British Journal of Mathematics & Computer Science
3(4): 617-634, 2013

SCIENCEDOMAIN *international*
www.sciencedomain.org

An Empirical Approach to Optimize Design of Backpropagation Neural Network Classifier for Textile Defect Inspection

Md. Tarek Habib[1*] and M. Rokonuzzaman[2]

[1]Department of Computer Science and Engineering, Prime University, North East of Darussalam Road, Mirpur-1, Dhaka-1216, Bangladesh.
[2]Department of Electrical and Electronic Engineering, Independent University, Aftabuddin Ahmed Road, Bashundhara, Dhaka-1229, Bangladesh.

Research Article

Received: 28 March 2013
Accepted: 20 June 2013
Published: 12 July 2013

Abstract

Automated fabric inspection systems have been drawing plenty of attention of the researchers in order to replace manual inspection. Two difficult problems are mainly posed by automated fabric inspection systems. They are defect detection and defect classification. Backpropagation is a popular learning algorithm and very promising for defect classification. In general, works reported to date have claimed varying level of successes in detection and classification of different types of defects through backpropagation model. In those published works, no investigation has been reported regarding to the variation of major performance parameters of neural network (NN) based classifiers such as training time and classification accuracy based on network topology and training parameters. As a result, application engineer has little or no guidance to take design decisions for reaching to optimum structure of NN based defect classifiers in general and backpropagation based in particular. Our work focuses on empirical investigation of interrelationship between design parameters and performance of backpropagation based classifier for textile defect classification. It is believed that such work will be laying the ground to empower application engineers to decide about optimum values of design parameters for realizing most appropriate backpropagation based classifier.

Keywords: Textile defect, machine vision, defect classification, artificial neural network (ANN), backpropagation algorithm, optimization problem, optimum design parameter.

1 Introduction

Automated textile inspection systems have been drawing a lot of attention of the researchers of many countries for a number of years. Automated textile inspection systems mainly involve two challenging problems, namely defect detection and defect classification.

**Corresponding author: md.tarekhabib@yahoo.com;*

Automated textile inspection systems are real-time applications. So they require real-time computation, which exceeds the capability of traditional computing. Artificial neural networks (ANNs) are suitable enough for real-time systems because of their parallel-processing capability. Moreover, ANNs have strong capability to handle classification problems with good classification accuracy. They vary in network architecture as well as training or learning algorithm. There is a number of performance metrics of ANN models. Classification accuracy, model complexity and training time are three of the most important performance metrics of ANN models.

Backpropagation is a popular learning algorithm, which is capable of handling large learning problems. It has been used by different research communities in different contexts, including textile defect classification, but sufficient investigation has not been performed in the context of textile defect classification. So an ANN trained by backpropagation algorithm appears to be a very good choice as a classifier in order to address the problem of textile defect classification.

Although there have been some reports about the feasibility of ANN based classifier development for textile defect classification, but there has been no reported work investigating interrelationship between design parameters and performance of ANN based classifier. Concept demonstration alone is not sufficient to empower an application engineer to design optimum classifier. Therefore, this work not only focuses on the study of the feasibility of backpropagation model in the context of textile defect classification, but also reports the findings of empirical investigation about the implications of backpropagation design parameters on the training and classification performance. In particular, we empirically discover the interrelationship between the performance metrics, accuracy and training time and the backpropagation design parameters, learning rate (γ), momentum rate (α) and model complexity (number of computing units in the hidden layer). Finally, we compare the performance of the backpropagation model with that of the classification models described in different articles.

2 Literature Review

Many efforts have been given for automated textile defect inspection [1-16]. Most of them have focused on defect detection, where some of them have focused on classification. ANNs have been used as classifiers in a number of articles. Different learning algorithms have been used in order to train the ANNs. The main intent of such works was not addressed to find an appropriate ANN model. That means none of them has performed a thorough investigation on interrelationship between design parameters and performance of ANN model.

Habib and Rokonuzzaman [2] have deployed counter propagation neural network (CPN) in order to classify four types of defects. Basically, they concentrated on feature selection rather than giving attention to the backpropagation model. They have not performed in-depth investigation on interrelationship between design parameters and performance of backpropagation model.

Backpropagation learning algorithm has been used in [3-7]. Habib and Rokonuzzaman [3] mainly focused on feature selection rather than focusing on the ANN model. They have used four types of defects and two types of features. Saeidi et al. [4] have first performed off-line experiments and then performed on-line implementation. In both cases, they have used six types of defects. Karayiannis et al. [5] have used seven types of defect. They have used statistical texture features. Kuo and Lee [6] have used four types of defect. Mitropulos et al. [7] have used seven types of

defects in their research. Detailed investigation on interrelationship between design parameters and performance of ANN model has not been performed in any of these works discussed.

Resilient backpropagation learning algorithm has been used to train ANN in [8,9]. They have worked with several types of defects considering two of them as major types and all other types of defects as a single major type. They have not reported anything detailed regarding the investigation of finding an appropriate ANN model.

Shady et al. [10] have used learning vector quantization (LVQ) algorithm in order to train their ANNs. They have used six types of defects. They have separately worked on both spatial and frequency domains for defect detection. Kumar [13] has used two ANNs separately. The first one was trained by backpropagation algorithm. He has shown that the inspection system with this network is not cost-effective. So he has further used linear ANN trained by least mean square error (LMS) algorithm. The inspection system with this ANN is cost-effective. Karras et al. [15] have also separately used two ANNs. They have trained one ANN by backpropagation algorithm and the other one by Kohonen's Self-Organizing Feature Map (SOFM). Thorough investigation on interrelationship between design parameters and performance of ANN model has not been reported in any of these reviewed works.

