Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X
IMPACT FACTOR: 6.199

IJCSMC, Vol. 8, Issue. 4, April 2019, pg.127 – 135

Building Accurate and Efficient Color Image Recognizer

Prof. Ziad A. AlQadi[1]; Eng. Ahmad S. AlOthman[2]; Eng. Mahmoud O. Alleddawi[3]; Eng. Mohammad AlHiary[4]; Eng. Osama T. Ghazal[5]; Eng. Sameh S. Salman[6]
Al-Balqa Applied University, Jordan
http://www.bau.edu.jo
dr.ziad.alqadi@bau.edu.jo; alothmanahmad@std.bau.edu.jo; Mahmoud.alleddawi@gmail.com; Eng.hiary@gmail.com; e.osa_gh@yahoo.com; eng.sameh@std.bau.edu.jo

Abstract: Digital color image recognition is one of the most important tasks used in digital image processing, due to the various applications requiring this task. Color image recognition system deals with huge data, thus minimizing the memory space and recognition time will be an important issue. In this paper a method of color image features extraction will be introduced, the extracted features will be minimized and will be unique for each image. A data set if color images features will be built and passed to artificial neural network to be used for recognition purposes,, deferent ANN architectures will be trained , tested and implemented , the obtained results will be used as a guide for selecting the suitable , efficient and accurate ANN.
Keywords: Features array, ANN, FFANN, CFANN, architecture, activation function

1- Introduction

Digital color image is a 3D matrix, the first dimension represents the red color; the second represents the green color; while the third one represents the blue color [1], [2], [3].
Color image can be represented by 2D matrix by reshaping the original color image to 2D matrix as shown in figure 1.

Figure 1: Reshaped color image

Digital color image recognition is one of the most important task used in digital image processing [3], due to the various applications requiring this task such as automobile industry, gamming, healthcare industry, merged reality, retail industry, security industry, social media platforms, visual search engines.

Color images usually have high resolution, and this leads to a huge image size, thus the recognition phase based on pixel by pixel matching will require a lot of processing time and more storage space, table 1 shows the required time using pixel by pixel matching:

Table 1: Recognition time based on pixel matching

Image dimensions	Image size(pixels)	Matching time(seconds)
1854x1473x3	8192826	1.272000
819x1024x3	2515968	0.394000
423x476x3	604044	0.091000
187x269x3	50303	0.011000
543x800x3	1303200	0.203000
395x500x3	592500	0.099000
1600x1600x3	7680000	1.188000
163x288x3	46944	0.014000
400x600x3	720000	0.116000
526x639x3	1008342	0.155000

The matching time increases when the image size increases and there is a linear relationship between them as show in figure 2:

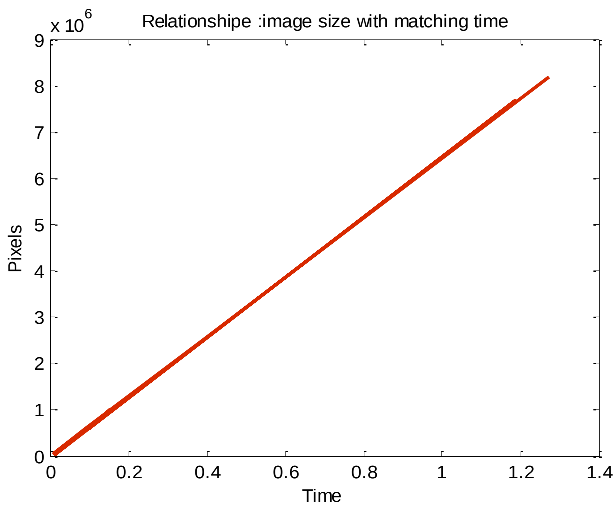

Figure 2: Relationship between matching time and image size

To build an efficient recognition system [4] we have to consider the following important issues:

- Selecting a method to create a features for each image, these features must be unique for every image to make it possible to use a features as a signature or key to retrieve or recognize the image,
- Selecting an efficient tool to be used as a recognizer, and here we can recommend artificial neural network (ANN) [5], [6], [7].

