, Feb. 1995.

[10] M. J. Swain, and D. H. Ballard, “Color indexing,” *International Journal of Computer Vision*, Vol. 7, No. 1, pp.11-32, 1991.

[11] H. J. Zhang, *et al*, "Image retrieval based on color features: An evaluation study," *SPIE Conf. on Digital Storage and Archival*, Pennsylvania, Oct. 25-27, 1995.

[12] P. Kerminen, J. T. Tanttu and M. Gabbouj, *Utilization of luminance information on color-based image retrieval* , Institute of Signal Processing, in Proceedings for the Nordic MATLAB Conference 2003, October 21-22.2003, pp. 252 - 256, Comsol A/S, ISBN 87-989426-0-3

[13] Rumelhart, D. E., G. E. Hinton, and R. J. Williams, Learning representations by back-propagating errors, Nature, Vol. 323, pp. 533-536, 1986.

[14] L. Yu, S. Wang, and K. Lai. An integrated data preparation scheme for neural network data analysis. *IEEE Trans. on Knowledge and Data Engineering*, 18(2):217–230, 2006.

[15] L. S. Camargo and T. Yoneyama. Specification of training sets and the number of hidden neurons for multilayer perceptrons. *Neural Computation*, 13(12):2673–2680, 2001.

[16] I. Rivals and L. Personnaz. A statistical procedure for determining the optimal number of hidden neurons of a neural model. In *Symposium on Neural Computation*, 2000.

[17] K. H. Schindler and M. Sanguineti. Bounds on the complexity of neuralnetwork models and comparison with linear methods. *Int. Journal of Adaptive Control and Signal Processing*, 17:179–194, 2003.

[18] L. Yujian. A simple method to estimate the size of feedforward neural networks. In *IEEE Int. Conference on Systems, Man and Cybernetics*, volume 2, pages 1322–1326, 2003.

[19] Y. S. Goh and E. C. Tan. Pruning neural networks during training by backpropagation. In *IEEE Region 10’s Int. Conference*, volume 2, pages 805–808, 1994.

[20] E. D. Karnin. A simple procedure for pruning back-propagation trained neural networks. *IEEE Trans. on Neural Networks*, 1(2):239–242, 1990.

[21] M. Shahjahan and K. Murase. A pruning algorithm for training cooperative neural network ensembles. *IEICE Trans. on Information and Systems*, E89UD(3):1257–1269, 2006.

[22] G. Thimm and E. Fiesler. Pruning of neural networks. Technical Report RR 97-03, Institute for Perceptive Artificial Intelligence, 1997.