Furferi et al. [17] have used Levemberg-Marquardt algorithm, a variant of backpropagation learning algorithm, in order to train their ANN for the grading of car seat fabric quality. They have used five quality classes. Furferi and Governi [18] have used the combination of a statistical method of a SOFM and a feed forward backpropagation ANN based approach to correctly classify woollen clothes to be recycled.

3 Backpropagation Neural Network Model

We use three-layer fully connected feedforward ANN for this model as shown in Fig. 1, where it is assumed that input layer contributes to the first layer.

3.1 Choice of Activation Function

Backpropagation algorithm searches the minimum of error function (E) in weight space using the gradient descent method. Since this method demands computation of the gradient of E at each iteration step, such an activation function has to be used that is continuous and differentiable, because a discontinuous activation function leads E to be discontinuous [19].

One of the most popular activation functions for backpropagation algorithm is the sigmoid function, s_c: $IR \rightarrow (0, 1)$, which is defined as follows:

$$s_c(x) = \frac{1}{1+e^{-cx}}. \tag{3.1}$$

The constant c can be selected arbitrarily. Higher the value of c is, closer is the shape of the sigmoid function to that of the step function, and in the limit $c \rightarrow \infty$, the sigmoid function converges to a step function, f:$IR \rightarrow \{0, 1\}$, which is defined as follows [19,20]:

$$f(x)=\begin{cases}0, & if\ x<0\\ \frac{1}{2}, & if\ x=0\\ 1, & if\ x>0\end{cases}. \tag{3.2}$$

3.2 Initialization of Weights

Training begins with randomly chosen weight values. Large values of weight may lead the output of the computing units in hidden layer to saturation, which causes large amount of training time to emerge from the saturated state. This happens because of the behavior of the sigmoid function [20]. So, randomly chosen initial weight values should be small.

3.3 Choice of Learning Rate (γ)

Learning rate, γ, is an independent parameter that determines how quickly learning being performed. A large value of γ causes rapid learning, but there is a risk that the learning, i.e. search process may oscillate. Again, a small value of γ leads to slow learning [20].

In fact, the right value of γ depends on the application. In a large number of applications, value between 0.1 and 0.9 is used for γ [20].

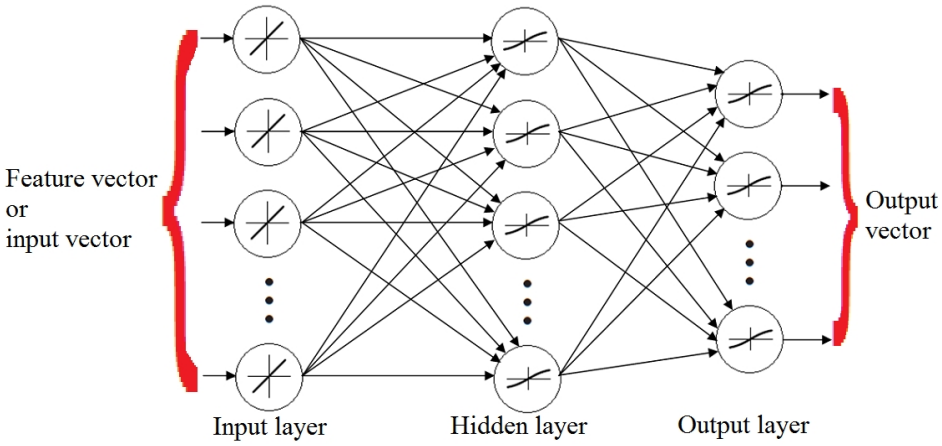

Fig. 1. Three-layer fully connected feed forward backpropagation ANN architecture

3.4 Introduction of Momentum Term

In the backpropagation algorithm, if the minimum of E lies in a narrow valley of weight space, following negative gradient direction can lead to wide oscillations of learning, i.e. search process, which makes the learning process take a large amount of time to converge. Moreover, the surface of E in the weight space can be highly uneven and jagged, which contains a large number of local minima. In that case, the learning, i.e. search process can get stuck in one of the local minima [19,20].

A simple solution to the problems discussed in the previous paragraph is the introduction of momentum term. This technique helps the learning, i.e. search process to avoid unwanted oscillations in narrow valleys of the surface of E in the weight space [19]. Moreover, the weighted average of the current gradient and the previous correction direction allow the weights to be changed in the general direction of decrease in E without getting stuck in a local minimum [20].

3.5 Reduction of Computing Units

Computation is too expensive with a large number of computing units. Again, training process does not converge with too small number of computing units. That means the ANN will not be powerful enough to solve the classification problem with too small number of computing units. In fact, the right size of ANN depends on the specific classification problem that is being solved using ANN.

