IJMI
International Journal of Machine Intelligence
ISSN: 0975–2927 & E-ISSN: 0975–9166, Volume 3, Issue 2, 2011, pp-62-73
Available online at http://www.bioinfo.in/contents.php?id=31

CLASSIFICATION AND GRADING OF BULK SEEDS USING ARITIFICIAL NEURAL NETWORK

ANIL KANNUR[1*], ASHA KANNUR [2], VIJAY S RAJPUROHIT[3]
[1]Department of Computer Science & Engineering, Shaikh College of Engineering & Technology, Belgaum, India
[2]Department of Electronics & Communication, Shaikh College of Engineering & Technology, Belgaum, India
[3]Department of Computer Science & Engineering, Gogte Institute of Technology, Belgaum, India
*Corresponding Author(s): Email- anilkannur@indiatimes.com[1],goranal.asha@rediffmail.com[2], vijaysr2k@yahoo.com[3]

Received: August 10, 2011; Accepted: September 06, 2011

Abstract- This paper describes a different neural network model for classification and grading of bulk seeds samples using different artificial neural network models. Algorithms are developed to acquire and process color images of bulk seeds samples. Different seeds like Groundnut, Jowar, Wheat, Rice, Metagi, Red gram, Bengal gram, and Lentils etc. are considered for the study. The developed algorithms are used to extract over 11 (9 color, area and equidiameter) features, 18 (color only) features and 20 (18 color and 2 boundary) features. The area and equidiameter features are extracted from the watershed segmentation. Different types of Neural Network based classifier is used to identify the unknown seeds samples. The classification is carried out using different types of features sets, viz., color, area and equidiameter. Classification accuracies of over 85% are obtained for all the seeds types using all the three feature sets. And also different neural network gives different accuracies and time period taken for training all the three feature sets.
Key words – Classification, Grading, Watershed, Extraction, Seeds, Neural, Elman's, Cascade-Forward, Feed-Forward.

1. Introduction

In the past few years, automation and intelligent sensing technologies have revolutionized our food production and processing routines. Machine vision systems (MVS) provide an alternative to manual inspection of seeds samples for kernel characteristic properties and the amount of foreign material. During seeds handling operations, information on seeds type and seeds quality is required at several stages before the next course of operation system; seeds type and quality are rapidly assessed by visual inspection. These initiatives have been accredited to the rising concerns about food quality and safety. Also, rising labor costs, shortage of skilled workers, and the need to improve production processes have all put pressure on producers and processors (Van Henten et al. 2006)[5]. In such a scenario, automation can reduce the costs by promoting production efficiency. Automated solutions, such as quality grading and monitoring (Chao et al. 2000)[9], post-harvest product sorting (Wen and Tao 2000)[8], and robotics for field operations (Van Henten et al. 2006)[5]. often integrate machine vision technology for sensing due to its non-destructive and accurate measurement capability (Chen et al. 2002)[10]. The information acquired using imaging sensors contains geometric, color, and texture characteristics of the objects and these physical attributes have to be extracted using image processing algorithms.

Determining the potential of morphological features to classify different seeds species, classes, varieties, damaged seeds, and an impurity, using statistical pattern recognition techniques, has been the main focus of much of the published research. Some researchers have tried to use color features for seeds identification. Only limited work has been done to incorporate textural features for classification purposes. Efforts have also been made to integrate all these features in terms of a single classification vector for seeds kernel identification. Most of the published research mainly focuses on identifying a seeds type from digital images acquired by placing kernels in a non-touching fashion. Such experiments are comparatively easier to perform in controlled situations, as in a research lab, but would be difficult to implement on site because of cumbersome setup requirements. Such systems generally require a device to present kernels in a non-touching manner, an independent conveyor belt assembly, and the typical imaging devices in order to perform the task in real-time. The algorithms for classification of seeds type used with such images are based on morphological, color, boundary features, and combination of two or more features. The pre-processing operations such as background removal, and object extraction, which are a prerequisite to determining morphological features, are some of the most time-consuming operations.

2. Problem Statement

Recent research has shown that machine vision has the potential to become a viable tool to identify seeds types (Majumdar and Jayas 1999)[11]. Most of these studies have utilized well-defined images of seeds kernels acquired under controlled conditions. Under controlled

situations, seeds kernels are usually placed apart from each other during image acquisition. In an industrial setup, where these systems will finally be implemented, seed kernels touch each other or even overlap. Available imaging techniques utilize two-dimensional imagery to represent original three-dimensional objects. Such two-dimensional representation poses difficulty in recognizing individual items that touch or overlap with one another in a single image. Touching and overlapping of particles renders the extraction of shape and size parameters very difficult making recognition tasks impossible. Moreover, physical separation of particles using mechanical means prior to imaging is not always practical. Therefore, implementation of machine vision in seeds handling requires availability of algorithms that could separate occluding groups of seed kernels in digital images. Refer Figure 1.1, for the proposed methodology for classification and grading of bulk seeds.

