IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 13 September 2023, accepted 4 October 2023, date of publication 9 October 2023, date of current version 13 October 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3322924

RESEARCH ARTICLE

Quantitative Analysis of Worm Transmission and Insider Risks in Air-Gapped Networking Using a Novel Machine Learning Approach

MUHAMMAD SULAIMAN[1], AWAIS KHAN[1], ADDISU NEGASH ALI[2], GHAYLEN LAOUINI[3], AND FAHAD SAMEER ALSHAMMARI[4]

[1]Department of Mathematics, Abdul Wali Khan University, Mardan 23200, Pakistan
[2]Faculty of Mechanical and Industrial Engineering, Bahir Dar Institute of Technology, Bahir Dar University, Bahir Dar 6000, Ethiopia
[3]College of Engineering and Technology, American University of the Middle East, Egaila 54200, Kuwait
[4]Department of Mathematics, College of Science and Humanities in Alkharj, Prince Sattam Bin Abdulaziz University, Al-Kharj 11942, Saudi Arabia

Corresponding author: Addisu Negash Ali (addisuan@gmail.com)

This work was supported by the Prince Sattam Bin Abdulaziz University under Project PSAU/2023/R/1444.

ABSTRACT Researchers and practitioners in the fields of science and engineering encounter significant challenges when it comes to mitigating the proliferation of computer worms, owing to their rapid spread within computer and communication networks. This study delves into a comprehensive analysis of the mathematical model governing the hazard of worm propagation in such networks. Specifically, the mathematical framework employed herein encompasses a system of ordinary differential equations. In numerous instances, mathematical models have been employed to quantitatively investigate the propagation patterns of worms across computer networks. In this scholarly article, we present an enhanced Susceptible-Exposed-Infected-Quarantined-Vaccinated (SEIQV) model, denoted as Susceptible-Exposed-Infected-Quarantined-Patched (SEIQP), which effectively captures the dissemination dynamics of an insider threat within a network featuring air gaps. To facilitate the study, we leverage the power of feedforward neural networks that are trained using the backpropagated Levenberg-Marquardt optimization algorithm. These neural networks serve as surrogate tools, providing solutions to the SEIQP model. To evaluate the efficacy of our approach, we meticulously assess their performance across three distinct scenarios. Additionally, the stability of the mathematical model is examined by manipulating the probability of an insider threat removing a patch from the host, denoted as η. Our empirical findings conclusively establish the effectiveness of the proposed approach in addressing the intricate challenges associated with insider threats within network environments.

INDEX TERMS SEIQV model, insider threat, artificial neural networks, machine learning, system of differential equations, surrogate solutions, optimization algorithm, anti-virus, numerical solutions, patching.

I. INTRODUCTION

Computer networks have become crucial and necessary instruments for communicating information, cyber security, and research in today's environment. Most computers are linked to one another in some fashion, whether through the Internet or within an intranet. While this is an extremely beneficial feature for communication, Additionally, it lets malware, a type of cybercrime, flourish [1]. Malware includes a variety of malicious behavior, such as that displayed by worms, viruses, spyware, trojans, and rootkits [1], [2], [3]. It is a widespread security concern for the Internet infrastructure since it can compromise data integrity and even lead to theft [4], [5]. Moreover, several researchers have applied the concept of malicious to Android devices [6], [7]. Due to the distinctions between mobile phones and desktop computers, the currently used models for the transmission of computer worms cannot immediately begin their operation in

The associate editor coordinating the review of this manuscript and approving it for publication was Wei Wei.

a mobile network [8], [9]. Susceptible-Affected-Infectious-Suspended-Recovered (SAIDR) is a model that is mostly applicable to mobile networks [10]. However, a variety of models can be utilized for computer networks. As a result of the speed at which they spread, computer worms in computing systems and networks constitute a significant security risk [11], [12]. As an illustration, the 2001 Code Red worm infected 359,000 hosts in 24 hours [13], [14]. The Blaster, Witty, and Conficker worms are just a few of the newer, more complex worms that have been identified since 2001 [11], [14]. The integrity, confidentiality, and availability of the system can be compromised by computer worms used as a delivery mechanism for other malicious activities. To statistically comprehend spreading characteristics and develop effective countermeasures or mitigation strategies, there is a growing need to precisely model computer worm propagation characteristics.

To effectively combat computer worms, several strategies have been developed, including patching the vulnerability [15], eradicating the infection using software designed to combat viruses [16], and policing the flow of traffic using intrusion detection technologies [17]. Certain networks have been air-gapped to prevent them from having direct internet connectivity to avoid exterior exposure from occurring in the first place [18]. A networking connection between the internal networks and the intranet, or the outside world, is absent in an air-gapped network [19], [20]. Despite this, the network is still open to insider attacks such as the transmission of computer worms from within the network. The following steps might be very helpful for the model.

- For an air-gapped network with insider threats, a modified version of the SEIQV model is quantitatively analyzed. For the remainder of this study, we will refer to our modified SEIQV model as the Susceptible-Exposed-Infected-Quarantined-Patched (SEIQP) model to distinguish it from earlier SEIQV models described in the literature.
- To provide quantitative guidance for using containment measures against a worm with specific characteristics, additionally, we calculate the SEIQP model's fundamental reproduction number.
- To decide where resources should be used in the network to accomplish different goals and results, In the third stage, we analyze the insider threat's influence over the operating variables.

In [21], the worm propagation and insider threat in air-gapped network is modelled using a modified SEIQV mathematical framework. The RK-4 approach was used to solve the SEIQP model. This method is not appropriate for stiff differential equations because it is computationally expensive, lacks a built-in error monitoring system, and is not computationally efficient. It is only stable conditionally. If there are higher-order ODEs, this procedure may make its implementation more difficult and perhaps create new sources of error. However, machine learning algorithms outperforms other numerical techniques because it is more user-friendly than conventional approaches, and can handle data-driven approaches, non-linearity, complexity, high-dimensional data handling, and complicated patterns. In addition, compared to other methodologies, it provides us with a solution that is much closer to the actual solution [22], [23]. These advantages of machine learning compelled the authors of this manuscript to the SEIQP model discussed in this paper.

This study's objective is to examine a modified version of the SEIQV model for "worm propagation and insider threats in the air-gapped network" by making use of Artificial Neural Networks (ANNs) to solve the problem. The machine learning approach provides an additional way to solve the problem through the use of data. It provides precise numerical estimates, makes model building easier, lowers processing costs, and provides robust and adaptable model abilities. The use of an ANN-based system makes the analysis more effective, and it also reduces the amount of error. ANNs have the potential to develop in a number of distinct ways, each of which is determined by the data that moves through the network during the course of the learning process, whether that data is external or internal. An artificial neural network will employ the Back Propagation method to carry out simultaneous training in order to improve the overall effectiveness of a Multilayer Perceptron (MLP) network. It is the paradigm for complicated multi-layered networks that are utilized the most since it is efficient, effective, and easy to understand. The Levenberg-Marquardt algorithm, is a new converging reliability strategy for artificial neural networks (ANNs) [24]. It offers a numerical solution to a wide variety of problems that are caused by computer worms. To investigate the dispersal of worms and the impacts of insider risks in a network, we employed a backpropagated neural network optimized using the Levenberg-Marquardt approach LMB-NN. The following is an overview of the most important components that have been proposed in this manuscript:

- The Levenberg-Marquardt back propagation neural networks architecture is used to solve an application-oriented system of ODEs.
- The model for worm propagation and insider threats in air-gapped networking is analyzed using novel intelligent computing.
- Solutions of ODEs are predicted and then analyzed by different performance indices.
- Accuracy, efficiency, validation, convergence analysis, Mean Square Error (MSE), error histograms, and regressions, were obtained for the design scheme, validating the accuracy and repeatability of the designed solution.

The rest of the article is structured as follows. The relevant research on worm transmission and insider threats is covered in Section II. The problem-solving mathematical model is in Section III. Moving on to mathematical modeling in Section IV. Section V, Deciding on the suggested approach to problem-solving. The design process is provided in

TABLE 1. Abbreviations and its descriptions.

Abbreviation	Description
FFNN	Feedforward Neural Networks
MLP	Multilayer perceptron
LMB	Levenberg-Marquardt backpropagated
NN	Neural Network
SEIUR	Susceptible-Exposed-Infectious-Undetected-Recovered
SIS	Susceptible-Infected-Susceptible
SEIS-V	Susceptible-Exposed-Infectious-Susceptible-Vaccination
SIR	Susceptible-Infected-Recovered
SIRS	Susceptible-Infectious-Removed -Susceptible
SEIR	Susceptible-Exposed-Infectious-Recovered
SEIQV	Susceptible-Exposed-Infected-Quarantined-Vaccinated
SEIRS	Susceptible-Exposed-Infected-Recovered-Susceptible
SEIRS-V	Susceptible-Exposed-Infectious-Recovered-Susceptible with Vaccination
SEIQRS	Susceptible-Exposed-Infected-Quarantined-Recovered-Susceptible

Section VI, which also details the feedforward neural network (FFNN) based approach to problem-solving. Section VII describes the findings and discusses them, and section VIII then makes a conclusion.

II. PRELIMINARIES

Spread of malicious objects in computer network and their control are area of concern for the researcher. The study of virus spread in computer networks has frequently used biologically based epidemic models. The earliest mathematical model for epidemics was created by Daniel Bernoulli in 1760 [25], [26], but today's commonly used models are based on the work of Kermark and McKendrick, who employed compartmental epidemiological models to capture [21] the dynamical aspect of epidemics [27], [28]. Their original Susceptible-Infectious-Recovered (SIR) model has undergone many changes, including the addition of new compartments including quarantined (Q), exposed (E), undiscovered (U), and vaccinated (V). A number of researchers and scientists have experimented with and developed a variety of models, some of which include Susceptible-Exposed-Infectious-Undetected-Recovered (SEIUR) [29], Susceptible-Infected-Susceptible (SIS) [30], Susceptible-Exposed-Infectious-Susceptible with Vaccination (SEIS-V) [31], Susceptible-Infectious-Recovered (SIR) [32], Susceptible-Infectious-Removed-Susceptible (SIRS) [33], Susceptible-Exposed-Infectious-Recovered (SEIR) [34], [35], (SEIQV) [11], Susceptible-Exposed-Infectious-Recovered-Susceptible (SEIRS) [36], Susceptible-Exposed-Infectious-Recovered-Susceptible with Vaccination (SEIRS-V) [37], Susceptible-Exposed-Infectious-Quarantined-Recovered-Susceptible (SEIQRS), and so on. Every single one of them is an updated version of a biological spreading disease model. The SEIR model is employed to mimic the dissemination of the virus over the network [34], [35]. Using several forms of command and control, the SEIUR model is being developed to reduce the prevalence of malicious codes spreading throughout a computer system [29], [38]. For worms to spread vertically in a network, researchers had to create a dynamic e-epidemic SEIS-V model. Applying this model to the study of antivirus programs will be a massive benefit [31]. Wireless sensor network viral dynamics were previously studied using SIS models that concentrated on wired networks [30], [39]. It was also put forward by [40] and study the factors that contribute to the transmission of malware. that is recurrent in multilayer topologies that combine two distinct kinds of networks. Different types of wireless networks in the Internet of Things [41], [42], where the complexity and computational capacity of the devices vary widely, are studied using the SIR model to analyze the spread of jamming assaults that can damage many levels of communication for all nodes in the network [32], [43]. The SIRS model helps to better understand and forecast the scope and rate of Internet worm propagation and offers practical solutions for halting its spread [33]. To account for the spread of malware throughout networks, we develop the SEIRS model of epidemic transmission, which assumes a constant death rate for infective nodes and a death rate that is independent of the source of infection. In the network of computers, the death of a node is synonymous with the isolating of that node from the computer network, which prevents the transmission of dangerous objects even when the anti-malicious software is continuously running [36], [44], [45]. A malware-spreading model called SEIRS is dependent on a rumor-dissemination model that is used to investigate the movement of malware propagating on scale-free networks (SFNs). This takes into consideration the distribution of various software packages across network nodes preventing the spread of malware [46], [47]. The military has a variety of uses for wireless sensor networks, including monitoring the activity of militants in remote places and providing force protection. Wireless sensor networks can also be utilized in a variety of other ways [48], [49]. For these kinds of applications, the SEIRS-V model may be utilized to investigate the behavior of worms when they attack sensor nodes [37], [50].The study of distributed cloud data center delivers real-time cloud services with resilience, dependability, and security despite the possibility of failure and the discrete dispersion of data center users. Network failures harm cloud computing significantly because they result in widespread service delays and interruptions due to the inter connectivity of data centers. The ability to do a conspicuous worldwide search without regard to gradient allows it to achieve the breakthrough in fault location accuracy [51], [52]. A brain-like productive service provisioning scheme with federated learning (BrainIoT) for IIoT. The BrainIoT scheme is composed of three algorithms, including industrial knowledge graph-based relation mining, federated learning-based service prediction, and globally optimized resource reservation. BrainIoT combines production information into network optimization, and utilizes the interfactory and intrafactory relations to enhance the accuracy of service prediction. The globally optimized resource reservation algorithm suitably reserves resources for predicted services

considering various resources [53]. A SEIQRS model for Internet worms, complete with graded infection rates; in order to mount an effective defense against worms [54]. With the use of the SEIQV model, we are able to calculate the fundamental reproduction amount that determines whether or not a worm has become extinct [11]. The majority of currently available models for worm propagation are geared toward finding solutions to the issues caused by computer worms.

