Image Finder Mobile Application Based on Neural Networks

Nabil M. Hewahi
Department of Computer Science, College of Information Technology, University of Bahrain,
Sakheer P.O. Box 32038, Kingdom of Bahrain
nhewahi@uob.edu.bh

Saira Rashid
Department of Computer Science, College of Information Technology, University of Bahrain,
Sakheer P.O. Box 32038, Kingdom of Bahrain
monrichi@msn.com

Abstract

Nowadays taking photos via mobile phone has become a very important part of everyone's life. Almost each and every person who has a smart phone also has thousands of photos in their mobile device. At times it becomes very difficult to find a particular photo from thousands of photos, and it takes time. This research was done to come up with an innovative solution that could solve this problem. The solution will allow the user to find the required photo by simply drawing a sketch on the objects in the required picture, for example a tree or car, etc. Two types of supervised Artificial Neural Networks are used for this purpose; one is trained to identify the handmade sketches and other is trained to identify the images. The proposed approach introduces a mechanism to relate the sketches with the images by matching them after training. The experimentation results for testing the trained neural networks reached 100% for the sketches, and 84% for the images of two objects as a case study.

Keywords: image recognition, neural networks, sketches.

1. Introduction

This research is a base for a mobile application which will allow users to find specific pictures from their mobile devices by drawing a sketch of the object. The aim of this research is to provide a possible solution and techniques to develop such an application.

Artificial Neural Networks (ANN) is one of the best solutions available for pattern recognition, handwriting recognition, image recognition, and machine learning (Egmont-Petersen et al., 2002; Ramírez-Quintana et al., 2012; Yang et al., 2012). In our research work we us ANN as a recognizing tool. A number of sketches and real images are collected and treated independently using neural networks. ANN can be used in classification and clustering, where classification can be performed using supervised learning, whereas clustering can be performed using unsupervised learning. A supervised learning based on ANN is used in our research. In ANN supervised learning, the system has to first be trained using several training examples, and only after we expect it to be able to recognize any problem instance related to the training examples and the system domain. The backpropagation algorithm is being used to train our neural networks (Rumelhart et al., 1986). Two ANNs are used, one for sketches training and another for real images training.

1.2 Background

Bergen (2011) stated that from the invention of cameras to the present day, around 3.5 trillion photos have been taken over the course of history. Today, people are taking four times more photos than they were 10 years ago. It is estimated that nowadays, every two minutes, people snap as many photos as the entire world took in the 1800s. So the number of photos is

dramatically increasing and day-by-day it is becoming difficult for people to find some specific photos among the huge number of photos stored in their devices.

When the user wants to find out a certain picture in his mobile, he/she has to search it image by image, which takes too much time. Another option is to tag each image, but this process is not suitable either because the user has to remember the tags, not to mention that giving tags to images also takes time. In this work we propose another solution to solve the problem by allowing the user to sketch an object for it to be found among the real images saved in the mobile.

The proposed solution is based on supervised Neural networks using the backpropgation algorithm. Egmont-Petersen (2002) raised a major question in their research and tried to answer it. The question was the following: "What are the major applications of neural networks in image processing now and in nearby future" (Egmont-Petersen et al., 2002).