3. Different Artificial Neural Network Models

3.1. An Elman's Neural Network

The training function can be any of the backprop training functions such as TRAINGD, TRAINGDM, TRAINGDA, TRAINGDX, etc… Algorithms which take large step sizes, such as TRAINLM, and TRAINRP, etc., are not recommended for Elman networks. Because of the delays in Elman networks the gradient of performance used by these algorithms is only approximated making learning difficult for large step algorithms. The learning function can be either of the backpropagation learning functions such as LEARNGD, or LEARNGDM. The performance function can be any of the differentiable performance functions such as MSE or MSEREG. Elman networks consist of Nl layers using the weight function, net input function, and the specified transfer functions. The first layer has weights coming from the input. Each subsequent layer has a weight coming from the previous layer. All layers except the last have a recurrent weight. All layers have biases. The last layer is the network output. Each layer's weights and biases are initialized with INITNW. Adaption is done with TRAINS which updates weights with the specified learning function.

3.2. Cascade-Forward Neural Network

The transfer functions can be any differentiable transfer function such as TANSIG, LOGSIG, or PURELIN. The training function can be any of the backprop training functions such as TRAINLM, TRAINBFG, TRAINRP, TRAINGD, etc… TRAINLM is the default training function because it is very fast, but it requires a lot of memory to run. If you get an "out-of-memory" error when training try doing one of these:

(1) Slow TRAINLM training, but reduce memory requirements, by reducing Network training Parameter by 2 or more.

(2) Use TRAINBFG, which is slower but more memory efficient than TRAINLM.

(3) Use TRAINRP which is slower but more memory efficient than TRAINBFG.

The learning function can be either of the backpropagation learning functions such as LEARNGD, or LEARNGDM. The performance function can be any of the differentiable performance functions such as MSE or MSEREG. A cascade-forward network consists of Nl layers using the weight function, net input function, and the specified transfer functions. The first layer has weights coming from the input. Each subsequent layer has weights coming from the input and all previous layers. All layers have biases. The last layer is the network output. Each layer's weights and biases are initialized with INITNW. Adaption is done with TRAINS which updates weights with the specified learning function.

3.3. Feed-Forward Neural Network

The transfer functions can be any differentiable transfer function such as TANSIG, LOGSIG, or PURELIN. The training function can be any of the backprop training functions such as TRAINLM, TRAINBFG, TRAINRP, TRAINGD, etc… TRAINLM is the default training function because it is very fast, but it requires a lot of memory to run. If you get an "out-of-memory" error when training try doing one of these:

(1) Slow TRAINLM training, but reduce memory requirements, by reducing Network training Parameter by 2 or more.

(2) Use TRAINBFG, which is slower but more memory efficient than TRAINLM.

(3) Use TRAINRP which is slower but more memory efficient than TRAINBFG.

The learning function BLF can be either of the backpropagation learning functions such as LEARNGD, or LEARNGDM. The performance function can be any of the differentiable performance functions such as MSE or MSEREG. Feed-forward networks consist of Nl layers using the weight function, net input function, and the specified transfer functions. The first layer has weights coming from the input. Each subsequent layer has a weight coming from the previous layer. All layers have biases. The last layer is the network output. Each layer's weights and biases are initialized with INITNW. Adaption is done with TRAINS which updates weights with the specified learning function.

4. Feature Extraction

In this, RGB components are separated (see figure 3), and then the color features extracted. And two more features namely area and equidiameter are extracted using watershed segmentation as explained in section 4.2.

4.1 Color Feature Extraction

The extraction of RGB and HSI Color features (for the list see Table 1.1, 1.2, & 1.3) is as follows:

Algorithm 1: Eighteen Color Feature Extraction

Input: Original 24-bit color image.

Output: 18 color features.

Start

Step 1: Separate the RGB components from the original 24-bit input color image. First step in extraction of RGB

features is separation of RGB components from the original color image sample (Figure 4.1 a, b, c).

Step 2: The Red, Green, and Blue Components of original image of a sample grain and their respective histogram are shown in figure 4.1(d, e, f). The RGB mean, variance, and range are computed using the following expressions:

$$\text{Mean } m = \sum_{i=0}^{L-1} z_i p(z_i) \quad \text{..... eq. 4.1}$$

$$\text{Standard deviation } \sigma = \sqrt{\mu_2(z)} \quad \text{..... eq. 4.2}$$

(4.2)

$$\text{Variance} = \sigma x \, \sigma \quad \text{..... eq. 4.3}$$

Maximum element and minimum elements from given input image is:

max1=max(image), max2=max(max1)
min1=min(image), min2=min(min1)

The above function returns the row vector containing maximum element from each column, similarly find minimum element from whole matrix. Range is the difference between the maximum and minimum elements

Range = *max2-min2* ..... eq. 4.4

Step 3: Obtain the HSI components from RGB components using the equations:

$$H = \begin{cases} \theta & \text{if } B \le G \\ 360 - \theta & \text{if } B > G \end{cases} \quad \text{with } \theta = \cos^{-1}\left\{ \frac{\frac{1}{2}[(R-G)+(R-B)]}{[(R-G)^2 + (R-B)(G-B)]^{1/2}} \right\} \text{ ..eq. 4.5}$$

$$S = 1 - \frac{3}{(R+G+B)}[\min(R,G,B)] \quad \text{.. eq. 4.4}$$

$$I = \frac{1}{3}(R+G+B) \quad \text{.. eq. 4.4}$$

Step 4: Find the mean, variance, and range for each RGB and HSI components using the equations 4.1, 4.3, and 4.4

Stop.

4.2. Watershed Segmentation

The watershed segmentation algorithm and features extraction is introduced. The features like area and equidiameter which computed using region prop function after the watershed segmentation. And also the analysis of the algorithm is done using a sample image of groundnut.