In order to halt the proliferation of malicious software, network countermeasures like patching and antivirus software were used before these different portions were implemented. A substantial amount of study has gone into assessing the results of various countermeasures in order to offer networks effective defense tactics [55], [56]. A large-scale foliaceous epidemic model is used to estimate the best patching strategies, and an adaptive mitigation technique is developed to stop the spread of malware. In order to provide the defense with the optimum approach, the goal of ''FLIPIT,'' a game theoretical model, is to perform a cost-benefit analysis of both patching and virus eradication (i.e., anti-virus) [57], [58]. The drawbacks of network-based antivirus programs are examined in [59], which presents a model to do so. Even though every model has a unique set of limitations, these models nonetheless yield valuable insights regarding the role that countermeasures play in preventing the spread of features.

The assumption of homogeneity acts as a limiting condition for the vast majority of suggested models of malware epidemics. [59]. Heterogeneity is taken into account by certain models, though. The effect that the interval between infections has on the rate of worm reproduction is investigated in [60], In [61], network topology fluctuations are accounted for by means of a spatial-temporal model, There is a description of the dynamics of multi-group transmission in [62] and [63] takes into account the possibility of variations in infection routes.

III. MATHEMATICAL MODEL FOR THE PROBLEM

The exploration of malware propagation in computer networks has typically used biologically based epidemic models. In 1760, Daniel Bernoulli developed the first epidemic model [25]. However, the models that are widely used now are based on the work of Kermark and McKendrick [21], whose compartmental epidemiological models best captured the inherent fluidity of disease transmission [27]. Different modifications have emerged from their initial Susceptible-Infectious-Recovered (SIR) model, incorporating new partitions like undiscovered (U), vaccinated (V), quarantined (Q), and exposed (E). As a result of the fact that malware models use these chambers, the proposed models include SIR, SIS, SEIR, SIRS, SEIRS, SEIQRS, SEIQS, SEIRS-V, SEIQV, SEIUR, SEIS-V, and so on. Network countermeasures like patching and anti-virus programs were used to prevent the dissemination of malware, which prompted the construction of these separate compartments. In order to provide networks with adequate strategies of defense, a significant amount of research has been put into evaluating the consequences of a variety of counter-responses. It is advised to use a patching-based adaptive mitigation method [17] as a means of slowing the spread of malware, and a broad stratifying pandemic model [15] is utilized in order to select the most effective patching rules to put into effect. References [57] and [64] suggests using a gaming theoretical model called ''FLIPIT'' to perform a cost-benefit analysis of both removing (also known as anti-virus software) and patching in order to provide the defense the best possible plan of action. The flaws in virus scanners on a network are studied via the model described here [59]. Even though each model has its own unique set of drawbacks, they all shed light on the important function of countermeasures in halting the spread of features. The assumption of homogeneity is made by the overwhelming majority of the models that have been presented to explain the spread of malware. However, some models presumptively assume heterogeneity. The variance in worm spreading caused by the timing of the subsequent infection, [60], [61] employs a spatial-temporal model to account for network topology fluctuations, [62] describes the dynamics of multi-group propagation and [63], [65] takes into consideration potential changes in infection pathways. In current history, there has been a discernible rise in the amount of research conducted on heterogeneous models for malware propagation. Contrary to the biological models that served as the basis for their development, not all worm propagation models suppose an unlimited immune response after vaccination. Specifically, the concept of SEIQRS-V that was introduced in [59] makes the assumption that the absence of anti-virus software updates will, in due course, result in the computers located within the protected compartment being free of viruses that are susceptible to infection. The transition from the recuperated chamber to the susceptible chamber is presumed to happen similarly in [22] and [66]. The system in [37] and [67] also makes the assumption that there is no permanent immunity in the cybersphere, and as a result, the model offers a transition from compartments that have been immunized to chambers that are vulnerable. However, because they assume that there are numerous worms with the same rate of spreading and network properties, these models are only useful for understanding the effects of immunity loss. Although worm propagation models have drawn a lot of interest, most works in modeling insider risks have concentrated on escalation, objectives, and assault setup [68]. Only [69] has used a probabilistic epidemic model for insider threat assessments to demonstrate coordinated external and internal network strikes. We suggested SEIQP model deviates from all previously described models in three significant ways:

1) The original infection's source, also known as the seed, is an insider threat that occurs within a network that has been air-gapped.
2) It is possible for the host to lose its immunity if the patch is maliciously removed from the host by an insider threat.

TABLE 2. Explanation of the notations used in the referenced differential equation.

Notations	Definitions
N	Quantity of hosts overall in the network
t	Unit of time
S(t)	Amount of hosts that are susceptible at time t
E(t)	Amount of hosts that are exposed at time t
I(t)	Infected host quantity at time t
Q(t)	Number of hosts under quarantine at time t
P(t)	Amount of hosts with patches at time t
β	Contact rate for infections
σ	A susceptible host's rate of infection by insider threats
θ	Rate of antivirus scans
ρ	Patching rate for susceptible hosts
γ	The likelihood of an exposed host becoming infected at a certain rate
δ	Rates of quarantining hosts infected with a virus
ϵ	Rate of "cleaning" and patching an infected node
ξ	Human mistake removing host patch probability
η	The likelihood of an insider threat causing a host patch to be disabled.

3) The elimination of the patch from the host as a result of human error in the application of policy can cause the host to lose immunity.

IV. PROCEEDING TO MATHEMATICAL MODELING

The SEIQP approach, which represents an insider hazard as the uploading of a worm into an air-gapped network, is offered in this section of the article. In more detail, we first describe the model's underlying presumptions. Then, the seven transitions and five compartments of the SEIQP model are introduced. In the final step of this process, we develop a set of ODEs that will be utilized to describe the dynamical system of the model. In the mentioned system of ODE, the nomenclature is displayed in Table (2).

A. ASSUMPTIONS

The following network presumptions serve as the foundation for our model: (1) Since the contact network is a full graph, each host is susceptible to infection by any other host [21]. (2) Due to the homogeneous nature of the network [27], all hosts are susceptible to the worm's exploits until they are patched. (3) The only way for a worm to get into the air-gapped network is for a malicious insider to get into one of the hosts on the network and upload it there. (4) The total amount of hosts that make up the network as a whole remains unchanged and unchanging over the course of time [21]. (5)There is only one threat posed by an insider within the network. (6) There is only one worm involved in the insider threat, hence numerous worms are not present. (7) Having the required authorizations and network access, the insider threat can alter host and/or network policies. (8) The patch eliminates the threats, but it does not eradicate the worm from a computer that has already been infected (i.e.; Immunity can only be acquired from susceptible, contained hosts) [57].

Furthermore, the following assumptions underpin the properties of our model: Every host is vulnerable to attacks and worms as long as there hasn't been an occasion of

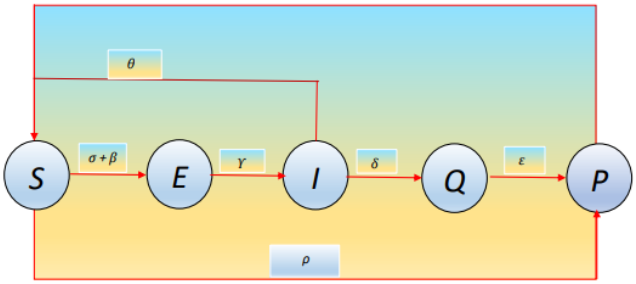


FIGURE 1. Diagrammatic representation of worm movement in an air-gapped network with an insider threat.

an insider potential risk successfully transferring a worm into a host [59]. (2) Infection rates, denoted by *beta*, are constant across time [70]. (3) The worm is always detected and eliminated by an anti-virus when it detects the host. (4) After being patched, a host gains immunity and is no more susceptible to infection [57]. (5) A host may lose immunity if an insider threat intentionally removes a patch from it [71]. (6) The unintended removal of a patch and the destruction of host immunity can both be caused by human error in the application of network policies.

Although these presumptions restrict the model's applicability to a certain situation, it nevertheless offers insight into the dynamic nature of worm behavior spreading under a number of conditions that could develop as a consequence of insider risk.

B. PROBLEM FORMULATION

Figure (1) reveals the SEIQP model, which was based on our network, human error, insider threat, and worm infection assumptions. The dynamics of the air-gapped network are described by the model's seven transitions and five sections. The SEIQP model's five sections have the same amount of nodes (N) as the network's total [70] that is:

$$N = N(t) = E(t) + I(t) + P(t) + S(t) + Q(t) \qquad (1)$$

The network becomes infected with the worm when an insider threat infects any weak host there. The worm then makes the specified transitions described below to move through the network of hosts.

- From section S to section E as a consequence of the worm being exposed, either by the insider threat or by other nodes that were already infected.
- Due to the vulnerability being patched from chamber S to chamber P on the vulnerable node.
- After the worm was uploaded and installed, it spread from compartment E to compartment I and then became contagious.
- The worm was removed from the node, from chamber I to chamber S, as a result of the anti-virus program executing, detecting, and deleting the worm.
- It was transferred from partition I to partition Q once the infection was found and the node was placed

TABLE 3. State transitions used in system of differential equations and their values [21].

Terminologies	Symbols	Values
Rate at which a susceptible host becomes infected by insider threat	σ	0
Infection contact rate	β	0.1
Anti-virus scan rate	θ	0.2
Rate at which a susceptible host is patched	ρ	0.1
Rate at which exposed host becomes infected	γ	0.45
Quarantine rate of infected hosts	δ	0.3
Rate at which an infected node is "cleaned" and patched	ϵ	0.3
Probability of human error resulting in removal of patch on host	ξ	0.01

in quarantine to get rid of the worm and patch the vulnerability.

- After the worm was eradicated and the vulnerability was patched, data was transferred from compartment Q to compartment P.
- Due to insider threat malevolent behavior or unintentional human error, from compartment P to compartment S, the node becomes susceptible to infection once more.

Based on these validated transitions, the following system of ODEs represents the transmission between sections.

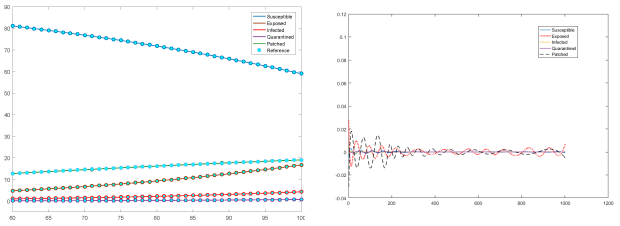
$$\begin{cases} \dfrac{dS(t)}{dt} = P(t)(\eta+\xi) + I(t)\theta - S(t)\sigma - S(t)I(t)\beta - S(t)\rho, \\ \dfrac{dE(t)}{dt} = S(t)\sigma + S(t)I(t)\beta - E(t)\gamma, \\ \dfrac{dI(t)}{dt} = E(t)\gamma - I(t)\theta - I(t)\delta, \\ \dfrac{dQ(t)}{dt} = I(t)\delta - Q(t)\epsilon, \\ \dfrac{dP(t)}{dt} = S(t)\rho + Q(t)\epsilon - P(t)(\eta+\xi). \end{cases} \tag{2}$$

The rates of change in state that are shown in Table (2) for each chamber is here denoted by the letters θ, γ, ξ, η, σ, ρ, β, ϵ and δ and their values are shown in Table (3).

By modifying the value of η, we may establish three situations to investigate deeply the impact of insider threats on patch removal.

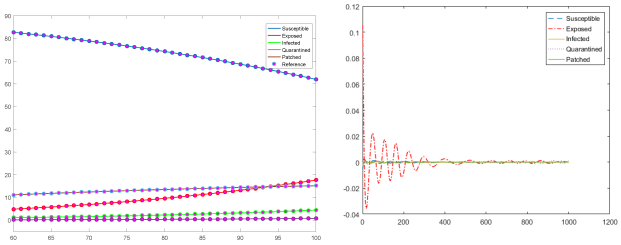
V. MOVING TOWARD THE PROPOSED METHODOLOGY

This section includes the selection of the optimal method for solving the given differential equation system. Proposed methods (numerical technique and machine learning technique) are judged by using graphs and tables. For assessment comparing both techniques [72]. The first is the numerical technique, which requires a mathematical model along the initial condition or boundary conditions. It delivers focused data by employing the mathematical model for a specific problem and its initial condition, and targeted data can be made closer to the actual solution as the number of iterations



(a) case 1: Dynamical behavior(b) Error graphs for Dynamical behavior of system with parameters S_0 =havior of system 99, $E_0 = 1$

FIGURE 2. Dynamical behavior of system (2) and its error.



(a) case 2: Dynamical behavior of(b) Error graphs for Dynamical system with parameters $S_0 = 99$,behavior of system $E_0 = 1$

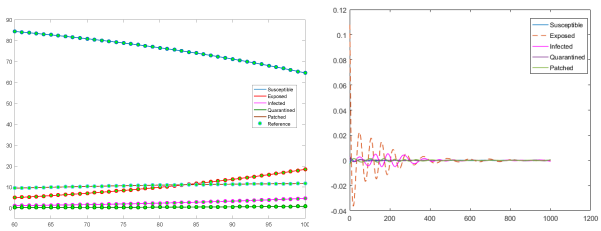
FIGURE 3. Dynamical behavior of system (2) and its error.

grows. The second plot is produced by a machine-learning technique that can only be obtained from the targeted data. We don't require a mathematical model of the specified problem for this kind. In this case, the targeted data and weights are processed to create a surrogate model [73], [74]. The number of weights in this technique depends on the amount of neurons, as the amount of neurons increases the number of weights increases because each neuron contains three weights. In this study, the machine learning strategy is favored over the numerical method since it is more efficient. Furthermore, when compared to other methodologies, it is most appropriate for real-world challenges. We are examining the plots of the following three situations along the tables in this section.