4 Approach and Methodology

We are to address the problem of empirically discovering the interrelationship between performance metrics, accuracy and training time, and the network design parameters, learning rate (γ), momentum rate (α) and model complexity (number of computing units in the hidden layer). Our intention is to maximize accuracy and minimize training time. Both accuracy and training time depend on model complexity, learning rate and momentum rate. If accuracy, training time, model complexity, number of computing units in the input, hidden and output layer are represented by A, T, C_M, N_I, N_H and H_O respectively, then

$$A = f_2(C_M, \gamma, \alpha), \tag{4.1}$$

$$\text{and } T = f_2(C_M, \gamma, \alpha), \tag{4.2}$$

$$\text{where } C_M = (N_I, N_H, N_O). \tag{4.3}$$

So the optimization problem is defined as follows:

$$\begin{aligned} &\textit{maximize } f_1(C_M, \gamma, \alpha) \text{ and } \textit{minimize } f_2(C_M, \gamma, \alpha) \\ &\textit{subject to } N_I = 4 \\ &\qquad N_H \geq 6 \\ &\qquad N_O = 6 \\ &\qquad 0 < \gamma < 1 \\ &\qquad 0 < \beta < 1 \end{aligned}$$

4.1 Types of Defect

In this paper, we have dealt with four types of defect, which frequently occur in knitted fabrics, namely color yarn, hole, missing yarn (horizontal and vertical) and spot. All of the defects are shown in Fig. 2.

4.2 Feature Set

An appropriate feature set is selected for classifying the defects. The set consists of the features, which are as follows:

i) *Height of Defect Window,* H_{DW}.
ii) *Width of Defect Window,* W_{DW}.
iii) *Height to Width Ratio of Defect Window,* $R_{H/W} = H_{DW} / W_{DW}$.
iv) *Number of Defective Regions,* N_{DR}.

All of these four features can be found in detail in [2] for readers.

5 Implementation

The starting point of our approach is an inspection image of knitted fabric. About one hundred color images of defective and defect-free knitted fabrics of seven colors are captured using a 9.5-megapixel Fuji camera whose model is FinePix S9500. We proceed with inspection images of size 512×512 pixels, which are converted into a gray-scale image. In order to smooth these images and remove noises, they are filtered by 7×7 low-pass filter convolution mask. Then gray-scale histograms of the images are formed. Two threshold values θ_L and θ_H are calculated from each of these histograms using histogram peak technique [21]. Using the two threshold values θ_L and θ_H, images with pixels *P(x, y)* are

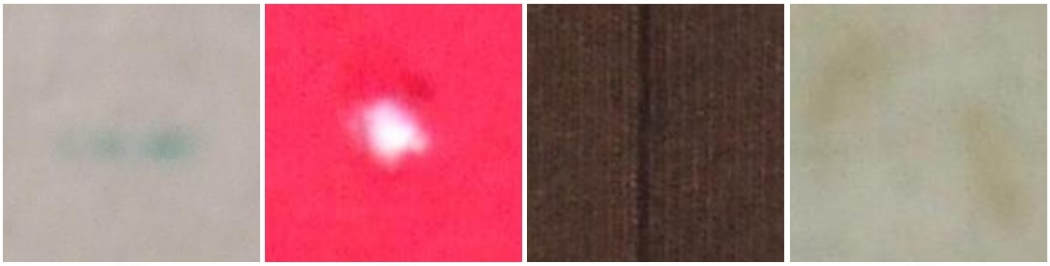

Fig. 2. Different types of defect occurred in knitted fabrics. (a) Color yarn. (b) Hole. (c) Missing yarn. (d) Spot

converted to binary images with pixels $I_B(x, y)$, where

$$I_B(x,y) = \begin{cases} 1, & if\ \theta_L \le P(x,y) \le \theta_H \\ 0, & otherwise \end{cases} \tag{5.1}$$

These binary images contain objects (defects) if any exists, background (defect-free fabric), and some noises. These noises are smaller than the minimum defect wanted to detect. In our approach, we want to detect a defect of minimum size 3mm×1mm. So, any object smaller than the minimum-defect size in pixels is eliminated from the binary images. If the minimum-defect size in pixels is θ_{MD} and an object with pixels $Obj(x, y)$ is of size N_{obj} pixels, then

$$Obj(x,y) = \begin{cases} 1, & if\ N_{obj} \ge \theta_{MD} \\ 0, & otherwise \end{cases}. \tag{5.2}$$

Then a number of features of defects are calculated, which forms feature vectors corresponding to defects present in images.

The classification step consists of the tasks of finding proper backpropagation model from a number of backpropagation models. Building a backpropagation model involves two phases, namely training phase and testing phase. For this purpose, one hundred color images of defective and defect-free knitted fabrics of seven colors are captured. So, the number of calculated features or input vectors is 100. That means our sample consists of 100 feature vectors. Table 1 shows the frequency of each defect and defect-free class in our sample of 100 images.

Table 1. Frequency of each defect and defect-free class

No.	Class	Frequency
1	Color Yarn	6
2	Vertical Missing Yarn	16
3	Horizontal Missing Yarn	16
4	Hole	11
5	Spot	18
6	Defect-Free	33
	Total	100