Algorithm 2: Watershed Segmentation

Input: 24-bit color image

Output: Segmented image and two boundary features.

Step1: Reading of original image which is an atomic force microscope image of a surface coating (Figure 4.2a).

Step2: Type Conversion: Converting the RGB image to gray color image (Figure 4.2b).

Step3: Maximize Contrast: The top-hat transform is defined as the difference between the original image and its opening (Figure 4.2c). The opening of an image is the collection of foreground parts of an image that fit a particular structuring element. The bottom-hat transform is defined as the difference between the closing of the original image and the original image (Figure 4.2d).

Step4: Subtract Images: To maximize the contrast between the objects and the gaps that separate them from each other, the "bottom-hat" image is subtracted from the "original + top-hat" image (Figure 4.2e).

Step5: Convert Objects of Interest: Recall that watershed transform detects intensity "valleys" in an image. We use the complement function on our enhanced image to convert our objects of interest to intensity valleys (Figure 4.2f).

Step6: Detect Intensity Valleys: The location (Figure 4.2g) rather than the size of the regions in the imextendedmin image is important. The imimposemin function will modify the image to contain only those valleys found by the imextendedmin function. The imimposemin function will also change a valley's pixel values to zero (Figure 4.2h).

Step7: Watershed Segmentation: Watershed segmentation of the imposed minima image is accomplished with the watershed function (Figure 4.2i).

Step8: Extract Features from Label Matrix.

5. TRAINING PHASE FOR NEURAL NETWORK

The Neural Network is designed and implemented using the MATLAB 6p5 with Neural Network Toolbox 4.0. Jayas et al. (2000). There are four steps in the training process:

(i) Assemble the training data
(ii) Create the network.
(iii) Train the network.
(iv) Test and validate network response to new inputs.

The number of nodes in the hidden layer is calculated using the formula:

$$n = \frac{I + O}{2} + y^{0.5} \quad \text{..... eq. 5.1}$$

Where n=number of nodes in hidden layer,
I=number of input nodes (features),
O=number of output nodes, and
y=number of input patterns in the training set

For all types of Neural Network a common model is used, which consists of four layers with eight nodes at the output layer and number of nodes in input layer depend on the number of features used (See Figure 5.1). In this study, we have used three different sets of features (11, 18, & 20 Features). For all training images, color and boundary features are extracted and stored in the databases.

Algorithm 3: **Neural Network Learning Algorithm**

Input: Initial network with random weights.

Output: Update network with weights modified according to training patterns which are used for testing new samples for classification.

Start Step 1: Initialize the weights in the network (randomly).

Step 2: repeat

For **each** example e in the training set do

Forward pass

O=actual neural net out-put for e; T= desired output (target) for e; Calculate error (T-O) at the output

Backward pass

Compute delta_wi (changes in weight) for all weights from hidden layer to output layer;

Compute delta_wi for all weights from input layer to hidden layer;
Update the weights in the network;
End Until all examples classified correctly or stopping criterion satisfied
Return (network).
Stop
During training for identification of a grain, the ANNs are presented with binary output data. There are eight output variables representing eight grain types and two grades. For a particular grain images, its corresponding value should be one and others should be zero.

6. RESULTS AND DISCUSSION

This section gives the results of exhaustive experimentation of developed methodology. A total of 480 images of seeds samples (60 of each seeds type, see figure 2.1) are used in this experiment. Different Neural Network is used for training, learning, testing, and validating the images.

6.1 Different ANN Model Comparisons

Different ANN works in a different way, some of these reduce memory size and some reduces training time period. The comparison of all the networks is as shown in the figure. The Elman's network takes more time in training the neural network for the features sets mentioned in this study. So Elman's network is not feasible to use compared to others. Next is Cascade-Forward network is better than Elman's network. The Cascade-Forward network takes more in training but reduces the memory size. Among the different ANN, the better one is Feed-Forward network, which trains neural network fast and takes less memory compared to Elman's network. The performance (blue line in figure 6.1) of all three neural networks is as shown in the figure 6.1, the feed-forward network reaches goal (black line in figure 6.1) within 20-30 iterations whereas the other two networks cascade-forward and Elman's network requires more than 1000 iterations to reach the goal for the same set of features.

6.2. Different Feature Set Comparisons

6.2.1. Classification based on 11 features: The feature set used consists of 9 color features and 2 boundary features for the analysis. The 9 color features are mean, variance & range of RGB components and 2 boundary features are area and equi-diameter. A four layer Neural Network is used to develop a classifier. It consists of 11 input nodes, 12 nodes in each of the two hidden layers, and 8 output nodes, one for each grain type (11-18-18-8 neural network).

6.2.2. Classification based on 18 features: The feature set used consists of 18 color features for the analysis. The 18 color features are mean, variance & range of RGB and HSI components. A four layer Neural Network is used to develop a classifier. It consists of 18 input nodes, 12 nodes in each of the two hidden layers, and 8 output nodes, one for each grain type (18-19-19-8 neural network).

6.2.3. Classification based on 20 features: The feature set used consists of 18 color features and 2 boundary features for the analysis. The 18 color features are mean, variance & range of RGB and HSI components and 2 boundary features are area and equi-diameter. A four layer Neural Network is used to develop a classifier. It consists of 20 input nodes, 12 nodes in each of the two hidden layers, and 8 output nodes, one for each grain type (20-20-20-8 neural network).

