Proc. of the European Safety and Reliability Conference (ESREL 2026)
Edited by José C. Matos, Paulo B. Lourenço, Michael Beer, Daniel Oliveira,
Edoardo Patelli, Hélder S. Sousa, Rui A. Silva and Monica Santamaria
 Published by Research Publishing, Singapore.
doi: 10.3850/ESREL2026061419_esrel26-p26126-cd

An active-learning neural network for structural reliability analysis

Henok Tefera Ayele

Mechanical Engineering, University of Electronic Science and Technology of China, Ethiopia. E-mail: henoktefera1998@gmail.com

Hong-Xing Zhan

School of Physics and Electronic Engineering, Qujing Normal University, China. E-mail: myzhx926@163.com

Structural reliability analysis faces persistent challenges due to the computational complexity and time-consuming nature of numerical simulations, particularly when applied to real-world engineering systems with high-dimensional input variables. Reducing the number of function evaluations while maintaining sufficient accuracy remains a critical objective in this field. Although many methods, such as Monte Carlo Simulation (MCS), provide a simple solution, their computational burden makes them unsuitable. To address this issue, methods based on active learning surrogate models have received widespread attention. These surrogate model methods iteratively approximate complex models using machine learning, selecting informative samples to improve the accuracy of the surrogate model. Among existing approaches, Kriging models are widely used because they can provide prediction variance for unsampled points. However, they struggle with high-dimensional input spaces. This paper proposes an Active-Learning Neural Network (ALNN) framework that combines the flexibility of back-propagation neural networks with the adaptive capabilities of active-learning strategies. The proposed method enables the estimation of prediction variance directly within the neural network structure, allowing for dynamic sample selection and efficient model refinement. The ALNN enables adaptive sampling and efficient model updates while handling high-dimensional and multi-output structural systems. The effectiveness and efficiency of the proposed method are validated through numerical examples, demonstrating improved adaptability, reduced computational costs, and enhanced accuracy in structural reliability analysis.

Keywords: Surrogate model, active-learning, learning function, backpropagation, activation function, probability of failure.

1. Introduction

Structural reliability theory has been applied to many engineering problems in the last few decades, with the primary objective of quantifying the safety level of structures. Surrogate models have been widely employed as a useful tool to keep the computational burden acceptable. An active-learning Kriging model for structural reliability analysis is a well-known surrogate model. The main advantage of the Kriging model is that the prediction variance for an unsampled point can be used to guide the selection of subsequent training samples, but the Kriging model is not effective for high-dimensional problems and cannot be applicable for multi-output problems. (Bichon et al. 2008)

To overcome these limitations, this paper proposed an active-learning approach for reliability analysis based on neural networks as a surrogate model. The methodology is employed in the solution of two examples and compared with the MCS and active learning reliability method (AK-MCS). In all cases, the active-learning neural network yields results close to those obtained by MCS and requires fewer limit state function evaluations. Neural network modeling has emerged as a valuable technique in reliability analysis, offering a promising alternative to traditional methods like MCS. By leveraging the capacity of neural networks to approximate complex functions, it can efficiently capture the intricate relationships between inputs and outputs. (Chojaczyk et al. 2015)

This approach entails training a neural network based on a dataset generated by costly simulations, which can effectively learn the underlying characteristics of the system. Once the neural network is trained well, it can serve as a surrogate for the costly model,

enabling rapid and accurate estimation of failure probability. (McCulloch and Pitts 1943)

The use of neural network surrogate modeling in reliability analysis can not only reduce the huge computational burden but also maintain high efficiency. An active-learning neural network (ALNN) in reliability analysis refers to a specialized model capable of actively adjusting its structure and parameters to effectively capture complex relationships between inputs and outputs of the system performance. By continuously updating the architecture of the neural network, ALNN offers a powerful way for reliability engineers to accurately estimate the failure.

