کارگاه‌های آموزشی مرکز اطلاعات علمی

مقاله نویسی علوم انسانی

اصول تنظیم قراردادها

آموزش مهارت های کاربردی در تدوین و چاپ مقاله
An artificial Neural Network approach to monitor and diagnose multi-attribute quality control processes

S. T. A. Niaki*

Professor, Department of Industrial Engineering, Sharif University of Technology, Tehran, Iran

M. Nafar

Department of Industrial Engineering, Sharif University of Technology, Tehran, Iran

Abstract

One of the existing problems of multi-attribute process monitoring is the occurrence of high number of false alarms (Type I error). Another problem is an increase in the probability of not detecting defects when the process is monitored by a set of independent uni-attribute control charts. In this paper, we address both of these problems and consider monitoring correlated multi-attributes processes following multi-binomial distributions using two artificial neural network based models. In these processes, out-of-control observations are due to assignable causes coming from some shifts on the mean vector of the proportion nonconforming of the attributes. Model one, which is designed for positively correlated attributes, consists of three neural networks. The first network not only detects whether the process is out-of-control, but also determines the direction of shifts in the attribute means. In this situation, the second and the third networks diagnose the process attribute/s that has/have caused the out-of-control signal due to increase or decrease in proportion nonconforming, respectively. Model two is designed for negatively correlated attributes and consists of two neural networks. The first network is designed to detect whether the process is out-of-control and the second one diagnoses the attribute/s that make/s the signal. The results of five simulation studies on the performance of the proposed methodology are encouraging.

Keywords: Neural Networks; Monitoring; Multi-attribute; Quality control

1. Introduction and literature review

Monitoring is an effort to reduce variability and improve processes in industries. In many manufacturing or service companies, some characteristics of processes cannot be measured numerically, but may be classified as “conforming” and “nonconforming.” These characteristics are called attributes and the monitoring task becomes difficult when there are more than one attribute of interest, especially when some correlation exist between them.

There exist many possible applications of multi-attribute monitoring in service or industries, in which there are more than one correlated defect-type in the product. As an example, in an electronic-board production company the numbers of weakly soldered and over-soldered bases in a board are two negatively correlated quality attributes that must be monitored simultaneously. In another example, the numbers of leak points on either the side seam or the bottom panels of 100 orange juice concentrate cans are two positively correlated quality attributes. In a service industry, the numbers of employee paychecks that are erroneous or distributed late during a pay period are usually two negatively correlated quality attributes that are of interest to monitor simultaneously. Moreover, in many multi-variate processes, in order to reduce the inspection costs, we may be interested in classifying the variables with go and no-go gauges as conforming or nonconforming attributes. In these cases, if there is a correlation between the quality variables it will project itself in the classified attributes; justifying the applications of multi-attribute control charts.

Although multi-variate statistical process control is receiving an increasing attention in the literature, lit-
little work has been done to deal with multi-attribute processes. Patel [26] considered time dependent and time independent observations following multi-variate Poisson and multi-variate binomial distributions, respectively. Assuming normality for the distribution of the observations by large sample sizes and fixed covariance structure, he proposed a $G$ statistic, approximately distributed as $\chi^2$ distribution, as the base of his proposed control chart. Lu et al. [19] presented a multi-variate $np$ control chart ($MNP$) based on the $X$ statistic defined as the weighted sum of the non-conforming units. They also proposed a score statistic ($Z$) to recognize the out-of-control signal and showed that the $MNP$ chart was more sensitive than a set of uni-attribute $np$ charts. The limitation of this technique was the dependencies between the sample sizes and the value of non-conforming items of all process attributes. Jolayemi [17] addressed an economic design of multi-attribute control chart based on $J$ approximation (Larpkiattaworn [18] and Jolayemi [16]) and Gibra ([11] and [12]) model, who assumed that the assignable causes are independent and that they do not overlap each other. When the proportions of nonconformities in each quality category were known or estimated using a base period, Marcucci [20] used a Multi-Nomial distribution to develop a control chart; but since not all multi-attribute processes follow Multi-Nomial distribution, this method is not always applicable. Garde and Rattihalli [10] by assumption of Multi-Nomial distribution for multi-attribute processes used a $MP$-test to determine if the parameters of the distribution changed. In their method to do the $MP$-test, the magnitude of the parameters of interest must be defined in advance.

Decisions in the design of multi-attribute process monitoring due to its complexity cannot be easily attained with classical statistical process monitoring. For this reason, the research studies in this field are progressively directed toward the use of new approaches and methods developed in some areas of the Artificial Intelligent (AI) world like Artificial Neural Networks (ANN) and fuzzy logic.

Many quality engineers and researchers are familiar with the successful applications of Artificial Neural Networks, which have been applied to statistical process control (SPC) since late 1980s in monitoring univariate and multi-variate processes (see Pugh [28]. Cheng [6], Sakai et al. [30], and Niaki and Abbasi [22]).