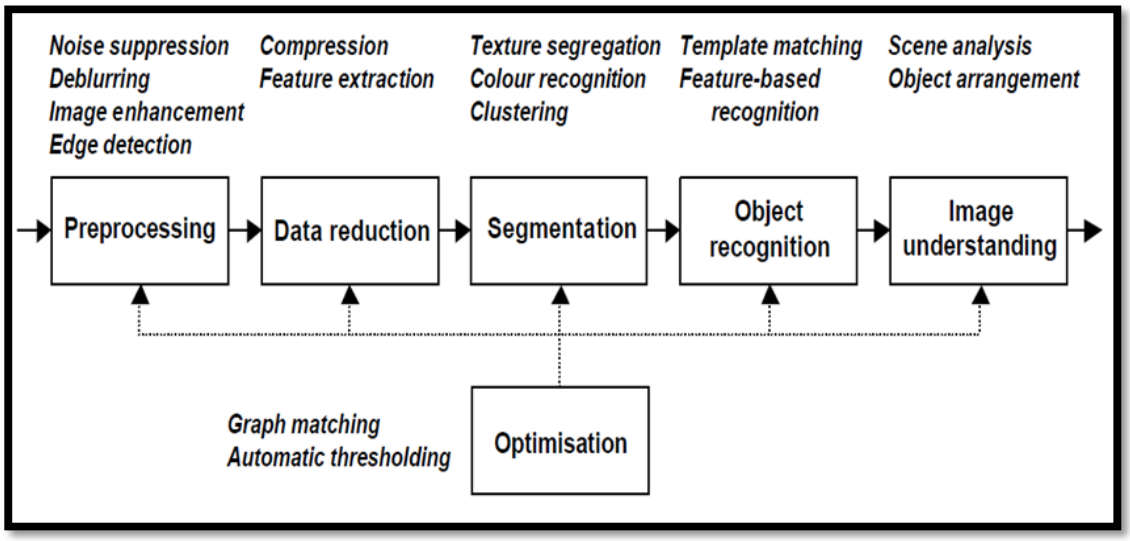

Figure 1. Methodology used in the research by (Egmont-Petersen et al., 2002)

Figure 1 shows the methodology used by Egmont-Petersen et al. (2002) to come up with an answer to their research question. Their study says that image recognition using neural networks goes through the stages shown in Figure 1. In our research we shall use this general proposed approach.

Yang et al. (2012) conducted a research on the character feature extraction method based on the integrated neural network. In their research, the hand written recognition system was developed using integrated neural networks. In order to do so, to train the network, there was conducted an experiment to demonstrate the use of the integrated back-propagation neural network algorithm, and good results were achieved. Basu et al. (2010) studied the use of artificial neural network in pattern recognition. In their study, they tried to summarize and compare few of the well-known methods that are used in various phases of a pattern recognition system using artificial neural network. They also tried to identify research topics and applications of artificial neural network. They summarized and compared the research done on interactive voice response (IVR), stock price pattern recognition, load forecasting for electric power system, face recognition, electrocardiogram (ECG) pattern recognition, rain attenuation model, optical character recognition (OCR), fault detection of induction motor and linear matched filtering, all based on artificial neural network. They found that after applying artificial neural networks to each of the above pattern recognition methods, they showed better results as compared to the those obtained without applying artificial neural networks.

Ramírez-Quintana et al. (2012) presented a very good survey on applying neural networks on image processing applications in various areas such as engineering, science, medicine, industry and security. This study covered 160 related papers and concluded that neural networks perform very well in medicine, especially with MRI segmentation and detection. However, the performance was better with applications related to biometrics. This study has led to the conclusion that neural networks will last for a long period for image processing applications.

Based on various applications of neural networks on image recognition and classifications, we select neural networks as a tool to help us in developing our application.

2. Methodology

In this section we present in details the methodology used for this research. Figure 2 depicts the used methodology. The adopted approach is based on first training the system to be able to recognize certain sketches by providing it with various hand-drawn examples such as trees, cars and mountains. Secondly, the system has to be trained again to recognize real images by providing it with real images such as trees, cars and mountains.