2. Proposed Method

The main challenges of many reliability analysis methods for structural systems are the computationally expensive implicit performance functions and multiple failure modes. To overcome these limitations, an active learning neural network method is proposed. Unlike many current methods restricted to Kriging models, this approach leverages back-propagation (BP) neural networks for efficient reliability analysis. By employing BP neural networks, which are capable of dynamically adjusting their structure and parameters based on training data, the proposed method aims to enhance the efficiency and accuracy of reliability analysis. (Hornik et al. 1990)

The proposed method in this paper provides the prediction variance for unsampled points using a neural network, so it can be used in active-learning with the aid of the Uncertainty learning function (U-learning function). This integration of neural networks and active-learning techniques enhances the adaptability and efficiency of the reliability analysis process, allowing for a more informed and targeted selection of training samples to improve model accuracy and reduce computational costs. The method uses four different non-adaptive neural networks built with the same hidden layer activation function but different output layer activation functions, which gives four different results for providing the prediction mean and variance. The method computes both the mean and variance of the unsampled point using the predictions obtained from these four neural networks. After finding mean and variance values, the proposed method employs a U-learning function, which guides the selection of the next sample in the iterative process. Then the proposed stopping criterion, which is the loop termination condition, is used to stop the iteration. There are four main steps in this proposed method, which are as follows:

- Constructing four non-adaptive neural networks
- Finding the mean and variance of the unsampled point
- Use the U-learning function for the identification of the best next point
- The stopping condition (loop termination) is used to stop the process

2.1. *Neural network architecture and Backpropagation*

In this paper, all surrogate models are constructed using fully connected feedforward artificial neural networks (ANNs). The networks consist of multiple hidden layers and are trained using the scaled conjugate gradient backpropagation algorithm (trainscg) with a fixed number of training epochs to ensure consistent convergence behaviour. Four independent non-adaptive neural networks are constructed with the same hidden layer activation function, which is the sigmoid activation function, and different output layer activation functions for each neural network, which gives different values of probability of failure. The output layer activation function for the four neural networks is Linear Function (LA), Sigmoid Function (SA), Hyperbolic Tangent Function (TA), and SoftMax Function (SA). The four neural networks are designed to create and train feedforward neural networks. These functions utilize the Neural Network Toolbox in MATLAB to define network architectures and perform training. The four neural networks are constructed with the same number of hidden layers and the same number of nodes however, The specific number of hidden layers and neurons per layer may vary depending on the complexity of each benchmark problem and is therefore detailed within the corresponding example sections. For training, all functions employ the

$'trainscg'$ algorithm, a scaled conjugate gradient backpropagation method. Figure 1 demonstrates how the non-adaptive neural networks with input, hidden, and output layers are used to calculate the mean and variance of unsampled points.

Activation functions play a crucial role in shaping the behavior and performance of neural networks. One fundamental distinction among activation functions is their effect on the linearity of network computations. Non-linear activation functions enable neural networks to capture intricate relationships and nonlinear patterns present in the data. In contrast, linear activation functions limit the network's capacity to represent complex patterns by imposing a linear transformation on the input. (Sharma et al. 2020)

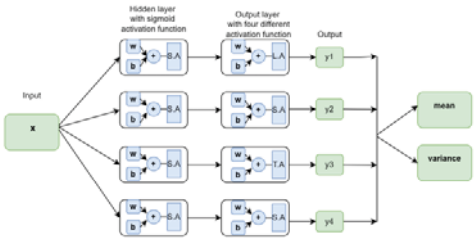

Fig. 1. Demonstration of the four different independent non-adaptive neural networks.

Overall, the selection of activation functions is a critical decision in designing neural networks, with profound implications for their performance and ability to learn complex patterns from data. (Bucher and Most 2008)

2.2.*Proposed prediction uncertainty*

The AK-MCS method leverages the capabilities of Kriging models to deliver mean and prediction variances. By incorporating the Kriging models into the Monte Carlo Simulation framework, AK-MCS can efficiently estimate the failure probability utilizing the prediction mean and variance of the Kriging model. (Echard et al. 2011)

The proposed method introduces an approach by replacing Kriging model with neural networks to address the limitations associated with the Kriging model, particularly for high-dimensional problems. By utilizing neural networks, the method aims to enhance the modeling capability and flexibility required to effectively handle complex and high-dimensional data. Four neural networks with identical architecture and training settings are constructed for each case. The only distinction among these networks lies in the output-layer activation function. This controlled variation in output nonlinearity constitutes the core idea of the proposed framework. The diversity in output responses allows computation of the ensemble mean and variance, which are subsequently used within a learning function to guide adaptive sampling. Therefore, the performance improvement of the proposed adaptive neural network framework originates from output-level diversity and uncertainty-driven learning rather than structural differences in network architecture.