One of the main reasons for the applications of ANN to SPC is to automate SPC charts interpretation. However, while most of these applications have focused primarily on univariate control charts, little effort has been devoted to multi-variate control charts. Pugh [29] proposed a back-propagation neural network for detecting univariate process mean shifts. Cheng [5] studied performance comparison between artificial neural network and Shewhart-CUSUM scheme in detecting unnatural patterns of a process using a multi-layer perceptron (MLP). Guo & Dooley [13] recommended network models that can identify positive changes in the process mean and variance. While Hwarng and Hubele [15] developed back-propagation networks to identify unnatural patterns on Shewhart $X$ control charts, Chang and Aw [3] proposed a neural-fuzzy network to detect and classify mean shifts. Moreover, Chang & Ho [4] expanded neural network models to detect and classify the magnitude of variance shifts and also developed a combined neural networks control scheme for monitoring mean and variance shifts with a combination of Chang & Ho [4] and Chang & Aw [3] models.

Artificial neural networks have also been applied to control and monitor multi-variate processes. Martin & Morris [21] proposed a fuzzy neural network as an alternative approach for identifying out-of-control causes in a multi-variate process. Wilson et al. [31] used applied Radial Basis Function (RBF) network in a multi-variate process. Niaki & Abbasi [22] used a multi-layer perceptron to pattern classification when there was a signal in multi-variate control charts.

In multi-variate process monitoring there are two broad research categories:

1. Monitoring the proportions of several correlated nonconformities
2. Controlling the number of several correlated defect-types

While it is common practice to assume a multi-variate binomial distribution for the first category, in the second category we usually assume a multi-variate Poisson distribution.

Larpkiattaworn [18] proposed a neural network to monitor processes with two positively correlated attributes and compared its performance with the ones from the $MNP$ and multi-variate normal approximation techniques. He assumed that the variance-covariance matrix associated with the process attributes was unchanged from process to process. He employed the design of experiments technique and presented three scenarios illustrating decision-making rules to select the best method to be used in a particular process. In a more recent research in this area, Niaki and Abbasi [23] based on a simple approach that almost eliminates the existing correlations be-
between the attributes, presented a methodology to monitor multi-attribute processes, and presented a rectangular region for the monitoring purposes. Moreover, Niaki and Abbasi [24] developed a methodology to derive control limits on the attributes based on the bootstrap method in which they build simultaneous confidence intervals on the attributes. Then, based upon the in-control and out-of-control average run length criteria they investigated the performance of the proposed method using simulation.

This paper describes the applicability of ANN approach in multi-attribute process monitoring. This tool, in fact, introduces an innovative approach, which is fundamentally based on a knowledge not directly visible by the user. However, it is able to be stored through a simpler and more intuitive training process. We propose two neural network based models to both monitor and diagnose multi-attribute processes in which there exist more than one correlated process attributes. These models consist of two and three neural networks, respectively. The first network in both models investigates whether the process is in in-control or out-of-control condition. The second and the third network in model 1 and the second network in model 2 show the status of the attributes.

Although it is correct that the use of the control charting method is easy, there is not any control charting methods available in the literature that counts for the possible correlation between the attributes in multi-attribute processes. The one that exists uses the approximate multi-normal probability distribution, which may not be appropriate. Besides, the proposed neural network method not only is designed to detect mean shifts, but also it is capable of diagnosing the attributes that have been shifted. The latter case is an important contribution of the proposed methodology in process monitoring. It is true that it may require more effort to train the network, but once it is trained, it can easily be applied in practice. The practitioners may view the method as a black box such that when they input the values of the attributes they can see if the process is out-of-control, and if yes, which attribute(s) is in out-of-control condition.

In Section 2, we briefly introduce the neural network modeling and their training processes. The data generation and pre-processing comes in section three. Section 4 contains the neural network modeling of the multi-attribute quality control problem, the proposed network architecture, and its training process. The performance evaluation by simulation technique comes in Section 5. Finally, conclusion and recommendations for future research is presented in Section 6.

2. Neural Networks

Artificial Neural Networks (ANN) modeling is an optimization tool for the output processes (responses). They mimic biological neural networks to model and solve a variety of problems arising in prediction or forecasting, function approximation, pattern classification, clustering, and categorization. A neural network, which consists of a number of interconnected nodes called neurons, plays like a computational algorithm for information processing (Pao [25] and Sakai et al. [30]).

In pattern classification application of the ANN, we usually assign an observation to a pre-identified pattern. There are two stages in pattern classification, namely, characteristics identification stage and classification stage. In the former, we select the basic characteristics of a pattern that distinguishes it from the others. We will later use them as a criterion for decision-making process. In the latter, we design a classification machine such that it takes the characteristics as input and produces patterns as output.

As many researchers have proposed several classification machines so far, there are many topologies of the ANN in pattern classification. Studies show that the Multilayer Perceptron (MLP) network with error back propagation has better performance than that of traditional statistical classification methods (Sakai et al. [30]). MLP is a neural network that does nonlinear classification and has three layers: the input layer, the output layer, and the hidden layer. Input layer distributes input to all neuron in the hidden layer, which contains the sigmoid transfer function. The task of the output layer is to determine the pattern (Pao [25]).

Design and implementation of an ANN is fulfilled in three steps. In the first phase, we train the network using a set of inputs and desired outputs (known as training set). After training, we employ another set of inputs and outputs to check the validity of the model. If the obtained error of the validity specification phase is acceptable, the implementation phase will begin.

