Object Recognition and Classification

Taylor Christine Johnson

MAY 2012

A Thesis Submitted to The Honors College

In Partial Fulfillment of the Bachelors of Science Degree

with Honors in

Mathematics

THE UNIVERSITY OF ARIZONA

Approved by

Dr. Marek Rychlick

Professor of Mathematics

Department of Mathematics

STATEMENT BY AUTHOR

This thesis has been submitted in partial fulfilment of requirements for a degree at the University of Arizona and is deposited in the University Library to be made available to borrowers under rules of the library.

Signed: TAYLOR CHRISTINE JOHNSON

Abstract

Object recognition and classification is a common problem facing computers. There are many shortcomings in proper identification of an object when it comes to computer algorithms. A very common process used to deal with classification problems is neural networks. Neural networks are modelled after the human brain and the neuron firings that occur when an individual looks at an image and identifies the objects in it. In this work we propose a probabilistic neural network that takes into account the regional properties of an image of either an ant or an egg as determined by edge segmentation and an extraction of geometric features specific to the object. To do this the algorithm calculates the regional properties of a black and white representation of the object and then gives these properties to the probabilistic neural network which calculates the probability of the object being an ant or an egg.

Contents

1 Statement of Purpose and Relevance

The purpose of creating an algorithm that could take an image and distinguish an object in the image based on its regional properties is to further the progress of computer object recognition. There are many difficulties facing computers in their effort to identify objects. In the past an object was represented by a single node. This simple representation allows for much overlap in terms of the computer recognizing the object. For example, consider a table that is represented by a node as a surface with four legs. Given a set of images, one image of a table and the other images of chairs it is quite likely that all of the images would be recognized as a table since all of the images have a surface with four legs.

In order to combat this limited representation of complex images we have employed the use of a probabilistic neural network. A neural network uses a single node to represent a simple property of the image and many nodes to represent an object [6]. For example a table might be represented by six nodes, one for each leg, one for the table top and one for the area of the table top. A chair would have the same number of nodes and the node representing the area of the chair top and the table top would help to distinguish the difference between a table and a chair for a computer. The neural network makes use of factorial or distributed representations of objects which allows larger problems to be tackled.

Neural networks have allowed researchers to focus on creating algorithms to better the processes of artificial intelligence for computers. These networks are based on the functions of the human brain in that it uses 10^{11} simple processors, or neurons, that all work concurrently to compute the probability that the object is something. This model, created to follow the function of the human brain, is nowhere near complete nor is understanding the probabilistic neural network equivalent to understanding natural intelligence; it is merely a model that has made object recognition more accurate. This improvement has not made the computational abilities of neural networks classifications perfect or in any way equal to human classification.

Overall, this paper is relevant as neural networks are used in a variety of important computational tasks such as automatic control, pattern completion, pattern classification, and function optimization, approximation, and prediction. It is also important for security purposes for computer operated cameras or other devices to accurately identify people as opposed to objects for the safety and security of those the cameras are trying to protect. Furthermore computers are far behind the recognition capabilities of humans which limits their usefulness to society. Humans are able to properly recognize objects in an image with just a quick glance while computers struggle to recognize an object if there is any inconsistency (such as noise or pixel variation) in an image [9]. In order to make proper use of probability neural networks it is important to be able to produce results where it properly recognizes objects in an image and classifies them correctly with an insignificant error in order to be useful.

Even with the improvements that neural networks have made to object recognition, poor quality images are not helpful in creating a better recognition system as they only complicate the number of issues the network has to deal with. Poor quality images can cause measurements of regional properties to be wrong, which in turn could cause overlapping classes creating ambiguities that are currently impossible for neural networks to solve. Due to these issues the purpose is to make the best possible classifications of the objects in images based on the information that is available in the form of regional properties. Since probability neural networks do not yet have the capacity to be error free, trial and error are needed in order to make strides towards improving these functions that have the potential to be very powerful tools in the future.

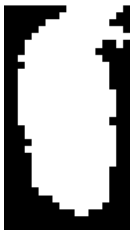

Figure 1: Black and white egg.

2 Methodology

Through the use of a probabilistic neural network a program was created that would take images of a single ant or a single egg and convert these images into a form in which the computer could train itself to recognize an ant from an egg based on various regional properties.