Input dataset (X_{mc}) with n_{mc} samples, represented as a matrix with n_{mc} rows and m columns. The four neural networks represented by net_i for $i = 1,2,3,4$ are first used to calculate the mean of each neural network. Then computed predictions of each neural network mean are shown in Eq. (1), (2), (3), and (4), as below:

$$\bar{y}_1 = net_1(X_{mc}^T) \quad (1)$$

$$\bar{y}_2 = net_2(X_{mc}^T) \quad (2)$$

$$\bar{y}_3 = net_3(X_{mc}^T) \quad (3)$$

$$\bar{y}_4 = net_4(X_{mc}^T) \quad (4)$$

Concatenate predictions into the matrix as shown in Eq. (5).

$$Y = \begin{bmatrix} \bar{y}_1(1) & \bar{y}_2(1) & \bar{y}_3(1) & \bar{y}_4(1) \\ \bar{y}_1(2) & \bar{y}_2(2) & \bar{y}_3(2) & \bar{y}_4(2) \\ \vdots & \vdots & \vdots & \vdots \\ \bar{y}_1(n_{mc}) & \bar{y}_2(n_{mc}) & \bar{y}_3(n_{mc}) & \bar{y}_4(n_{mc}) \end{bmatrix} \quad (5)$$

Computed mean and standard deviation across all neural networks are shown in Eq. (6) and Eq. (7).

$$\bar{y} = \frac{\sum_{i=1}^{4} y_i}{4} \quad (6)$$

$$\delta_{\bar{y}} = \sqrt{\frac{\sum_{i=1}^{4} (y_i - \bar{y})^2}{4}} \quad (7)$$

Each of the four trained neural networks predicts output values for the MC samples. The predictions from these neural networks are concatenated into a matrix. Each row contains the predictions from the four networks for all samples in the MC samples. The mean is calculated by predicting across all four networks for each sample in the MC set. This results in a vector where each element represents the mean prediction of a particular sample from the ensemble of the four networks. The standard deviation is the discrepancy across the four networks of each sample in the MCS.

By aggregating predictions from multiple neural networks and calculating their mean and variance, the proposed method aims to estimate one more prediction, the variance for each sample in the MC set, to measure the uncertainty. This process helps in understanding the prediction consensus and the level of confidence or uncertainty associated with the predictions.

2.3.*Learning function*

AK-MCS was used to enhance the efficiency of reliability analysis. The U-learning function is an uncertainty-based learning function, which quantifies the model's uncertainty at different points in the input space. Typically, it is computed as the ratio of the absolute predicted mean to the standard deviation (uncertainty) of the surrogate model's predictions at each data point, and its expression is shown in Eq. (8).

$$\mathrm{U}(\mathbf{x}) = \frac{\left|\mu_{\hat{g}}(\mathbf{x})\right|}{\sigma_{\hat{g}}(\mathbf{x})} \qquad (8)$$

The point that minimizes U(x) has the most probability of making a wrong sign prediction, and thus it should be selected to be the new training sample to update the surrogate model.

3. Steps of the proposed method

The proposed method aims to establish an accurate surrogate model to determine the system's probability of failure and can be employed in ten main steps. The flowchart is shown in Figure 2, each of which is briefly described below:

Step 1: Generate the Monte Carlo population S. In this stage, the distribution of the design variables is used to generate S.

Step 2: Define an initial DoE random selection of N points. The selected points undergo evaluation using the performance function and serve as the initial training samples for the neural network. Based on experience, a relatively small number proves to be sufficient.

Step 3: Construct four non-adaptive neural networks. During this phase, four separate non-adaptive neural networks are constructed, each utilizing the sigmoid activation function for the hidden layer. However, distinct output layer activation functions are employed for each neural network. These varying output layer activation functions yield different probability of failure values for each network.

Step 4: Find the mean and variance of the unsampled point. The four trained neural networks are used to predict output values for the MC set. These predictions are then combined into a matrix, with each column representing predictions from each network of all samples. The mean is computed across all networks for each sample, yielding a vector of mean predictions. Similarly, the variance is calculated across the networks, resulting in a vector of variances for each sample.

Step 5: Identify the next sample in S by the U-learning function. This stage involves implementing the learning function, computed for each point in the sample set S.