The network training process is an important task that needs to be addressed before network implementation. There are two general types of training process, namely, the supervised and the non-supervised process. In the supervised training process, in which the user plays an important role as the network learns, a set of training inputs with a corresponding set of targets are given and the network is trained to adjust the weights such that a measure of performance is satisfied. However, in a non-supervised process, which is of two types (reinforcement learning and unsuper-
vised learning), a set of inputs without any target is available and the network is trained by letting it continually adjust itself to new inputs and finding relationships within data.

2.1. Multi-Layer Perceptron and Back Propagation neural networks

Perceptron Neural Network (PNN) is perhaps the most popular network architecture in use today and is discussed at length in most neural network textbooks (e.g., Bishop [1]). An important class of the PNN is multi-layer feed forward perceptron (MLP) network. In this type of network, we arrange the units in a layered feed forward topology, where the units each perform a biased weighted sum of their inputs and pass this activation level through a transfer function to produce their output. The network therefore has a simple interpretation as a form of input-output model, with the weights and thresholds (biases) as the free parameters of the model.

In order to train a MLP network with error back propagation (BP), in the first step one needs to generate sufficient data containing all of the classified patterns. In the second step, one uses the data as input to the network and compares the output of the network with a pre-specified target. Then, in the third step, based upon some performance criteria, such as the mean squared error; the error being the difference between the target value and the output, the error back propagation algorithm modifies the weights (W’s) and the threshold values or biases (b’s). The training process goes to the next step and so on until either an acceptable value for the mean squared error is achieved or a pre-specified number of cycles (epochs) is reached. In other words, Error backpropagation learning consists of two calculating passes, feed-forward and feed-back, through the different layers of the network. In the forward pass, an activity pattern (input vector) is applied to the sensory nodes of the network and its effect propagates through the network layer by layer to produce a set of output as the actual response of the network, finally. During the forward pass, the synaptic weights of the networks are all fixed and in the backward pass the weights are adjusted in accordance with an error-correction rule. This error is then propagated backwards through the network to make outputs close to the targets (Haykin [14] and Chowdary [7]). Backpropagation networks are often applied in pattern recognition and classification problems. Although the number of input and output units is defined by the problem, the number of hidden units to use is far from clear. A good starting point is to use one hidden layer and trade the number of units in it. For a good reference on the details of the feed-forward and the feedback equations of the BP algorithm, we may refer to Haykin [14].

Figure 1 shows a typical structure of a MLP with \( k \) neurons in its input layer, one hidden layer with one neuron and \( m \) neurons in the output layer, in which learning is supervised and error back propagation algorithm is used. Such networks can model functions of almost arbitrary complexity, with the number of layers and units in each layer, determining the function’s complexity. Important issues in MLP design include specification of both the number of hidden layers and the number of units in these layers (see Haykin [14] and Bishop [1]).

MLP networks have successfully solved some difficult problems with a supervised error backpropagation training algorithm (see Patterson [27], Haykin [14], Fausett [9], and Chowdary [7]). Modern second-order algorithms such as conjugate gradient descent and Levenberg-Marquardt (see Bishop [1]) are substantially faster for many problems; however, back propagation is easier to understand and still has advantages in some circumstances.

2.2. Resilient backpropagation

Multi-layer networks typically use sigmoid transfer functions in their hidden layers. These functions compress an infinite input range into a finite output range. Sigmoid functions are characterized by the fact that their slopes must approach zero, as the input becomes large. This causes a problem when we use the steepest descent to train a multi-layer network with sigmoid functions, because the gradient can have a very small magnitude and, therefore, cause small changes in the weights and biases, even though they are far from their optimal values.
The purpose of the resilient backpropagation training algorithm is to eliminate the harmful effects of the partial derivatives’ magnitudes. Only the sign of the derivative is used to determine the direction of the weight update; the magnitude of the derivative has no effect on it.

The size of the weight change is determined by a separate update value. Whenever the derivative of the performance function with respect to that weight has the same sign for two successive iterations, the update value for each weight and bias is increased by a factor. However, it is decreased by a factor whenever the derivative with respect to that weight changes sign from the previous iteration. If the derivative is zero, then the update value remains the same. Whenever the weights are oscillating, the weight change is reduced. If the weight continues to change in the same direction for several iterations, then the magnitude of the weight change increases (Demuth et al. [8]). We have chosen to use this algorithm for this research.

3. Data generation and pre-processing

In order to train a neural network, a set of proper data along with its target values is required. To generate data from say a $\mathbb{R}^d$ valued random vector $X$ with marginal cumulative distribution functions

$$F_i(.) = P(X_i \leq .), \quad i = 1, ..., d$$

and correlation matrix

$$\Sigma_X = (\sigma_X(i, j) : 1 \leq i, j \leq d).$$

Cario and Nelson [2] presented the NORTA procedure based on the following transformation function:

$$X = \begin{bmatrix} F_{x_1}^{-1}[\Phi(Z_1)] ,& F_{x_2}^{-1}[\Phi(Z_2)] ,& ... ,& F_{x_d}^{-1}[\Phi(Z_d)] \end{bmatrix}^T$$

(3)

where $Z = (Z_1, Z_2, ..., Z_d)^T$ is a multi-normal vector with correlation matrix $\Sigma_Z$, $\Phi(.)$ is the cumulative probability distribution function of a standard normal random variable, and $F_{X_1}, F_{X_2}, ..., F_{X_d}$ are the desired marginal distributions. The correlation matrix ($\Sigma_X$) is set to ensure that $X$ will have prescribed correlation matrix of desirable data and the transformation technique ensures that $X_i$ has the desired marginal distribution $F_{X_i}$.