In this paper, and specifically in my methods, I use MATLAB and its various image processing capabilities. I assume in the following text that the reader is familiar with the terms of this program. One can refer to the MATLAB reference [5] to glean more information on the functions being used and referred to in this paper.

2.1 Identifying Objects

Given an image containing a single ant or a single egg the image is first converted from its original color form to black and white using **im2bw**. Once the image is in black and white, using a series of morphing operations, the various objects in the image, such as the ant or the egg, become a grouping of white pixels. The noise behind the objects in the image becomes a collection of black pixels as shown in Figure 1. After the image is in this form the regional properties of the image are calculated. 8-connectivity (mod8) is used to determine the area of the objects in the binary black and white image. This means that given a pixel x, each of the pixels that share an edge or a vertex with x are used in the calculation of area. The object in the image with the largest area (which ought to be the ant or the egg)

Figure 2: Original Image. [1]

is then analyzed further and the object's major axis length, minor axis length, eccentricity, and area are stored in a vector.

2.2 Edge Detection

In the problem of edge detection, the goal is to determine areas in which the intensity of the image is rapidly changing. In other words, we are looking for places in the image where the contrast between the object and the background is the largest. The zero-cross method, a specific gradient-based method, searches for regions in the image in which the second derivative of a given image, let's call it f, crosses the origin, in other words, has a zero-cross. Given an image Figure 2, the computer must find the object it wants to assess and then based on the probabilistic neural network it can determine if the object is an ant, an egg, or neither. Edge detection is used to segment a given image f, into objects. The image is converted into a black and white image first, creating the image field $\nabla f(x, y)$. The lower threshold $T_L(x, y)$ is set to be zero [7]. Thus, when the magnitude of the image field is greater then the threshold, $|\nabla f(x, y)| > T_L(x, y)$ an edge exists. The threshold can be calculated

using the following equation.

$$\nabla^2 f(x,y) = \nabla\nabla(f(x,y)) = \frac{\partial^2 f(x,y)}{\partial x^2} + \frac{\partial^2 f(x,y)}{dy^2} \tag{1}$$

For a two dimensional sequence $f(n_1, n_2)$, the partial derivatives can be represented by the convolution of the impulse response of a filter (denoted by $h(n_1, n_2)$) and $f(n_1, n_2)$. To show $f(n_1, n_2) * h(n_1, n_2)$ is a discrete approximation of $\nabla^2 f(x,y)$, approximate $\frac{\partial f(x,y)}{\partial x}$ by:

$$\frac{\partial f(x,y)}{\partial x} \rightarrow f_x(n_1, n_2) = f(n_1 + 1, n_2) - f(n_1, n_2). \tag{2}$$

The scaling factor does not affect the outcome because the zero-cross method is being implemented and the scaling factor can be omitted.

$$\frac{\partial^2 f(x)}{\partial x^2} \rightarrow f_x x(n_1, n_2) = f_x(n_1, n_2) - f_x(n_1 - 1, n_2) \tag{3}$$

From equations 2 and 3

$$\frac{\partial^2 f(x,y)}{\partial x^2} \rightarrow f_x x(n_1, n_2) - 2f(n_1 + 1, n_2) - 2f(n_1, n_2) + f(n_1 - 1, n_2) \tag{4}$$

By approximating $\frac{\partial^2 f(x,y)}{dy^2}$ and equations 1 and 4

$$\begin{aligned} \nabla^2 f(x,y) \rightarrow \nabla^2 f(n_1, n_2) &= f_x x(n_1, n_2) + f_y y(n_1, n_2) \\ &= f(n_1 + 1, n_2) + f(n_1 - 1, n_2) + f(n_1, n_2 + 1) + f(n_1, n_2 - 1) - 4f(n_1, n_2) \end{aligned} \tag{5}$$

The zero-cross points of $\nabla^2 f(x,y)$ are shown in the image of f, represented by $f(n_1, n_2)$. Since the threshold level is zero, the function **edge_segmentation** makes use of contour coding, following the boundaries of the various gray levels on the image field $\nabla f(x,y)$ [4]. In the function **edge_segmentation**, written expressly for ants and eggs, a green line is

Figure 3: Example of edge_segmentation.

superimposed above the grayscale image, denoting edges.