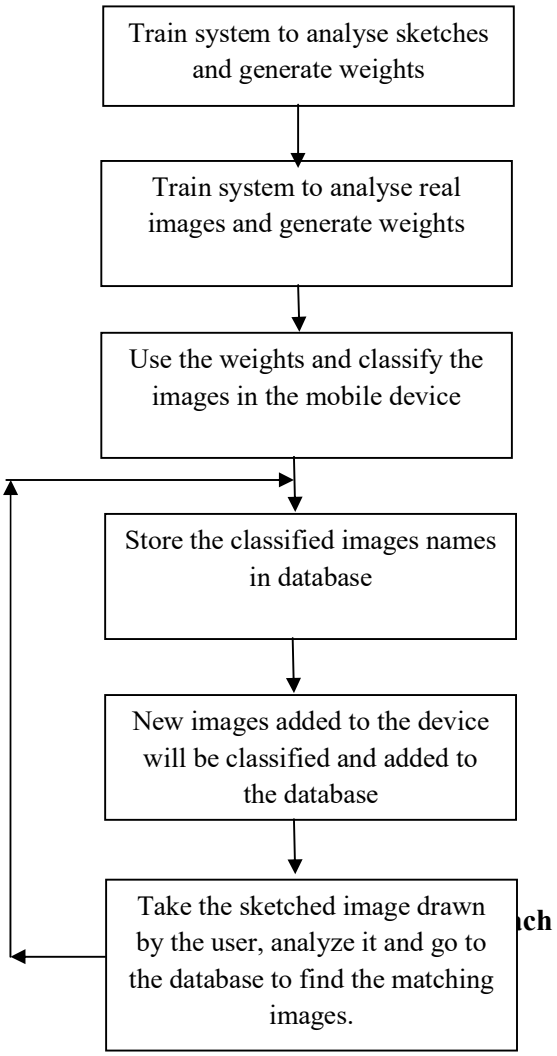

Figure 2. The used approach

The training process we use in both trainings is based on neural networks, using the backpropgation algorithm. By using this algorithm, we are going to finally obtain the weights that will be used in our model to make the system recognize the input given by the user.