In other words, the NORTA procedure to generate random deviates from a $d$-dimensional random vector $X$ takes the following steps:

1) Generate an $\mathbb{R}^d$ standard multi-variate normal random vector $Z = (Z_1, ..., Z_d)$ with correlation matrix $\Sigma_Z$.

2) Compute the vector $X = (X_1, X_2, ..., X_d)^T$ where $X_i = F_{x_i}^{-1}(\Phi(Z_i))$; $i = 1, 2, ..., d$ and $\Phi(.)$ is the cumulative probability distribution function of a standard normal random variable. Then $F_{x_i}^{-1}(u) = \inf \{ x : F_i(x) \geq u \}$.

In this research, we assume the marginal probability distribution of the correlated quality attributes to be binomial.

3.1. Data preprocessing

In order to improve the efficiency of the training process, the generated input data are usually preprocessed. One of these preprocessing methods is scaling the input data to fall within the interval of [−1,1] by the following equation:

$$pn = \frac{2[p - \min(p)]}{\max(p) - \min(p)} - 1,$$

(4)

where $p$ is the input matrix, $pn$ is the scaled input matrix, $\min(p)$ is the minimum input vector, and $\max(p)$ is the maximum one. Then, the un-scaling task may be performed using the following equation:

$$p = 0.5(pn + 1)[\max(p) - \min(p)] + \min(p).$$

(5)

4. Neural Network modeling of a multi-attribute quality control process

Although ANNs have been introduced for several decades, their use in multi-attribute process monitoring area is quite recent and their applications are still very few. In this research, in order to monitor a multi-attribute quality control process and diagnose the shifts in the attribute means we propose two models. The first is designed for situations in which the attributes are positively correlated and have a common direction in the mean shifts. Otherwise, the second model is suggested for negatively correlated attributes.
In this study, multi-layer backpropagation perceptron networks are chosen. The number of input nodes in all architectures represents the number of characteristics of the process under consideration. The only node in the output layer of the first network represents the status of the process (in-control or out-of-control). In the second and the third network, the output nodes represent the status of the attributes under study; hence, the number of output nodes in these networks is equal to the number of attributes. The default number of the layers in the hidden layer is set to be two (Haykin [14]). However, by a trial-and-error method, in some instances in the training process one layer seemed to be good enough. Figure 2 shows the proposed neural networks to monitor correlated multi-attributes processes.

In order to monitor a multi-attribute quality control process, first the preprocessed observations are applied to the network 1. This network is trained such that when the process is in-control, the output becomes very close to “0”. However, if the process is out-of-control and at least one attribute-mean has positive shift, it returns a value close to “1”, and finally the output is almost “-1” if the process is out-of-control and at least one attribute-mean has a negative shift. Then, networks 2 and 3 determine which attribute mean(s) has/have a positive or a negative shift, respectively.

In processes in which some attributes have positive and some have negative mean-shifts, as in cases of negatively correlated attributes; model 1 is unable to detect the shifts. In this situation, the observations are applied to model 2, which consists of two networks. The first network returns almost “0” and “1” when the process is in-control or out-of-control, respectively.

The second one determines the attribute(s) causing out-of-control signal. In other words, the output of this network shows the status of the attributes. For example an output vector of approximately [1 -1 0] correspond to a situation in which the first attribute has positive mean-shift, the second one has negative mean-shift, and the third one is under control.

Figure 3 shows the proposed neural networks (model 2) to monitor negatively correlated multi-attributes processes.

![Figure 2](image-url)

**Figure 2.** The neural network model to monitor and diagnose positively correlated multi-attribute processes (Model 1).

![Figure 3](image-url)

**Figure 3.** The neural network model to monitor and diagnose negatively correlated multi-attributes processes (Model 2).
4.1. Network training process

In this research, the resilient back-propagation-training algorithm introduced in Section 2.2 is chosen for the training process. Not only this algorithm is generally much faster than the standard algorithms based on the steepest descent methods, it also requires a modest amount of memory. Moreover, it has shown good performances in pattern recognition problems (Demuth et al. [8]).

The resilient backpropagation algorithm is not very sensitive to the setting of the training parameters, so some of the networks parameters such as the number of epochs and the network MSE are chosen randomly for best results. Adaptive learning rate and momentum constant are selected according to a computer experiment that has been conducted in pattern classifier to illustrate the learning behavior of a multi-layer perceptron. These values are 0.1 for the learning-rate and 0.5 for the momentum constant. The transfer function in the hidden layers is selected to be the sigmoid function.

The flow chart of the training process for multi-attribute monitoring shown in Figure 4 has been employed while developing the Resilient Back-Propagation Artificial Neural Networks in Matlab computer package environment. The trained networks for monitoring have been simulated and tested for their validity. While testing the networks, various patterns have been applied to it. The output generated from the neural network models is compared to the desired output.