2.3 Probabilistic Neural Network

The concept of a neural network is based on the neurons in the brain. Its purpose is to imitate the base level of neurons that fire in the human brain. These networks were designed using the same functions determined for neuron brain activity and were implemented in the same order, in hopes that this would mimic the process of the human brain and create artificial intelligence. Although neural networks are in no way an acceptable replacement of the human brain they are able to provide the functions of first-order characteristics of neurons found in the brain. These functions include the ability to receive, understand, process, and transmit signals over the neural pathways. Speaking in terms of the computer program, the neural network is able to take a set of inputs and represent all of the outputs of the corresponding neurons.

Looking more specifically, a probabilistic neural network is a supervised learning environment. This means that the network is provided with a set of inputs and a corresponding

set of desired outputs in the form of a target vector. The probabilistic neural network takes in the input matrix, calculates the output, compares the output with the target vector, and calculates the difference between the outputs and the target vector, or the error. This error rate is fed back through the system and the network changes the weights applied to the input matrix in order to minimize that error [3]. This repeats until the probabilistic neural network achieves results with an acceptably low level of error, or error that is statistically insignificant.

Given a matrix P, with each column representing an input such as area or eccentricity, each input is applied to an artificial neuron, or the distribution layer. In this way P corresponds to the signals sent to the synapses of the neurons, where each signal represents an input. These signals are then multiplied by an assigned weight, $w_1, w_2, ...w_n$, where the weight signifies the strength of the signal and its synaptic connection. The weights each have a value that is a component of the target vector. Let us call the matrix of weights W. The network then groups the pattern layer neurons by their classification as given by the target vector. Now all of the weighted inputs are summed and produce an output we will refer to as NET [8].

$$NET = PW \tag{6}$$

Then the non-linear function $f\odot$ is applied to NET and produces Z_{ci} where c denotes the class of the target vector and i is the pattern layer neuron that is calculating the class. The pattern layer neuron computes the dot product of the target vector and the input vector.

$$T_{Ri} = (x_{R1}, x_{R2}, \ldots, x_{Rn}), \tag{7}$$

$$T_{Ri}^t * X_i = x_{R1} * x_1 + x_{R2} * x_2 + \cdots + x_{Rn} * x_n \tag{8}$$

Using this dot product Z_{ci} can be calculated by subtracting 1 and dividing by the variance

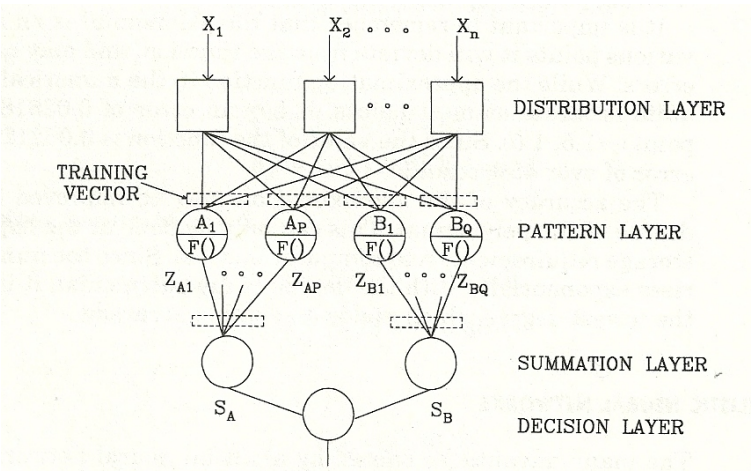

Figure 4: Probabilistic Neural Network.

σ^2 and applying the exponential function.

$$Z_{ci} = \exp\left[\frac{(X_{Ri}^t X_i - 1)}{\sigma^2}\right] \tag{9}$$

Following this, the summation layer will sum all pattern layer outputs associated with the same class. In this way each class will be summed individually in the summation layer of the probabilistic neural network, denoted by S_c.

$$S_c = \sum_{i=1}^{??????} \exp\left[\frac{(X^t X_{Ri} - 1)}{\sigma^2}\right] \tag{10}$$

The final step is the decision layer where a comparison is formed and the class of the input is represented by a '0' or a '1' respectively. This is a description of a two class probabilistic neural network as implemented in this paper. Using this same method the algorithm can be extended to a problem involving any number of classes [9]. A probabilistic neural network is a specific way to solve classification problems such as the one we present; classifying an object as an ant or an egg. It is a direct test classifier that requires no training, which is utilized in our case to determine if the image is of an ant or an egg using only geometric properties.