Step 6: Check the stopping criterion. The stopping criterion of the U-learning function is changed to the loop termination condition in this proposed method. Once the size condition is met, the loop breaks, and the process moves to subsequent reliability metric calculations. If it has met the stopping criterion it proceeds to Step 8. If not, proceed to Step 7 and keep the active-learning process.

Step 7: Evaluate the next best point and update DoE. If the stopping condition in Stage 6 is not satisfied, the active learning process carries on, and the next new training sample x^* in S, and the true response of x^* is evaluated by the true performance function. Following this, the next new training sample and its true response are added to the training samples.

Step 8: Evaluate the value of $C.O.V_{Pf}$ Verify whether the value of $C.O.V_{Pf}$ is reached at the required level or not. If $C.O.V_{Pf} < 0.05$, proceed to Step 10; otherwise, return to Step 9.

Step 9: Generate another MCS sample. During this step, an additional set of N_s MCS samples are generated.

Step 10: Output the predicted $\hat{P}_f$.

Half-tone pictures must be sharp enough for reproduction.

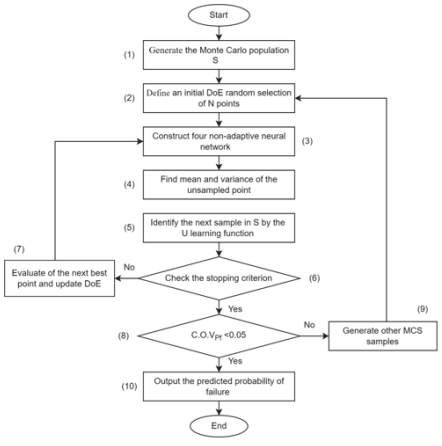

Fig. 2. Flow chart of the proposed adaptive neural network method for reliability analysis.

4. Examples, Results, and Discussion

The effectiveness of the proposed method is demonstrated through two examples. The first example entails a series system comprising four performance functions, while the second example involves a highly nonlinear oscillator. For both examples, the four back-propagation neural networks are constructed. To assess the performance of the proposed method, accuracy and efficiency comparisons are conducted with classic approaches, including AK-MCS+U, MCS, and four non-adaptive neural networks, which are NN1, NN2, NN3, and NN4 that use Linear activation function, Sigmoid activation function, Hyperbolic Tangent activation function, SoftMax activation function, respectively, for the output layer. All the non-adaptive neural network uses sigmoid activation function in the hidden layers for the two examples

4.1.*Examples and Results*

Example 1 Four branch function

This example involves a series system comprising four branch. The system incorporates two standard normally distributed random variables, x_1 and x_2. The value of k is taken to 6. The limit state functions are defined in the Eq. (9). (Grooteman 2008)

$$\mathrm{g}(x_1, x_2) = \min \begin{Bmatrix} 3 + 0.1(x_1 - x_2)^2 - \frac{(x_1+x_2)}{\sqrt{2}} ; \\ 3 + 0.1(x_1 - x_2)^2 + \frac{(x_1+x_2)}{\sqrt{2}} ; \\ (x_1 - x_2) + \frac{k}{\sqrt{2}} ; \\ (x_2 - x_1) + \frac{k}{\sqrt{2}} ; \end{Bmatrix} \quad (9)$$

For this example, a fully connected feedforward neural network with three hidden layers containing 10, 20, and 15 neurons was employed. The hyperbolic tangent sigmoid (tansig) activation function was used in all hidden layers. The network was trained using the scaled conjugate gradient algorithm (trainscg) for a maximum of 2000 epochs. Four models were constructed with identical architecture and training settings, differing only in the output-layer activation function.

Table 1 displays the results of the proposed method. Notably, the failure probability estimated by the proposed ALNN closely matches that obtained by MCS. Additionally, the proposed ALNN demonstrates a reduction in the number of calls to the limit state functions, aligning with accuracy.

Table 1. Results of the four branches function with different methods, including a non-adaptive neural network, AK-MCS+U, and the proposed method.

Methods	$\mathrm{N_{call}}$	$\hat{P}_f$($\times 10^{-3}$)	$C.O.V_{Pf}$($\times 10^{-2}$)
MCS	10^6	4.445	1.49
AK-MCS+U	126	4.416	1.49
NN1	50,000	4.442	1.49
NN2	50,000	4.541	1.47
NN3	50,000	4.391	1.49
NN4	50,000	4.601	1.45
Proposed ALNN+U	100	4.438	1.50

Furthermore, the performance of the proposed ALNN is comparable to other surrogate methods. These results highlight the effectiveness of

ALNNs in terms of both accuracy and efficiency to existing surrogate techniques while requiring fewer computational resources. Figure 3 shows the training performance and indicates how well the model fits the training data.