In the training process of the first network in Model 1, first we tried 200 in-control and 100 out-of-control patterns for processes with two and three attributes. An Out-of-control pattern is a set of observations with both positive and negative shifts around the process mean. However, the performance of the network in terms of both in-control and out-of-control process mean. While the first network in Model 2 was trained by 2000 in-control and 200 out-of-control patterns, the second network was trained by 800 out-of-control data sets on two attributes. Out-of-control patterns in both networks are a set of observations with both positive and negative shifts around the process mean.

In all training processes, we estimated the variance-covariance matrix of the attribute means assuming changes in the variances and no changes in the correlations and reached the MSE of approximately $10^{-4}$ by trial and error.

5. Performance evaluation

In Model 1 and 2, the performance of the network 1 is measured by in-control and out-of-control average run length (ARL). An in-control average run length (ARL$_i$) is the average number of samples that must be taken before a sample indicates an out-of-control condition when, in fact, the process is in control. An out-of-control average run length (ARL$_t$) is the average number of samples taken to detect a shift in the mean of a process, when the process is in a particular out-of-control condition. For network 2 in Model 1 and 2 and network 3 in Model 1, the percent of the times the networks detect the true out-of-control attribute(s) is selected as a measure to evaluate their performance in the following simulation experiments.

5.1. Simulation experiment 1 (Model 1)

In the first simulation experiment we investigate a Multi-Binomial distribution with two positive correlated attributes and parameters as $\rho = 0.2$, $\mu = 20$, $\sigma = 0.25$, $p_1 = 0.15$, and $p_2 = 30$. Tables 1, 2 and 3 show the performance of network 1, 2, and 3 of Model 1, respectively.

In these tables and the ones for the other simulation experiments, there are empty cells referring to situations in which their corresponding data could not be observed from a process. For example in Table 1 the data following a Multi-Binomial distribution whose attribute means are shifted by $-3\sigma$ could not be generated because their nonconforming proportions become negative. The $-2\sigma$ and $-3\sigma$ rows in the Table 12 have the same description as mentioned. In addition, the entries in these tables are estimated from 10000 replications.

The results of Tables 1, 2 and 3 show that in this simulation experiment the networks possess good performances.
Figure 4. The flow chart of the training process.
Table 1. The performance of network 1 in simulation experiment 1.

<table>
<thead>
<tr>
<th>Shift</th>
<th>-3σ₁</th>
<th>-2σ₁</th>
<th>-σ₁</th>
<th>No-shift</th>
<th>+σ₁</th>
<th>+2σ₁</th>
<th>+3σ₁</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARL</td>
<td>-</td>
<td>2.47</td>
<td>68.49</td>
<td>312</td>
<td>16.42</td>
<td>3.33</td>
<td>1.4</td>
</tr>
</tbody>
</table>

Table 2. The performance of network 2 in simulation experiment 1.

<table>
<thead>
<tr>
<th>Shift</th>
<th>-σ₂</th>
<th>+σ₁</th>
<th>+2σ₁</th>
<th>-σ₂</th>
<th>+2σ₁</th>
<th>+3σ₁</th>
</tr>
</thead>
<tbody>
<tr>
<td>Correct detection</td>
<td>0.49</td>
<td>0.53</td>
<td>0.64</td>
<td>0.74</td>
<td>0.71</td>
<td>0.81</td>
</tr>
</tbody>
</table>

Table 3. The performance of network 3 in simulation experiment 1.

<table>
<thead>
<tr>
<th>Shift</th>
<th>-σ₁</th>
<th>-σ₂</th>
<th>-2σ₁</th>
<th>-σ₁</th>
<th>-2σ₁</th>
<th>-σ₁</th>
</tr>
</thead>
<tbody>
<tr>
<td>Correct detection</td>
<td>0.59</td>
<td>0.57</td>
<td>0.65</td>
<td>0.78</td>
<td>0.80</td>
<td>0.98</td>
</tr>
</tbody>
</table>

Table 4. The performance of network 1 in simulation experiment 2.

<table>
<thead>
<tr>
<th>Shift</th>
<th>-3σ₁</th>
<th>-2σ₁</th>
<th>-σ₁</th>
<th>No-shift</th>
<th>+σ₁</th>
<th>+2σ₁</th>
<th>+3σ₁</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARL</td>
<td>1.02</td>
<td>1.78</td>
<td>27.15</td>
<td>338</td>
<td>10.00</td>
<td>1.78</td>
<td>1.06</td>
</tr>
</tbody>
</table>

Table 5. The performance of network 2 in simulation experiment 2.

<table>
<thead>
<tr>
<th>Shift</th>
<th>-σ₁</th>
<th>-σ₂</th>
<th>-2σ₁</th>
<th>-σ₁</th>
<th>-2σ₁</th>
<th>-σ₁</th>
</tr>
</thead>
<tbody>
<tr>
<td>Correct detection</td>
<td>0.48</td>
<td>0.48</td>
<td>0.63</td>
<td>0.55</td>
<td>0.57</td>
<td>0.90</td>
</tr>
</tbody>
</table>

Table 6. The performance of network 3 in simulation experiment 2.