- In the first case ($\eta = 0.05$) [21], we analyzed the output obtained by using a machine learning technique and targeted data using Table (4). Their difference is shown in Table (8) and Figure (2).
- In the Second case ($\eta = 0.25$) [21], we have examined the result of the machine learning technique and targeted data using Table (5). Their difference is indicated in Table (9) and Figure (3).
- In the third case ($\eta = 0.5$) [21], we used a table to assess the outcome of the machine learning technique and targeted data (6). Their differences are shown in Table (10) and Figure (4).

VI. DESIGN METHODOLOGY

Before diving into an optimization method for the neuron learning operation in FFNN architecture, Here, we'll start



(a) case 3: Dynamical behavior of system with parameters $S_0 = 99$, $E_0 = 1$ (b) Error graphs for Dynamical behavior of system

FIGURE 4. Dynamical behavior of system (2) and its error.

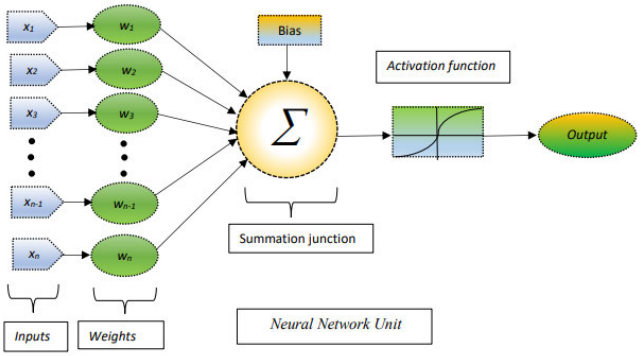


FIGURE 5. A three-layer feed-forward neural network's architecture.

with the basics structure, and how a feed-forward neural network works will be outlined.

A. ARTIFICIAL NEURAL NETWORKS USING FEEDFORWARD NETWORK

McCulloch's computer model of the human brain, which he constructed in 1943, was the impetus for the beginning of research into artificial neural networks (ANN). An extensive range of complex problems can be learned, recognized, and handled by ANNs. Only FFNN models are frequently employed in a variety of applications among all ANN models. A noteworthy aspect of the architectural representation of an FFNN is that it enables the recognition of a computational function in network form. This is a feature that is worth mentioning. Furthermore, an FFNN's foundation makes it a well-liked function approximator, with the result that it can approximate and solve any function or task. A computer model known as an FFNN is made up of several neurons connected by weights and layered layer by layer [24]. As displayed in Figure (5), FFNNs have a unique structural architecture where nodes in one layer are linked to nodes in the subsequent layer. An FFNN node can process data using connection weights. The mathematical formula for calculating the output of a node y_i is:

$$y_j = A_i(\sum_{j=1}^{n} w_j x_j + b_j), \tag{3}$$

here, x_j denotes the inputs, n for the amount of samples, w_j for the connection weights, the bias vector is indicated by b_j and activation function is symbolized by A_i. The weighted vectors w_1, w_2, w_3,…., w_n, weights and the n-dimensional input vectors x_1, x_2, x_3…., xn are used to parameterize the activation function $A(x, w)$. The activation function in this case is an S-shaped curved sigmoid function (log sigmoid).

$$A_i = \frac{1}{1 + e^{-(wx+b)}}. \tag{4}$$

Implementing a log-sigmoid prevents output values from spiking since it produces a smooth gradient.

B. METHODOLOGY

This section explains how the connection weights within the FFNN framework were adjusted to best accommodate the approximate solutions of the model. The first step in solving a problem in Mathematica is to build a reference solution with a thousand and one points. After that, the FFNN model is calibrated using relevant parameters such as the amount of iterations, the activation function, and the amount of hidden neurons. Thereafter, inputs and outputs are sent to the FFNN to conduct supervised machine learning. Figure (6) illustrates the FFNN model's architecture. One of the most popular ANN paradigms is the use of multilayer perceptron (MLP) networks. The FFNN's input-hidden-output layers represent "strengths," which may be characterized as two-dimensionally stacked quantities called "weights," as weights in interneuron interactions. New information is learned by the FFNN and saved in these links. These weights are used to determine the output signal quantities for recently tested input signal quantities. Every layer is connected by a certain sort of computer unit called a neuron. Backpropagation feed-forward algorithms are the most often used kind in multilayer perceptron networks [75]. The "input layer" of the backpropagation feed-forward mechanism supplies neurons to the neural network that is arranged. The network is then connected by a minimum of one "hidden layer", where its actual computation is carried out via a system of "weighted linkages". The outcome is displayed in the "output layer," which is connected to the following hidden layers. By returning to the input nodes during backpropagation, an error between the prediction and the targeted data is corrected. Errors are reduced at the end of the training procedure. As a result, the data used in ANN models have to be correctly categorized and optimized. For the input weights specified as an input restriction in the presence of hidden neurons, the estimated results have been gathered in the form of performance, regression analysis, fitness, gradient, accuracy evaluation, and histogram assessment at the output layer. The fundamental structure for setting up the suggested ANN models is shown in Figure (5). The data used to create the ANN model must be optimized for usage in the prediction node of the artificial neural networks paradigm [76]. From this forward point, the data used to train the ANN model have been refined to yield the best results. The training section was optimized for 70% of the data, whereas the testing and validation

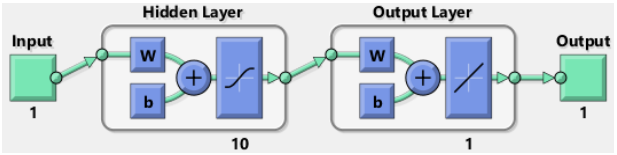


FIGURE 6. FFNN modeling framework for modeling approximative solutions.

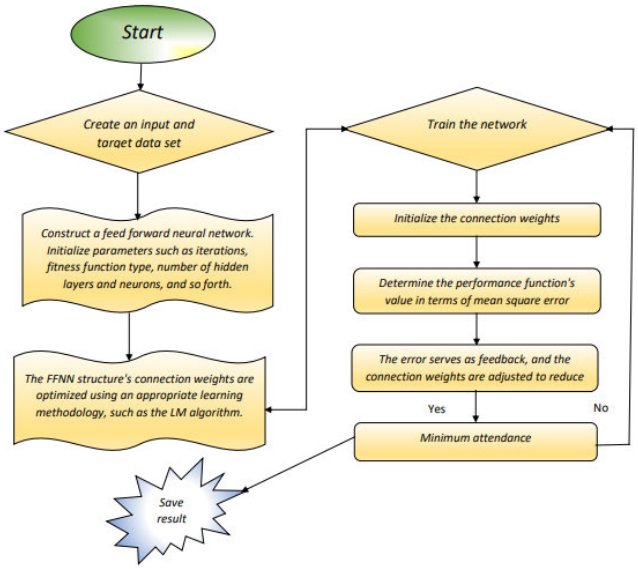


FIGURE 7. Flowchart of the design algorithm.

sections were targeted for 15%, and 15%, respectively, of the data used to build the ANN models. There is no widely accepted approach for deciding how many neurons to use in ANNs. As a consequence, several models with different node counts were looked at, and their prognostication capability was evaluated throughout the design stage of both MLP networks. The present experiment's findings show that the total amount of nodes with the highest expected efficacy has been determined. Suggested neural networks (NNs) employ an architecture made up of 10 highly active neurons that interact in parallel to resolve the worm-spreading issue with the sigmoid function of activation. As seen in Figure (6), the sigmoid activation function's input values span from +1 to 0. LMB-NN is created by combining the Gauss-Newton method with the steepest descent. The methodology acts in a manner similar to the steepest descent method when the current solution is far from optimal; Although it is sluggish, it will definitely converge. But if the present solution is near the optimal solution, then the Gauss-Newton method is utilized. The figure (7) gives a thorough description of how the LMB-NN method functions. This technique is often regarded as the most effective algorithm for training artificial neural networks since it performs well. The LM technique was designed to approach second-order training rates without the need to compute the Hessian matrix, similar to quasi-Newton approaches. When a sum of squares is the shape that the performance function takes, that is;

$$f(W) = \frac{1}{2}e^T e, \tag{5}$$

where $e^T = [e_{1,1}, e_{2,1}, \ldots, e_{k[M],1}, e_{1,2}, \ldots, e_{k[M],Q}]$, Q = L· k[M] and W consists of all weights of the network. The well-known recurring calculation creates Newton's approach for minimizing performance function.

$$W_{i+1} = W_i - H^{-1}\nabla f(W), \tag{6}$$

where ∇f(W) is the gradient of f(W) and $f(W) = \frac{1}{2}e^T$e, then

$$\nabla f(W) = J^T(x), \tag{7}$$

Thus, the Hessian matrix may be characterised as follows:

$$H(x) = J^T(x)J(x) + S(x), \tag{8}$$

where the Jacobian matrix J comprises the first derivatives of the network errors with respect to weights and biases, and e is a vector of network errors. If it can be supposed that S(x) is small when matched to the product of the Jacobian, then the Hessian matrix can be approximated by the following

$$H(x) \approx J^T(x)J(x), \tag{9}$$

so the Gauss Newton algorithm is;

$$W_{i+1} = W_i - [J^T(x)J(x)]^{-1}J^T(x)e, \tag{10}$$

The simplified Hessian matrix may need to be invertible, a potential drawback of this technique. A modified Hessian matrix can be utilized to solve this issue,

$$H(x) \approx J^T(x)J(x) + \mu I, \tag{11}$$

I stands for the identity matrix, and μ is the quantity that causes H(x) to be positive definite and thus invertible. The LM method reflects the most recent alteration in the Hessian matrix.

$$W_{i+1} = W_i - [J^T(x)J(x) + \mu kI]^{-1}J^T(x)e. \tag{12}$$

In order to demonstrate that μ can vary while the algorithm is being executed, μ is now represented as μ_k. Because it determines stability (by ensuring that the Hessian can be inverted) and converging speed, the choice of μ is crucial to the algorithm's operation.

C. PERFORMANCE INDICES

The Levenberg-Marquardt neural network technique's innovative design is implemented in two phases. In the first stage, the Runge-Kutta technique of order 4 is used to assess a mathematical model for the propagation of worms and insider threats. To do this, the reference solution of 1001 data points is generated using the "NDSolve" function included with Mathematica. In the second stage, the Levenberg-Marquardt approach–a proven method in the field of soft computing–is used with MATLAB's "nftool" function to correctly train, validate, and test the issue. The training, validation, testing, and parameter settings for the LMB-NN method are shown

TABLE 4. Setting η=0.05.

Input	0	3	6	9	12	15	18	21	24
Susceptible hosts	99	41.9557	2.559	2.84496	3.49995	4.1854	4.6853	4.9439	5.0322
LMB-NN	99.006	41.9553	2.5591	2.84498	3.49994	4.1855	4.6854	4.944	5.03201
Exposed hosts	1	27.309	30.031	18.311	13.999	12.273	11.7	11.618	11.692
LMB-NN	0.9724	27.315	30.032	18.312	13.9995	12.275	11.69	11.614	11.695
Infected hosts	0	7.9239	27.0486	21.25	15.4629	12.399	11.058	10.5971	10.5168
LMB-NN	0.00065	7.9238	27.0485	21.251	15.463	12.4	11.05845	10.597	10.5169
Quarantined hosts	0	1.5264	13.5692	19.8721	18.574	15.57443	13.17403	11.73364	11.0214
LMB-NN	-0.0012	1.5263	13.5691	19.8722	18.574	15.57442	13.174	11.73365	11.02139
Pached hosts	0	21.284	26.791	37.72	48.464	55.567	59.382	61.107	61.737
LMB-NN	0.03	21.28	26.792	37.722	48.462	55.566	59.38	61.109	61.735

TABLE 5. Setting $\eta = 0.25$.