The features provided by the feature extractor are of values of different ranges, which causes imbalance among the differences of feature values of different defect types and makes the training phase difficult. The scaling, shown in (5.3), (5.4), (5.5) and (5.6), of the features is made in order to have proper balance among the differences of feature values of defect types. If H'_{DW}, W'_{DW}, $R'_{H/W}$ and N'_{DR} represent the scaled values of the features provided by the feature extractor H_{DW}, W_{DW}, $R_{H/W}$, and N_{DR}, respectively, then

$$H'_{DW} = \frac{H_{DW}}{512} \times 100, \tag{5.3}$$

$$W'_{DW} = \frac{W_{DW}}{512} \times 100 , \tag{5.4}$$

$$R'_{H/W} = 100 \times R_{H/W}, \tag{5.5}$$

$$N'_{DR} = \sqrt[500]{(N_{DR} - 1) \times 10^{999}} . \tag{5.6}$$

We split all feature vectors into two parts. One part consisting of 53 feature vectors is for both testing and training the backpropagation model and the other part consisting of the rest of the feature vectors is for testing only. The target values are set to 1 and 0s for the corresponding class and the rest of the classes, respectively. That means if a feature vector is presented to the backpropagation model during training, the corresponding computing unit in the output layer of the corresponding class of the feature vector is assumed to fire 1 and all other units in the output layer are assumed to fire 0. The backpropagation model is trained with maximum number of training cycle 10^6, maximum amount of training time 1 hour and maximum tolerable error less than 10^{-3}. That means training continues until 10^6 training cycles and 1 hour is elapsed and error less than 10^{-3} is found. After the training phase is completed, the backpropagation model is tested with all the feature vectors of the both parts. Then all feature vectors are again split into two parts. The first fifty percent of the part for training comes from the previous part for training and the rest fifty percent comes from the previous part for only testing. All other feature vectors form the new part for only testing. The backpropagation model is trained with these new parts and then is tested. In this way, for a specific combination of backpropagation design parameters, the model is trained and tested from 3 to 5 times in total. We take the results which mostly occur. If the results are uni-modal, we take the results that are the closest to their averages.

We use three-layer feed forward ANN for our model assuming that input layer contributes one layer. We started with a large ANN that has 4 computing units in the input layer, 48 computing units in the hidden layer and 6 computing units in the output layer (since we have six different classes according to Table 1). We describe the entire training in detail in the following parts of this section, i.e. Section 5.

5.1 Activation Function Chosen

We choose the following sigmoid function, s_c: $IR \rightarrow (0, 1)$, where $c = 1$:

$$s_1(x) = \frac{1}{1 + e^{-x}} . \tag{5.7}$$

5.2 Weights Initialized

We randomly choose initial weight values of small range, i.e. between -1.0 and 1.0.

5.3 Learning Rate (γ) and Momentum Rate (α) Chosen

We first train the ANN letting γ equal 0.01. Then we test the ANN with the feature vectors, which comprise our entire sample. We gradually increase the value of γ, and train as well as test the ANN for that value of γ. The achieved results are shown in Table 2 and Fig. 3. Here is to mention that the elapsed training time of each training is much less than 5 hours.

Table 2. Results of tuning learning rate γ

Network size (Number of computing			Learning rate (γ)	Error function (E)	Number of elapsed epoch	Accuracy
Input layer	Hidden layer	Output layer				
4	48	6	0.01	9.999967×10^{-4}	785617	96.91%
			0.1	9.999991×10^{-4}	316448	96.91%
			0.2	6.500011	1000000	74.23%
			0.3	17.740225	450493	37.11%
			0.4	19.000553	870657	32.99%

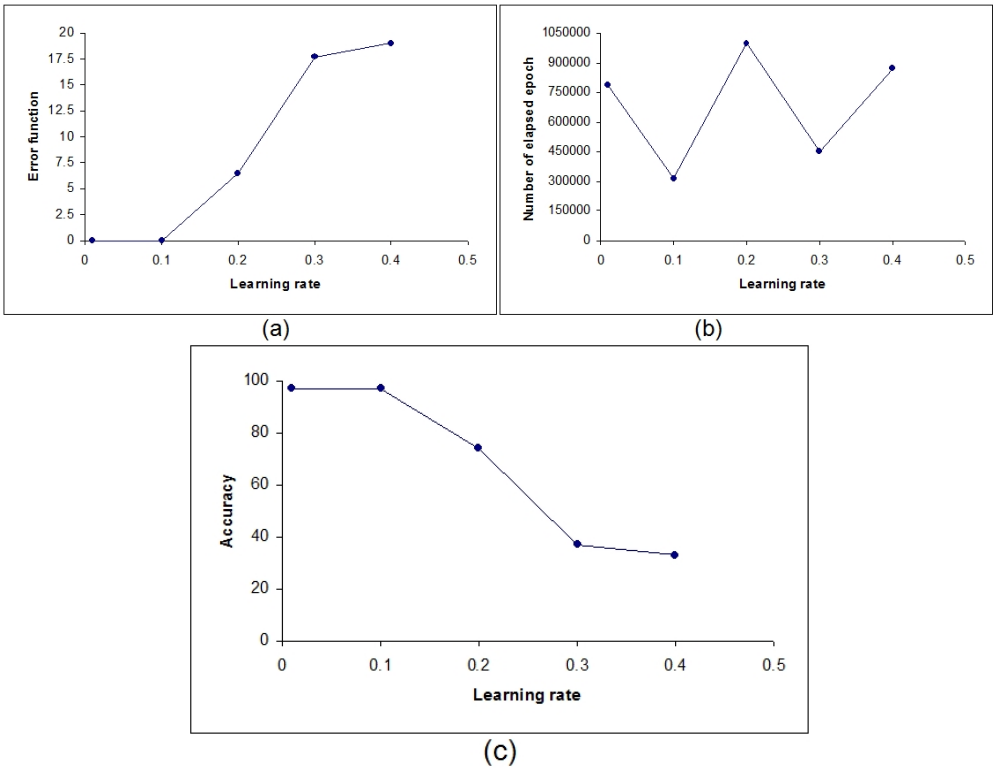

Fig. 3. Results of tuning learning rate γ. (a) Error function. (b) Number of elapsed epoch. (c) Accuracy