6.3. Discussions

The summarized results shown in Figure 6.2 and Figure 6.3, the classification accuracy, as expected, is very high for all the seeds types. In this study, the three different types of ANN models are developed and tested, the better result is given by Feed-forward network and other types are very slow in training. The performance of the feed-forward network is good for the study and for other applications as it takes less time in training and uses less memory space.

7. CONCLUSION

The maximum and minimum grain classification accuracy is found to be 100% and 85% respectively. The results from this study can be used for rapid identification of grain types when they arrive in railcars at the terminal grain elevators. The work carried out has relevance to real world classification of seeds and it involves both image processing and pattern recognition techniques. The watershed segmentation is used for more accuracy in classifying different types of seeds. The images in this study are acquired from clean seeds samples. For future study, the effect of percentage of foreign material on classification accuracy can be investigated using images of bulk samples and separation of seeds using images of touched seeds. Among the different ANN, the Feed-forward network is better to use for this kind of study.

References

[1] Cyril Voyant, Marc Muselli, Christophe Paoli, Marie-Laure Nivet (2010) *CNRS, Corte*.

[2] Jayanta Kumar Basu, Debnath Bhattacharyya, Tai-hoon Kim(2010) *IJSEA* Vol. 4, No. 2.

[3] Rojalina Priyadarshini, Nillamadhab Dash, Tripti Swarnkar, Rachita Misra (2010) *Special Issue of IJCCT* Vol.1 Issue 4.

[4] Frauke Günther and Stefan Fritsch (2010) *The R Journal* Vol. 2/1.

[5] Van Henten E.J., Van Tuiji B.A.J., Hoogakker G.J., Van Der Weerd M.J. (2006) *Biosystems Engineering,* 94.

[6] Zhang M., Laszlo L. Mark M., Krutz G. and Cyrille (1999) *The International Society for Optical engineering*, vol. 3543, 208-219.

[7] Wang W. and Paliwal J. (2004) *North central CSAE/ASAE Conference Canada.*

[8] Wen Z. and Tao Y. (2000) *Transactions of the ASAE*, 43(2); 449-452.
[9] Chao K.P., Su Y.C., Chen C.S. (2000) *J. Appl. Phycol*. 12, 53–62.
[10] Chen J. et al. (2002) *Proc. Natl Acad. Sci. USA*, 99, 12257–12262.
[11] Majumdar S. and Jayas D.S. (1999) *Journal of Agricultural Engineering Research*. 73(1):35-47.

Table-1.1 – Eleven Features set consisting of RGB components of images and Boundary features.

Sr. No.	Component of the images	Features
1	Red Component	Mean
2		Variance
3		Range
4	Green Component	Mean
5		Variance
6		Range
7	Blue Component	Mean
8		Variance
9		Range
10	Seeds	Area of the Seeds
11		Equidiameter of the Seeds

Table-1.2 – Eighteen Features set consisting of RGB components and HSI components of images

Sr. No.	Component of the images	Features
1	Red Component	Mean
2		Variance
3		Range
4	Green Component	Mean
5		Variance
6		Range
7	Blue Component	Mean
8		Variance
9		Range
10	Hue Component	Mean
11		Variance
12		Range
13	Saturation Component	Mean
14		Variance
15		Range
16	Intensity Component	Mean
17		Variance
18		Range

Table-1.3 – Twenty Features set consisting of RGB components, HSI components of images & Boundary features.

Sr. No.	Component of the images	Features
1	Red Component	Mean
2		Variance
3		Range
4	Green Component	Mean
5		Variance
6		Range
7	Blue Component	Mean
8		Variance
9		Range
10	Hue Component	Mean
11		Variance
12		Range
13	Saturation Component	Mean
14		Variance
15		Range
16	Intensity Component	Mean
17		Variance
18		Range
19	Seeds	Area of the Seeds
20		Equidiameter of the Seeds

Table Appendix – Abbreviations used in this paper and their descriptions

Sr. No.	Short Forms	Descriptions
1	**TRAINGD**	**Gradient descent backpropagation:** Traingd is a network training function that updates weight and bias values according to gradient descent.
2	**TRAINGDM**	**Gradient descent with momentum backpropagation:** Traingdm is a network training function that updates weight and bias values according to gradient descent with momentum.
3	**TRAINGDA**	**Gradient descent with adaptive lr backpropagation: T**raingda is a network training function that updates weight and bias values according to gradient descent with adaptive learning rate.
4	**TRAINGDX**	**Gradient descent w/momentum & adaptive lr backpropagation:** Traingdx is a network training function that updates weight and bias values according to gradient descent momentum and an adaptive learning rate.
5	**TRAINLM**	**Levenberg-Marquardt backpropagation:** Trainlm is a network training function that updates weight and bias values according to Levenberg-Marquardt optimization.
6	**INITNW**	A layer initialization function which initializes a layer's weights and biases according to the Nguyen-Widrow initialization algorithm. This algorithm chooses values in order to distribute the active region of each neuron in the layer evenly across the layer's input space.
7	**LEARNGD**	Gradient descent weight/bias learning function
8	**LEARNGDM**	The gradient descent with momentum weight/bias learning function.
9	**MSEREG**	**Mean squared error with regularization performance function:** msereg is a network performance function. It measures network performance as the weight sum of two factors: the mean squared error and the mean squared weights and biases.

