Interdisciplinary Journal of Information, Knowledge, and Management

*An Official Publication of the Informing Science Institute*
*InformingScience.org*

*IJIKM.org*

**Volume 16, 2021**

# SECURITY AS A SOLUTION: AN INTRUSION DETECTION SYSTEM USING A NEURAL NETWORK FOR IOT ENABLED HEALTHCARE ECOSYSTEM

| | | |
|---|---|---|
| Anshul Jain* | AIIT, Amity University, Noida, India | anshuljain13@gmail.com |
| Tanya Singh | ASET, Amity University, Noida, India | tsingh2@amity.edu |
| Satyendra K Sharma | Modern Institute of Technology & Research Center, Alwar, India | skpacific323@gmail.com |

* Corresponding author

## ABSTRACT

| | |
|---|---|
| Aim/Purpose | The primary purpose of this study is to provide a cost-effective and artificial intelligence enabled security solution for IoT enabled healthcare ecosystem. It helps to implement, improve, and add new attributes to healthcare services. The paper aims to develop a method based on an artificial neural network technique to predict suspicious devices based on bandwidth usage. |
| Background | COVID has made it mandatory to make medical services available online to every remote place. However, services in the healthcare ecosystem require fast, uninterrupted facilities while securing the data flowing through them. The solution in this paper addresses both the security and uninterrupted services issue. This paper proposes a neural network based solution to detect and disable suspicious devices without interrupting critical and life-saving services. |
| Methodology | This paper is an advancement on our previous research, where we performed manual knowledge-based intrusion detection. In this research, all the experiments were executed in the healthcare domain. The mobility pattern of the devices was divided into six parts, and each one is assigned a dedicated slice. The security module regularly monitored all the clients connected to slices, and machine learning was used to detect and disable the problematic or suspicious devices. We have used MATLAB's neural network to train the dataset and automatically detect and disable suspicious devices. The different network architectures and different training algorithms (Levenberg–Marquardt and Bayesian Framework) in MATLAB software have attempted to achieve |

Accepting Editor Man Fung (Kelvin) LO | Received: May 26, 2021 | Revised: July 12, July 20, July 22, 2021 | Accepted: July 23, 2021.
Cite as: Jain, A., Singh, T., & Sharma, S. K. (2021). Security as a solution: An intrusion detection system using a neural network for IoT enabled healthcare ecosystem. *Interdisciplinary Journal of Information, Knowledge, and Management*, *16,* 331-369. https://doi.org/10.28945/4838

| | |
|---|---|
| | more precise values with different properties. Five iterations of training were executed and compared to get the best result of R=99971. We configured the application to handle the four most applicable use cases. We also performed an experimental application simulation for the assessment and validation of predictions. |
| Contribution | This paper provides a security solution for the IoT enabled healthcare system. The architectures discussed suggest an end-to-end solution on the sliced network. Efficient use of artificial neural networks detects and block suspicious devices. Moreover, the solution can be modified, configured and deployed in many other ecosystems like home automation. |
| Findings | This simulation is a subset of the more extensive simulation previously performed on the sliced network to enhance its security. This paper trained the data using a neural network to make the application intelligent and robust. This enhancement helps detect suspicious devices and isolate them before any harm is caused on the network. The solution works both for an intrusion detection and prevention system by detecting and blocking them from using network resources. The result concludes that using multiple hidden layers and a non-linear transfer function, logsig improved the learning and results. |
| Recommendations for Practitioners | Everything from offices, schools, colleges, and e-consultation is currently happening remotely. It has caused extensive pressure on the network where the data flowing through it has increased multifold. Therefore, it becomes our joint responsibility to provide a cost-effective and sustainable security solution for IoT enabled healthcare services. Practitioners can efficiently use this affordable solution compared to the expensive security options available in the commercial market and deploy it over a sliced network. The solution can be implemented by NGOs and federal governments to provide secure and affordable healthcare monitoring services to patients in remote locations. |
| Recommendations for Researchers | Research can take this solution to the next level by integrating artificial intelligence into all the modules. They can augment this solution by making it compatible with the federal government's data privacy laws. Authentication and encryption modules can be integrated to enhance it further. |
| Impact on Society | COVID has given massive exposure to the healthcare sector since last year. With everything online, data security and privacy is the next most significant concern. This research can be of great support to those working for the security of health care services. This paper provides "Security as a Solution", which can enhance the security of an otherwise less secure ecosystem. The healthcare use cases discussed in this paper address the most common security issues in the IoT enabled healthcare ecosystem. |
| Future Research | We can enhance this application by including data privacy modules like authentication and authorisation, data encryption and help to abide by the federal privacy laws. In addition, machine learning and artificial intelligence can be extended to other modules of this application. Moreover, this experiment can be easily applicable to many other domains like e-homes, e-offices and many others. For example, e-homes can have devices like kitchen equipment, rooms, dining, cars, bicycles, and smartwatches. Therefore, one can use this application to monitor these devices and detect any suspicious activity. |

Keywords IoT, Network Slicing, Software Defined Network (SDN), IoT Ecosystem, IoT Security, Healthcare, Intrusion Detection System (IDS), Machine Learning (ML), Artificial Neural Network (ANN)

## INTRODUCTION

Healthcare is the backbone of any country, and it empowers the economy and population. Access to the standard healthcare system is the fundamental right of any citizen, and securing the individual's healthcare data is equally important. The growing usage of healthcare services and its high dependence on telecom and the internet becomes equally essential to secure its ecosystem. With massive exposure to healthcare and the economic disruption caused by COVID-19 in 2020 and 2021, it becomes crucial for us to analyse and secure the data utilised and find a sustainable solution. COVID-19 has made remote working a new normal. Everything from offices, schools, colleges, and consultations is currently happening remotely, and it has caused extensive pressure on the network where the data flowing through it has increased multifold. Excessive data in the air and on the wire has opened a pandora's box for us and bliss for black hat hackers. Managing this data is very costly, and there are very few open-source solutions available that can work on the amount of data being generated. Geekflare (2020) shares 12 open source IoT tools and applications that can be integrated into our IoT platforms or develop our solutions using standard security protocols. Moreover, they do not provide any specialised security solution to ensure modular security as provided in our solution. Therefore, it becomes our joint responsibility to provide a cost-effective and sustainable solution for everyone to implement, improve and add new security attributes to our healthcare services.