We see from Table 2 and Fig. 3 that E is tolerable for $0.01 \leq \gamma \leq 0.1$ and E increases for $\gamma > 0.1$. As a result, accuracy is maximum, i.e. 96.91%, for $0.01 \leq \gamma \leq 0.1$ and it decreases for $\gamma > 0.1$. Number of elapsed epoch is minimum, i.e. 316448, for $\gamma = 0.1$. Like many applications, performance is good here for $0.01 \leq \gamma \leq 0.1$. There is a risk for a large value of γ that the learning, i.e. search process may oscillate that makes the learning process take a large amount of time to converge. Again, the backpropagation learning, i.e. search process can get stuck in a local minimum of E in the weight space. Here, for $\gamma > 0.1$, this kind of oscillations happens, or the search process gets stuck in a local minimum of E. Since larger value of γ causes rapid learning, number of elapsed epoch is minimum for $\gamma = 0.1$, rather than for $\gamma = 0.01$.

Number of elapsed epoch, 316448, is large enough and takes large amount of time. Moreover, accuracy, 96.91%, is not good enough for the sample size used. So, we introduce a momentum term to the backpropagation algorithm.

We first train the ANN letting α equal 0.01 and γ equal 0.1. Then we test the ANN with the feature vectors, which comprise our entire sample. We gradually increase the value of α, and train as well as test the ANN for that value of α keeping the value of γ fixed. Again, we gradually decrease the value of α from 0.01, and train as well as test the ANN for that value of α keeping the value of γ fixed so that number of elapsed epoch can be reduced further. The results achieved are shown in Table 3 and Fig. 4. Here is to mention that the elapsed training time of each training is much less than 5 hours.

Table 3. Results of tuning momentum rate α around 0.01

Network size (Number of computing units)			Learning rate (γ)	Momentum rate (α)	Error function (E)	Number of elapsed epoch	Accuracy
Input layer	Hidden layer	Output layer					
4	48	6	0.1	0.005	9.999695×10^{-4}	175712	100%
				0.006	9.998915×10^{-4}	161824	100%
				0.007	9.999927×10^{-4}	161075	100%
				0.008	9.999356×10^{-4}	137737	100%
				0.009	9.999839×10^{-4}	132267	100%
				0.01	9.999674×10^{-4}	122425	100%
				0.05	9.999992×10^{-4}	98724	100%
				0.1	9.999899×10^{-4}	92638	100%
				0.15	9.999392×10^{-4}	93729	100%
				0.2	9.999878×10^{-4}	94406	100%
				0.25	9.999812×10^{-4}	77159	100%
				0.3	9.999933×10^{-4}	49355	100%
				0.35	9.999822×10^{-4}	54589	100%
				0.4	9.999878×10^{-4}	67081	98.97%
				0.45	9.999866×10^{-4}	108984	97.94%

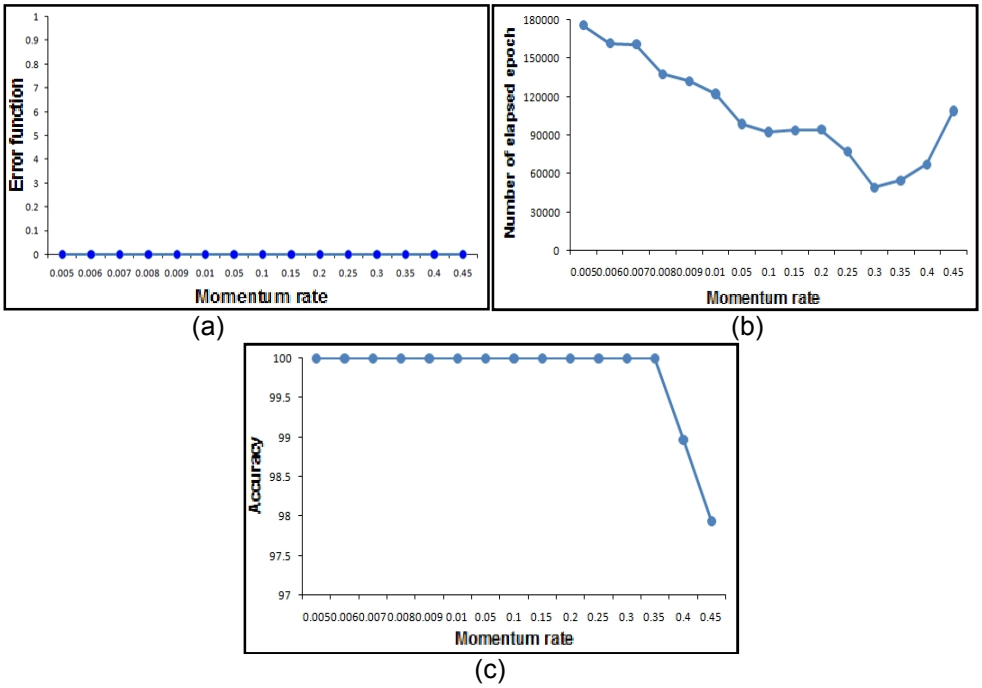

Fig. 4. Results of tuning momentum rate α. (a) Error function. (b) Number of elapsed epoch. (c) Accuracy

We see from Table 3 and Fig. 4 that E is tolerable for $\alpha \geq 0.01$, but accuracy is maximum, i.e. 100% for $0.01 \leq \alpha \leq 0.35$. Accuracy starts decreasing for $\alpha > 0.35$. Number of elapsed epoch is minimum, i.e. 49355, when $\alpha = 0.3$. Although number of elapsed epoch, 49355, is reduced after introducing momentum term, it is not small enough. E is tolerable and accuracy is maximum, i.e. 100%, for $\alpha \leq 0.01$. Number of elapsed epoch increases for $\alpha \leq 0.01$.