Input	0	3	6	9	12	15	18	21	24
Susceptible hosts	99	44.6998	3.199	3.7325	4.581	5.00589	5.07315	5.0374354	5.00864
LMB-NN	99.005	44.6997	3.2001	3.7327	4.5811	5.005898	5.07311	5.0374353	5.00863
Exposed hosts	1	29.4	36.002437	25.6469	23.534	23.6574	24.045	24.2321	24.27
LMB-NN	0.894	29.39	36.002434	25.6483	23.535	23.6575	24.044	24.2325	24.269
Infected hosts	0	8.3214	30.8781	26.8022	22.775	21.54454	21.52463	21.7076	21.80778
LMB-NN	0.0008	8.3213	30.8782	26.8021	22.775	21.54459	21.52467	21.7076	21.80779
Quarantined hosts	0	1.573	15.049	23.3644	23.8709	22.7049	21.984	21.7754	21.7741
LMB-NN	-0.0041	1.574	15.05	23.3642	23.8705	22.7048	21.9844	21.7752	21.7748
Pached hosts	0	16.0047	14.869	20.4538	25.2383	27.08708	27.37243	27.24734	27.138839
LMB-NN	0.0025	16.0045	14.869	20.4539	25.2382	27.08702	27.372474	27.24738	27.1388393

in Figure (7). Figure (6) shows the computer model for the design strategy that employs two neural networks [77]. The suggested method's effectiveness is evaluated using the fitness function of the model's mean square error (MSE), regression R^2, error histograms, and absolute errors (AE). The MSE, R^2, and AE are mathematically described as follows:

$$Minimize \;\; MSE = \frac{1}{m}\left(\sum_{i=1}^{m}(x_i(t) - \hat{x}_i(t))^2, \right. \tag{13}$$

in the above equation, the reference solution is $x_i(t)$, while the updated solution is $\hat{x}_i(t)$. Additionally, the connection weights are optimized using a LMB-NN method by diminishing the power function as much as possible provided in equation (13). For perfect modeling of approximative solutions, the MSE value becomes closer to zero. Locating the point where a multivariable function is at its lowest point, which is defined as the sum of squares of non-linear real-valued functions, is done iteratively using the LMB algorithm. With widespread application in numerous domains, it has established itself as a common technique for non-linear least-squares issues.

$$R^2 = 1 - \frac{\sum_{i=1}^{m}(\hat{x}_i(t) - \bar{x}_i(t))^2}{\sum_{i=1}^{m}(x_i(t) - \bar{x}_i(t))^2)}, \tag{14}$$

$$AE = |x_i(t) - \hat{x}_i(t)|. \tag{15}$$

m is the amount of mesh points, while x_i, $\bar{x}_i$, and $\hat{x}_i$ stand for the reference, approximation, and mean of the solution at the ith input. The MSE and AE should both equal zero for perfect fitting, whereas R^2 should equal one.

VII. RESULTS AND DISCUSSIONS

The LMB-NN design strategy is implemented in this portion to look into how to stop worms from spreading and deal with insider threats in an air-gapped network. Weight training for LMB-NN is performed [78]. To address this issue of solving ordinary differential equations, a neural network is employed

TABLE 6. Setting $\eta = 0.5$.

Input	0	3	6	9	12	15	18	21	24	25
Susceptible hosts	99	46.69235	3.2371	3.9887	4.8184	5.0519	5.0352	5.00651	4.9987	4.9985
LMB-NN	99.002	46.69292	3.2372	3.9888	4.81849	5.0521	5.0353	5.006554	4.9988	4.9981
Exposed hosts	1	31.3421	38.859	28.62	27.49	27.952	28.2179	28.2397	28.24788393	28.2442
LMB-NN	0.89	31.3429	38.865	28.617	27.4891451	27.951	28.2178	28.2398	28.24775	28.2447
Infected hosts	0	8.704	33.26	29.202	25.849	25.1837	25.2963	25.3084	25.421	25.4211
LMB-NN	1.50E-05	8.7006	33.27	29.2612	25.8495	25.1835	25.2961	25.3084	25.4208	25.4218
Quarantined hosts	0	1.6201	16.1369	25.237	26.2822	25.7222	25.4332	25.3017	25.405	25.4097
LMB-NN	-0.0032	1.62	16.1367	25.238	26.2821	25.722	25.4331	25.3015	25.406	25.4091
Pached hosts	0	11.6403	8.49715	12.8001	15.5595	16.0892	16.0172	15.9435	15.9264	15.9262
LMB-NN	0.0034	11.6402	8.49771	12.8005	15.5596	16.0891	16.0171	15.9436	15.9266	15.9258

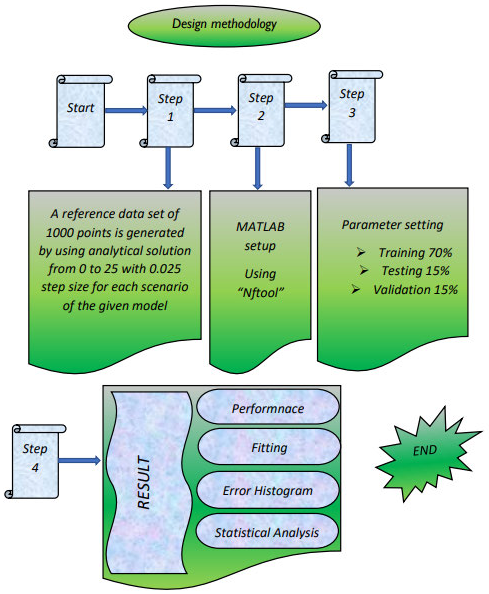


FIGURE 8. Design for a given model.

TABLE 7. Convergence of means square error for different scenarios.

Scenarios 1	Scenarios 2	Scenarios 3
3.4828×10^{-07}	1.8231×10^{-07}	6.8835×10^{-08}
9.766×10^{-06}	8.2145×10^{-05}	8.539×10^{-05}
4.3424×10^{-09}	3.2653×10^{-07}	9.251×10^{-08}
2.3595×10^{-05}	6.4862×10^{-08}	2.8655×10^{-07}
8.281×10^{-09}	1.8853×10^{-08}	2.4144×10^{-06}

(2) with a boundary condition numerically by LMB-NN. To achieve the solution numerically, three different scenarios for the model of worm propagation and insider threat in an air-gapped network are developed. The effect of insider threat (η) is assumed to be 0.05 in the first case, 0.25 in the second case, and 0.5 in the third case, and the other terminologies are kept constant mentioned in Table (2). The details of several cases are provided as The reference data for neural LMB-NN are calculated using the Runge-Kutta numerical method. For each value of η, use NDSolve in the Mathematica framework. The 1001 data points from the numerical technique are tested, trained, and validated at 70%, 15%, and 15%, respectively. The convergence of the mean square error (MSE) function for the cases I, II, and III is depicted in Figures (9), (10), and (11), respectively. A close examination of the graphical representation reveals MSE values, which are at peak at the beginning of the training process, decrease as the number

TABLE 8. error for setting $\eta = 0.05$.

Input	Absolute Error in Susceptble hosts	Absolute Error in Exposed hosts	Absolute Error in Infected hosts	Absolute Error in Quarantined hosts	Absolute Error in Patched hosts
0	6.397×10^{-03}	2.759×10^{-02}	6.55×10^{-04}	1.188×10^{-03}	3.0×10^{-02}
3	4.098×10^{-04}	5.343×10^{-03}	1.549×10^{-04}	3.65×10^{-05}	3.96×10^{-03}
6	8.93×10^{-05}	3.62×10^{-04}	1.30×10^{-04}	5.48×10^{-05}	9.51×10^{-04}
9	2.84×10^{-05}	1.154×10^{-03}	6.43×10^{-05}	4.22×10^{-05}	2.358×10^{-03}
12	3.96×10^{-06}	5.528×10^{-04}	4.42×10^{-05}	4.73×10^{-06}	1.43×10^{-03}
15	9.30×10^{-05}	2.04×10^{-03}	3.97×10^{-05}	1.47×10^{-05}	8.66×10^{-04}
18	7.45×10^{-05}	2.012×10^{-03}	3.88×10^{-06}	2.34×10^{-05}	1.7×10^{-03}
21	6.73×10^{-05}	3.39×10^{-03}	6.18×10^{-05}	1.69×10^{-05}	1.83×10^{-03}
24	2.79×10^{-04}	2.48×10^{-03}	3.28×10^{-05}	3.27×10^{-05}	1.23×10^{-03}

TABLE 9. Error for setting $\eta = 0.25$.

Input	Absolute Error in Susceptble hosts	Absolute Error in Exposed hosts	Absolute Error in Infected hosts	Absolute Error in Quarantined hosts	Absolute Error in Patched hosts
0	5.059×10^{-03}	1.054×10^{-01}	8.79×10^{-04}	4.145×10^{-03}	2.59×10^{-03}
3	1.24×10^{-04}	8.97×10^{-03}	9.73×10^{-05}	5.735×10^{-04}	1.28×10^{-04}
6	4.690×10^{-04}	3.11×10^{-06}	1.182×10^{-04}	3.629×10^{-04}	4.45×10^{-05}
9	2.00×10^{-04}	1.43×10^{-03}	1.0×10^{-04}	1.44×10^{-04}	1.37×10^{-04}
12	1.32×10^{-04}	7.3×10^{-04}	4.57×10^{-05}	3.79×10^{-04}	1.19×10^{-04}
15	8.22×10^{-05}	3.51×10^{-05}	4.88×10^{-05}	8.60×10^{-05}	6.39×10^{-05}
18	4.48×10^{-05}	1.046×10^{-03}	3.94×10^{-05}	4.182×10^{-04}	3.72×10^{-05}
21	1.30×10^{-07}	3.193×10^{-04}	6.40×10^{-06}	1.21×10^{-04}	4.01×10^{-05}
24	1.27×10^{-05}	6.49×10^{-04}	1.36×10^{-05}	7.149×10^{-04}	2.79×10^{-07}

TABLE 10. Error by setting $\eta = 0.5$.

Input	Absolute Error in Susceptble hosts	Absolute Error in Exposed hosts	Absolute Error in Infected hosts	Absolute Error in Quarantined hosts	Absolute Error in Pached hosts
0	2.94×10^{-03}	1.07×10^{-01}	1.50×10^{-05}	3.233×10^{-3}	3.432×10^{-3}
3	3.01×10^{-05}	8.00×10^{-04}	3.714×10^{-03}	1.563×10^{-04}	1.11042×10^{-04}
6	6.17×10^{-05}	5.40×10^{-03}	1.998×10^{-03}	2.027×10^{-04}	5.599×10^{-04}
9	6.86×10^{-05}	2.786×10^{-03}	1.094×10^{-03}	1.034×10^{-04}	4.369×10^{-04}
12	8.89×10^{-05}	1.499×10^{-03}	5.04×10^{-04}	1.02×10^{-04}	8.44×10^{-05}
15	1.18×10^{-04}	1.031×10^{-03}	2.624×10^{-04}	2.321×10^{-04}	1.16×10^{-04}
18	9.72×10^{-05}	2.23×10^{-05}	2.198×10^{-04}	1.465×10^{-04}	1.303×10^{-04}
21	3.58×10^{-05}	3.90×10^{-05}	4.93×10^{-06}	1.149×10^{-04}	6.85×10^{-05}
24	5.47×10^{-05}	1.35×10^{-04}	2.69036×10^{-04}	3.186×10^{-04}	1.618×10^{-04}

TABLE 11. Statistical analysis.

		functions	Hidden Neurons	Training	Validation	Testing	Gradient	Mu	Epochs	Regression	Time(s)
Case 1	n=0.05	S(t)	10	2.6×10^{-07}	3.4×10^{-07}	4.07×10^{-07}	3.8×10^{-04}	1×10^{-06}	1000	1	<1
		E(t)	10	9.43×10^{-06}	9.76×10^{-06}	1.06×10^{-05}	3.5×10^{-04}	1×10^{-05}	1000	1	<1
		Q(t)	10	8.86×10^{-09}	4.34×10^{-09}	7.5×10^{-09}	4.76×10^{-07}	1×10^{-08}	1000	1	<1
		P(t)	10	2.31×10^{-05}	2.3×10^{-05}	2.63×10^{-05}	2×10^{-03}	1×10^{-07}	1000	1	<1
		I(t)	10	8.24×10^{-09}	8.2×10^{-09}	8.68×10^{-09}	4.9×10^{-05}	1×10^{-08}	1000	1	<1
Case 2	n=0.25	S(t)	10	1.74×10^{-07}	1.82×10^{-07}	1.6×10^{-07}	8.1×10^{-05}	1×10^{-07}	1000	1	<1
		E(t)	10	6.45×10^{-05}	8.2×10^{-05}	9.1×10^{-05}	2.7×10^{-04}	1×10^{-06}	1000	1	<1
		Q(t)	10	2.66×10^{-07}	3.2×10^{-07}	2.5×10^{-07}	9.4×10^{-06}	1×10^{-06}	1000	1	<1
		P(t)	10	5.96×10^{-08}	6.48×10^{-08}	1.2×10^{-07}	1.4×10^{-05}	1×10^{-07}	1000	1	<1
		I(t)	10	1.97×10^{-08}	1.8×10^{-08}	2.20×10^{-08}	8.5×10^{-05}	1×10^{-08}	1000	1	<1
Case 3	n=0.5	S(t)	10	5.93×10^{-08}	6.8×10^{-08}	5.11×10^{-08}	7×10^{-04}	1×10^{-08}	1000	1	<1
		E(t)	10	6.65×10^{-05}	8.5×10^{-05}	9.4×10^{-05}	2.4×10^{-02}	1×10^{-07}	973	1	<1
		Q(t)	10	7.52×10^{-08}	9.29×10^{-08}	1.39×10^{-07}	2.6×10^{-04}	1×10^{-08}	1000	1	<1
		P(t)	10	2.6×10^{-07}	2.8×10^{-07}	3.9×10^{-07}	3.8×10^{-04}	1×10^{-09}	1000	1	<1
		I(t)	10	3.1×10^{-06}	2.4×10^{-06}	3.20×10^{-06}	4.4×10^{-03}	1×10^{-08}	391	1	<1

of training epochs rises. Further observation reveals that the lines generated from data obtained throughout the testing, validation, and training stages of the ANN converge to the most effective line, which is denoted by dotted lines in the (1000, 385, 967) epochs. When the ANN achieves the lowest possible value for MSE in the (1000, 385, 967) epochs, which signifies the conclusion of the training phase after the repetition of countless epochs, the model's training is optimally complete. This process shows that the ANN models' high-performance training period is over.