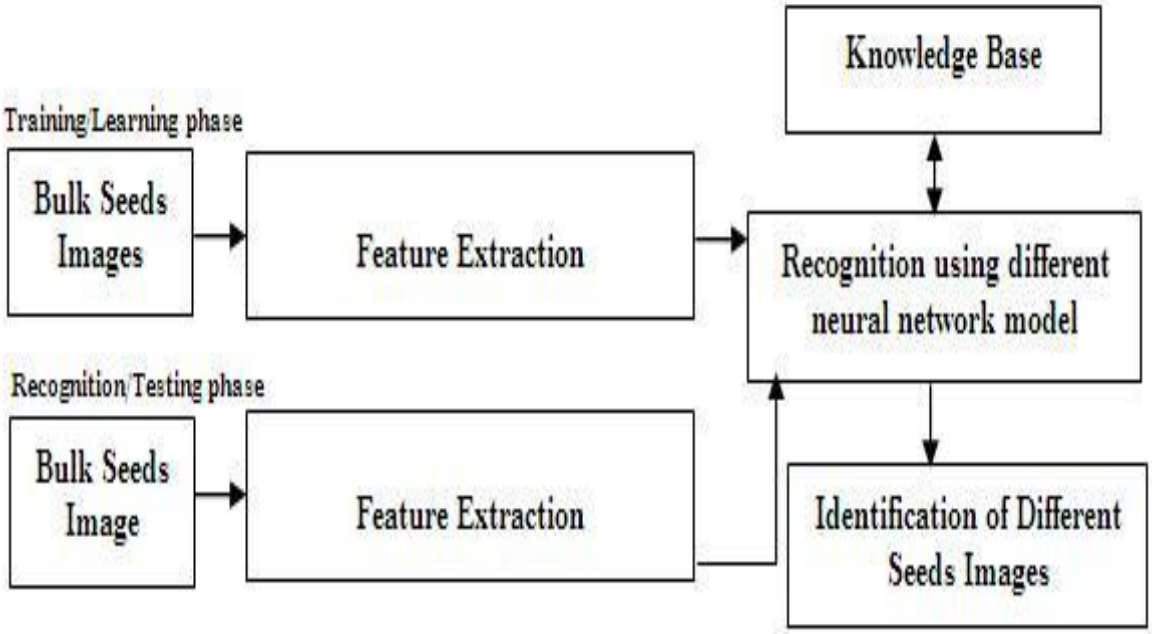

Figure 1.1 Proposed methodology block diagram

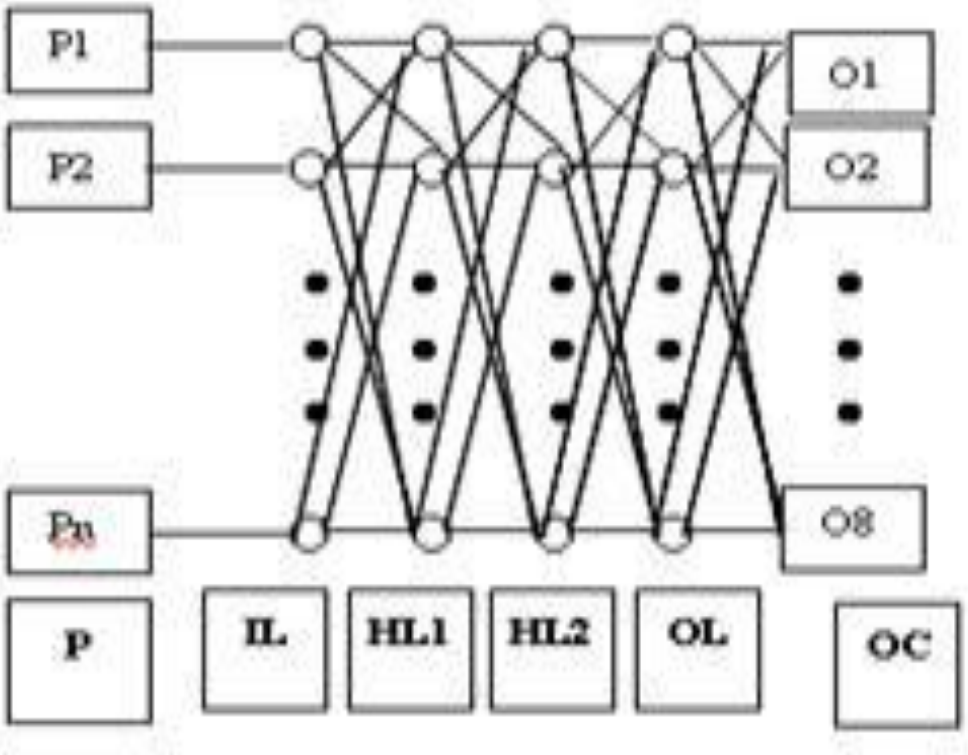

Figure 5.1 – An Artificial Neural Network Model block diagram

Figure 2.1(a) Groundnut Image of Bulk Seeds Samples used in this study

Figure 2.1(d) Jowar Image of Bulk Seeds Samples used in this study

Figure 2.1(b) Redgram Image of Bulk Seeds Samples used in this study

Figure 2.1(e) Metagi Image of Bulk Seeds Samples used in this study

Figure 2.1(c) Greengram Image of Bulk Seeds Samples used in this study

Figure 2.1(f) Bengalgram Image of Bulk Seeds Samples used in this study

Figure 2.1(g) Rice Image of Bulk Seeds Samples used in this study

Figure 2.1(h) Wheat Image of Bulk Seeds Samples used in this study

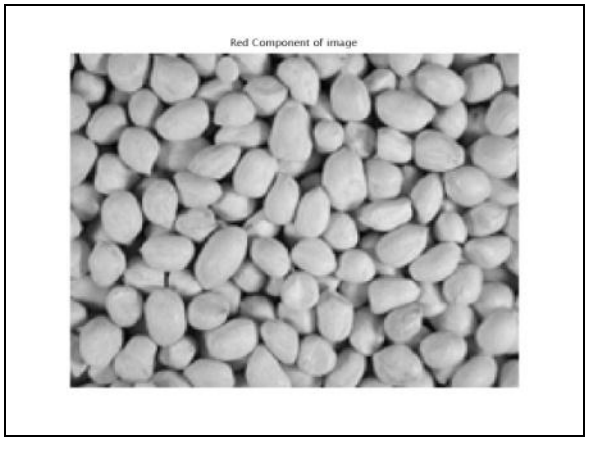

Figure 4.1(a) Red Component of sample – RGB Components and their Histograms Images separated using Algorithm 1