A probabilistic neural network takes an input matrix containing regional properties of an object in the image and a target vector, and uses them in the neural network to determine the margin of error between the input matrix and the target vector. The input matrix contains the regional properties and the target vector assigns the image a 1 or a 0. The number '1' signifies that the object being represented in the input vector is an ant, while a '0' classifies the object as an egg. The probabilistic neural network then minimizes the margin of error based on the given input matrix and target vector in order to be able to classify future input matrices. This is how the program learns to be more accurate in classifying future images that are fed to the program.

Once the neural network has minimized the error the network is given an input matrix containing the regional properties of random objects. The first thing the probabilistic neural network does is compute the distance between the input matrix and the target vector, where the target vector contains all possible outcomes, which, in this case, are a 1 and a 0, or in words and ant and an egg. Once this distance is computed a vector is produced, illustrating how close the input is to the training input. The network then sums the distances for each individual input in order to create a net output vector of each of the regional properties probabilities respectively. Then, the network applies a compete transfer function to the net output vector and assigns a 1 to the class with the maximum value of the various probabilities and a 0 to the remaining classes [5]. The class assigned the 1 has the highest probability of being correct and thus classifies the image as an ant or an egg based on the black and white images regional properties.

2.4 Functions

In order to create a program that could accept an image it was necessary to write a series of functions in MATLAB. The following are the names of the functions written and their purpose.

filename_array When given a folder of images, **filename_array** creates a list of the file names in the given folder.

feature_extractor When given a file name and a protocol **feature_extractor** creates a row vector containing the eccentricity, major axis length, minor axis length and area of the largest object found in the image.

input_and_target Creates an input matrix, containing all row vectors determined for the folder of images initially presented to **filename_array** and creates a target column vector containing the protocol. The row index of the input matrix corresponds to the protocol with the same row index.

pt_return Calls **input_and_target** in order to run through the entire folder of images.

pnn This is a Probabilistic Neural Network to determine if the image is an ant or an egg. When given the input matrix and target vector from **pt_return**, pnn uses this data to train the network to distinguish between an ant or an egg based on the images region properties.

Using these functions in the proper order allows the user to give MATLAB entire folders of images; in this case the images must be of ants and eggs as the probabilistic neural network is set up to only identify ants and eggs. When this program is given a folder of images it will take that folder, convert all of the file names into an array, and, using edge-detection and the abilities of **feature_extractor**, will output a vector of regional properties for each image. These functions were written with the express purpose of finding the largest object in the image. In this way it is required that each image presented to the functions in MATLAB it contains only one identifiable object, as determined by the user. So **feature_extractor** finds the largest object in the image and returns a vector containing that one object's eccentricity, major axis length, minor axis length, and area.

Then **input_and_target** combines these vectors to create a matrix of each of the image's regional properties. Following this **pt_return** will combine the input matrices of all of the folders given to MATLAB in order to create one large matrix containing all of the regional properties of all of the images from all of the folders that were given to MATLAB for the purpose of determining if the images contain ants or eggs. Finally this matrix will be given to the probabilistic neural network and the results will be output in the form of a vector, assigning a '1' to each image that the **pnn** believes signifies an ant and a '0' to each image the **pnn** classifies as an egg. Along with this vector a graph is created, showing the distribution of ants and eggs throughout space as seen in Figure 5.

2.5 Testing

The preliminary test of the **pnn** used two folders; one containing images of only eggs, the other containing images of only ants, so the results of the network could be compared to any persons visual classification of the objects. First, each folder was input to **filename_array**, one at a time, with the protocol of '0' or '1' already attached to each file in the folder. These protocols were determined based on the programmers visual interpretation of the image. This array was then sent to **input_and_target**, which called **feature_extractor** and created two matrices. The first matrix, P, or input matrix, contained the eccentricity, major axis length, minor axis length, and area of the largest object discovered in the image being assessed. The second matrix, T, or target matrix, contained the protocol of the image. Each row of P corresponded with the equivalent row of T. Now both folders had an input and target matrix. **pt_return** combines the two input matrices and target matrices respectively, maintaining the correspondence between rows. The transpose of the final input matrix and transpose of the final target matrix is then fed to the probabilistic neural network to train it.