Services in the healthcare ecosystem consist of two parts: (a) providing uninterrupted service, and (b) securing the data flowing through it. The two healthcare requirements mentioned above are the existing problems, and the objective of this paper is to address both of them. The explanation provided in this paper provides "Security as a Solution", which can enhance security using neural networks and provide uninterrupted service for critical and life-saving devices. The healthcare use cases applied in this paper handle both these open issues. COVID-19 has given enormous light to the healthcare sector since last year. Its integration with IoT and ANN has already changed the culture and standard operating procedure of this field. The studies conducted by Jain et al. (2020) and Jain et al. (2021) have provided a solution for embedding security in an IoT Ecosystem. The execution of the study was over 5G, network slicing and the security solution by introducing Pattern Matched Intrusion Detection System. This paper implements that solution to the healthcare domain and enhances it by introducing security using a neural network.

Lam and Abbas (2020) propose a similar anomaly detection mechanism for the 5G network, where a software-defined security system is implemented as a defence system for the network. This solution used Machine Learning to design the neuron network that detects the suspicious device on the network. The system developed by them identified benign traffic with a 100% accuracy rate and anomalous traffic with a 96.4% detection rate. Dey and Rahman (2019) worked on two similar approaches for a flow-based Intrusion detection system. Initially, they used the NSL-KDD dataset and machine learning algorithms like the random forest, projective adaptive resonance theory, radial basis function network, decision tree, and Bayesian network. They used these algorithms for detecting various attacks like denial of service and probing. In the next phase, the deep neural network was used to improve the classifier output and improve the accuracy. Our paper also used a similar approach to implement the Neural network on the dataset to get the desired results and accuracy.

This paper extends the study conducted by Jain et al. (2020), where they implemented security on a sliced 5G network. We have introduced machine learning to detect suspicious devices and restrict them further. In our previous papers, we introduced pattern matched intrusion detection methodology to detect suspicious devices, which was restricted to the pattern written in the files. In this paper, the system is made intelligent by using a neural network. Data generated by the device was collected, and the neural network trained the dataset on the collected samples. We conducted five different

training iterations to show how results transformed with different neurons and hidden layers. As a proof of concept, machine learning in this paper is implemented only in one module. We can extend it to other modules as per our requirements and the availability of resources. Our solution detected the suspicious devices with 99.4% accuracy in the last iteration and 100% for the benign devices. We proposed a network architecture to manage the healthcare network. A layered architecture is proposed to design the required application and placement of different modules. We have discussed four different use cases that are applicable in the healthcare domain.

We have organised this paper into multiple sections. The first section, the introduction, is followed by the literature review, which provides a brief overview of the general security issues in IoT enabled healthcare services. Next, we have a review of related work in the field to find the existing gaps. The following section briefs the four most relevant use cases applicable in health care. After that, we propose the IoT layer and its network architecture. Next, the design and training methodology section briefs all the technical details about the design, tools, and techniques used in this paper. The following sections provide details of the five training iterations and the use cases used to train the dataset using MATLAB's neural network. After that, we performed a comparative analysis of the results obtained by dataset training using neural network and application performance and behaviour. Finally, we discuss the conclusion and the future scope available to move this research ahead.

## LITERATURE REVIEW AND BACKGROUND

This section briefly discusses the research conducted in IoT, healthcare and Neural Network domain. Most of the discussions in this paper is around these topics and the simulation performed. Artificial Neural Network is already providing some of the best solutions, and its integration with IoT indeed makes a significant positive impact. This section also discusses different research done in intrusion detection and prevention, and the security issues in the healthcare domain. We also take a short glimpse of network slicing before listing related research and gaps in this field.

### *SECURITY ISSUES OF USING IOT IN THE HEALTHCARE ECOSYSTEM*

Most of the security issues found in the IoT healthcare field are inherited either from traditional IT networks or from IoT networks. These security issues can vary from data privacy to generic network attacks like denial of services. Basics of security, as discussed by Maple (2017), i.e., Confidentiality, Integrity and Availability(CIA), is also the primary and most important need for the healthcare ecosystem. Securing the ecosystem at one level is not sufficient moreover, it should be a multilayer. Like other domains, healthcare also requires a secured infrastructure at all levels, be it a network, application, data processing or storage.

The data generated in healthcare is sensitive; therefore, it is under many restrictions. The healthcare domain is monitored and regulated mainly by government agencies. Developed countries have their data privacy standards, especially for healthcare. One such law made by the USA government is the Health Insurance Portability and Accountability Act of 1996 (HIPAA). HIPPA, as explained by the Centers for Disease Control and Prevention (CDC, 2018), says this law protects the patient's medical records and ensures privacy until the patient's permission. This law covers four parties involved in the exchange, processing, or storage of medical records. Entities coved under HIPPA are healthcare providers, insurance companies, medical logistics suppliers, and business associates who deal with the medical records. HIPPA classifies health records a Protected Health Information (PHI) and put them under several other privacy laws governed by the federal government.

Jain et al. (2018) explains the different types of attacks possible in the IoT ecosystem and provides the solution available at each layer. Chacko and Hayajneh (2018) mention that these layers can include attacks like distributed denial of services, tampering, and viruses. As shown in Figure 1, Jain et al. (2018) mention some of the security issues of the traditional IoT network and these issues are also

inherited in healthcare. The solution to these issues is also the same, i.e., hardening network configurations to building defence in depth.

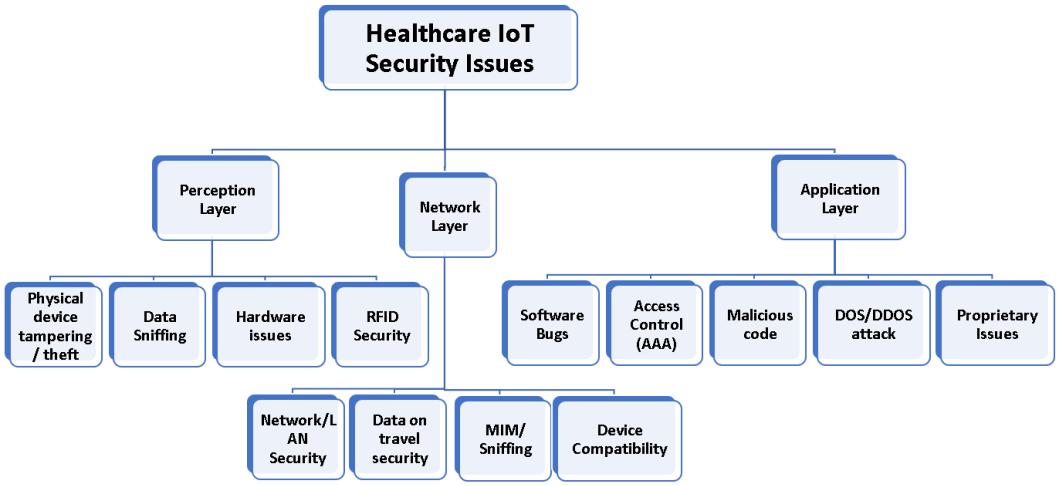


**Figure 1. Security issues in healthcare based IoT Ecosystem**

Other security issues specific to healthcare are:

1. *Device management:* Selvaraj and Sundaravaradhan (2020) write that IoT devices allow patients to stay at home and get monitored. However, the required device may be uncomfortable to use or may not be installed or inappropriately used, sending incorrect data to doctors.
2. *Inventory management:* Change and advancement of technology require introducing new devices and excluding old devices, increasing inventory management challenges for administrators. Gloss (2020) mentions that inventory management software is not updated so frequently; therefore, some devices get leftover, increasing the risk of getting compromised.
3. *Device reliability:* Usage of non-standard medical devices in the IoT Ecosystem increases the risk of getting compromised. Such devices can also be implanted intentionally by hackers to get the data from the network.
4. *Device interoperability:* Mavrogiorgou et al. (2019) mention that the interoperability of devices is the biggest concern. Devices getting manufactured by different vendors are sometimes incompatible with other vendors' software.
5. *Impersonation:* Adams (2005) and Younghwa (2012) write that when an attacker can easily imitate and trick the administrators into gaining access to private data or the network, it is called "impersonation".
6. *Data leakage and destruction:* Tao et al. (2019) write that intentional and unintentional leakage of data impact the privacy principle. Data leakages are unauthorised because they pass the otherwise private information to an unauthorised third party.

These security issues can build a robust ecosystem that can help fulfil all the basic security principles. A secure ecosystem's architecture should be designed to incorporate security at all layers. We have proposed and discussed one such security layered architecture in this paper's IoT layer architecture section.

## *Intrusion Detection Using Neural Network in Healthcare*

In the previous section, we discussed the general security issues in IoT and the healthcare domain. Some of these issues are inherited from IoT, and others are a combination of both domains. Most

of the issues in the healthcare domain are related to privacy because this data is regulated by government laws whereas, in IoT, most of them are inherited. Therefore, the traditional IT techniques might not be applicable or even effective in solving these issues. New innovative and out-of-box techniques are required to conquer these issues. Machine learning is one such technique that can help resolve these issues and maintain the privacy standards regulated by the government. This section looks at different studies on machine-learning or neural networks, when combined with our traditional techniques, as an efficient way to help detect and prevent intrusions in our network. These studies help us quantify the methodology used in this paper to help detect and disable suspicious devices using neural networks.

Security architects always recommend deploying security in multiple layers that can provide 'Defence in Depth'. Defence in depth helps detect and isolate attacks at multiple layers. Makani and Reddy (2018) write about a machine learning based intrusion detection system as the first layer of defence, which can help detect the attack by differentiating between normal and abnormal behaviour. The same differentiation can be deployed as the second layer where the device with abnormal behaviour is blocked, preventing an attack. Mitchell and Cheng (2014) mention speed and accuracy as the essential tool for any intrusion detection mechanism to be successful. Machine learning serves this purpose where normal and abnormal behaviour is detected using a round of learning, and recommendations are sent to the application to take appropriate action. There are many methods for training our data using machine learning like artificial neural networks, decision trees, Bayesian networks, and K-means clustering. The artificial neural network was used to train our network and get the results.

Mitrokotsa et al. (2007) explain that artificial neural networks' structure is designed on the human brain foundation. In this structure, training is a continual process, and more data means enhanced training. This process requires three steps for completion: data collection, training, and validation and testing. However, the data training can take a lot of processing time on a noisy and incomplete dataset. Perfection and the result of training also depend on the algorithms used. This paper has used two algorithms for training the provided data set, i.e., Levenberg-Marquardt and Bayesian Regularisation.

Huch et al. (2018) present a machine learning based anomaly behaviour detection mechanism to predict a system's health. This research proves that regular neural networks with long short-term memory (LSTM) are more effective in sensing irregularities than other classifiers. Zolanvari et al. (2019) present a vulnerability assessment report of the Industrial Internet of Things identifying threats, vulnerabilities, and risks. They use the machine learning approach to solve the problem and prepare a testbed system to detect and isolate intrusion. In their research, Cui et al. (2019) develop an anomaly detection method using a neural network to forecast the load and then detect the cyber-attacks using k-means clustering. P. et al. (2021) discussed the IoT devices as the easy target for hackers citing their heterogeneous nature. Their research designed an intrusion detection system to reveal Distributed denial of service attacks using IoT botnets. The deep neural network was used to design the solution and the real environment data to train the network. Finally, Casas et al. (2016) propose a machine learning based anomaly detection tool for cellular networks. The authors assessed the solution using the inexplicably generated data using an active cellular internet service provider. Generated data matched real network traffic based on which suggestion was provided to improve performance in a specific scenario.

In this paper, we present 'Security as a Solution' for IoT based healthcare systems. Organisations can implement this solution on an already existing infrastructure where security is not embedded in the design. Security as a Solution is the most prevalent option because many existing devices and infrastructures do not have any inbuilt security or are outdated. Chawla and Thamilarasu (2018) propose security as a service framework for real-time intrusion detection in their paper. The paper presents a similar solution in a different approach where they also use a machine learning module and a mitigation module to detect and handle the alert. In their paper, Hodo et al. (2016) proposed an artificial neural network-based intrusion detection system. The proposed system detects denial and distributed

denial of services with 99.4% accuracy. This solution works both for host-based and network-based intrusion detection.

## *Overview and Evolution of Network Slicing*

Today, Network Softwarization is playing a significant role. Condoluci et al. (2016) and Ibrahim et al. (2018) explain how software is used to design, implement, and manage network components. Extensive deployment of software for most of the services brings the cloud and virtualisation into the picture. Barakabitze, Ahmad, et al. (2020) write how 5G provides end-to-end services based on network softwarization and enhances customer experience. Andrews et al. (2014) mention the 5G network as a cloud-based, fully virtualised network that is scalable per user and environment requirements.