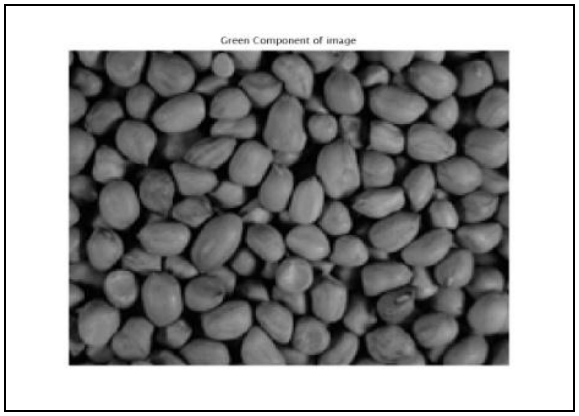

Figure 4.1(b) Green Component of sample – RGB Components and their Histograms Images separated using Algorithm 1

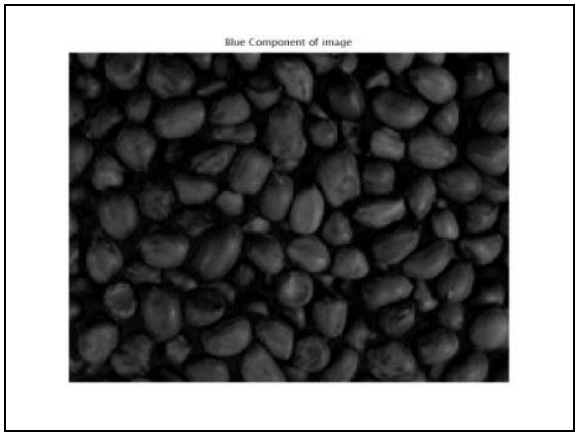

Figure 4.1(c) Green Component of sample – RGB Components and their Histograms Images separated using Algorithm 1

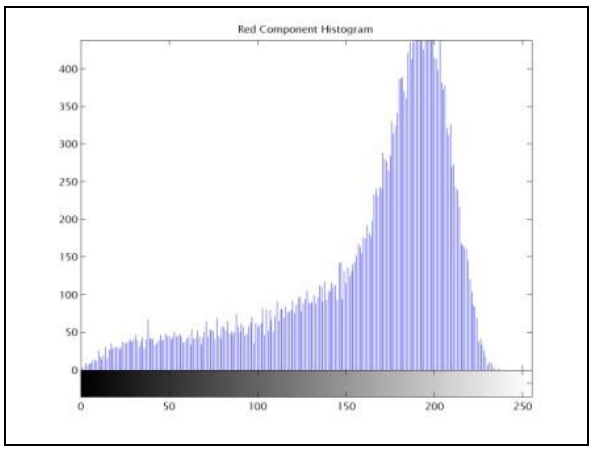

Figure 4.1(d) Red Component Histogram of sample – RGB Components and their Histograms Images separated using Algorithm 1

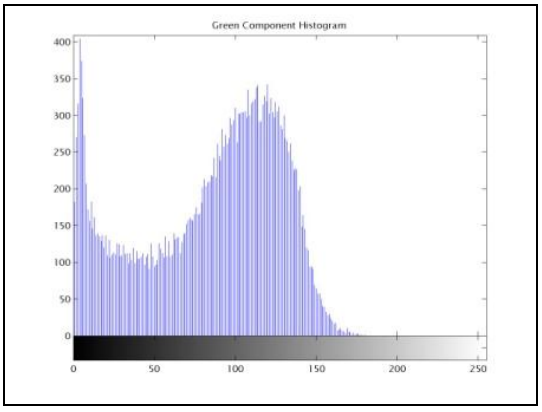

Figure 4.1(e) Green Component Histogram of sample – RGB Components and their Histograms Images separated using Algorithm 1

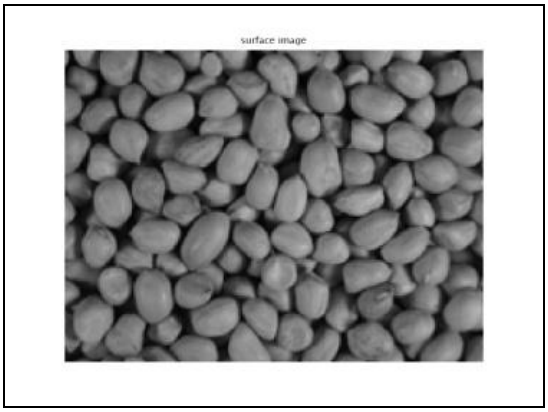

Figure 4.2(b) Surface Image – Images generated by Watershed Segmentation for extracting boundary features using Algorithm 2