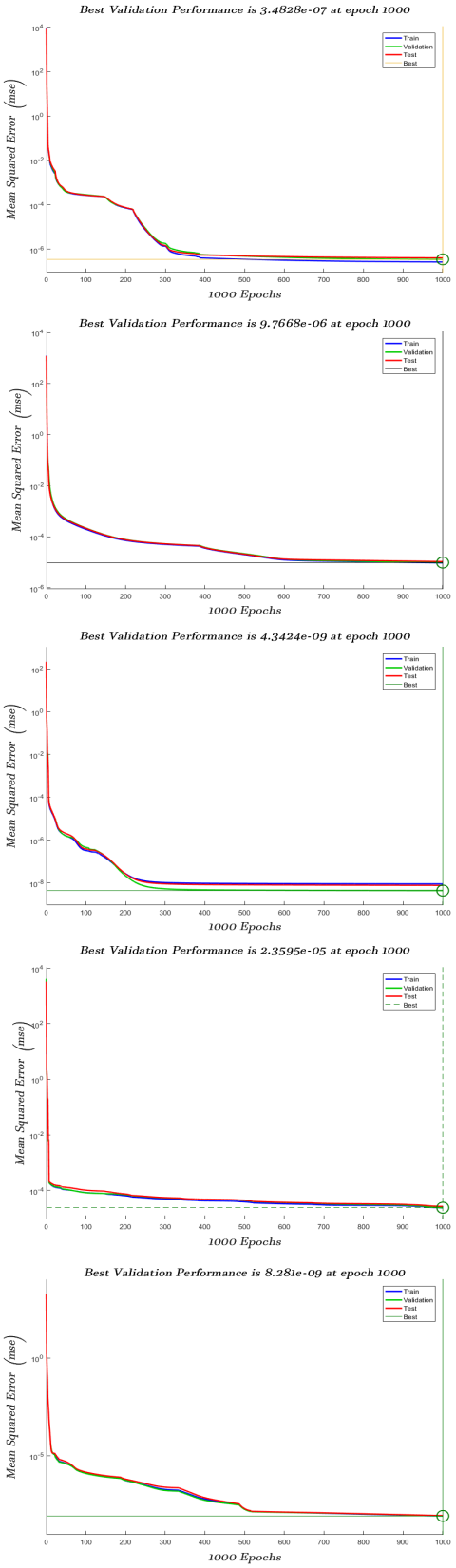


FIGURE 9. The mean square error of the LMB-NN for the consequences of an insider threat on the removal of patches for case 1.

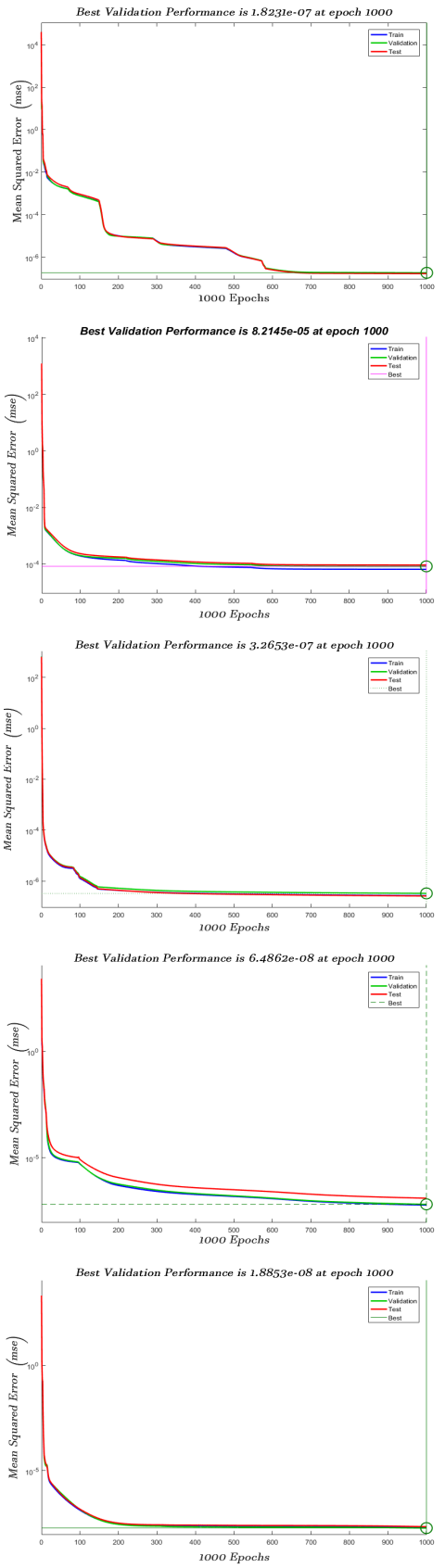


FIGURE 10. The mean square error of the LMB-NN for the consequences of an insider threat on the removal of patches for case 2.

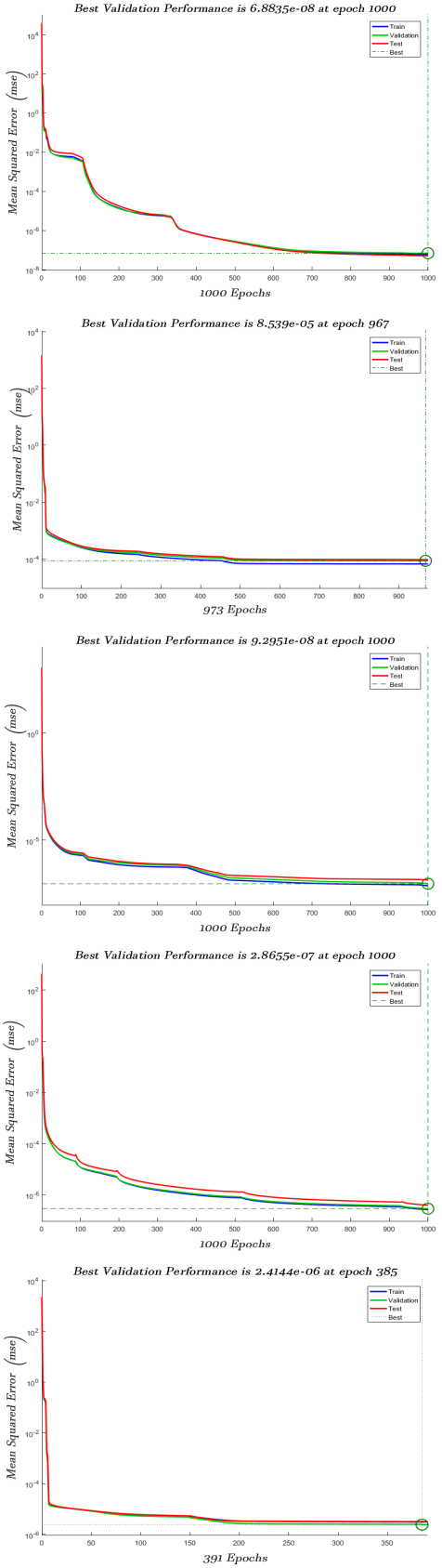


FIGURE 11. The mean square error of the LMB-NN for the consequences of an insider threat on the removal of patches for case 3.

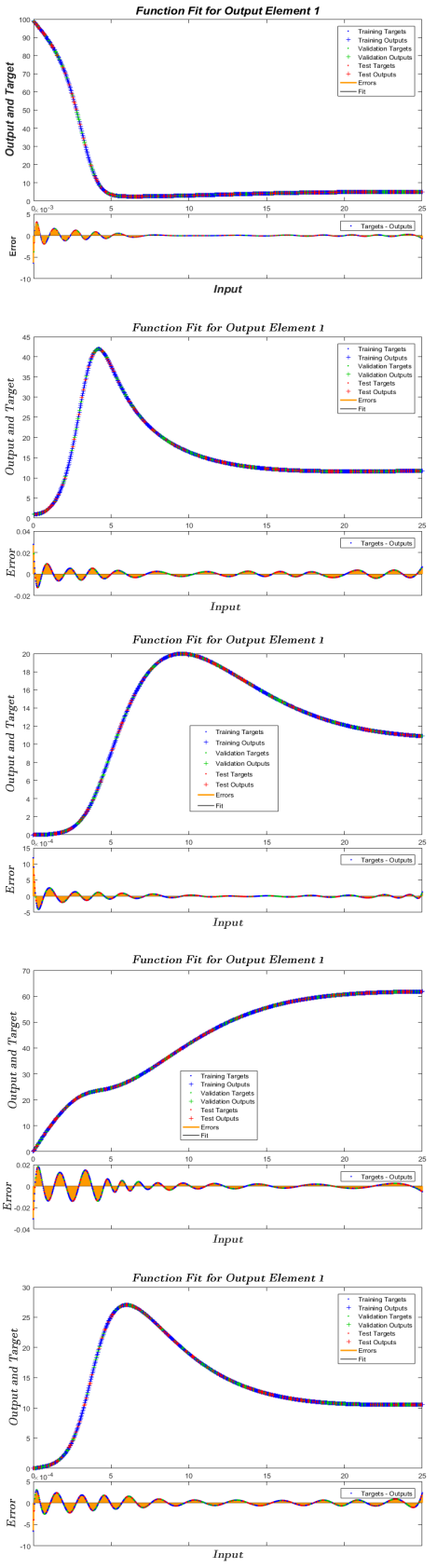


FIGURE 12. Performing a comparative analysis of the approximate solutions that were generated by LMB-NN and the numerical solutions for case 1.

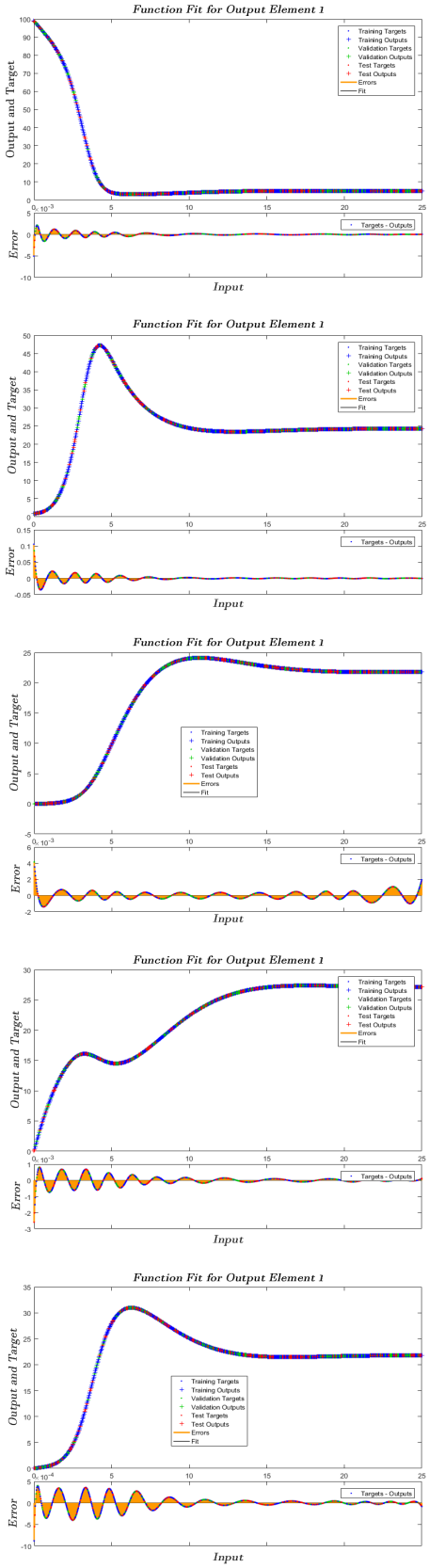


FIGURE 13. Performing a comparative analysis of the approximate solutions that were generated by LMB-NN and the numerical solutions for case 2.

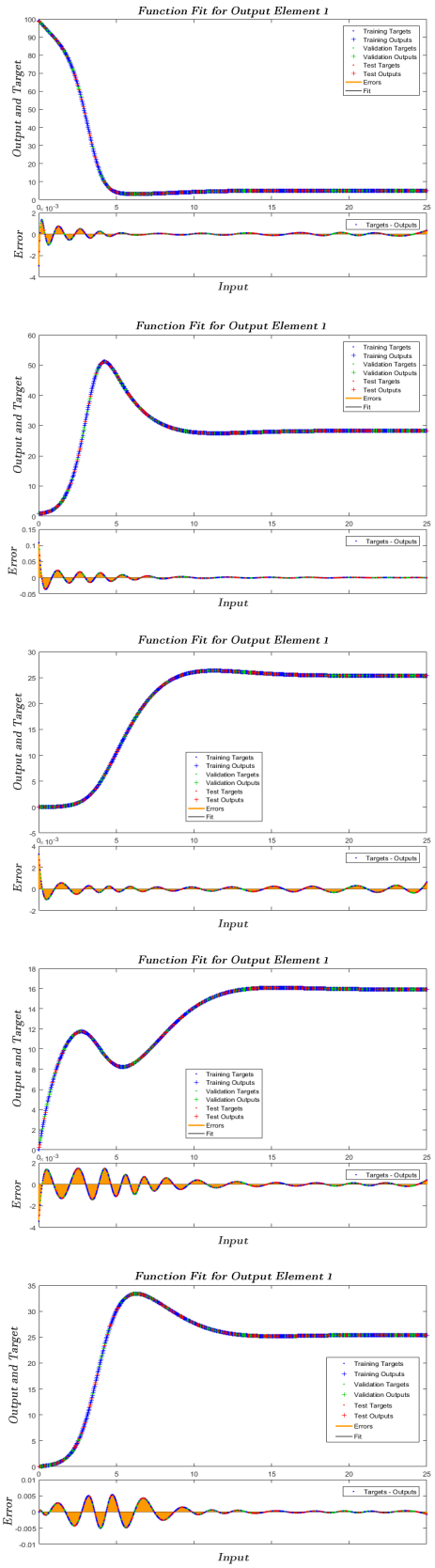


FIGURE 14. Performing a comparative analysis of the approximate solutions that were generated by LMB-NN and the numerical solutions for case 3.