After this, tests of a random sampling of images was performed in much the same way

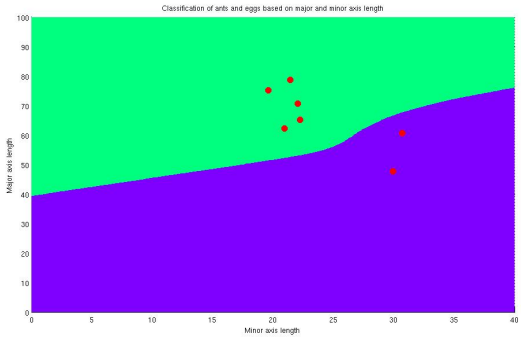

Figure 5: Graph of Results.

as the preliminary test with the exception that the protocol was not assigned by the user but rather determined based on regional properties of the image and the accuracy of the probabilistic network. All of the random images had ants and eggs on a white background. All of these images were obtained from the research of Anna Dornaus.

3 Results

Realizing the limitations of the scope of **feature_extractor** and **edge_segmentation** due to noise, select images of ants and eggs were used. The ant and egg images tested were not random in the sense that they all were images with a solid white background. Although images of eggs on a dirt-like background were available the noise was too much for the algorithm to handle in its current state. Also, we did not test any ants or eggs that were overlapping other ants or eggs. A sample of five ants and two eggs were tested; Figure 5 illustrates the results. Figure 5 represents the distribution of the results of the probabilistic neural network. The top portion of the graph is indicative of objects being classified as ants, the bottom portion of the graph, the eggs. The distribution throughout the space represents the probability of the object being correctly identified in terms of the maximum likelihood estimator (MLE). The dividing line, or ridge, is illustrating the points in space when the

Figure 6: Egg.

likelihood of the image being an ant or an egg has the exact same probability.

This test proved successful in that all of the images were correctly recognized and classified. In order to further test this method a more robust solution to dealing with noise must be added. Running more tests on a larger selection of images would also increase the credibility of this model. Finally, being able to test images containing more than one object and images with eggs and ants overlapping would greatly increase the applicability and scope of this method.

4 Discussion

It is necessary to know if the image is an ant or an egg prior to running it through **edge_segmentation** in order to successfully convert it to a black and white image where the area of the ant or egg is properly represented. This is due to the nature of the eggs. Ant eggs appear transparent as seen in Figure 6. Due to this transparency it is very difficult to clear up noise in the image without completely getting rid of the egg, since it takes the appearance of the surface that it is on, including any noise in the image. Due to the transparency of the egg, images of ants and eggs without a solid background are difficult to detect due to the increased noise in the image. On the other hand, images of ants and eggs with a solid background such as the one shown in figure 7, allow the **edge_segmentation** function to work properly and succeed at identifying ants and eggs as objects. A possible solution to this

Figure 7: Original Image. [1]

problem, though not tested, is to create a universal method of edge detection. This method would first use the **edge_segmentation** function to attempt to determine objects and their regional properties and then also use **feature_extractor** to catch any missed objects that **edge_segmentation** did not pick up on due to noise.

5 Literature Review

In terms of statistical analysis of an image there are many various ways to approach object recognition. The probabilistic neural network is one way in which statistics and probability are used in connection with regional properties to determine what the image is portraying. Even though this process attempts to model the functions of the neurons in the human brain, it is very basic, making it a step toward achieving artificial intelligence, though not a complete solution. Some concepts proposing possible fixes to these errors are discussed.

5.1 Noise

Object recognition using the methods of **feature_extractor** and **edge_segmentation** are not error free. One of the major factors that influence the mis-characterization of an ant or an egg is noise. In the method above there is no procedure to minimize noise in any way. As you can see in Figure 1 noise in the background of the image of an egg makes the black and white image of the egg in Figure 1 appear larger than it actually is. Using statistical edge detection methods there is a way to decrease the amount of noise in an image, if only by a minimal amount.