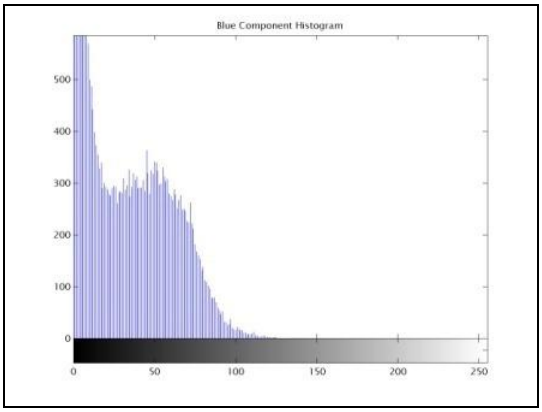

Figure 4.1(f) Green Component Histogram of sample – RGB Components and their Histograms Images separated using Algorithm 1

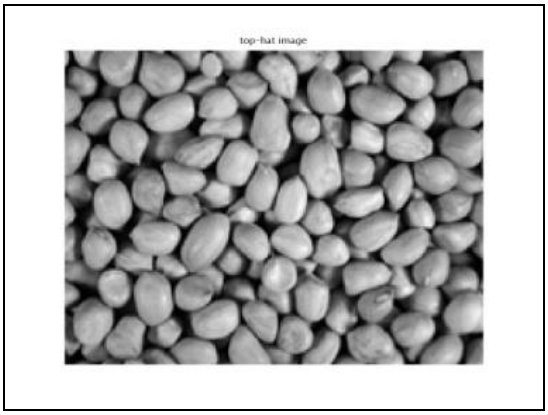

Figure 4.2(c) Top-Hat Image – Images generated by Watershed Segmentation for extracting boundary features using Algorithm 2

Figure 4.2(a) Original Image – Images generated by Watershed Segmentation for extracting boundary features using Algorithm 2

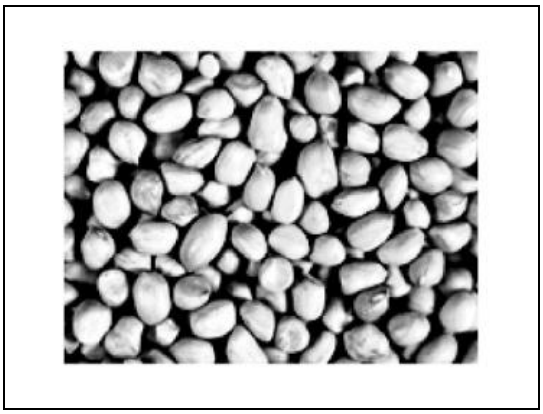

Figure 4.2(d) Bottom-Hat Image – Images generated by Watershed Segmentation for extracting boundary features using Algorithm 2

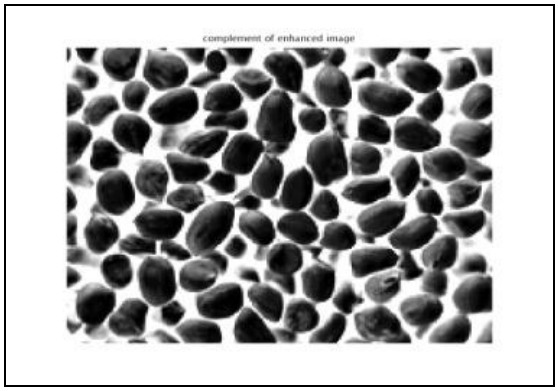

Figure 4.2(e) Subtract Image – Images generated by Watershed Segmentation for extracting boundary features using Algorithm 2

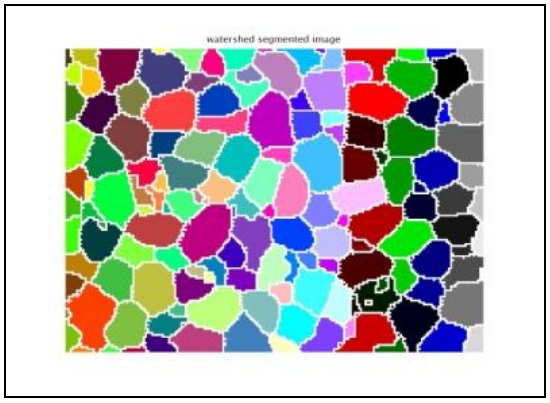

Figure 4.2(h) Watershed Image – Images generated by Watershed Segmentation for extracting boundary features using Algorithm 2

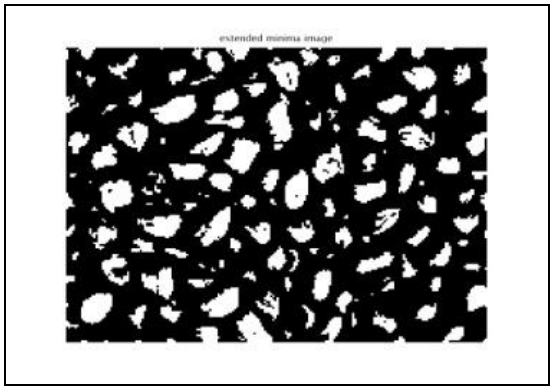

Figure 4.2(f) Binary Image – Images generated by Watershed Segmentation for extracting boundary features using Algorithm 2