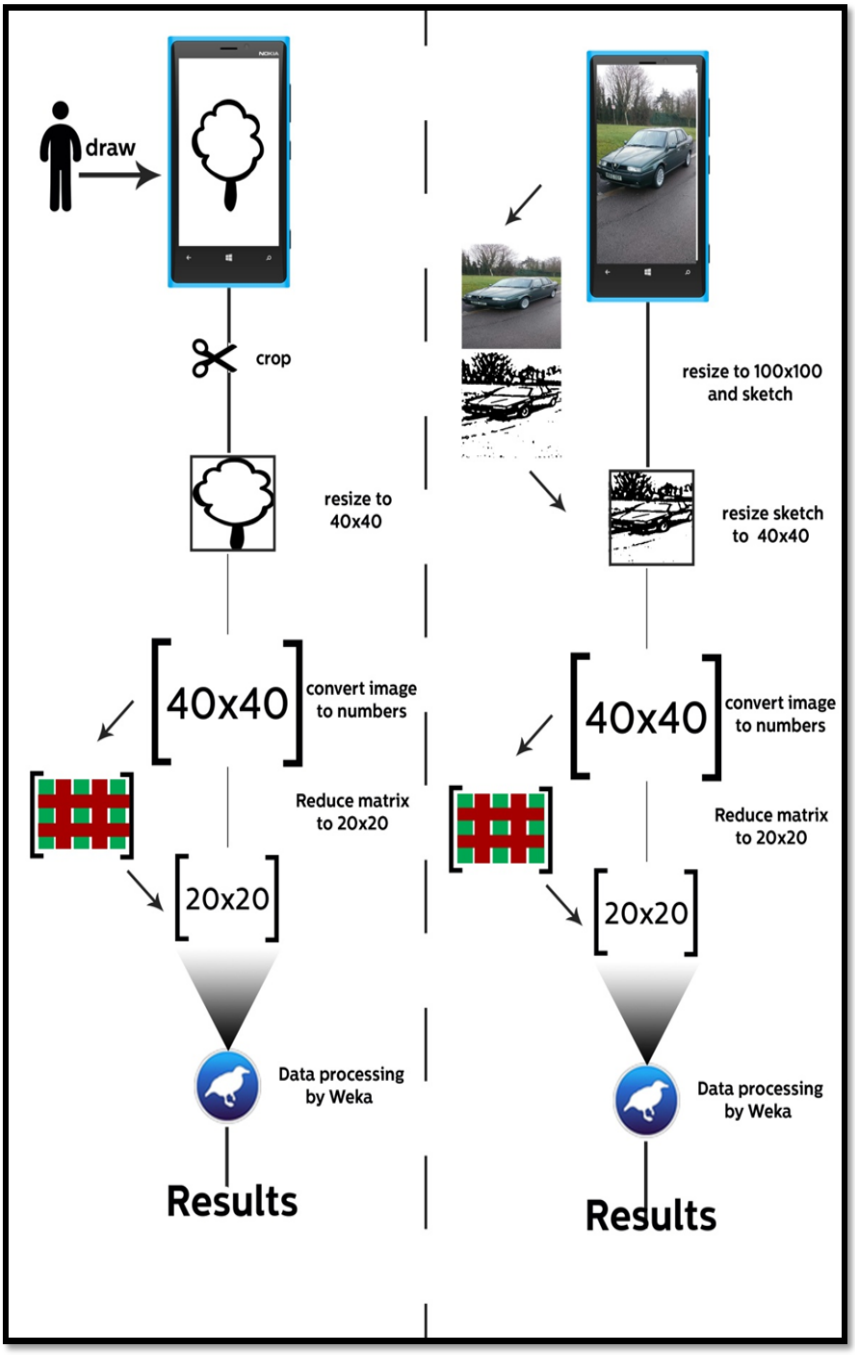

Figure 3. The used methodology

Once the training for the hand drawn sketches is over, we are going to get weights that will be used to recognize any inputted hand drawn sketches. For example, if the user draws a tree, the system will be able to recognize what has been drawn as a tree, using the obtained weights from the neural networks. On the other hand, once the training for the real images is

over, it means that if we provide our system with a real image, for example a tree, the system will be able to recognize it using the obtained weights from the training phase.

The general mechanism to be followed is:

1.Train the system to recognize hand drawn sketches using neural networks and get the final weights of the neural network.

2.Train the system to recognize real images using neural networks, and get the final weights of the neural network.

3.Using the weights obtained in step 2, recognize all the images in the mobile.

4.Store the recognized images in step 3 in a special database in the mobile phone containing information about the image and its class/category (tree or a car for example).

5.Any new image saved in the mobile phone can be recognized by our system and added to the database.

6.If the user draws a sketch, the system will recognize the sketch, and then it searches in the database to find and show all the corresponding drawings. It means, if the drawing is a tree, the application will show all the images with a tree saved in the mobile phone.

The detailed steps of the methodological approach are shown in Figure 2 and Figure 3. It is to be noticed that in order to apply our approach there are various steps that need to be followed, as explained in the next section.

3. The Proposed Approach

The full details of every step in the proposed approach will be explained in this section.

3.1. Data for Neural Network Training

We use two types of data, one is related to hand drawn sketches and the other is related to real images. This yields to use two neural networks as below:

- Data to train the neural network to recognize handmade sketches;
- Data to train the neural network to recognize objects in the pictures taken by the camera.

3.2. Handmade Sketches

A mobile application is developed to collect the data of the sketches of the required objects. Two types of objects are considered as a case study to be drawn onto the mobile application in various ways, to help the system get trained on various types of hand drawing related to these objects. The sketches are then cropped minutely and are converted to integer numbers based on the color intensity of each pixel. The sketches are cropped to improve the reliability of the data. Instead of taking the whole area of the drawing canvas, only the part where the user made the drawing is taken into account and saved. All these steps are described below.

3.2.1. Mobile Application for Sketches

Figure 5 shows the interface of the mobile application that is created to collect the data of the sketches. It contains a drawing canvas; where drawings are made and it also contains few text boxes which are filled with the information of the drawing. For example, if the drawing is a tree, '1' is written in the tree text box. Furthermore, the undo and redo buttons help modify the drawing and the clear button clears everything drawn onto the drawing canvas.