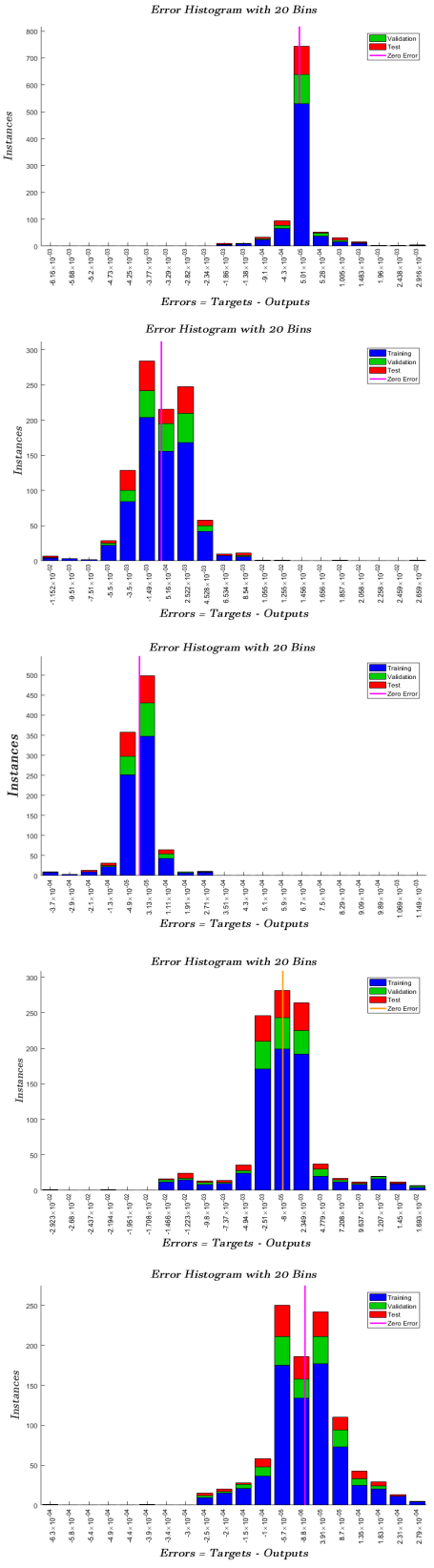


FIGURE 15. Histogram study of the differences between the target data and approximative solutions for the case 1.

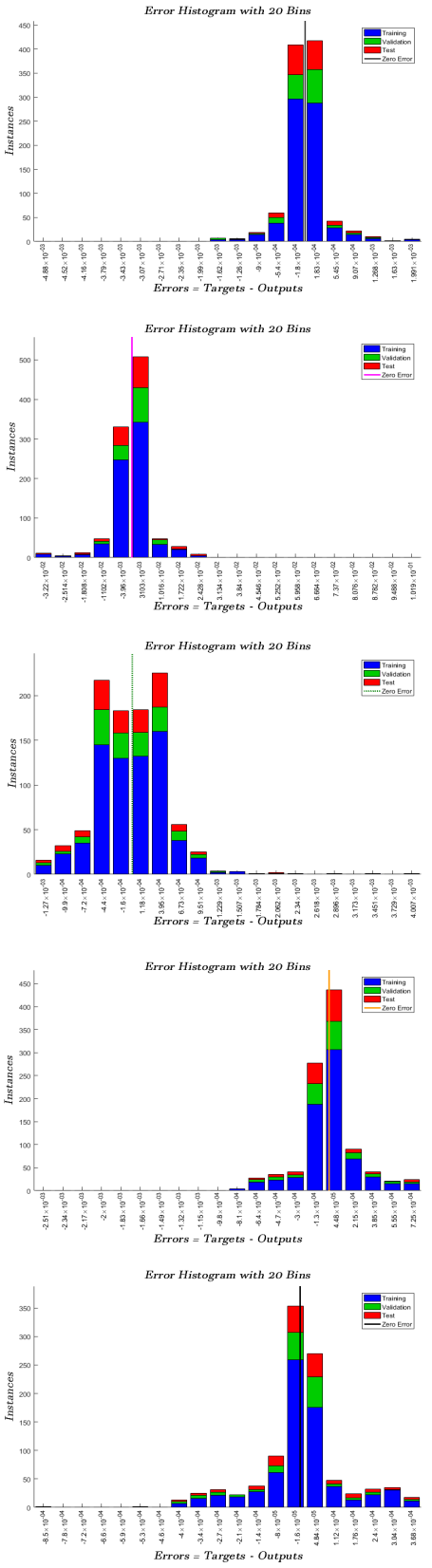


FIGURE 16. Histogram study of the differences between the target data and approximative solutions for the case 2.

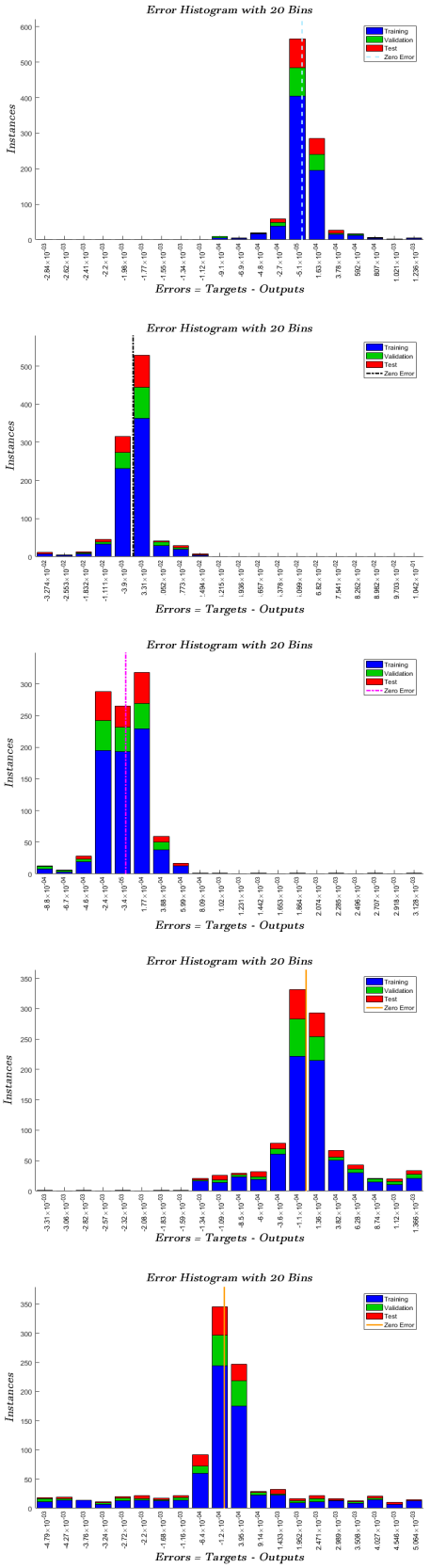


FIGURE 17. Histogram study of the differences between the target data and approximative solutions for the case 3.

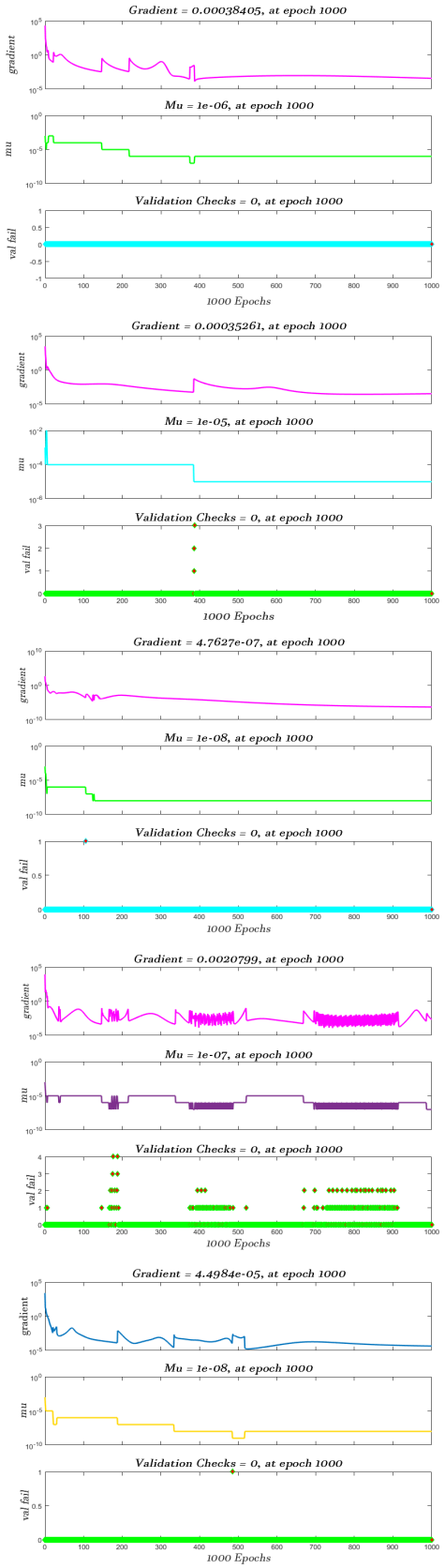


FIGURE 18. LMB-NN Performance depending on gradient, mu, and validation failures throughout the optimization process for case 3.

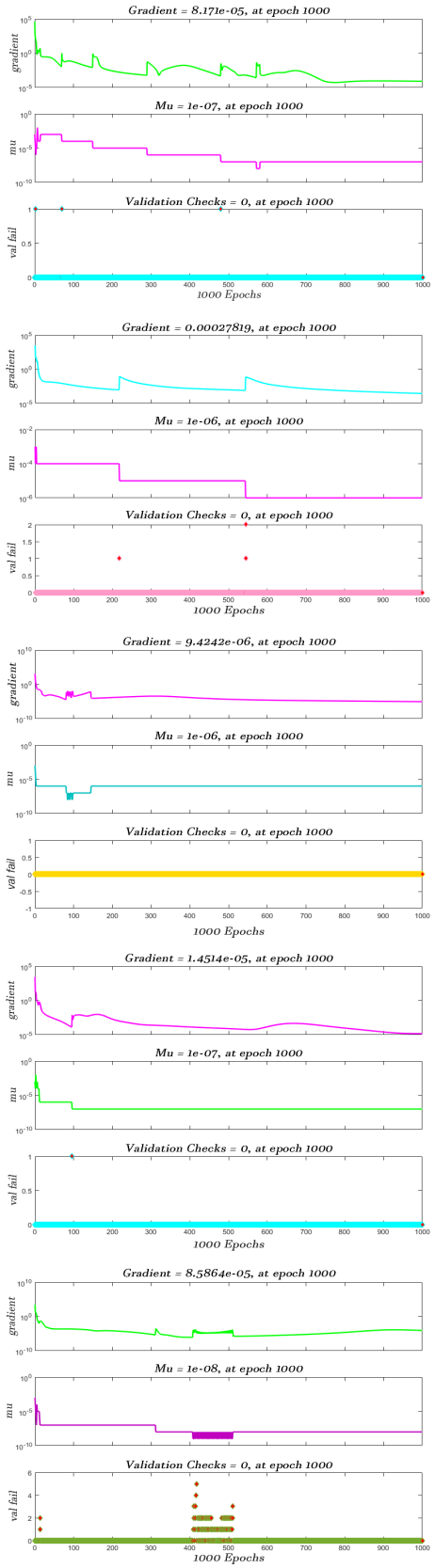


FIGURE 19. LMB-NN Performance depending on gradient, mu, and validation failures throughout the optimization process for case 2.