Taking these issues into consideration we can apply the approach of creating recognition system to be built and designed performing the following tasks (as shown in figure 3):

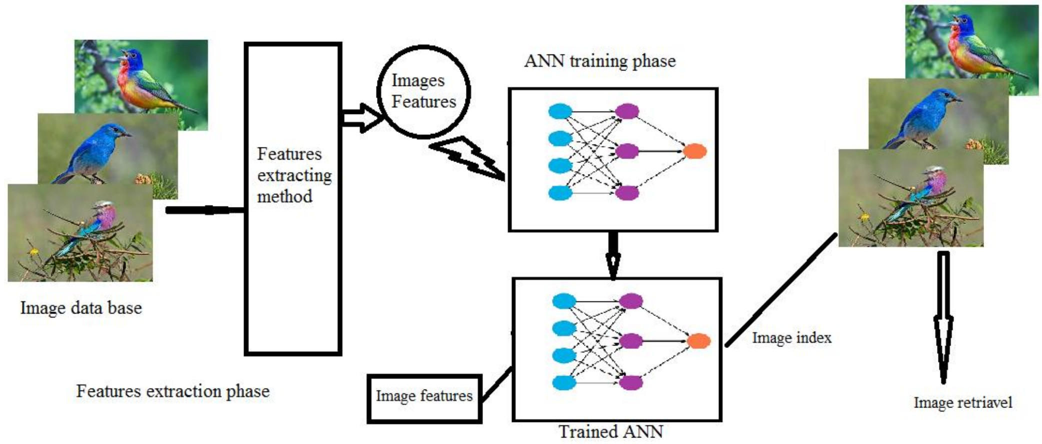

Figure 3: Phases of creating image recognition system

- Building image features data set: this data set will be a 2D matrix, one column for each image features, here we have to select an accurate and efficient method to generate unique features for each image.
- Creating and training ANN to be used later as a recognizer, here we must pay attention on defining the type, and the architecture of ANN [8].
- The good trained ANN must be saved to be used later as a recognition tool.

2- Creating Color Images Features

Many methods of color image features[9], [10], [11] extraction are based on local binary pattern (LBP) [12], [13], these method are implemented in deferent variations [12], [13] to produce a features array of 256, 32, 16, and 4 values[14], [15],[16],[17],[18] .

Here in this paper we will introduce features are method which is based on pixels 8-neighbours and 16-neighbours as shown in figure 4:

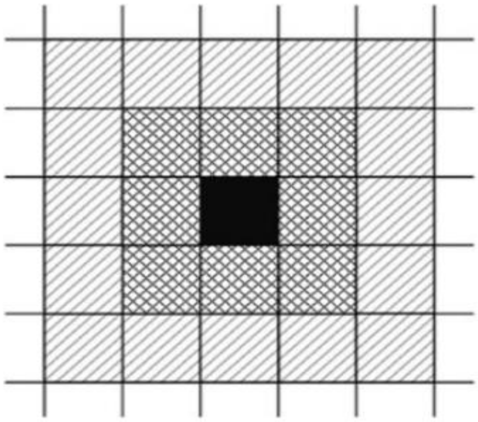

Figure 4: Pixel 8 and 16-neighbours

To extract the features array for each image we have to follow the following steps:

1. For each image do the following
2. Get the original color image.
3. Reshape the 3D color image to 2D image.
4. Initialize the features array (one column with 4 values) to zeros.
5. For each pixel do the following:
6. Find the average of the 8-neighbours (av8).
7. Find the average of the 16-neighbours (av16).
8. If av8>= pixel value let c0=1, else let c0=0.
9. If av16>= pixel value let c1=1, else let c1=0.
10. Find the features array index, index=c0+2xc1.
11. Add 1 to the repetition of this index.
12. End for
13. Add the features array to features matrix.
14. Save features matrix.
15. End for

The following matlab code was implemented using various color images in type and size:

```
a=imread('C:\Users\win 7\Desktop\com-dec\im1.jpg');
[n1 n2 n3]=size(a);
fe=zeros(4,1);
b=reshape(a,n1*n3,n2);
tic
for i=3:n1*n3-3
    for j=3:n2-3
        b8=[b(i-1:i+1,j-1:j+1)];b16=[b(i-2:i+2,j-2:j+2)];
        m8=mean(mean(b8));m16=mean(mean(b16));
        if(m8>b(i,j))   c0=1;
        else            c0=0;
        end
        if(m16>b(i,j))  c1=1;
        else            c1=0;
        end
        k=c0+c1*2;  fe(k+1,1)=fe(k+1,1)+1;
    end
end
toc
fe
```