3.2.2. Cropping the Sketch

The drawing of a tree is shown on the drawing canvas in Figure 4. It is clear from the figure that there is some empty area on top, right, left and bottom of the drawing sketch, which can cause problems while using this image for training the neural network. Therefore, there is

implemented a crop function which crops the image very minutely and removes the empty space as shown in Figure 5.

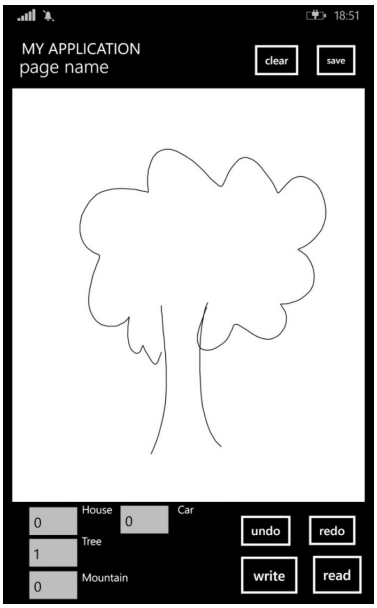

Figure 4. Mobile Application to collect Sketch data

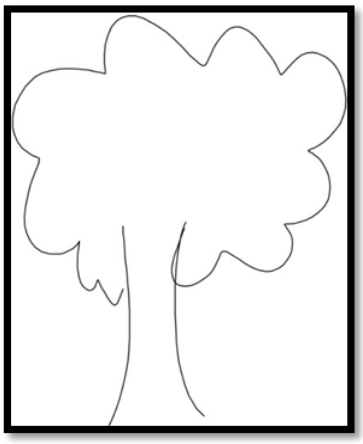

Figure 5. Cropped Sketch

3.2.3. Resizing the Cropped Sketch

After the Sketch is cropped it is converted to a fixed size of 40x40 pixels. This is done to keep all the sketches of same size to get the same input from every sketch for the neural network. This means we are going to have 1600 inputs for the neural networks, which is too much and will lead to a long time of training.

3.2.4 Image to Numbers

Finally, the cropped and resized image is taken and it is passed to a function that converts the image to an integer matrix of size 40x40, so a total of 1600 integer values is taken. The integer values are between 0-255, which is the color intensity of each pixel of the image.

3.2.5 Reducing the Size

As previously described, there are a total of 1600 integers, which can be the input to the dataset for neural networks. But it is a huge number, so in order to minimize the size of inputs to

the neural network the 40x40 matrix is reduced to 20x20 by skipping odd rows and columns of the original matrix. Figure 6 shows a matrix containing green and red rows and columns. If this was the 40x40 matrix, then the red part of this matrix would be skipped to convert it into a 20x20 sized matrix. Now there are only 20x20=400 values, which is a reasonable input size for the neural network.

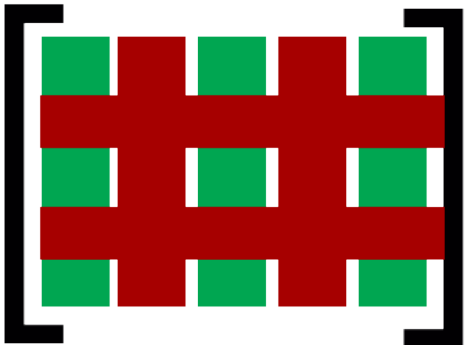

Figure 6. Reducing size of Sketch

3.3 Real Pictures

The other part of the developed approach is to collect the data about the same two objects of the real pictures taken by the camera. These images are converted into black and white pictures and then treated the same way as the sketches, i.e. black and white pictures are also converted into integers based on the color of each pixel.