Let us assume we have an image $f(n_1, n_2)$. Then we can say that the observed image, $y(n_1, n_2)$ is composed of both the actual image f and a noise component, lets call it $N(n_1, n_2)$. So $y(n_1, n_2) = f(n_1, n_2) + N(n_1, n_2)$. After applying the edge-detection function to the image MATLAB will return the enhanced image, let us call it g, where $g(n_1, n_2) = y(n_1, n_2) \odot (h(n_1, n_2))$ where h represents the impulse response of edge detection. Due to noise, g contains an error component which is $h(n_1, n_2) \odot (N(n_1, n_2))$. If linear processing was used it would be ideal to perform Wiener estimation techniques on the original image to reduce noise. However, since edge detection in the context of this paper is using thresholding, the only way to reduce noise is to apply a mean-square estimation of the original image f before running the image through the edge detection function. Although this method decreases the accuracy of the image y it has a high probability of producing better, more accurate results [7].

5.2 Threshold

The method that was used in our model for edge detection involved arbitrarily setting the threshold at zero. The threshold is computationally cheap but has a large effect on the boundaries that the edge detection function will choose for objects being extracted. The use

of an automated threshold selector could allow for more accurate data to be drawn from the objects extracted from the image, diminishing the error of our object recognition algorithm.

There are two different ways to approach extracting an object from an image. One method is thresholding, which is used in **edge_segmentation**, and the other method uses gradient operators that look at the contour of the image and determine an object based on the complete boundaries created, similar to the method used in **feature_extractor**. A method has been proposed to combine these two different concepts into one ultimate solution, called the Morphological Gradient Threshold (MGT) segmentation.

Consider an image X. The first step of MGT will perform the basic morphological operations of erosion, $\varepsilon_B(X)$ and dilation, $\delta_B(X)$ to determine the morphological gradient of the image. A structuring element, referred to as $\rho_B(X)$ can be used to define this.

$$\rho_B(X) = \frac{1}{2}\left(\delta_B(X) - \varepsilon_B(X)\right). \tag{11}$$

Next, the morphological Laplacian, or second derivative, is calculated using 11, thus cutting down on computational time

$$\Lambda_B(X) = \frac{1}{2}\left(\delta_B(X) - \varepsilon_B(X)\right) - X. \tag{12}$$

where $\Lambda_B(X)$ is the morphological second derivative of the image X by a structuring element $\rho_B(X)$. It is then necessary to create a pixel map $m(x, y)$ such that for every pixel (x, y) in X it can be classified as an object, a border, or part of the background depending on its gray-level.

$$m(x,y) = \begin{cases} 128 & \text{if } \rho_B(x,y) < MGT, \\ 255 & \text{if } \rho_B(x,y) \geq MGT \text{ and } \Lambda_B(x,y) \geq 0, \\ 0 & \text{if } \rho_B(x,y) \geq MGT \text{ and } \Lambda_B(x,y) \leq 0. \end{cases} \tag{13}$$

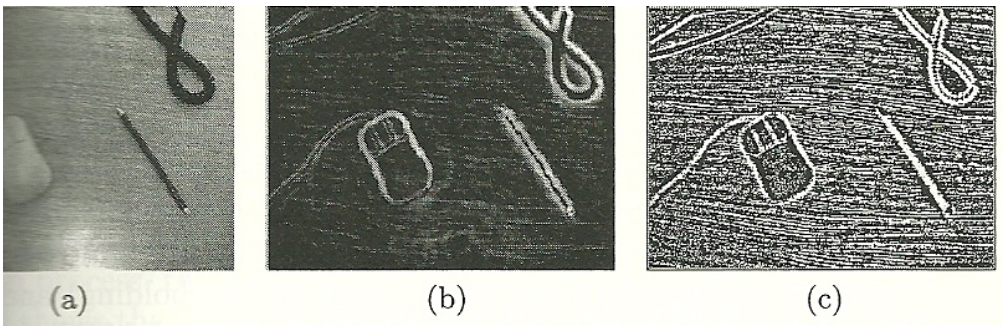

Figure 8: MGT of a Real Image. (a) Original Image, (b) $\rho_B(X)$, (c) $\Lambda_B(X)$