Along with 5G, network slicing also comes in the picture, where the network is distributed based on virtualisation. Nanda and Tzi-cker (2005) mention that network slicing is based on network virtualisation with its origin back to the 1960s. Goldberg (1974) mentions that IBM introduced virtualisation in an operating system and, since then, it has evolved to capture the whole market today.

The Software Defined Network (SDN) and Network Function Virtualization (NFV) have enabled 5G as a software-based virtualised network. Barakabitze, Barman, et al. (2020) mention SDN and NFV as the service enablers for the 5G network that promotes automation. ONF (2018) defines SDN as "the physical separation of the network control plane from the forwarding plane, and where a control plane controls several devices". SDN brings intelligence and flexibility to the network, which helps enhance its capacity. NFV transforms the traditional networking hardware equipment into a virtualised environment. It helps to decouple software from the hardware platform and provides greater flexibility and dynamic network operations. This paper provides a security solution over network slicing that can be quickly deployed on the network.

## *Related Work and Gaps Identified*

IoT devices providing healthcare services have grown exponentially in the last couple of years, especially after the impact of COVID throughout the world. An exponential surge in the usage of healthcare IoT devices has also amplified the security risk arising out of this evolution. As privacy laws back healthcare fields, many commercial organisations and researchers are also continually working towards nullifying the security risk. Many such kinds of research are already implemented for the welfare of humankind, and others are on their way. However, solutions provided by commercial organisations have a monetary impact where the cost of solutions is recovered from the end customer and, in most cases, unaffordable for ordinary people. Therefore, it becomes crucial for academia to provide solutions that can easily be implemented and have no significant cost impact. Many such solutions are already available, and we have discussed a few of them in this section.

Tao et al. (2019), in their research, provide a solution for the secure collection of healthcare data. Their research has mitigated the risk of data leakage or attack at the time of conversion from cipher to text and vice versa. The solution named "SecureData" provides a four-layer solution where the first layer comprises end-user sensors or devices, the second layer is the Fog layer, the third layer is the cloud computing layer, and the fourth layer is dedicated to healthcare service providers. The authors also discuss, in brief, the security challenges faced by healthcare along with the threat model, which includes collusion attacks and eavesdropping. They build a hardware-based lightweight cipher solution that can mitigate the risk of data leakage during the conversion phase. Their paper presents a new lightweight cipher algorithm called 'KATAN' implemented on specially designed hardware. The solution suggested in this paper works fine for the encryption of data. The study does not provide an end-to-end solution for vulnerabilities present at different layers. However, the solution suggested in our research provides an end-to-end solution that can be easily implemented and deployed in many different domains like in healthcare in this paper.

Hamza et al. (2020) present a privacy-preserving solution designed for IoT healthcare services. This encryption algorithm preserves images of the patient's medical records from compromised brokers. The proposed algorithm is a chaos-based cryptosystem designed using symmetric block encryption methodology and operates in one round of confusion and diffusion. Furthermore, the solution is designed using an additional randomisation mechanism, i.e., it generates different output if the same key is used twice for encryption. This paper presents a state of art cryptosystem solution designed to encrypt and decrypt data in a secure way. This solution can be implemented on the application layer and can provide a solution to privacy issues. After proper testing, we can implement this solution on other layers like the network layer.

Moosavi et al. (2015), in their research, share an authentication and authorisation based solution for a healthcare IoT environment. This paper emphasises the medical data privacy factor and presents a solution for secure authentication of remote healthcare professionals using a distributed smart e-health gateways called SEA. The authors claim SEA is the first such solution to provide authentication and an authorisation approach for IoT-based healthcare solutions. The authors have elaborated on the solution and provided a proof-of-concept to demonstrate the prototype. This solution is an amicable solution used in future projects to implement authorisation and authentication solutions. However, it does not provide an end-to-end solution. Therefore, we can use this solution as an additional module for user authentication. Furthermore, research done in this paper can be combined with the one discussed above to strengthen the authentication and authorisation module.

Jimenez and Torres (2015) present a prototype for monitoring remote patients and alerting the hospitals, doctors, and their families if any critical condition arises. Various other patient monitoring solutions are provided by Chandra-Sekaran et al. (2009), D'Souza et al. (2008), Occhiuzzi (2014), and Redondi et al. (2013). These solution designs help monitor a patient's health, but they compromise the security or privacy of health data. The solution suggested in this paper can be combined with many other prevalent options to ensure security over the sliced network.

Luo et al. (2018) present a privacy solution for collecting patients' medical records and preventing sophisticated attacks like collusion attacks and data leakage. Their paper also presents an access control mechanism where patients can securely share their data over multiple clouds without exposing the actual contents. The authors here first performs a detailed study of the challenges prevailing the secured collection of data and proposes a framework to prevent threats and attacks. The authors name the designed solution as "PrivacyProtector". Patients' data are stored on multiple cloud servers to ensure that it is accessible even if some servers are unavailable. The 'Slepian-Wolf-coding-based Secret Sharing' (SW-SSS) algorithm is used in the solution for ensuring data security and key exchange. The proposed framework provides an end-to-end data security solution and prevents several attacks like collusion, insider, data leakage and destruction. However, this framework does not provide any solution for network failure for critical or life-saving services. The solution provided in our paper is that we have handled the scenario where network failure can lead to complications, especially in critical healthcare services.

Most of the solutions prevalent in the market provide solutions for the privacy of medical data, and there are very few that mention securing the whole healthcare ecosystem. The simulation presented in this paper provides an end-to-end solution to this problem. The algorithm used to detect suspicious devices is an artificial neural network.

## DESIGN METHODOLOGY

The application's architecture has added a new module on top of the architecture discussed in Jain et al. (2021). This paper has included the machine learning module using neural networks to train the network and handle the intrusion automatically. In this paper, only one module of intrusion detection was trained and tested. We have used the minimum and maximum bandwidth allocated to each pattern, device and layer, and the actual bandwidth used by the devices. This data helps capture the

bandwidth usage trend by any device and the average bandwidth used by a device in a unique mobility pattern. The training started with ~500 and increased to ~410,000 to get the desired results.

This section discusses the changes made in the functioning of the application to include machine learning methods and algorithms to train the data. The three parameters used to train the network are minimum, maximum, and actual allocated bandwidth. The trained data calculates the average bandwidth used by a device and proactively detect the devices behaving abnormally while consuming the bandwidth. However, there is still a chance of error where a false positive or false negative may occur. Such errors are reduced by continually training more data on the pattern and behaviour of the devices.