3.4 Mobile Application for Images

Another very simple mobile application is developed, one which goes through all the pictures in the mobile device and selects the pictures with the names (tree0, tree1, …) and (car0, car1, …)only, one by one, doing the following:

- Resizes the picture into 100x100 pixels;
- Applies a black and white sketch filter on the picture;
- Converts the picture to integer numbers according to the color intensity of each pixel;
- As the object is always in the middle of the picture, out of 100x100 only the 40x40 pixels in the middle are taken into account;
- The 40x40 pixels size array is then reduced to 20x20 by skipping odd rows and columns. Now there are only 20x20=200 values, which is a reasonable input size for the neural network.

By the end of this stage we have two datasets, one for the handmade sketches and the second one for the pictures. So at this point we are ready to start training our neural network, a process that is explained in the next chapter.

4. Experimentation and Results

In this section, we introduce some of the implementation issues and we describe the experimentation performed to train both neural networks.

4.1. Implementation

The two mobile applications described in the previous section are developed using c# language and are developed for windows phone 8.1 Silverlight. Some of the main functions of the application are described below.

4.1.1. Crop Function

This function takes five parameters, a WriteableBitmap type of object, the location of the starting point on x coordinate, the location of the starting point on y coordinate, width and height. It crops the image according to these parameters as shown in Figure 7.

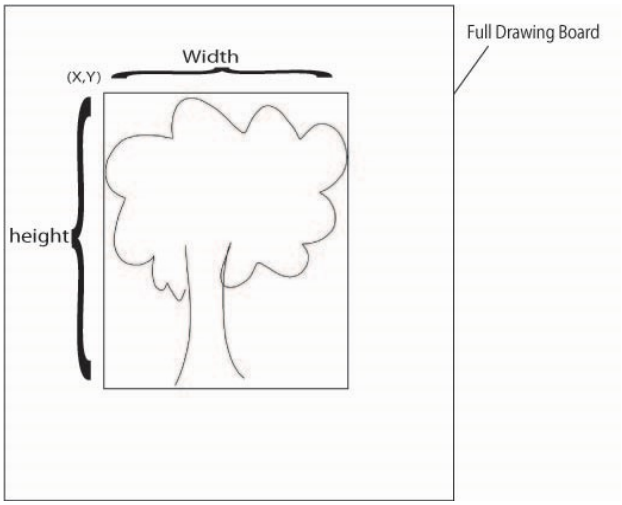

Figure 7. Cropping image

4.1.2. Image to Integer Function

This Function takes only one parameter that is a WriteableBitmap type of object and it returns a long array which contains the value of the color intensity of each pixel of the image. This is how it converts from image to integer.

4.1.3. Reducing image size from 40x40 to 20x20

The above mentioned function is to reduce the image size from 40x40 to 20x20 by skipping one row and one column and keep each one of them.

4.2. Experimentation

All the experimentation is done using the open source Weka. "Weka is a collection of machine learning algorithms for solving real-world data mining problems. It is written in Java and runs on almost any platform. The algorithms can either be applied directly to a dataset or called from your own Java code"(Weka-machine-learning-software-in-java.soft112.com).

For each of the both neural networks for sketches and images, Weka created 400 nodes for the input layer, 201 for the hidden layer and two nodes for the output layer.

4.2.1. Sketch Recognition

It is described in the previous section that a mobile application is used to collect data about the sketches and the data is converted to 20x20=400 integer numbers to give it as input to Weka.

Our experimentation includes only two objects for recognition i.e. trees and cars.

Total tree sketches used = 175
Total car sketches used = 72
Learning rate = 0.3
Momentum = 0.2
Number of epochs = 500

70% of data is used to train the neural network and the remaining 30% is used for testing the trained neural network. Figure 8 shows the neural network for sketches. The results are shown in Figure 9 and are as follows:

Total Correct Recognition = 100%

Total Incorrect Recognition = 0%
Error Per epoch = 0.0000007

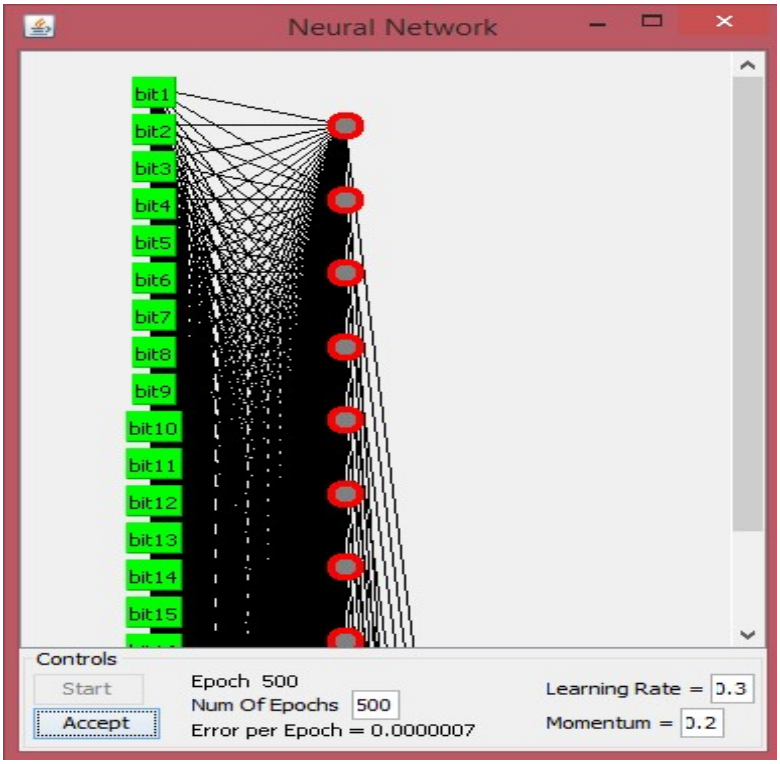

Figure 8. Training Neural Network for Sketches

```
Time taken to build model: 2.06 seconds

=== Evaluation on test split ===
=== Summary ===

Correctly Classified Instances          74               100      %
Incorrectly Classified Instances         0                 0      %
Kappa statistic                          1
Mean absolute error                      0.0009
Root mean squared error                  0.0018
Relative absolute error                  0.2074 %
Root relative squared error              0.4039 %
Total Number of Instances               74

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 1         0          1         1         1          1        tree
                 1         0          1         1         1          1        car
Weighted Avg.    1         0          1         1         1          1

=== Confusion Matrix ===

  a  b   <-- classified as
 52  0 |  a = tree
  0 22 |  b = car
```

Figure 9. Weka Results for Sketches

4.3.2. Image Recognition

The dataset of the real images is also treated the same way as the dataset of the sketches.

Total tree images used = 86
Total car images used = 54
Learning rate = 0.3
Momentum = 0.2
Number of epochs = 500

70% of data is used to train the neural network and the remaining 30% is used for testing the trained neural network. Figure 10 shows the neural network for images.

The results are shown in Figure 11 and are as follows:

Total Correct Recognition = 83.7838%
Total Incorrect Recognition = 16.2162%
Error Per epoch = 0.0235851