Here, number of elapsed epoch is significantly reduced for each training after introducing momentum term since unwanted oscillations have been avoided. Moreover, E is found tolerable for each training after introducing momentum term because of not getting stuck in a local minimum. Consequently, accuracy is maximum, 100%, for almost each training.

5.4 Computing Units Reduced

One approach to find the right size of ANN is to start training and testing with a large ANN. Then some computing units and their associated incoming and outgoing edges are eliminated, and the ANN is retrained and retested. This procedure continues until the network performance reaches an unacceptable level [20,22]. Following the approach, we train a large three-layer fully connected feed forward ANN, which has 4, 48 and 6 computing units in the input, hidden and output layer respectively. We test that ANN with the feature vectors comprising our entire sample. Then we successively eliminate 3 computing units in the hidden layer, and train as well as test the reduced neural network. We carry on the procedure until the network performance reaches an unacceptable

level. The results achieved are shown in Table 4 and Fig. 5. Here is to mention that the elapsed training time of each training is much less than 5 hours.

Table 4. Results of reducing computing units in hidden layer

Network Size (Number of computing			Learning rate (γ)	Momentum rate (α)	Error function (E)	Number of elapsed epoch	Accuracy
Input layer	Hidden layer	Output layer					
4	48	6	0.1	0.3	9.999933×10^{-4}	49355	100%
	45				9.999748×10^{-4}	34582	100%
	42				9.999901×10^{-4}	50835	100%
	39				9.999675×10^{-4}	21470	100%
	36				9.999912×10^{-4}	50945	100%
	33				9.999874×10^{-4}	31313	100%
	30				9.999935×10^{-4}	66077	100%
	27				9.999693×10^{-4}	34933	100%
	24				9.999999×10^{-4}	72820	100%
	21				9.999972×10^{-4}	62763	100%
	18				9.999984×10^{-4}	74643	100%
	15				9.999873×10^{-4}	69937	100%
	12				9.999858×10^{-4}	137043	100%
	9				2.490976	53027	91.75%
	8				6.191334	129981	82.47%

The classifier design objective of an application engineer is to choose such network topology that requires minimum training time and produces maximum classification accuracy. From this investigation, it appears that the network topology with $3i$ ($4 \leq i \leq 16$) computing units in hidden layer produces the highest accuracy and the network topology with i ($8 \leq i \leq 9$) computing units in hidden layer produces less accuracy. All of these investigations are summarized in Table 5.

Table 5. Summary of the results of all investigations

Design parameter	Optimum band	Number of elapsed training cycle	Accuracy
Learning rate (γ)	0.01 – 0.1	316448 – 785617	96.91%
Momentum rate (α)	0.005 – 0.35	49355 – 175712	100%
Network topology (number of computing units)	4-3i-6 ($4 \leq i \leq 16$)	21470 – 137043	100%

6 Comparative Analysis of Performance

In order to assess merits of our implemented backpropagation model for classifying textile defects, let's compare some recently reported relevant research results. It is to be noted that

assumptions taken by researchers in collecting samples and reporting results of their research activities in processing those samples will have serious implications on our attempt of comparative performance evaluation. The review of literature reveals that most of research reports are limited to the demonstration of concepts of machine vision based approach to textile defect classification without the support of adequate numerical results and their comparison with similar works. Moreover, the absence of use of common database of samples of textile defects makes it difficult to have a fair comparison of merits of different algorithms. Similar observation has been reported by Kumar in a comprehensive survey [23]. Kumar has also mentioned in his conclusion that although last few years have shown some encouraging trends in textile defect inspection research, systematic/comparative performance evaluation based on realistic assumptions is not sufficient. Despite such limitations, we have made an attempt to review numerical results related to textile defect classification to assess comparative merits of our work.

A number of learning algorithms have been used in order to train the ANNs. Backpropagation learning algorithm has been used in [3-7]. Habib and Rokonuzzaman [3] have worked with knitted fabrics. Their sample consisted of 100 images. They have used a three-layer feedforward ANN, which had 4, 12 and 6 computing units in the input, hidden and output layers respectively. It took 88811 cycles for the ANN to be trained. A 100%-accuracy has been found. Although the accuracy and model complexity (number of computing units) have been good and medium respectively, the training time has been long. Saeidi et al. [4] have worked with knitted fabrics. They have first performed off-line experiments and then performed on-line implementation. In case of off-line experiments, the sample size was 140. They have employed a three-layer feed forward ANN, which had 15, 8 and 7 computing units in the input, hidden and output layers, respectively. It took 7350 epochs for the ANN to be trained. An accuracy of 78.4% has been achieved. The model complexity has been modest. Moreover, the training time has been long and the accuracy has been poor. In case of on-line implementation, the sample size was 8485. An accuracy of 96.57% has been achieved by employing a feed forward ANN. The accuracy has been good although the model complexity and training time have not been mentioned. Karayiannis et al. [5] have worked with web textile fabrics. They have used a three-layer ANN, which had 13, 5 and 8 computing units in the input, hidden and output layers, respectively. A sample of size 400 was used. A 94%-accuracy has been achieved. Although the accuracy and model complexity have been good and small, respectively, nothing has been mentioned about the training time. Kuo and Lee [6] have used plain white fabrics and have got accuracy varying from 95% to 100%. The accuracy has been modest. Moreover, the model complexity and training time have not been reported. Mitropulos et al. [7] have used web textile fabrics for their work. They have used a three-layer ANN, which had 4, 5 and 8 computing units in the input, hidden and output layers, respectively. They have got an accuracy of 91%, where the sample size was 400. The accuracy has been modest although the model complexity has been small. Nothing has been mentioned about the training time.