We have divided the design methodology section into five parts and discussed it in detail in the upcoming section. In the first part, the four healthcare uses discussed are implemented in this research and are applicable in a sliced network. Then, we have proposed network and IoT layer architecture for the healthcare domain. We have then explained the high-level application design and the data flow methodology. Finally, the next part gives a brief overview of the application design algorithm and the methodology and architecture used in training the data using a neural network.

## *Healthcare Use Cases in Sliced Network*

Healthcare services can vary from monitoring essential health elements like BP, blood sugar, pulse, or oxygen to providing critical life-saving services and surgeries. Today, the healthcare domain is highly dependent on telecom and internet service providers, enabling patients to get seamless access to remote doctors, hospitals, and services. Enhancement in any service has its side effects, one of which is cybercrime. The healthcare domain must comply with several privacy laws, like HIPPA, making it essential for them to comply with IT security. Today, healthcare is a vast field where security is required at the hardware, application, or network level. There can be multiple use cases that we can implement in the healthcare domain. This section discusses those four significant use cases along with their solution.

### Case 1. Life-critical services

This use case has life-saving criticality; therefore, we allocated two slices to critical care. Any one of them can act as a backup. If any slice is compromised in a base station, traffic is immediately migrated to the backup slice. Figure 2 shows our proposed slice architecture where slice six and slice seven are dedicated to critical services. These slices work in a load-sharing mode as a backup for each other.

### Case 2. Non-critical services

In this use case, if there is a failure on any slice providing non-critical services, the migration of clients is decided based on the priority allocated to that device. A high priority device would be migrated first.

### Case 3. Suspicious devices

Devices are marked suspicious based on several criteria, such as abnormal bandwidth usage, above-average handover for still devices, and switching QoS. In this use case, all the suspicious devices are immediately disabled and removed from the network, whereas the suspicious devices using critical care services are not disabled; instead, the administrator is alerted.

### Case 4. Base station

This use case happens if a base station is compromised. However, this scenario is rare, but all the clients of that base station are migrated on a best effort and priority basis. The migration of clients is done to the next available and matching slice of the nearest base station, i.e., critical care patients are

migrated to the critical care slice and vice-versa. If free slots are not available on matching slices, they are migrated to the slots available on non-matching slices.

Figure 2, as described by Jain et al. (2021), shows the sample of seven slices applicable in the healthcare domain and the possible attack scenarios.

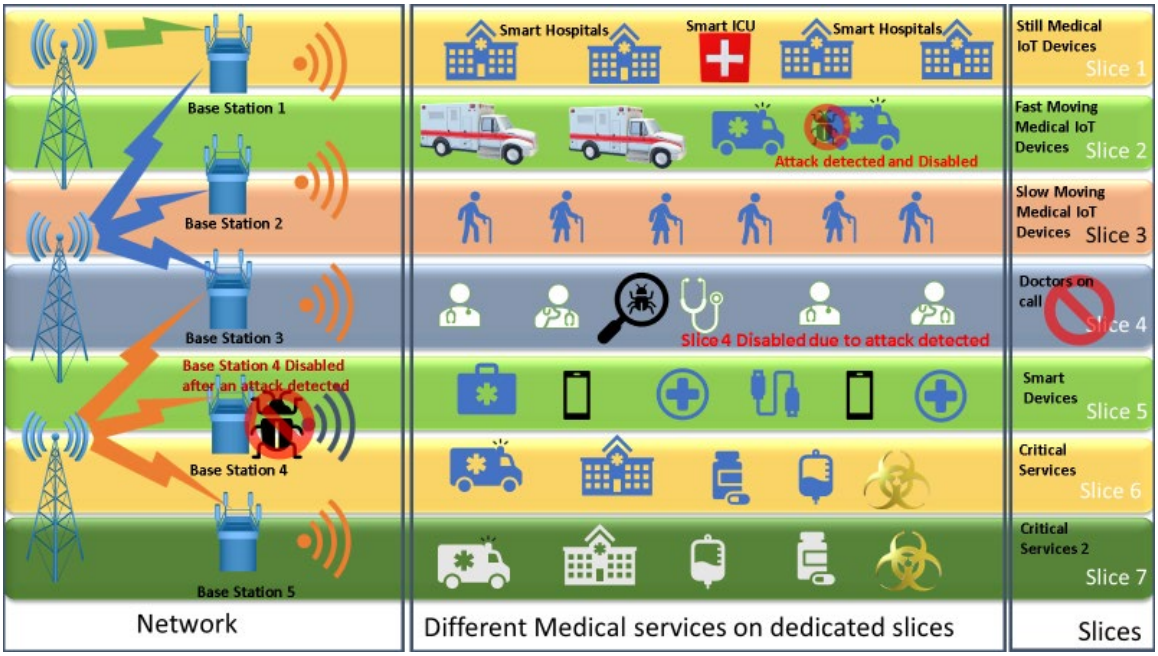


**Figure 2. Healthcare slice architecture**

## *Proposed Architecture for Healthcare*

This paper proposes the health care architecture at two levels; the first one is on the network level, and the second shows the IoT layer architecture. The network architecture shows the high-level view of the whole ecosystem and the way it connects the different devices on the network and the modules deployed on each layer. The proposed architecture shows the whole ecosystem and the working model at both levels. This section discusses both architectures in brief.

### Proposed network architecture of an IoT healthcare ecosystem

The network architecture proposed in this section is an extension and a subset of the architecture discussed by Jain et al. (2021). They explain a generic and a combined architecture of the 5G, IoT and slice network ecosystem. In this paper, we make changes to that architecture to make it specific to the healthcare ecosystem. As shown in Figure 3, the architecture shows a combined healthcare system consisting of IoT enabled hospitals, IoT ambulances, operation theatres, or ICUs as a part of hospitals and other healthcare devices used by the public. The network layer is the 3G/4G/5G network, connecting the endpoint and transferring data to the cloud. ISP network is the edge layer that connects different networks and helps pass the data from an endpoint to the cloud-based processing unit. This architecture consists of two cloud-based data processing and analytics sections which act as a backup for each other in a disaster situation. An organisation locates these units in a distant geographical location to survive a disaster. Figure 4 shows the data processing section of a network architecture as a machine learning module in IoT layer architecture.