<table>
<thead>
<tr>
<th>Shift</th>
<th>-σ₁</th>
<th>-σ₂</th>
<th>-2σ₁</th>
<th>-σ₁</th>
<th>-2σ₁</th>
<th>-σ₁</th>
</tr>
</thead>
<tbody>
<tr>
<td>Correct detection</td>
<td>0.47</td>
<td>0.49</td>
<td>0.71</td>
<td>0.56</td>
<td>0.62</td>
<td>0.97</td>
</tr>
</tbody>
</table>
Table 7. The performance of network 1 in simulation experiment 3.

<table>
<thead>
<tr>
<th>Shift</th>
<th>-3σ₁</th>
<th>-2σ₁</th>
<th>-σ₁</th>
<th>σ₁</th>
<th>2σ₁</th>
<th>3σ₁</th>
<th>No-shift</th>
<th>+σ₁</th>
<th>2σ₁</th>
<th>3σ₁</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARL</td>
<td>1.44</td>
<td>2.01</td>
<td>227.27</td>
<td>12.99</td>
<td>2.43</td>
<td>1.24</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 8. The performance of network 2 in simulation experiment 3.

<table>
<thead>
<tr>
<th>Shift</th>
<th>-3σ₁</th>
<th>-2σ₁</th>
<th>-σ₁</th>
<th>σ₁</th>
<th>2σ₁</th>
<th>3σ₁</th>
<th>+2σ₁</th>
<th>+2σ₁</th>
<th>+2σ₁</th>
</tr>
</thead>
<tbody>
<tr>
<td>Correct detection</td>
<td>0.28</td>
<td>0.29</td>
<td>0.27</td>
<td>0.39</td>
<td>0.42</td>
<td>0.45</td>
<td>0.23</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Shift</th>
<th>-3σ₁</th>
<th>-2σ₁</th>
<th>-σ₁</th>
<th>σ₁</th>
<th>2σ₁</th>
<th>3σ₁</th>
<th>+2σ₁</th>
<th>+2σ₁</th>
<th>+2σ₁</th>
</tr>
</thead>
<tbody>
<tr>
<td>Correct detection</td>
<td>0.44</td>
<td>0.41</td>
<td>0.39</td>
<td>0.63</td>
<td>0.65</td>
<td>0.64</td>
<td>0.44</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Shift</th>
<th>-3σ₁</th>
<th>-2σ₁</th>
<th>-σ₁</th>
<th>σ₁</th>
<th>2σ₁</th>
<th>3σ₁</th>
<th>+3σ₁</th>
<th>+3σ₁</th>
<th>+3σ₁</th>
</tr>
</thead>
<tbody>
<tr>
<td>Correct detection</td>
<td>0.45</td>
<td>0.61</td>
<td>0.50</td>
<td>0.73</td>
<td>0.83</td>
<td>0.71</td>
<td>0.63</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 9. The performance of network 3 in simulation experiment 3.

<table>
<thead>
<tr>
<th>Shift</th>
<th>-3σ₁</th>
<th>-2σ₁</th>
<th>-σ₁</th>
<th>σ₁</th>
<th>2σ₁</th>
<th>3σ₁</th>
<th>-3σ₁</th>
<th>-3σ₁</th>
<th>-3σ₁</th>
</tr>
</thead>
<tbody>
<tr>
<td>Correct detection</td>
<td>0.34</td>
<td>0.38</td>
<td>0.32</td>
<td>0.52</td>
<td>0.51</td>
<td>0.42</td>
<td>0.33</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Shift</th>
<th>-3σ₁</th>
<th>-2σ₁</th>
<th>-σ₁</th>
<th>σ₁</th>
<th>2σ₁</th>
<th>3σ₁</th>
<th>-2σ₁</th>
<th>-2σ₁</th>
<th>-2σ₁</th>
</tr>
</thead>
<tbody>
<tr>
<td>Correct detection</td>
<td>0.55</td>
<td>0.63</td>
<td>0.42</td>
<td>0.81</td>
<td>0.82</td>
<td>0.76</td>
<td>0.93</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 10. The performance of network 1 in simulation experiment 4.

<table>
<thead>
<tr>
<th>Shift</th>
<th>$-3\sigma_1$</th>
<th>$-2\sigma_1$</th>
<th>$-\sigma_1$</th>
<th>$+\sigma_1$</th>
<th>$+2\sigma_1$</th>
<th>$+3\sigma_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$-3\sigma_2$</td>
<td>$-2\sigma_2$</td>
<td>$-\sigma_2$</td>
<td>$+\sigma_2$</td>
<td>$+2\sigma_2$</td>
<td>$+3\sigma_2$</td>
</tr>
<tr>
<td></td>
<td>$-3\sigma_3$</td>
<td>$-2\sigma_3$</td>
<td>$-\sigma_3$</td>
<td>$+\sigma_3$</td>
<td>$+2\sigma_3$</td>
<td>$+3\sigma_3$</td>
</tr>
</tbody>
</table>

| ARL   | 13.23        | 332.23       | 9.2          | 1.8          | 1.13         |

Table 11. The performance of network 2 in simulation experiment 4.