Table 2 shows the obtained features for some images:

Table 2: Color image features

Image #	Features			
1	5767182	220633	386340	1783521
2	2010798	46037	56059	385694
3	490646	10260	21816	72622
4	39060	503	2444	6041
5	1146117	14005	19742	111216
6	373626	1223	104831	104420
7	4486472	219952	409882	1456194
8	1665651	85482	101296	646159
9	6521045	91498	184372	851110
10	21461	1368	4479	17406
11	65421	11351	11103	59829
12	61706	8284	8823	68067
13	1067276	55157	247499	715961
14	69523	9040	10698	57619
15	294857	40402	53438	322328

From the results shown in table 2 we can raise the following facts:

- Each image feature array is unique, thus we can use this array as a key or signature to retrieve or recognize the image, and this key must be passed to ANN for recognition purposes.
- The feature array is very sensitive to any changes in the image pixels values, this will lead to deferent image and deferent features array as shown in table 3:
- The used method of features extraction is very simple and flexible, any rotation of the image does not affect the features array values, and here the original image and the rotated versions can be treated as one image.

Table 3: Changing pixels values leads to changing features array

Image features	Changing 1 pixel changed	Changing 2 pixel changed	Changing 3 pixel changed	Changing 4 pixel changed
65421	65421	65421	65406	65406
11351	11353	11352	11350	11350
11103	11104	11104	11111	11112
59829	59826	59827	59837	59836

The obtained results using the proposed method of color image features extraction were compared with other LBP based methods (LBP, CSLBP, and RLBP) [9],[15], [16],[17] and here we can raise the followings which can be considered as advantages of the proposed method:

- Image features array is reduced to 4 values, this will reduce the size of the input data set to be passed to ANN recognizer, and the smaller data set size will simplify ANN architecture, which leads to memory space reduction and training time reducing.
- The feature array is sensitive to any change in the image; this will increase the recognition ratio and make it equal or much closed to 100%.
- The features array of the image is not sensitive to process of image rotation, so the original image and the rotated version will have the same features array.

3- Building the Recognizer

The proposed color image recognizer is to be built using ANN.

Artificial neural network a model which contains a set of computational elements called neurons, neurons are organized in one or more layers, each layer must be specified by an activation function, and according this function the output of the neuron will be calculated.

Figure 5 shows how each neuron can operate: in the first step the summation of the products of the weights and the corresponding inputs must be find, then depending on the activation function the output can be generated [7], [8].

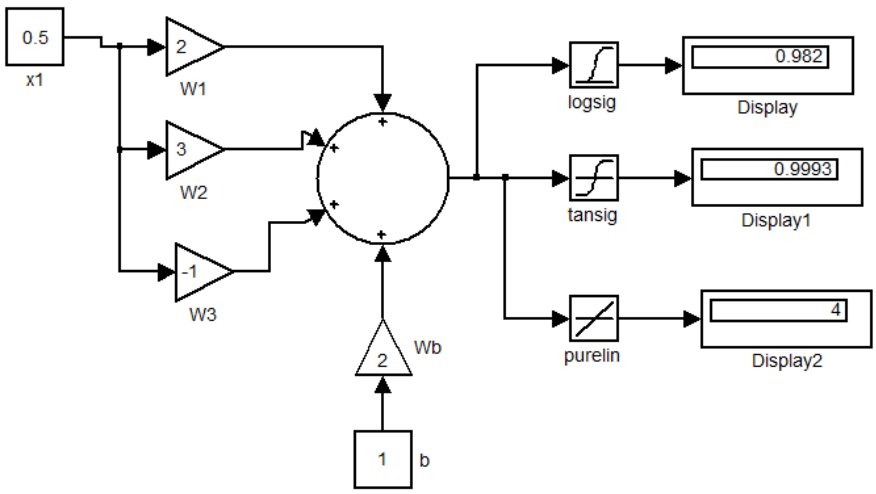

Figure 5: Neuron operations

There are many types of ANN, and here we will focus on feed forward (FFANN), and cascade feed forward ANN (CFANN). Both types of ANN can be used as a tool to recognize the image or generate the image index or classifier which can be used to retrieve or recognize the image.