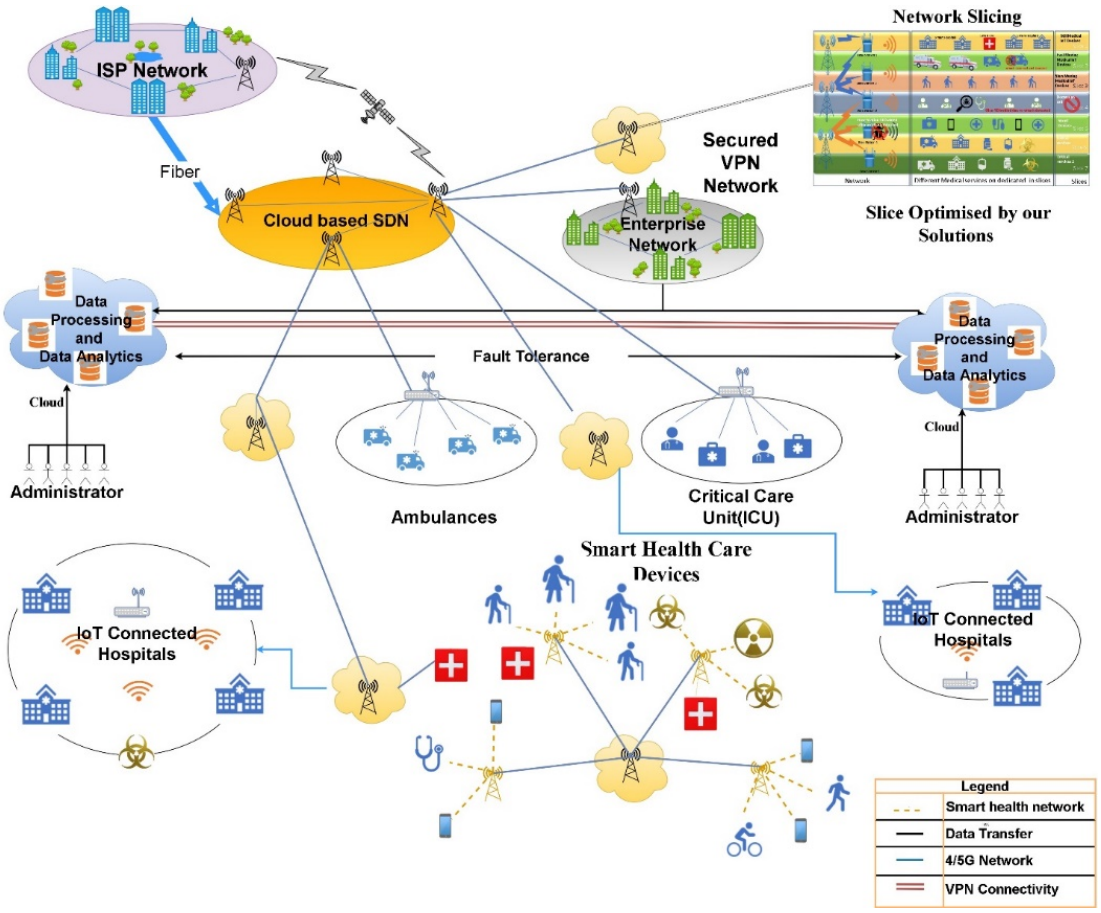


**Figure 3. Proposed healthcare network architecture**

## Proposed IoT layers architecture for health care using neural network

Figure 4 proposes the layered architecture for the IoT healthcare ecosystem. This figure shows the high-level layered architecture and different modules required for data processing. It shows an ecosystem combining different layers and the security mechanism implemented on them. Jain and Singh (2020) discussed the security mechanism showing the different intrusion detection techniques deployed on each layer.

The first layer of this architecture consists of endpoint IoT devices and corresponding applications installed on their mobile or static devices. Mobile devices can consist of intelligent health monitoring devices like pacemakers and smartwatches, collecting different health data like pulse, BP, and oxygen. Endpoint IDS solutions are installed on this layer to protect the applications and the end devices.

The second layer is the network layer that connects the endpoint and manages the network infrastructure of the ecosystem. Devices on the first layer can connect the network using an IP or non-IP network. Firewall, routers and other hardware or cloud-based network devices secure this layer. All the data passing through this layer are checked for the valid source and destination. Intelligent IDS/IPS devices are installed here to prevent unauthorised packets. The network layer is further connected to edge devices and passes the data packets to other networks. Edge intrusion detection systems are deployed over this layer. Finally, the edge layer is connected to the cloud layer, which collects all the data for storage and processing. All the data are collected and forwarded to other modules for processing. Other modules designed in this layer are device management, alert management, IDS module, and administrator modules to manage the whole system.

A cloud-based data processing module is connected to a machine learning module. The data processing module collects all the data from the endpoints. The machine learning module scans the raw data to learn and detect suspicious activities from the collected data. The intrusion detection and prevention module is also built within the machine learning module that alerts the monitoring devices to take appropriate actions. The administrators scan all the alerts generated to ensure that no false positive or false negative alerts impact the services provided to the end customers. As this is a healthcare system, we need to ensure that false alerts do not interrupt critical services.