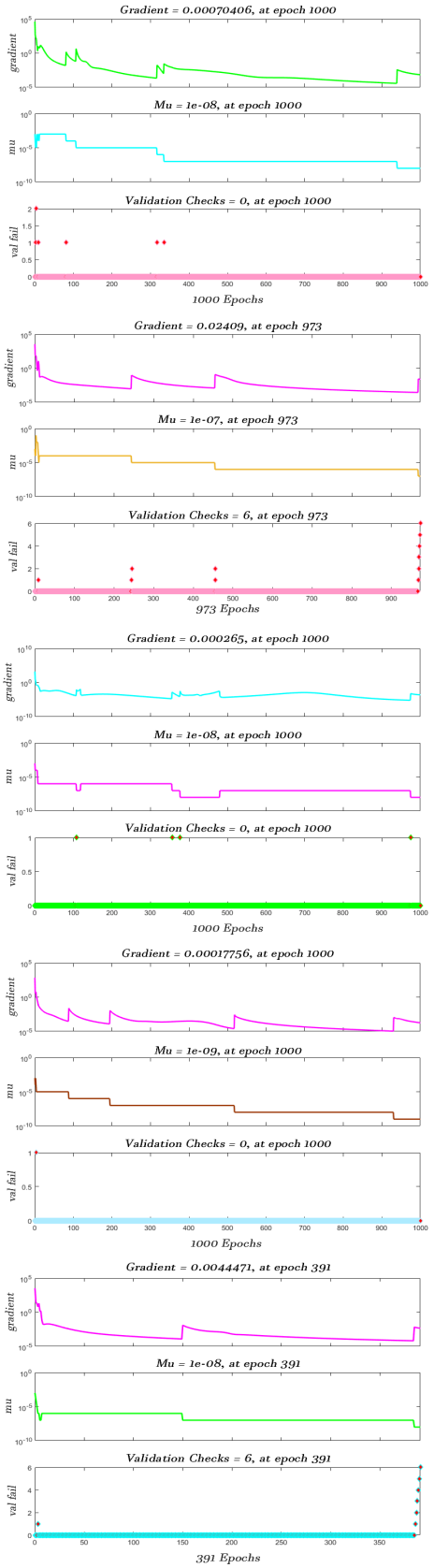


FIGURE 20. LMB-NN Performance depending on gradient, mu, and validation failures throughout the optimization process for case 3.

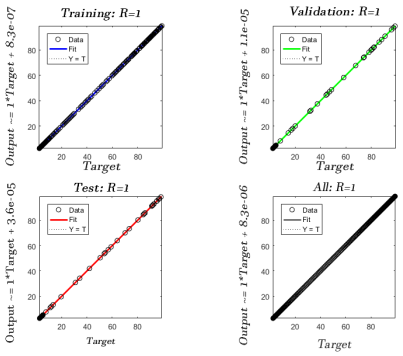


FIGURE 21. Analyzing the Role of Insider Threat in the removal of Patches Using Regression.

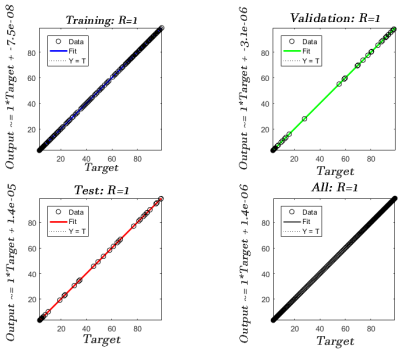


FIGURE 22. Analyzing the Role of Insider Threat in the removal of Patches Using Regression.

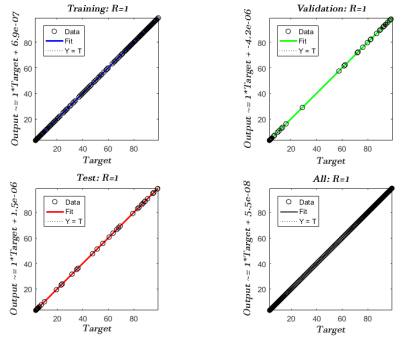


FIGURE 23. Analyzing the Role of Insider Threat in the removal of Patches Using Regression.

So the approximations for the performance values in each case are; 3.4828×10^{-07}, 9.7668×10^{-06}, 4.3424×10^{-09}, 2.3595×10^{-05}, 8.281×10^{-09} case I, 1.8231×10^{-07}, 8.2145×10^{-05}, 3.2653×10^{-07}, 6.4862×10^{-08}, 1.8853×10^{-07} for case II and similarly for case 3 the performance 6.8835×10^{-08}, 8.539×10^{-05}, 9.2951×10^{-08}, 2.8655×10^{-07}, 2.4144×10^{-06} are illustrated in Figure (9), (10) and (11) and Table (7). The effectiveness examination graphs for all MLP network frameworks are shown in Figures (12), (13), and (14), and the discrepancies between the targeted and reference solutions caused the error. The graphical depiction shows that the target outcome overlays the reference ANN model outlines for models with circumstances, demonstrating that the accuracy of the solution is validated by the (ANN) design framework. So we can say effect of changes in the worm propagation model on patch removal by varying the effects of insider threat. According to these numbers, an insider threat η has the potential to infect a bigger number of hosts over the course of time. The amount of hosts that have been patched declines while the amount of exposed and infected hosts rises as the system reaches equilibrium. This suggests that significantly impacts the equation's stability dynamics. (2). Error histogram examination is essential to assessing how well ANN models work. As demonstrated in Figures (15), (16), and (17), there is an overlap between the approximate solution's fitting and the target data, with only slight absolute errors. The error histogram graphs demonstrate that the errors acquired at each ANN model stage are relatively low. Despite this, it is still evident that mistakes tend to build up in the direction of the zero-error line. Typically, absolute errors in the solutions for S(t) are somewhere around 10^{-03} to 10^{-05}, 10^{-03} to 10^{-04}, 10^{-04} to 10^{-05} respectively. The solution of E(t) are lying around 10^{-02} to 10^{-04}, 10^{-02} to 10^{-03}, 10^{-02} to 10^{-03} respectively. Similarly the solution of Q(t) lies around 10^{-04} to 10^{-05}, 10^{-03} to 10^{-04}, 10^{-04} to 10^{-05}. Additionally P(t) solutions are all over the place 10^{-02} to 10^{-05}, 10^{-04} to 10^{-05}, 10^{-03} to 10^{-04} and likewise, the solution of Q(t) is located in the vicinity of 10^{-04} to 10^{-06}, 10^{-04} to 10^{-05}, 10^{-03} to 10^{-04}. In Figure (18), Figure (19), Figure (20), the training states (gradient, mu, validation checks) are visually shown. The plots use a progressively increasing epoch number to deliver the fluctuation in the gradient coefficient. It can be shown that the errors obtained from ANN models gradually reach optimal and optimum values after several test operations. These ANN model training results show that the created ANNs have successfully finished their training activities. Data from the training phases of artificial neural network (ANN) prototypes are shown in Figure (21), Figure (22) and Figure (23). The graph's goal values are shown on the horizontal axis, while the ANN output forecasts are shown visually on the vertical axis. The graphic representation of the data points collected during the training phase places them on the compatibility (fit) line. The solid line displays the outcomes result and the target values' best-fitting linear regression line, and the importance of R shows how they are related. The regression analysis R=1 throughout this calculation shows a precise linear relationship between the output and desired values, moreover statistical analysis is dictated in Tables (11).

VIII. CONCLUSION

Mathematical modeling of worm spread and insider risks in an air-gapped network was investigated in this study using a modified version of the SEIQV model. The model, supplied by a differential equation system, was derived from biological processes and modified for use in modeling the spread of worms via a computer network. A technique for soft computing that makes use of the supervised learning capabilities of Levenberg-Marquardt backpropagation neural networks is also used to compute the effects of insider

threats on patch removal and their impact on the provided differential equation model. The results that are presented in the figures lead one to the conclusion that as *eta* increases, the maximum amount of hosts that are both exposed and infected increases at a more gradual rate. This illustrates that an insider threat might evolve to infect a greater number of hosts by increasing its *eta*. Moreover, as the system becomes stable, fewer hosts are patched overall while an increase in exposed and infected hosts is observed. This demonstrates that *eta* has a substantial impact on the dynamics of stability of equation (2). A comprehensive graphical examination is carried out here making use of absolute errors, MSE, error histograms, regressions, and computing complexity to illustrate the robustness, efficiency, and accuracy of the constructed system.

REFERENCES

[1] J. Zhang, S. Peng, Y. Gao, Z. Zhang, and Q. Hong, "APMSA: Adversarial perturbation against model stealing attacks," *IEEE Trans. Inf. Forensics Security*, vol. 18, pp. 1667–1679, 2023.

[2] A. M. del Rey, "Mathematical modeling of the propagation of malware: A review," *Secur. Commun. Netw.*, vol. 8, no. 15, pp. 2561–2579, Oct. 2015.

[3] A. I. A. Alzahrani, M. Ayadi, M. M. Asiri, A. Al-Rasheed, and A. Ksibi, "Detecting the presence of malware and identifying the type of cyber attack using deep learning and VGG-16 techniques," *Electronics*, vol. 11, no. 22, p. 3665, Nov. 2022.

[4] Y. S. Abdulsalam and M. Hedabou, "Security and privacy in cloud computing: Technical review," *Future Internet*, vol. 14, no. 1, p. 11, Dec. 2021.

[5] K. Cao, B. Wang, H. Ding, L. Lv, R. Dong, T. Cheng, and F. Gong, "Improving physical layer security of uplink NOMA via energy harvesting jammers," *IEEE Trans. Inf. Forensics Security*, vol. 16, pp. 786–799, 2021.

[6] F. Akbar, M. Hussain, R. Mumtaz, Q. Riaz, A. W. A. Wahab, and K.-H. Jung, "Permissions-based detection of Android malware using machine learning," *Symmetry*, vol. 14, no. 4, p. 718, Apr. 2022.

[7] W. Zhang, N. Luktarhan, C. Ding, and B. Lu, "Android malware detection using TCN with bytecode image," *Symmetry*, vol. 13, no. 7, p. 1107, Jun. 2021.

[8] T. Li, T. Xia, H. Wang, Z. Tu, S. Tarkoma, Z. Han, and P. Hui, "Smartphone app usage analysis: Datasets, methods, and applications," *IEEE Commun. Surveys Tuts.*, vol. 24, no. 2, pp. 937–966, 2nd Quart., 2022.

[9] B. Li, X. Zhou, Z. Ning, X. Guan, and K.-F.-C. Yiu, "Dynamic event-triggered security control for networked control systems with cyber-attacks: A model predictive control approach," *Inf. Sci.*, vol. 612, pp. 384–398, Oct. 2022.

[10] E. Uçar, S. Uçar, F. Evirgen, and N. Özdemir, "A fractional SAIDR model in the frame of Atangana–Baleanu derivative," *Fractal Fractional*, vol. 5, no. 2, p. 32, Apr. 2021.

[11] F. Wang, Y. Zhang, C. Wang, J. Ma, and S. Moon, "Stability analysis of a SEIQV epidemic model for rapid spreading worms," *Comput. Secur.*, vol. 29, no. 4, pp. 410–418, Jun. 2010.

[12] Z. Lv, D. Chen, R. Lou, and H. Song, "Industrial security solution for virtual reality," *IEEE Internet Things J.*, vol. 8, no. 8, pp. 6273–6281, Apr. 2021.

[13] D. Moore, C. Shannon, and K. Claffy, "Code-Red: A case study on the spread and victims of an Internet worm," in *Proc. 2nd ACM SIGCOMM Workshop Internet Measurment (IMW)*, 2002, pp. 273–284.

[14] Y. Wang, S. Wen, Y. Xiang, and W. Zhou, "Modeling the propagation of worms in networks: A survey," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 2, pp. 942–960, 2nd Quart., 2014.

[15] S. Eshghi, M. H. R. Khouzani, S. Sarkar, and S. S. Venkatesh, "Optimal patching in clustered malware epidemics," *IEEE/ACM Trans. Netw.*, vol. 24, no. 1, pp. 283–298, Feb. 2016.

[16] D. A. Kumar and S. K. Das, "Machine learning approach for malware detection and classification using malware analysis framework," *Int. J. Intell. Syst. Appl. Eng.*, vol. 11, no. 1, pp. 330–338, 2023.

[17] P. Lee, "Passivity framework for modeling, composing and mitigating cyber attacks," M.S. thesis, Dept. Elect. Eng., Univ. Washington, USA, 2016.

[18] D. Zhou, M. Sheng, J. Li, and Z. Han, "Aerospace integrated networks innovation for empowering 6G: A survey and future challenges," *IEEE Commun. Surveys Tuts.*, vol. 25, no. 2, pp. 975–1019, 2nd Quart., 2023.

[19] M. Guri, M. Monitz, and Y. Elovici, "Bridging the air gap between isolated networks and mobile phones in a practical cyber-attack," *ACM Trans. Intell. Syst. Technol.*, vol. 8, no. 4, pp. 1–25, Jul. 2017.

[20] T. Li, Y. Li, M. A. Hoque, T. Xia, S. Tarkoma, and P. Hui, "To what extent we repeat ourselves? Discovering daily activity patterns across mobile app usage," *IEEE Trans. Mobile Comput.*, vol. 21, no. 4, pp. 1492–1507, Apr. 2022.

[21] J. L. Safar, M. Tummala, J. C. McEachen, and C. Bollmann, "Modeling worm propagation and insider threat in air-gapped network using modified SEIQV model," in *Proc. 13th Int. Conf. Signal Process. Commun. Syst. (ICSPCS)*, Dec. 2019, pp. 1–6.

[22] H.-Y. Jin and Z.-A. Wang, "Global stabilization of the full attraction-repulsion Keller–Segel system," 2019, *arXiv:1905.05990*.

[23] W. Lyu and Z.-A. Wang, "Global classical solutions for a class of reaction-diffusion system with density-suppressed motility," 2021, *arXiv:2102.08042*.

[24] M. Sulaiman, N. A. Khan, F. S. Alshammari, and G. Laouini, "Performance of heat transfer in micropolar fluid with isothermal and isoflux boundary conditions using supervised neural networks," *Mathematics*, vol. 11, no. 5, p. 1173, Feb. 2023.

[25] D. Bernoulli, "Essai d'une nouvelle analyse de la mortalité causée par la petite vérole, et des avantages de l'inoculation pour la prévenir," *Histoire De l'Acad., Roy. Sci. (Paris) Avec Mem*, pp. 1–45, 1760.