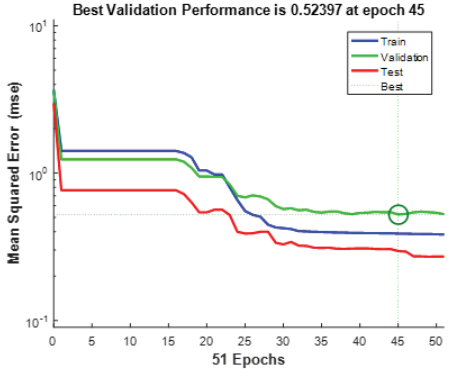

Fig. 3. The graph that shows the validation performance assesses how well the model generalizes to training, validation, and testing data during training.

In Figure 4, the inconsistency between the predicted values and the observed values may suggest a lack of precision in the predictions. However, it's important to recognize that this inconsistency doesn't necessarily indicate that the proposed method is inaccurate. Reliability analysis methods are typically concerned with assessing the probability of failure of a system operating within acceptable safety margins.

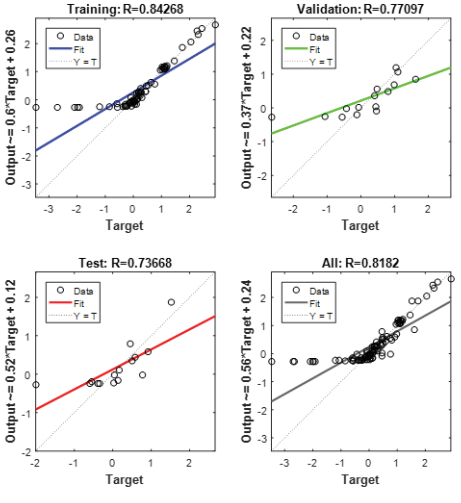

Fig. 4. The goodness of training, validation, and test.

Example 2: A nonlinear oscillator

This example consists of a nonlinear undamped system, the limit state function is as shown in Eq. (10). (Echard et al. 2011)

$$g(c_1, c_2, m, r, t_1, F_1) = 3r - \left|\frac{2F_1}{m\omega_0^2}\sin\left(\frac{\omega_0 t_1}{2}\right)\right| \quad (10)$$

where $\omega_0 = \sqrt{\frac{c_1+c_2}{m}}$, The distribution parameters of the six random variables are shown in Table 2. Figure 5 illustrates the nonlinear oscillator system considered in this study, defining the dynamic performance function used to evaluate the failure probability in the reliability analysis.

For Example 2, a fully connected feedforward neural network with three hidden layers containing 50, 70, and 60 neurons, respectively, was employed. The hyperbolic tangent sigmoid (tansig) activation function was used in all hidden layers. The networks were trained using the scaled conjugate gradient algorithm (trainscg) for a maximum of 2000 epochs. Four models with identical architecture and training settings were constructed, differing only in the output-layer activation function (purelin, logsig, tansig, and softmax) to enable ensemble-based uncertainty estimation within the adaptive learning framework.

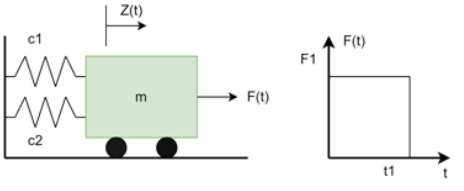

Fig. 5. The nonlinear oscillator

Table 2. nonlinear oscillator random variable

Variable	Distribution	mean	Standard deviation
m	10^6	2.83	-
c_1	102	2.84	0.21
c_2	50,000	2.86	0.58
r	50,000	2.91	0.57
t_1	50,000	2.84	0.58
F_1	50,000	2.84	0.57

The findings from Table 3 indicate that the adaptive neural network performs admirably even

in highly nonlinear functions, maintaining accuracy in estimating the probability of failure.

Table 3. Results of the nonlinear oscillator function with non-adaptive neural networks, AK-MCS+U, and the proposed method.