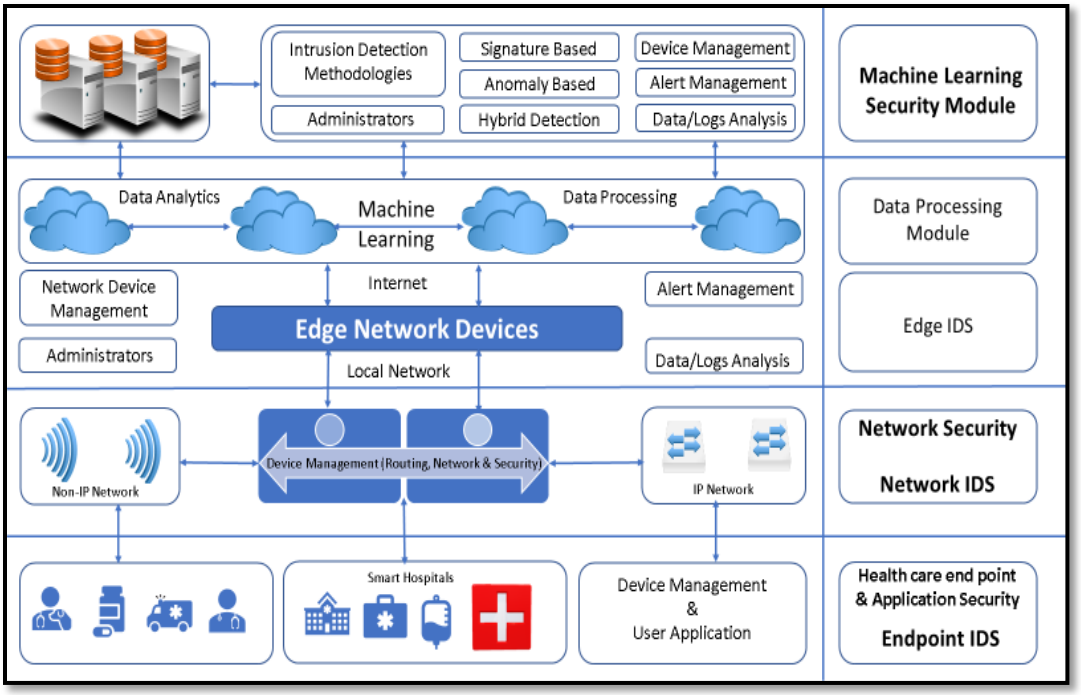


**Figure 4. Proposed IoT layered architecture**

## *Application Data Flow Design*

As discussed by Jain et al. (2020), the initial design of this application started with three modules and embedding security on top of it. This simulation's design was developed in Python (GitHub, 2020), which was later updated to include pattern matched intrusion detection. As shown in Figure 5, Jain et al. (2021) updated the design and increased the number of modules by adding a security implementation and attack detection module. This paper added machine learning using neural networks to detect suspicious devices based on bandwidth usage by a particular device category. Several machine learning techniques are available for data processing, like random forest detection tree (DT), Bayesian networks, and neural networks. This research has used a neural network machine learning technique to detect attacks on the network. The neural network was selected because it uses a similar approach to the human brain for processing and developing algorithms. The neural network is beneficial in the human brain, like predictions and especially in the security field. Wong et al. (2000) presented a detailed report on the use, applicability and research done on neural networks. This paper lists the application areas like finance, marketing, operations and information technology, where most research has been performed. Schmidhuber (2015) writes about the successful evolution and application of deep learning methodologies used in the neural network.

The data processing module's data was collected and trained using a neural network. We added two new sections to the design where raw data from the data processing module is forwarded to the neural network for training, and trained data is fed back to the security module. The security implementation module detects abnormal bandwidth usage by devices based on the learnings from the neural

network. The report containing details of the abnormal usage is sent to the attack detection module that blocks or marks the devices as suspicious. Figure 5 shows the high-level design and data flow of the application.

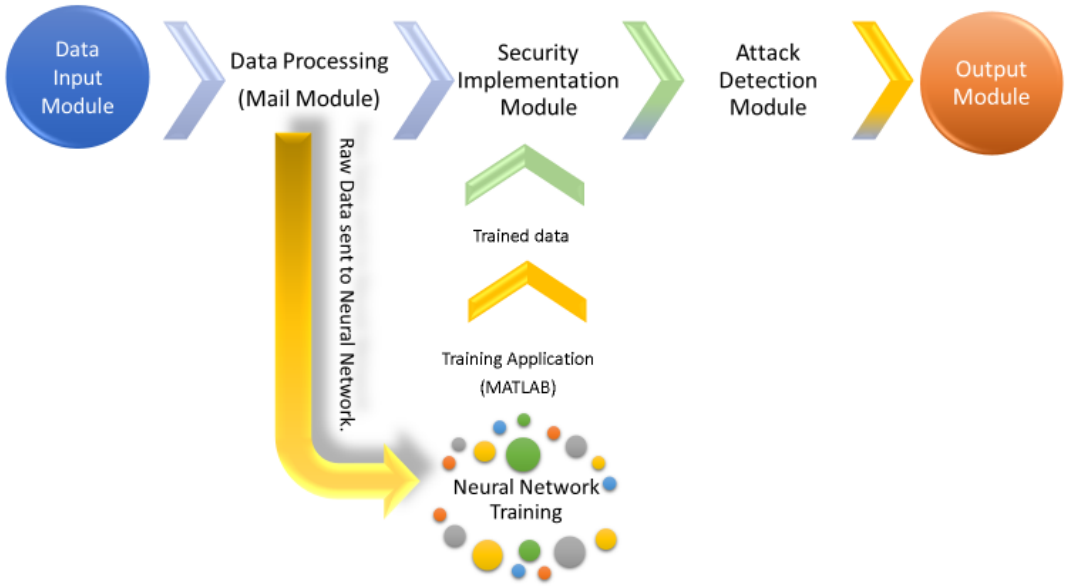


**Figure 5. Application data flow**

## *Design Algorithm*

The design algorithm is divided into ten steps, and the actions performed in each are mentioned below. The first important part of the algorithm is to decide the mobility pattern of the devices. Mobility patterns can vary as per requirements and based on the use case. Next, ANA (2017), Nursing Guide (2015), HCPRO (2019), and Perroca (2009) propose and classify the different types of patients and the services provided to them. These classifications helped us decide and distribute the types of patterns and the percentage shares allocated to them. The main concern was raised for critical patients where every moment is crucial to save a person's life. Therefore, we allocated two slices to critical care, which act as a backup for each other. After that, one decides the type of security required. In this research, we used bandwidth usage as a parameter to detect suspicious devices. Data of bandwidth usage was collected and trained on ANN to detect and disable suspicious devices.