FFANN architecture contains a set of fully connected neuron organized in one or more layers as shown in figure 6, CFANN likes FFANN, but the inputs of the input layer are connected to the output layer as shown in figure 7.

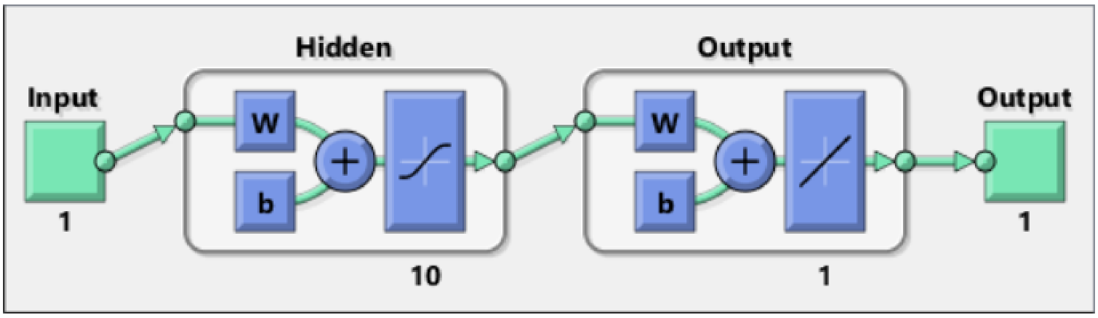

Figure 6: FFANN architecture

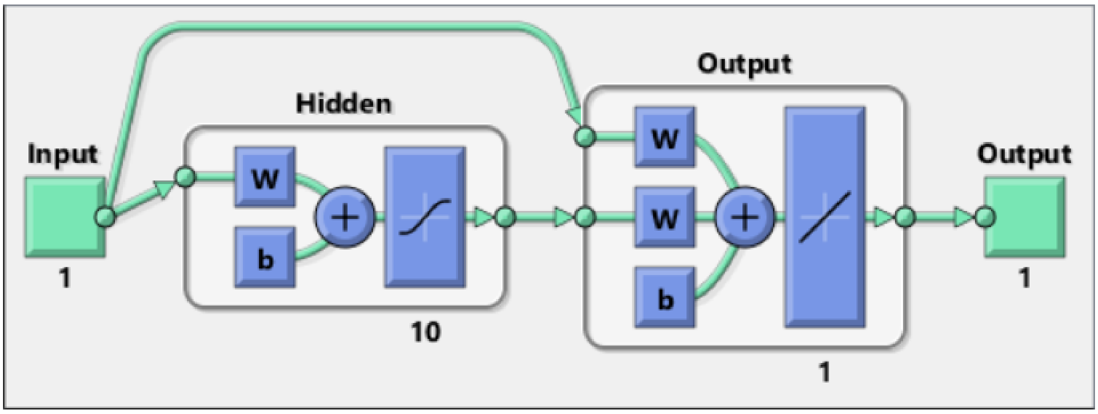

Figure 7: FFANN architecture

To select a suitable ANN we have to pay attention to:

- Selecting an input data set (in our case images features).
- Data normalization (in our case we divide the data by 10000000).
- Selecting the targets (in our case images classifiers or indexes.
- Selecting the type of ANN.
- Selecting ANN architecture by defining the number of layers, the number of neurons in each layer and the activation function for each layer.
- Initializing the weights to zeros.
- Setting some parameters for ANN such as the error(goal must lead to zero), and number of training cycles (epochs);

ANN must be created by considering the above mentioned factors, and then ANN must be trained.

The training phase may be repeated several times until we reach the goal. If ANN does not provide a zero or closed to zero error we can adjust ANN architecture by adding extra layer and/or change the activation function (AF) except for the output layer which must remain linear.

3-1 Using FFANN

The following matlab code was written to create and train FFANN, the code was implemented several times using various architectures and parameters to reach the optimal ANN architecture which will generate the accurate image index which is specified by a selected features array.