[26] A. Li, C. Masouros, A. L. Swindlehurst, and W. Yu, "1-bit massive MIMO transmission: Embracing interference with symbol-level precoding," *IEEE Commun. Mag.*, vol. 59, no. 5, pp. 121–127, May 2021.

[27] S. Eshghi, "Optimal control of epidemics in the presence of heterogeneity," Dept. Elect. Syst. Eng., Univ. Pennsylvania, Philadelphia, PA, USA, Tech. Rep. 1704, 2015.

[28] J. Ma and J. Hu, "Safe consensus control of cooperative-competitive multi-agent systems via differential privacy," *Kybernetika*, vol. 58, no. 3, pp. 426–439, Sep. 2022.

[29] A. Prajapati and B. K. Mishra, "Cyber attack and control techniques," in *Information Systems Design and Intelligent Applications*. Cham, Switzerland: Springer, 2015, pp. 157–166.

[30] S. Tang, D. Myers, and J. Yuan, "Modified SIS epidemic model for analysis of virus spread in wireless sensor networks," *Int. J. Wireless Mobile Comput.*, vol. 6, no. 2, p. 99, 2013.

[31] B. K. Mishra and S. K. Pandey, "Dynamic model of worm propagation in computer network," *Appl. Math. Model.*, vol. 38, nos. 7–8, pp. 2173–2179, Apr. 2014.

[32] Z. Masood, M. A. Z. Raja, N. I. Chaudhary, K. M. Cheema, and A. H. Milyani, "Fractional dynamics of Stuxnet virus propagation in industrial control systems," *Mathematics*, vol. 9, no. 17, p. 2160, Sep. 2021.

[33] D. Zhang and Y. Wang, "SIRS: Internet worm propagation model and application," in *Proc. Int. Conf. Electr. Control Eng.*, Jun. 2010, pp. 3029–3032.

[34] F. K. Batista, Á. M. del Rey, S. Quintero-Bonilla, and A. Queiruga-Dios, "A SEIR model for computer virus spreading based on cellular automata," in *Proc. Int. Joint Conf.*, León, Spain: Springer, 2017, pp. 641–650.

[35] G. Liu, J. Chen, Z. Liang, Z. Peng, and J. Li, "Dynamical analysis and optimal control for a SEIR model based on virus mutation in WSNs," *Mathematics*, vol. 9, no. 9, p. 929, Apr. 2021.

[36] B. K. Mishra and D. K. Saini, "SEIRS epidemic model with delay for transmission of malicious objects in computer network," *Appl. Math. Comput.*, vol. 188, no. 2, pp. 1476–1482, May 2007.

[37] B. K. Mishra and N. Keshri, "Mathematical model on the transmission of worms in wireless sensor network," *Appl. Math. Model.*, vol. 37, no. 6, pp. 4103–4111, Mar. 2013.

[38] A. Li, C. Masouros, B. Vucetic, Y. Li, and A. L. Swindlehurst, "Interference exploitation precoding for multi-level modulations: Closed-form solutions," *IEEE Trans. Commun.*, vol. 69, no. 1, pp. 291–308, Jan. 2021.

[39] B. Cheng, D. Zhu, S. Zhao, and J. Chen, "Situation-aware IoT service coordination using the event-driven SOA paradigm," *IEEE Trans. Netw. Service Manage.*, vol. 13, no. 2, pp. 349–361, Jun. 2016.

[40] S. Karageorgiou and V. Karyotis, "Markov-based malware propagation modeling and analysis in multi-layer networks," *Network*, vol. 2, no. 3, pp. 456–478, Sep. 2022.
[41] H. Jiang, Z. Xiao, Z. Li, J. Xu, F. Zeng, and D. Wang, "An energy-efficient framework for Internet of Things underlaying heterogeneous small cell networks," *IEEE Trans. Mobile Comput.*, vol. 21, no. 1, pp. 31–43, Jan. 2022.
[42] P. Chen, H. Liu, R. Xin, T. Carval, J. Zhao, Y. Xia, and Z. Zhao, "Effectively detecting operational anomalies in large-scale IoT data infrastructures by using a GAN-based predictive model," *Comput. J.*, vol. 65, no. 11, pp. 2909–2925, Nov. 2022.
[43] F. Guo, W. Zhou, Q. Lu, and C. Zhang, "Path extension similarity link prediction method based on matrix algebra in directed networks," *Comput. Commun.*, vol. 187, pp. 83–92, Apr. 2022.
[44] H. Zhu, M. Xue, Y. Wang, G. Yuan, and X. Li, "Fast visual tracking with Siamese oriented region proposal network," *IEEE Signal Process. Lett.*, vol. 29, pp. 1437–1441, 2022.
[45] Z. Xiong, X. Li, X. Zhang, M. Deng, F. Xu, B. Zhou, and M. Zeng, "A comprehensive confirmation-based selfish node detection algorithm for socially aware networks," *J. Signal Process. Syst.*, vol. 95, pp. 1–19, Apr. 2023.
[46] S. Hosseini and M. A. Azgomi, "A model for malware propagation in scale-free networks based on rumor spreading process," *Comput. Netw.*, vol. 108, pp. 97–107, Oct. 2016.
[47] D. Le, K. Dang, Q. Nguyen, S. Alhelaly, and A. Muthanna, "A behavior-based malware spreading model for vehicle-to-vehicle communications in VANET networks," *Electronics*, vol. 10, no. 19, p. 2403, Oct. 2021.
[48] B. Li, M. Zhang, Y. Rong, and Z. Han, "Transceiver optimization for wireless powered time-division duplex MU-MIMO systems: Non-robust and robust designs," *IEEE Trans. Wireless Commun.*, vol. 21, no. 6, pp. 4594–4607, Jun. 2022.
[49] D. Liu, Z. Cao, H. Jiang, S. Zhou, Z. Xiao, and F. Zeng, "Concurrent low-power listening: A new design paradigm for duty-cycling communication," *ACM Trans. Sensor Netw.*, vol. 19, no. 1, pp. 1–24, Feb. 2023.
[50] Q. Ni, J. Guo, W. Wu, H. Wang, and J. Wu, "Continuous influence-based community partition for social networks," *IEEE Trans. Netw. Sci. Eng.*, vol. 9, no. 3, pp. 1187–1197, May 2022.
[51] H. Yang, X. Zhao, Q. Yao, A. Yu, J. Zhang, and Y. Ji, "Accurate fault location using deep neural evolution network in cloud data center interconnection," *IEEE Trans. Cloud Comput.*, vol. 10, no. 2, pp. 1402–1412, Apr. 2022.
[52] X. Liu, T. Shi, G. Zhou, M. Liu, Z. Yin, L. Yin, and W. Zheng, "Emotion classification for short texts: An improved multi-label method," *Humanities Social Sci. Commun.*, vol. 10, no. 1, p. 306, Jun. 2023.
[53] H. Yang, J. Yuan, C. Li, G. Zhao, Z. Sun, Q. Yao, B. Bao, A. V. Vasilakos, and J. Zhang, "BrainIoT: Brain-like productive services provisioning with federated learning in industrial IoT," *IEEE Internet Things J.*, vol. 9, no. 3, pp. 2014–2024, Feb. 2022.
[54] F. Wang, F. Yang, Y. Zhang, and J. Ma, "Stability analysis of a SEIQRS model with graded infection rates for Internet worms," *J. Comput.*, vol. 9, no. 10, Oct. 2014, pp. 2420–2427.
[55] Z. Zhao, S. Yang, and D. Zhao, "A new framework for visual classification of multi-channel malware based on transfer learning," *Appl. Sci.*, vol. 13, no. 4, p. 2484, Feb. 2023.
[56] H. Jiang, M. Wang, P. Zhao, Z. Xiao, and S. Dustdar, "A utility-aware general framework with quantifiable privacy preservation for destination prediction in LBSs," *IEEE/ACM Trans. Netw.*, vol. 29, no. 5, pp. 2228–2241, Oct. 2021.
[57] T. Spyridopoulos, K. Maraslis, A. Mylonas, T. Tryfonas, and G. Oikonomou, "A game theoretical method for cost-benefit analysis of malware dissemination prevention," *Inf. Secur. J., A Global Perspective*, vol. 24, nos. 4–6, pp. 164–176, Oct. 2015.
[58] G. Liu, B. Peng, and X. Zhong, "Epidemic analysis of wireless rechargeable sensor networks based on an attack-defense game model," *Sensors*, vol. 21, no. 2, p. 594, Jan. 2021.
[59] M. Kumar, B. K. Mishra, and T. C. Panda, "Effect of quarantine & vaccination on infectious nodes in computer network," *Int. J. Comput. Netw. Appl.*, vol. 2, no. 2, pp. 92–98, 2015.
[60] D. M. Nicol, "The impact of stochastic variance on worm propagation and detection," in *Proc. 4th ACM workshop Recurring malcode*, Nov. 2006, pp. 57–64.
[61] Z. Chen and C. Ji, "Spatial–temporal modeling of malware propagation in networks," *IEEE Trans. Neural Netw.*, vol. 16, no. 5, pp. 1291–1303, Sep. 2005.
[62] Z. Wang, X. Fan, and Q. Han, "Global stability of deterministic and stochastic multigroup SEIQR models in computer network," *Appl. Math. Model.*, vol. 37, nos. 20–21, pp. 8673–8686, Nov. 2013.
[63] S. König, S. Schauer, and S. Rass, "A stochastic framework for prediction of malware spreading in heterogeneous networks," in *Proc. Nordic Conf. Secure IT Syst.* Berlin, Germany: Springer, 2016, pp. 67–81.
[64] F. Meng, X. Xiao, and J. Wang, "Rating the crisis of online public opinion using a multi-level index system," 2022, *arXiv:2207.14740*.
[65] X. Liu, G. Zhou, M. Kong, Z. Yin, X. Li, L. Yin, and W. Zheng, "Developing multi-labelled corpus of Twitter short texts: A semi-automatic method," *Systems*, vol. 11, no. 8, p. 390, Aug. 2023.
[66] L. Feng, X. Liao, Q. Han, and H. Li, "Dynamical analysis and control strategies on malware propagation model," *Appl. Math. Model.*, vol. 37, nos. 16–17, pp. 8225–8236, Sep. 2013.
[67] P. Mohan, N. Subramani, Y. Alotaibi, S. Alghamdi, O. I. Khalaf, and S. Ulaganathan, "Improved metaheuristics-based clustering with multihop routing protocol for underwater wireless sensor networks," *Sensors*, vol. 22, no. 4, p. 1618, Feb. 2022.
[68] Y. Chen, L. Zhu, Z. Hu, S. Chen, and X. Zheng, "Risk propagation in multilayer heterogeneous network of coupled system of large engineering project," *J. Manage. Eng.*, vol. 38, no. 3, May 2022, Art. no. 04022003.
[69] X. Li and S. Xu, "A stochastic modeling of coordinated internal and external attacks," *Submitted Dependable Syst. Netw.*, 2007.
[70] B. K. Mishra and N. Jha, "SEIQRS model for the transmission of malicious objects in computer network," *Appl. Math. Model.*, vol. 34, no. 3, pp. 710–715, Mar. 2010.
[71] Y. Shen, N. Ding, H.-T. Zheng, Y. Li, and M. Yang, "Modeling relation paths for knowledge graph completion," *IEEE Trans. Knowl. Data Eng.*, vol. 33, no. 11, pp. 3607–3617, Nov. 2021.
[72] I. Farag, M. A. Shouman, T. S. Sobh, and H. El-Fiqi, "Intelligent system for worm detection," *Int. Arab. J. e Technol.*, vol. 1, no. 1, pp. 58–67, 2009.
[73] K. Wang, B. Zhang, F. Alenezi, and S. Li, "Communication-efficient surrogate quantile regression for non-randomly distributed system," *Inf. Sci.*, vol. 588, pp. 425–441, Apr. 2022.
[74] J. Dong, J. Hu, Y. Zhao, and Y. Peng, "Opinion formation analysis for expressed and private opinions (EPOs) models: Reasoning private opinions from behaviors in group decision-making systems," *Expert Syst. Appl.*, vol. 236, Feb. 2024, Art. no. 121292.
[75] X. Qin, Z. Liu, Y. Liu, S. Liu, B. Yang, L. Yin, M. Liu, and W. Zheng, "User OCEAN personality model construction method using a BP neural network," *Electronics*, vol. 11, no. 19, p. 3022, Sep. 2022.
[76] Z. Khan, S. Zuhra, S. Islam, M. A. Z. Raja, and A. Ali, "Modeling and simulation of Maxwell nanofluid flows in the presence of Lorentz and Darcy–Forchheimer forces: Toward a new approach on Buongiorno's model using artificial neural network (ANN)," *Eur. Phys. J. Plus*, vol. 138, no. 1, p. 107, Feb. 2023.
[77] X. Xie, B. Xie, D. Xiong, M. Hou, J. Zuo, G. Wei, and J. Chevallier, "New theoretical ISM-K2 Bayesian network model for evaluating vaccination effectiveness," *J. Ambient Intell. Humanized Comput.*, vol. 14, pp. 12789–12805, Jul. 2022.
[78] Z. Lv, L. Qiao, J. Li, and H. Song, "Deep-learning-enabled security issues in the Internet of Things," *IEEE Internet Things J.*, vol. 8, no. 12, pp. 9531–9538, Jun. 2021.