<table>
<thead>
<tr>
<th>Shift</th>
<th>-</th>
<th>-</th>
<th>+\sigma_1</th>
<th>+\sigma_2</th>
<th>+\sigma_3</th>
<th>+\sigma_4</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>-</td>
<td>+\sigma_1</td>
<td>-</td>
<td>+\sigma_2</td>
<td>-</td>
<td>+\sigma_3</td>
</tr>
<tr>
<td></td>
<td>+\sigma_3</td>
<td>-</td>
<td>-</td>
<td>+\sigma_3</td>
<td>+\sigma_3</td>
<td>+\sigma_3</td>
</tr>
</tbody>
</table>

| Correct detection | 0.43 | 0.22 | 0.46 | 0.49 | 0.27 | 0.32 | 0.30 |

<table>
<thead>
<tr>
<th>Shift</th>
<th>-</th>
<th>-</th>
<th>+2\sigma_1</th>
<th>+2\sigma_2</th>
<th>+2\sigma_3</th>
<th>+2\sigma_4</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>-</td>
<td>+2\sigma_2</td>
<td>-</td>
<td>+2\sigma_2</td>
<td>-</td>
<td>+2\sigma_3</td>
</tr>
<tr>
<td></td>
<td>+2\sigma_3</td>
<td>-</td>
<td>-</td>
<td>+2\sigma_3</td>
<td>+2\sigma_3</td>
<td>+2\sigma_3</td>
</tr>
</tbody>
</table>

| Correct detection | 0.57 | 0.29 | 0.72 | 0.79 | 0.36 | 0.51 | 0.48 |

<table>
<thead>
<tr>
<th>Shift</th>
<th>-</th>
<th>-</th>
<th>+3\sigma_1</th>
<th>+3\sigma_2</th>
<th>+3\sigma_3</th>
<th>+3\sigma_4</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>-</td>
<td>+3\sigma_2</td>
<td>-</td>
<td>+3\sigma_2</td>
<td>-</td>
<td>+3\sigma_3</td>
</tr>
<tr>
<td></td>
<td>+3\sigma_3</td>
<td>-</td>
<td>-</td>
<td>+3\sigma_3</td>
<td>+3\sigma_3</td>
<td>+3\sigma_3</td>
</tr>
</tbody>
</table>

| Correct detection | 0.69 | 0.29 | 0.88 | 0.88 | 0.41 | 0.71 | 0.59 |
Table 12. The performance of network 3 in simulation experiment 4.

<table>
<thead>
<tr>
<th>Shift</th>
<th>$-\sigma_1$</th>
<th>$+\sigma_1$</th>
<th>$-\sigma_2$</th>
<th>$+\sigma_2$</th>
<th>$-\sigma_3$</th>
<th>$+\sigma_3$</th>
<th>$-2\sigma_1$</th>
<th>$+2\sigma_1$</th>
<th>$-2\sigma_2$</th>
<th>$+2\sigma_2$</th>
<th>$-2\sigma_3$</th>
<th>$+2\sigma_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Correct detection</td>
<td>0.48</td>
<td>0.41</td>
<td>0.40</td>
<td>0.67</td>
<td>0.49</td>
<td>0.56</td>
<td>0.51</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 13. The performance of network 1 in simulation experiment 5.

<table>
<thead>
<tr>
<th>Shift</th>
<th>$-\sigma_1$</th>
<th>$+\sigma_1$</th>
<th>$-\sigma_2$</th>
<th>$+\sigma_2$</th>
<th>$-\sigma_3$</th>
<th>$+\sigma_3$</th>
<th>$-2\sigma_1$</th>
<th>$+2\sigma_1$</th>
<th>$-2\sigma_2$</th>
<th>$+2\sigma_2$</th>
<th>$-2\sigma_3$</th>
<th>$+2\sigma_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARL</td>
<td>32.78</td>
<td>38.76</td>
<td>17.66</td>
<td>12.97</td>
<td>2.28</td>
<td>5.01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Shift</td>
<td>$-2\sigma_1$</td>
<td>$+2\sigma_1$</td>
<td>$-3\sigma_1$</td>
<td>$+3\sigma_1$</td>
<td>$-3\sigma_2$</td>
<td>$+3\sigma_2$</td>
<td>$-3\sigma_3$</td>
<td>$+3\sigma_3$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ARL</td>
<td>4.20</td>
<td>2.79</td>
<td>1.11</td>
<td>1.73</td>
<td>1.37</td>
<td>1.20</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Shift</td>
<td>No-shift</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ARL</td>
<td>271.00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 14. The performance of network 2 in simulation experiment 5.

<table>
<thead>
<tr>
<th>Shift</th>
<th>$-\sigma_1$</th>
<th>$+\sigma_1$</th>
<th>$-\sigma_2$</th>
<th>$+\sigma_2$</th>
<th>$-\sigma_3$</th>
<th>$+\sigma_3$</th>
<th>$-2\sigma_1$</th>
<th>$+2\sigma_1$</th>
<th>$-2\sigma_2$</th>
<th>$+2\sigma_2$</th>
<th>$-2\sigma_3$</th>
<th>$+2\sigma_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Correct detection</td>
<td>0.42</td>
<td>0.38</td>
<td>0.30</td>
<td>0.44</td>
<td>0.40</td>
<td>0.28</td>
<td>0.37</td>
<td>0.38</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Shift</td>
<td>$-2\sigma_1$</td>
<td>$-2\sigma_2$</td>
<td>$-2\sigma_3$</td>
<td>$+2\sigma_3$</td>
<td>$-2\sigma_2$</td>
<td>$+2\sigma_2$</td>
<td>$-2\sigma_3$</td>
<td>$+2\sigma_3$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Correct detection</td>
<td>0.94</td>
<td>0.55</td>
<td>0.46</td>
<td>0.90</td>
<td>0.79</td>
<td>0.39</td>
<td>0.49</td>
<td>0.78</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Shift</td>
<td>$-3\sigma_1$</td>
<td>$-3\sigma_2$</td>
<td>$-3\sigma_3$</td>
<td>$+3\sigma_3$</td>
<td>$-3\sigma_2$</td>
<td>$+3\sigma_2$</td>
<td>$-3\sigma_3$</td>
<td>$+3\sigma_3$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Correct detection</td>
<td>-</td>
<td>0.64</td>
<td>-</td>
<td>-</td>
<td>0.94</td>
<td>0.44</td>
<td>0.52</td>
<td>0.97</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
5.2. Simulation experiment 2 (Model 1)