Now that the choice of threshold has been automated, a comparison between the pixel map after gradient thresholding, let's call it G, and the ideal image A, will be calculated. The pixel map will be converted to yield a binary image so the distance between these two images will be measured in order to determine the best MGT value. Using the $p-th$ order mean difference to determine the error, $\delta^p(A,G)$ will denote the transforms of the threshold distance of the two images A and G. Assuming that X is the pixel raster, $A \subseteq X$ where $A = \{x \in X : A(x) = 1\}$, and $\varrho(x,y)$ represents the distance between the pixels x and y and we can calculate the shortest distance between a pixel x where $x \in X$ and $A \subseteq X$.

$$d(x,A) = \inf\{\varrho(x,y) : y \in A\} \tag{14}$$

$$\delta^p(A,G) = \left[\frac{1}{N}\sum_{x\in X}|w(d(x,A)) - w(s(x,G))|^p\right]^{\frac{1}{p}} \text{ for } 1 \leq \varrho \leq \infty \tag{15}$$

where N represents the total number of pixels in the image X and $w(t) = \min(t,c)$ when $c > 0$ [2]. This will determine if the estimated image is suitable to be used in place of the original image, making segmentation and object detection more personalized and accurate.

6 Conclusions

Using the best possible quality images we could find, with the least amount of noise, allowed for a fairly high rate of success in terms of proper classification of images. This is an exciting result as a small margin of error shows that the success rate of probabilistic neural networks is high, making this algorithm efficient and effective. With this being said it appears a bit backwards that the network was only given the best images possible since the algorithm is supposed to identify ants and eggs in any image. One reason for this has to do with the computing power. Since neural networks are supposed to recognize objects based on their regional properties, in order to create a fully functioning neural network, a very large input and target vector would be required in order to prepare the network to distinguish all significant features. These large vectors would cause an overload for the available computation platforms.

Seeing the possibilities of neural networks and their improvement upon the future of artificial intelligence, it is important to keep in mind that this method is in no way complete and unanswered questions still exist. For example, tests have not been exhaustive; there is still a question of the networks ability to generalize enough to correctly classify images that are similar to the input and target vector but are not identical. Another question that has an answer but no current solution, is about the adequacy of the input and target vector set. There is not adequate information for the initial test of the neural network merely because the computational capacities available to us are not powerful enough to handle inputs of the necessary size [9]. With all of these questions in mind, the neural networks are a powerful tool and will play a vital role in advancing toward artificial intelligence.

Much work remains to create a neural network that properly recognizes and classifies objects. There are many difficulties to consider in terms of creating a program that has the intelligence to inspect the regional properties of an image and based on mere geometry, prop-

erly classify the object, but yet it has been done. These results are very striking and further research should be conducted in order to take full advantage of this significant opportunity in the technological field of object classification.

References

[1] Anna Dornhaus. The social insect lab. `http://www.eebweb.arizona.edu/faculty/dornhaus/`.

[2] Mar Pujol Lopez Ramon Piza Aldeguer Francisco Antonio Pujol Lopez, Juan Manuel Garcia CHamizo and M. J. Pujol. *Selection of an Automated Morpholigical Gradient Threshold for Image Segmentation.* Springer.

[3] De-Shuang Huang. *Radial Basis Probabilistic Neural Networks: Model and Application.* 1999.

[4] Jae S. Lim. *Two-Dimensional Signal and Imaging Processing.* Prentice-Hall Inc., 1990.

[5] MathWorks. Product documentation. `http://www.mathworks.com/help/techdoc/`.

[6] Terrence J. Sejnowski Michael I. Jordan. *Graphical Models: Foundations of Neural Computation.* Massachusetts Institutue of Technology., 2001.

[7] William K. Pratt. *Digital Image Processing.* John Wiley & Sons, Inc., 1978.

[8] Phillip D. Wasserman. *Neural Computing Theory and Practice.* Van Nostrand Reinhold., 1989.

[9] Phillip D. Wasserman. *Advanced Methods in Neural Computing.* Van Nostrand Reinhold., 1993.