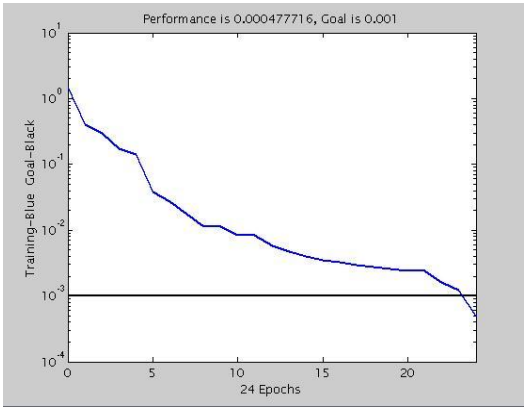

Figure 6.1 (a) *Feed-Forward Performance after 24 iterations – Performance of three Artificial Neural Networks*

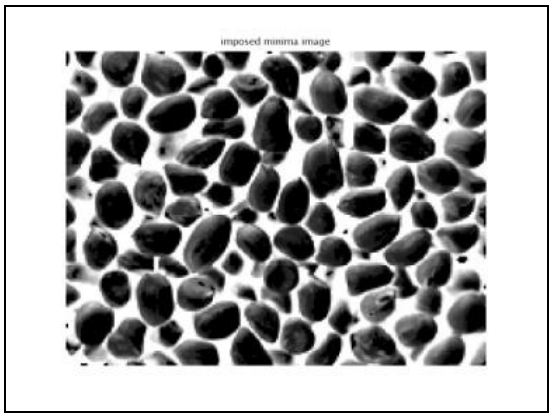

Figure 4.2(g) Imposed Image – Images generated by Watershed Segmentation for extracting boundary features using Algorithm 2

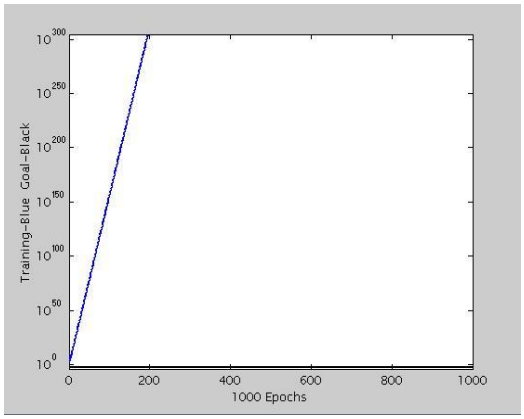

Figure 6.1 (b) Elman's Network Performance after 1000 iterations – Performance of three Artificial Neural Networks

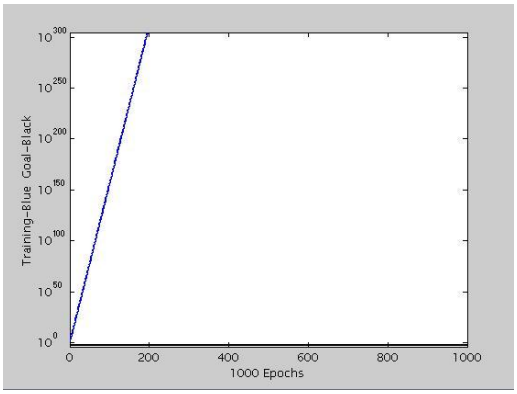

Figure 6.1 (c) *Cascade-Forward Network Performance after 1000 iterations – Performance of three Artificial Neural Networks*

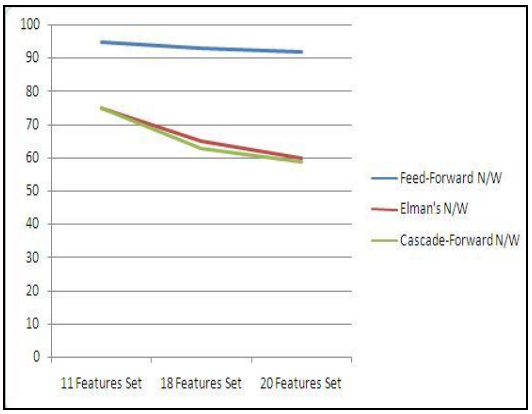

Figure 6.2 – Performance Comparisons of three Neural Network Models (blue line – Feed-forward, red line – Elman's N/W, green line – Cascade Forward N/W)

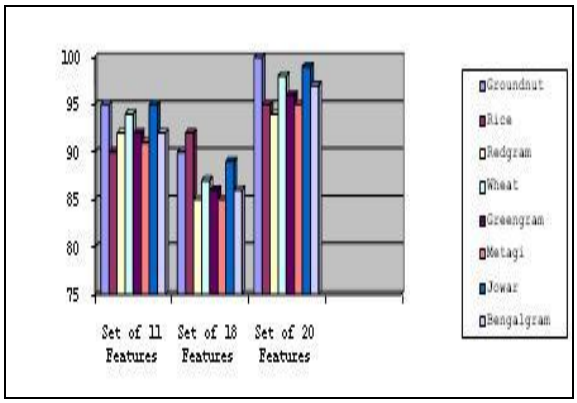

Figure 6.3 – Comparisons of three Features sets in classification & grading