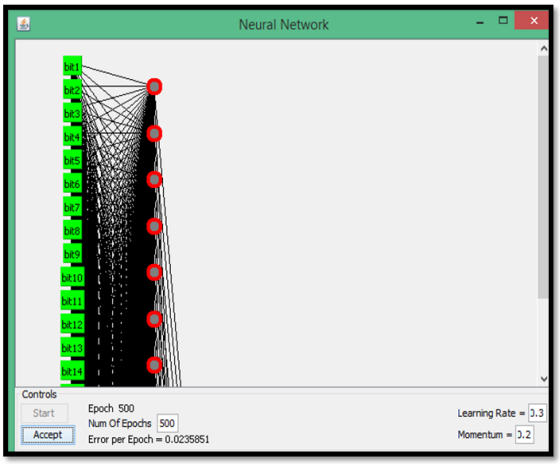

Figure 10. Training Neural Network for Images

```
Time taken to build model: 2.84 seconds

=== Evaluation on test split ===
=== Summary ===

Correctly Classified Instances          31               83.7838 %
Incorrectly Classified Instances         6               16.2162 %
Kappa statistic                          0.6764
Mean absolute error                      0.1597
Root mean squared error                  0.3592
Relative absolute error                 32.2798 %
Root relative squared error             72.0105 %
Total Number of Instances               37

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.8       0.118      0.889     0.8       0.842      0.862    tree
                 0.882     0.2        0.789     0.882     0.833      0.862    car
Weighted Avg.    0.838     0.155      0.843     0.838     0.838      0.862

=== Confusion Matrix ===

  a  b   <-- classified as
 16  4 |  a = tree
  2 15 |  b = car
```

Figure 11.Weka Results for Images

4.4. Discussion

Based on the obtained results, it is clear that the proposed approach is very effective in recognizing the sketches. About 84% for recognizing the images is also considered to be a very good percentage, which can be improved by providing more examples or by changing some of the parameter values, such as the learning rate and the momentum. If the system works with these two percentage values, the expected accuracy cannot be less than 84%, which is very good, and in case the percentage of image recognition increases, the overall accuracy of the system will also increase. It is expected that the more objects included, the lower the percentage of accuracy, especially if the objects look like each other, but the expected accuracy would be still acceptable.

The obtained results give a very good indication that the mobile application will work very well based on our approach. Before fully implementing the mobile application, further experiments need to be performed.

5. Conclusion

In this research we proposed a way to find images saved in a mobile phone by drawing sketches. The main approach is based on two supervised neural networks, one for sketches and the other for images. The experiments were performed using Weka. The accuracy of sketches recognition reached 100%, whereas the accuracy of real images recognition reached about 84% with two objects, a car and tree. The obtained results give a general indication that the mobile application would give good results, however, more there need to be done more experiments with more objects in order to scale the accuracy of the approach before implementing the mobile application. Some of the future directions are:

1. Conducting more experiments using various objects;
2. Improving the accuracy of image recognition;
3. Implementing the mobile application;
4. Upgrading the approach for it to be able to recognize more than one object in the sketch and image.

References

Basu, J. K., Bhattacharyya, D., & Kim, T. (2010). Use of Artificial Neural Network in Pattern Recognition. *International Journal of Software Engineering and Its Applications ,4*(2), 23-34. Retrieved April 2, 2010

Bergen, J. (2011, September 20). Facebook stores 10,000x more photos than the Library of Congress. Retrieved February 10, 2016, from http://www.geek.com/geek-cetera/facebook-stores-10000x-more-photos-than-the-library-of-congress-1422873/.

Egmont-Petersen, M., Ridder, D.D. and Handels, H. (2002). Image Processing with Neural Networks. *Pattern recognition*, *35*(10), pp. 2279-2301. doi:10.1016/s0031-3203(01)00178-9

Machine Learning Software in Java. (2011, March 24). Retrieved May 26, 2015, from http://weka-machine-learning-software-in-java.soft112.com/

Mustafidah, H., Hartati, S., Wardoyo, R., & Harjoko, A. (2014). Selection of Most Appropriate Backpropagation Training Algorithm in Data Pattern Recognition. *International Journal of Computer Trends and Technology,14*(2), 92-95. doi:10.14445/22312803/ijctt-v14p120

Ramírez-Quintana, J., A, Chacon-Murguia, M., I, & Chacon-Hinojos, J., F. (2012). Artificial Neural Image Processing Applications: A Survey . *Artificial Neural Image Processing Applications: A Survey ,20*(1).

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature,323*(6088), 533-536. doi:10.1038/323533a0

Yang, J., Yan, X., & Yao, B. (2012). Character Feature Extraction Method based on Integrated Neural Network. *AASRI Procedia*, 3, 197-202. doi:10.1016/j.aasri.2012.11.033