Resilient backpropagation learning algorithm has been used in [8,9]. Islam et al. [8] have used a fully connected four-layer ANN, which contained 3, 40, 4, and 4 computing units in the input, first hidden, second hidden and output layers, respectively. They have worked with a sample of over 200 images. They have got an accuracy of 77%. The accuracy has been poor and the model complexity has been large. Moreover, the training time has not been given. Islam et al. [9] have employed a fully connected three-layer ANN, which had 3, 44 and 4 computing units in the input, hidden and output layers, respectively. 220 images have been used as sample. An accuracy of 76.5% has been achieved. The accuracy and model complexity have been poor and large, respectively. Moreover, nothing has been mentioned about the training time.

Habib and Rokonuzzaman [2] have worked with CPN. Their sample consisted of 100 images of knitted fabrics. Their CPN had 4, 12 and 6 computing units in the input, hidden and output layers respectively. About 200 cycles was taken for the training of CPN. An accuracy of 100% has been achieved. Although the accuracy and training time have been good, the model complexity (number of computing units) has been too long in the context of CPN.

Shady et al. [10] have separately worked on both spatial and frequency domains in order to extract features from images of knitted fabric. They have used the LVQ algorithm in order to train the ANNs for both domains. A sample of 205 images was used. In case of spatial domain, they employed a two-layer ANN, which contained 7 computing units in the input layer and same number of units in the output layer. They achieved a 90.21%-accuracy. The accuracy has been modest although the model complexity has been small. Moreover, the training time has not been given. In case of frequency domain, they employed a two-layer ANN, which had 6 and 7 computing units in the input and output layers, respectively. An accuracy of 91.9% has been achieved. Although the model complexity has been small, the accuracy has been modest. Moreover, nothing has been mentioned about the training time.

Table 6 shows the comparison of our backpropagation model and others' ANN models. For our backpropagation model as shown in Table 6, we consider the best result found after entire implementation.

Kumar [23] has found that more than 95% accuracy appears to be industry benchmark. In that survey, it has been reported by Kumar in reviewing 150 articles that a quantitative comparison between the various defect detection schemes is difficult as the performance of each of these schemes have been assessed/reported on the fabric test images with varying resolution, background texture and defects.

With respect to such observation, our obtained accuracy of 100% appears to be good enough. Moreover, our model complexity (4, 15 and 6 computing units in the input, hidden and output layer respectively) and training time (69937 cycles) have been promising. As we have mentioned earlier, due to the lack of uniformity in the image data set, performance evaluation and the nature of intended application, it is not prudent to explicitly compare merits of our approach with other works. Therefore, it may not be unfair to claim that our implemented backpropagation model have enough potential to classify textile defects with very good accuracy.

Table 6. Results of the comparison of our backpropagation model and others' models

Reference	Type of Fabric	Number of Input Sites	Number of Classes	Sample Size (No. of Feature Vectors)	Performance Metrics			
					Training Time (Number of Elapsed Cycle)	Model Complexity		Accuracy
						Number of Computing Units	Connectivity	
Our work	Knitted fabric	4	6	100	69937	4-15-6	Fully connected feedforward	100%
[2]	Knitted fabric	4	6	100	191	4-12-6	Fully connected feedforward	100%
[3]	Knitted fabric	4	6	100	88811	4-12-6	Fully connected feedforward	100%
[4]	Knitted fabric	15	7	140	7350	15-8-7	Feedforward	78.4%
		NM[1]	NM	8485	NM	NM	Feedforward	96.57%
[5]	Web textile fabric	13	8	400	NM	13-5-8	NM	94%
[7]	Web textile fabric	4	8	400	NM	4-5-8	NM	91%
[8]	NM	3	4	Over 200	NM	3-40-4-4	Fully connected feedforward	77%
[9]	NM	3	4	220	NM	3-44-4	Fully connected feedforward	76.5%
[10]	Knitted fabric	7	7	205	NM	7-7	NM	90.21%
		6	7	205	NM	6-7	NM	91.9%

[1]*NM: Not mentioned*

7 Conclusion and Future Work

In this paper, we have not only found that the backpropagation model is suitable enough for automated textile defect classification, but we have also found an appropriate backpropagation model in the context of textile defect classification by empirically investigating the inter-relationship among the performance metrics, accuracy, training time and model complexity, and the network parameters, learning and momentum rate empirically. It's believed that such investigative approach will be laying the basis to guide application engineers to decide about optimum values of design parameters for realizing most appropriate backpropagation based classifier. Finally, we have compared the performance of the backpropagation model with that of the classification models described in different articles. In comparison to classification performances of reported research findings, our obtained accuracy of 100% appears to be quite good.