Table 4 shows the results on FFANN implementations:

```
load imfeat% image features matrix 4,15
data=imfeat/10000000;
target=1:15;
net=newff(minmax(data),[4 4 1],{'purelin','tansig','purelin'});
net=init(net);
net.trainParam.goal=0;
net.trainParam.epochs=1000;
net=train(net,data,target);
sim(net,data)
```

Table 4: FFANN implementation results

Layers[neurons]	AF	Number of epochs	Training phases	Error %
2[4 1]	Logsig, linear	100	9	0
2[4 1]	Tansig, linear	100	28	26.67
2[4 1]	Tansig, linear	500	12	0
2[4 1]	linear, linear	100	20	100
2[4 1]	linear, linear	1000	20	100
3[4 4 1]	Logsig,logsig,linear	100	24	95
3[4 4 1]	Logsig,logsig,linear	1000	3	0
3[4 4 1]	tansig,logsig,linear	1000	2	0
3[4 4 1]	logsig,tansig,linear	1000	3	0
3[4 4 1]	tansig,tansig,linear	100	23	Closed to target
3[4 4 1]	tansig,tansig,linear	1000	2	0
3[4 4 1]	linear,logsig,linear	1000	7	0
3[4 4 1]	linear,tansig,linear	1000	9	0

From table 4 we can see the following facts:

- We have to be careful when selecting FFANN architecture, some of the selected architecture does not give accurate results regardless the number of training cycles and the number of training phases (green rows in table 4).
- Some time the selected architecture requires more training cycles and training phases (Red rows in table 4).
- Different architectures may give accurate results but with more training time.

3-1 Using CFANN

The following matlab code was written to create and train CFANN, the code was implemented several times using various architectures and parameters to reach the optimal ANN architecture which will generate the accurate image index which is specified by a selected features array.

Table 5 shows the results on FFANN implementations:

```
load imfeat% image features matrix 4,15
data=imfeat/10000000;
target=1:15;
net=newcf(minmax(data),[4 1],{'logsig','purelin'});
net=init(net);
net.trainParam.goal=0;
net.trainParam.epochs=1000;
net=train(net,data,target);
sim(net,data)
```

Table 5: CFANN implementation results

Layers[neurons]	AF	Number of epochs	Training phases	Error %
2[4 1]	Logsig, linear	100	8	0
2[4 1]	Tansig, linear	100	11	0
2[4 1]	Tansig, linear	500	1	0
2[4 1]	linear, linear	100	20	100
2[4 1]	linear, linear	1000	20	100
3[4 4 1]	Logsig,logsig,linear	100	7	0
3[4 4 1]	Logsig,logsig,linear	1000	1	0
3[4 4 1]	tansig,logsig,linear	1000	1	0
3[4 4 1]	logsig,tansig,linear	1000	1	0
3[4 4 1]	tansig,tansig,linear	100	14	0
3[4 4 1]	tansig,tansig,linear	1000	1	0
3[4 4 1]	linear,logsig,linear	1000	3	0
3[4 4 1]	linear,tansig,linear	1000	1	0

From table 5 we can see the following facts:

- CFANN enhances the performance of FFANN, by comparing the results in table 4 and table 5 we can see much improvement when using CFANN
- As For FFANN linear activation function can't be used in hidden layers because it always leads to incorrect results.
- Different architectures may give accurate results but with more training time.

Conclusion

Color image features extraction method was introduced, tested and experimental results were obtained. The Obtained results were compared with LBP based methods results; the proposed method gave optimized data set which can be used to minimize ANN architecture which leads to minimizing training time and memory space size.

Two deferent ANN were selected, trained and tested and it was shown that CFANN provides more efficiency and accuracy in color image recognition process.

References

[1] Majed O. Al-Dwairi, Ziad A. Alqadi, Amjad A. AbuJazar and Rushdi Abu Zneit, Optimized True-Color Image Processing World Applied Sciences Journal 8 (10): 1175-1182, 2010 ISSN 1818-4952.

[2] Akram A. Moustafa and Ziad A. Alqadi, Color Image Reconstruction Using A New R'G'I Model, Journal of Computer Science 5 (4): 250-254, 2009 ISSN 1549-3636.