In this example, we consider a Multi-Binomial distribution with two negative correlated attributes and parameters of \( p_1 = 0.4, n_1 = 25, \rho = -0.16, p_2 = 0.2, \) and \( n_2 = 25. \) Tables 4, 5 and 6 show the performance of network 1, 2, and 3 in this experiment, respectively. Once again relatively good performances are observed in this experiment.

\[
\begin{align*}
\rho &= \begin{pmatrix}
1.00 & 0.19 & 0.14 \\
0.19 & 1.00 & 0.24 \\
0.14 & 0.24 & 1.00
\end{pmatrix}, \\
p_1 &= 0.18, \quad n_1 = 30 \\
p_2 &= 0.15, \quad n_2 = 30
\end{align*}
\]

studied in this experiment.

Tables 7, 8 and 9 show the performance of network 1, 2, and 3 in this experiment, respectively. Again good performances are observed. However, in terms of the probability of correct detection, sometimes the performances are not very good.

5.3. Simulation experiment 3 (Model 1)

Three positive correlated attributes with Multi-Binomial distribution and the parameters \( p_1 = 0.1, n_1 = 30, \) \( p_2 = 0.15, n_2 = 30, \) \( p_3 = 0.18, \)

\[
\rho = \begin{pmatrix}
1.00 & 0.12 & 0.57 \\
-0.12 & 1.00 & -0.19 \\
0.57 & -0.19 & 1.00
\end{pmatrix}, \\
p_1 &= 0.11, \quad n_1 = 22 \\
p_2 &= 0.12, \quad n_2 = 22 \\
p_3 &= 0.16, \quad n_3 = 22
\]

Although, Table 10 shows good performances of network 1 in terms of the in and out-of-control average run length, the results of networks 2 and 3 in Tables 11 and 12 show that sometimes these networks do not perform very well in terms of the probability of correct detection.

5.5. Simulation experiment 5 (Model 2)

In this simulation experiment we employ model 2 on the data obtained for the second simulation experiment in which there are two negatively correlated attributes and parameters \( p_1 = 0.4, n_1 = 25, \rho = -0.16, p_2 = 0.2, \) and \( n_2 = 25. \)

Tables 13 and 14 show the performances of network 1 and 2 in this experiment, respectively. The results of Table 5.1 show that the designed network works relatively well in terms of in and out-of-control average run lengths. However, from Table 14 we see that the probability of detecting the correct attribute(s) is not very good.

6. Conclusion and recommendations for future research

Monitoring processes in which there are several correlated quality attributes is a complex problem. In this paper, we proposed two neural-network-based models with three and two networks, to monitor multi-attributes processes. As the three networks in model 1 were trained using fewer data sets, they were not able to efficiently detect different scenarios of mean shifts in the mean of negatively correlated attributes. As a result, for the diagnosis purpose in negatively correlated multi-attributes processes, we designed and proposed model 2.

The first network in both models was designed to identify the status of the process under consideration; the second and the third network in model 1 were designed to detect the attribute(s) causing positive and negative shifts in the process mean, respectively. We showed that the proposed models possessed a desirable in-control average run length in different simulation experiments and could detect different scenarios of mean-shifts relatively fast. However, in terms of the probability of correct detection, sometimes they do not perform very well. As a result, this study has shown that a properly developed ANN model provides a valid alternative for monitoring multi-attribute quality control problems.

The resilient back propagation artificial neural network models demonstrated in this paper is an innovative approach fundamentally based on AI, which is not directly visible to the user. However, it is able to solve the complex problem through a simpler and supervised feed forward BP training process. The practitioners may view the method as a black box such that when they input the values of the attributes they can see if the process is out-of-control, and if yes, which attribute(s) is in out-of-control condition.

The potential of ANN as a multi-attribute monitoring tool is usually based on a large sample size. The generalization capabilities of ANNs are highly dependent on the number of patterns in the training set.
In term of the probability of correct detection, we saw that in some instances the proposed networks do not perform very well. Hence, there is scope to train and test the resilient BPANN model based on further larger sample size in the future. In addition, it could be interesting to the researchers to compare the performance of ANN approach with other available models.

References


کارگاه‌های آموزشی مرکز اطلاعات علمی

مقاله نویسی علوم انسانی

اصول تنظیم قراردادها

آموزش مهارت های کاربردی در تدوین و چاپ مقاله