Due to small sample size, our finding is not comprehensive enough to make conclusive comment about the merits of our implemented backpropagation model. Moreover, during acquiring images, lighting was not good enough to produce very high quality images. Further work remains to successfully classify commonly occurring textile defects for a sample of a very large number of high-quality images.

Moreover, there is a need of developing a common database of samples of textile defects to make a fair comparison of merits of different algorithms.

Competing Interests

Authors have declared that no competing interests exist.

References

[1] Abouelela A, Abbas HM, Eldeeb H, Wahdan AA, Nassar SM. Automated vision system for localizing structural defects in textile fabrics. Pattern Recognition Letters. 2005;26(10):1435-1443.

[2] Habib MT, Rokonuzzaman M. A set of geometric features for neural network based fabric defect classification. International Scholarly Research Network Artificial Intelligence. 2012;2012. http://dx.doi.org/10.5402/2012/643473.

[3] Habib MT, Rokonuzzaman M. Distinguishing feature selection for fabric defect classification using neural network. Academy Publisher Journal of Multimedia. 2011;6(5):416–24. http://dx.doi.org/10.4304/jmm.6.5.416-424

[4] Saeidi RG, Latifi M, Najar SS, Saeidi AG. Computer vision-aided fabric inspection system for on-circular knitting machine. Textile Research Journal. 2005;75(6):492-497. http://dx.doi.org/10.1177/0040517505053874

[5] Karayiannis YA, Stojanovic R, Mitropoulos P, Koulamas C, Stouraitis T, Koubias S, Papadopoulos G. Defect detection and classification on web textile fabric using multiresolution decomposition and neural networks. Proceedings of the 6th IEEE International Conference on Electronics, Circuits and Systems, Pafos, Cyprus. September 1999.

[6] Kuo C-FJ, Lee C-J. A back-propagation neural network for recognizing fabric defects. Textile Research Journal. 2003;73(2):147-151.
Available: http://dx.doi.org/10.1177/004051750307300209.

[7] Mitropoulos P, Koulamas C, Stojanovic R, Koubias S, Papadopoulos G, Karayiannis G. Real-time vision system for defect detection and neural classification of web textile fabric. Proceedings SPIE, San Jose, California. January 1999;3652:59-69.

[8] Islam MA, Akhter S, Mursalin TE. Automated fabric defect recognition system using computer vision and artificial neural networks. Proceedings World Academy of Science, Engineering and Technology. 2006;13:1-7.

[9] Islam MA, Akhter S, Mursalin TE, Amin MA. A suitable neural network to detect fabric defects. Neural Information Processing, SpringerLink. 2006;4233:430-438.
Available: http://dx.doi.org/10.1007/11893257_48.

[10] Shady E, Gowayed Y, Abouiiana M, Youssef S, Pastore C. Detection and classification of defects in knitted fabric structures. Textile Research Journal. 2006;76(4):295-300.
Available: http://dx.doi.org/10.1177/0040517506053906.

[11] Mak KL, Peng P, Lau HYK. A real-time computer vision system for detecting defects in textile fabrics. Proceedings IEEE International Conference on Industrial Technology, Hong Kong, China. December 2005.

[12] Baykut A, Atalay A, Erçil A, Güler M. Real-time defect inspection of textured surfaces. Real-Time Imaging. 2000;6(1):17–27.
Available: http://dx.doi.org/10.1006/rtim.1998.0153.

[13] Kumar A. Neural network based detection of local fabric defects. Pattern Recognition. 2003;36:1645-1659.
Available: http://dx.doi.org/10.1016/S0031-3203(03)00005-0.

[14] Cohen FS, Fan Z. Rotation and scale invariant texture classification. Proceedings IEEE Conf. Robot. Autom. April 1988;3:1394–1399.

[15] Karras DA, Karkanis SA, Mertzios BG. Supervised and unsupervised neural network methods applied to textile quality control based on improved wavelet feature extraction techniques. International Journal on Computer Mathematics. 1998;67:169-181.
Available: http://dx.doi.org/10.1080/00207169808804658.

[16] Stojanovic R, Mitropulos P, Koulamas C, Karayiannis YA, Koubias S, Papadopoulos G. Real-time vision based system for textile fabric inspection. Real-Time Imaging. 2001;7(6):507–518.
Available: http://dx.doi.org/10.1006/rtim.2001.0231.

[17] Furferi R, Governi L, Volpe Y. Neural network based classification of car seat fabrics. International Journal of Mathematical Models and Methods in Applied Sciences. 2011;5(3):696–703.

[18] Furferi R, Governi L. The recycling of wool clothes: an artificial neural network colour classification tool. International Journal of Advanced Manufacturing Technology. 2008;37(7-8):722–731.

[19] Rojas R. Neural networks: a systematic introduction. Germany: Springer-Verlag; 1996.

[20] Mehrotra K, Mohan CK, Ranka S. Elements of Artificial Neural Networks. India: Penram International Publishing; 1997.

[21] Phillips D. Image Processing in C. 2nd ed. Kansas: R & D Publications; 2000.

[22] Tan P-N, Steinbach M, Kumar V. Introduction to Data Mining. Boston: Addison-Wesley; 2006.

[23] Kumar A. Computer-vision-based fabric defect detection: a survey. IEEE Transactions on Industrial Electronics. 2008; 55(1):348-363.

Peer-review history:
The peer review history for this paper can be accessed here (Please copy paste the total link in your browser address bar)
www.sciencedomain.org/review-history.php?iid=240&id=6&aid=1634