Methods	N_{call}	$\hat{P}_f(\times 10^{-2})$	$C.O.V_{Pf}(\times 10^{-2})$
MCS	10^6	2.83	-
AK-MCS+U	102	2.84	0.21
NN1	50,000	2.86	0.58
NN2	50,000	2.91	0.57
NN3	50,000	2.84	0.58
NN4	50,000	2.84	0.57
Proposed ALNN+U	80	2.81	0.61

Notably, the proposed method requires the lowest number of calls compared to other approaches, such as the AK-MCS+U. While non-adaptive neural networks also deliver accurate results, their computational efficiency is high. This approach maintains computational effectiveness while preserving accuracy in this example. Figure 6 presents the training performance of the network and demonstrates the accuracy with which the model approximates the training data. Figure 7 compares the predicted responses of the neural network with the corresponding observed values for Example 2.

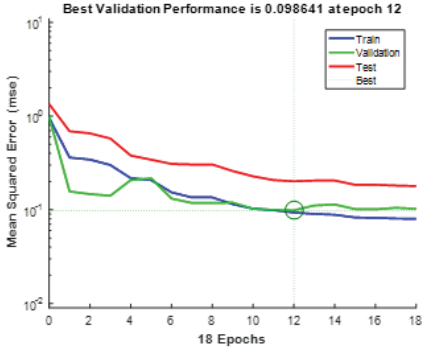

Fig. 6. The validation performance graph assesses the model's ability to generalize to training, validation and testing data during training.

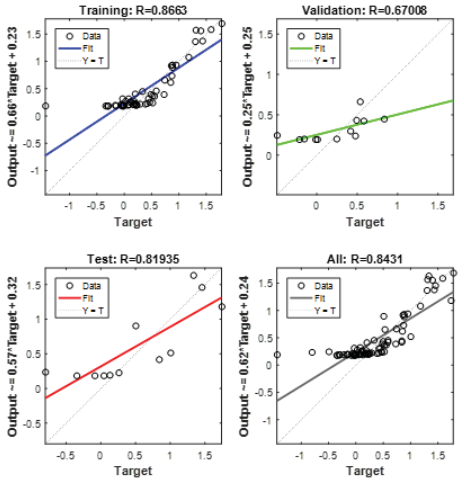

Fig. 7. The goodness of training, validation, and test.

6. Summary

The proposed active learning neural network method exhibits a clear advantage in computational efficiency over the classic non-adaptive neural network approach for the two examples. The results obtained from the proposed ALNN+U method are also better than those from the active learning Kriging model; they still represent a slight reduction in computational resources while retaining accuracy in failure probability estimation. This highlights that the proposed ALNN+U can offer substantial computational savings without compromising the precision of reliability assessments compared to the non-adaptive neural network approach. In practical applications, this underscores the method's effectiveness and its potential to streamline computational processes without sacrificing reliability assessment accuracy.

References

Bichon, B. J., M. S. Eldred, L. P. Swiler, et al. 2008. "Efficient Global Reliability Analysis for Nonlinear Implicit Performance Functions." AIAA Journal 46 (10): 2459–2468.

Bucher, C., and T. Most. 2008. "A Comparison of Approximate Response Functions in Structural Reliability Analysis." Probabilistic Engineering Mechanics 23 (2–3): 154–163.

Chojaczyk, A. A., A. P. Teixeira, L. C. Neves, et al. 2015. "Review and Application of Artificial Neural Network Models in Reliability Analysis of Steel Structures." Structural Safety 52 (PA): 78–89.

Echard, B., N. Gayton, and M. Lemaire. 2011. “AK-MCS: An Active Learning Reliability Method Combining Kriging and Monte Carlo Simulation.” Structural Safety 33 (2): 145–154.

Grooteman, F. 2008. “Adaptive Radial-Based Importance Sampling Method for Structural Reliability.” Structural Safety 30 (6): 533–542.

Hornik, K., M. Stinchcombe, and H. White. 1990. “Universal Approximation of an Unknown Mapping and Its Derivatives Using Multilayer Feedforward Networks.” Neural Networks 3 (5): 551–560.

McCulloch, W. S., and W. Pitts. 1943. “A Logical Calculus of the Ideas Immanent in Nervous Activity.” Bulletin of Mathematical Biophysics 5 (4): 115–133.

Sharma, S., S. Sharma, and A. Athaiya. 2020. “Activation Functions in Neural Networks.” International Journal of Engineering Applied Sciences and Technology 4 (12): 310–316.