[3] Musbah J. Aqel, Ziad ALQadi, Ammar Ahmed Abdullah, RGB Color Image Encryption-Decryption Using Image Segmentation and Matrix Multiplication, International Journal of Engineering & Technology, 7 (3.13) (2018) 104-107.

[4] Dr. Ghazi. M. Qaryouti, Dr.Saleh Khawatreh, Prof. Ziad A.A. Alqadi, Prof.Mohammed Abu Zalata, Optimal Color Image Recognition System (OCIRS), International Journal of Advanced Computer Science and Technology. ISSN 2249-3123 Volume 7, Number 1 (2017), pp. 91-99.

[5] Akram A. Moustafa1, Ziad A. Alqadi and Eyad A. Shahroury, Performance Evaluation of Artificial Neural Networks for Spatial Data Analysis, WSEAS TRANSACTIONS on COMPUTERS, Issue 4,Volume 10, pp 115-124 April 2011.

[6] Khaled M. Matrouk,Haitham A. Alasha'ary, Abdullah I. Al-Hasanat, Ziad A. Al-Qadi, Hasan M. Al-Shalabi, Investigation and Analysis of ANN Parameters, European Journal of Scientific Research, ISSN 1450-216X / 1450-202X Vol.121 No.2, 2014, pp.217-225.

[7] Jamil Al-Azzeh, Ziad Alqadi, Mohammed Abuzalata,Performance Analysis of Artificial Neural Networks used for Color Image Recognition and Retrieving, JCSMC, Vol. 8, Issue. 2, February 2019, pp .20–33.

[8] Abdullah Al-Hasanat, Haitham Alasha'ary, Khaled Matrouk, Ziad Al-Qadi, Hasan Al-Shalabi, Experimental Investigation of Training Algorithms used in Back propagation Artificial Neural Networks to Apply Curve Fitting , European Journal of Scientific Research ISSN 1450-216X / 1450-202X Vol.121 No.4, 2014, pp.328-335.

[9] AbuZneit, R., Z. AlQadi, and M. AbuZalata, *A Methodology to Create a Fingerprint for RGB Color Image*. International Journal of Computer Science and Mobile Computing, 2017. **16**(1): p. 205-212.

[10] AlQadi, Z. and H. M.Elsayyed, *Window Averaging Method to Create a Feature Victor for RGB Color Image*. International Journal of Computer Science and Mobile Computing, 2017. **6**(2): p. 60-66.

[11] AlZudool, M., S. Khawatreh, and Z. Alqadi, *Efficient Methods used to Extract Color Image Features*. International Journal of Computer Science and Mobile Computing, 2017. **16**(12): p. 7-14.

[12] Marko H., Matti P. and Cordelia S., "Description of interest regions with local binary patterns," Pattern Recognition, vol. 42 no. 3, (2009), pp. 425-436.

[13] T. Ojala, M. Pietikainen and D. Harwood, "A comparative study of texture measures with classification based on feature distributions," Pattern Recognition, vol. 29 no. 1, (1996), pp. 51-59.

[14] K. J. Priya and R. S. Rajesh, "A Local Min-Max Binary Pattern Based Face Recognition Using Single Sample per Class," International Journal of Advanced Science and Technology, vol. 36, (2011), pp. 41-50.

[15] Zhang J., Marszalek M., Lazebnik S. and Schmid C., "Local features and kernels for classification of texture and object categories: A comprehensive study," International Journal of Computer Vision June, vol. 73 no. 2, (2007), pp. 213-238.

[16] Rami, H., M. Hamri, and L. Masmoudi, *Objects Tracking in Images Sequence Using Center-Symmetric Local Binary Pattern (CS-LBP)*. International Journal of Computer Applications Technology and Research, 2013. **2**(5): p. 504-508.

[17] Gupta, R., H. Patil, and A. Mittal. *Robust order-based methods for feature description*. in *IEEE conference on computer vision and pattern recognition (CVPR)*. 2010. USA.

[18] Hong, X., et al., *Combining LBP difference and feature correlation for texture description.* IEEE TRANSACTIONS ON IMAGE PROCESSING, 2014. **23**(6): p. 2557–2668.