**Step 1:** Start healthcare security solution application.

**Step 2:** Decide the mobility pattern percentage of the healthcare devices connected to the network. The pattern used in this simulation:

| | |
|---|---|
| Critical_care | 2 |
| Critical_care2 | 5 |
| Smart_health_devices | 39 |
| Smart_hospitals | 10 |
| Doctors_on_call | 6 |
| Senior_e_health | 30 |
| Remote_surgery | 8 |

**Step 3:** Configure slice type for each slice.

**Step 3.1:** Backup slice declared for critical service in *Step 2: critical_care2*.

**Step 4:** Configure the type of slice and mobility pattern for each base station.

**Step 5:** Decide the security level required and enable/disable according to requirement.

*Flag = True/False.*

**Step 5.1:** *Security_Mode(flag);*

**Step 5.2:** *Attack_Mode(flag);*

**Step 5.3:** *Security_Attcak_Mode(flag);*

**Step 6:** Enable and execute the intrusion detection module with the required parameters.

**Step 6.1:** Collect bandwidth usage data.

**Step 7:** Train the network in MATLAB using the data collected in Step 6.

**Step 8:** Enable the attack detection module (ML-enabled).

**Step 8.1:**

```
if(attack==true){
        if (attack on critical care){
                forward client details to administrator; }
        else{
                block services and forward client details to administrator;   } }
```

**Step 8.2:** Check if any attack is detected on any Slice.

```
if(slice_attack==true){
        if (attack on critical care){
                move resources to backup critical care slice and forward client details to administrator; }
        else{
                move resources to free slice and forward client details to administrator; }}
```

**Step 8.3:** Application will detect if there is an attack on the base station/s.

***Priority 1:*** *move the critical resources to other base stations.*

***Priority 2:*** *if free slots are available on other base stations, move other resources too.*

**Step 8.4:** The administrator performs manual verification to detect whether the attack is False-positive or False-negative.

**Step 9:** Output generation.

**Step 10:** Log all the events and alert the monitoring team to take appropriate action.

## *Neural Network Architecture and Training Methodology*

An artificial neural network is a combination of input, hidden and output layers. Hidden layers in a neural network architecture can vary from one to multiple. Similarly, data sent to the input layer and

the output received can vary based on the nature of the problem. In this paper, we have used bandwidth usage data to train the network. As mentioned in the previous section, we have selected three parameters as the input for our training and one for the output fetched.

One other important decision was to select the software for training the network. Pat Research (n.d.) has suggested various software, like IBM Watson, MATLAB, AWS lambda, and NVIDIA DIGITS. However, we selected MATLAB software to train the data collected from the simulation. There is no specific reason for this selection, but we preferred MATLAB seeing its market predominance and proven track record in the field of research.

The input parameter is the limit defined for using a minimum and maximum bandwidth in a particular mobility pattern and the actual bandwidth used by the device during its usage.

We used two different methodologies to train our data and selected single and multiple hidden layers. A neuron count varied from 10 to 100, which was changed to get the best results. No written book rule states how many neurons should be used to train the network. However, as Ansari et al. (2019) and Nguyen et al. (2021) mention, a rule of thumb says that the number of neurons in the first hidden layer should be more than the input parameters. As per requirements, the input parameter used in this project is three, and the neuron count varied between 10 to 100. Accordingly, approximately 1,000 epochs were used in almost all iterations of training. During the training, input data was feed-forward, and the output was calculated based on the neural network weights and other characteristics. Table 1 shows the neural network characteristics.

**Table 1. Neural network characteristics**

| | |
|---|---|
| Network Type | Feed-Forward Backdrop |
| Training Function | Levenberg-Marquardt/ Bayesian Regularization |
| Adaption Learning Function | Learngdm |
| Performance Function | MSE |
| Layers | 3 |
| Transfer function | logsig/tansig |

Two training functions (Levenberg-Marquardt/Bayesian Regularization) were used alternatively during training. Hu (2019) suggests the Levenberg-Marquardt algorithm, stated as 'trainlm' in 'MATLAB', appropriately trains minor to medium-sized data network problems. Training using this algorithm takes less amount of time and requires more memory. It provides a numerical solution to the problem of minimising a non-linear function. Training using this algorithm is accomplished when generalisation stops improving, as indicated by an increase in the mean square of the validation samples. Bayesian Regularised Artificial Neural Networks (BRANNs) are more robust and can make a suitable generalisation of the complex and noisy dataset. Training using BRANNs is deemed as accomplished according to adaptive weight minimisation. BRANNs are tough to overfit because they calculate and train on several effective network parameters or weights, effectively turning off those non-relevant.

Garoosiha et al. (2019) explain one epoch as a combination of one feed-forward and one backpropagation. One epoch is considered completed when the error is returned to the ANN, and the initial weight is updated. Here, error is the difference between the actual and target values predicted by the neural networks. LEARNGDM is used as an adaption learning function to find the neural network's weights. LEARNGDM is a standard gradient descent method that starts with a presumption, and then the slopes at the preliminary speculations are premeditated to identify the track to local minima.

Mean square error (MSE) is used to evaluate the eminence of the trained model, and the same function terminates the training if no advance is observed. The performance function is the MSE, as shown in equation (1), where 'a' is the actual output, and 'p' is the predicted value.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_a - y_p) \qquad (1)$$

As shown in Figure 6, the logsig, purelin and tansig are the transfer functions used in this training. These functions are generally used together in a multilayer network. The function logsig, as shown in Figure 6, generates outputs between 0 and 1 as the neuron's net input goes from negative to positive infinity. The logsig function is generally used in part backpropagation networks, as it is differentiable. The linear transfer function purelin, also used as linear approximators, is used in backpropagation networks. Freitas et al. (2021) mentioned that a multilayer network also uses a tansig transfer function.

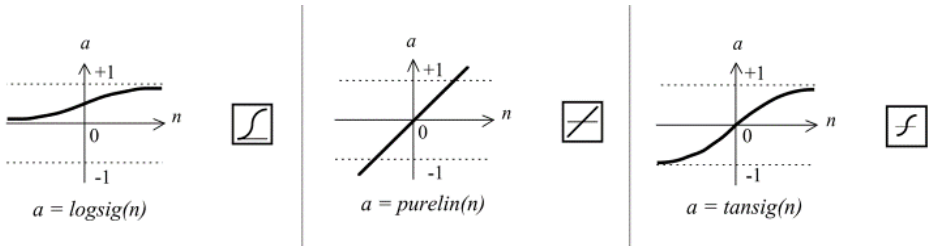


**Figure 6. Different transfer function**

The data used here for training an ANN is non-linear. Therefore, the usage of these three transfer functions becomes essential as it enables the program to detect the nonlinearity in the output.

Data partitioning is a critical and essential decision to be made during the whole training process. The data set used should be divided effectively to fit for training, validation, and testing. Such data is considered as 'Just Right'. If the dataset is divided ineffectively, the developed model can be termed 'Underfit' or 'Overfit'. This simulation avoids the 'Underfit' or 'Overfit' data issues using a dataset with standard data distribution where 70% of the data was used for training. The remaining 30% was equally divided for validation and testing. Table 2 shows the division of training data used in this simulation.

**Table 2. Data partitioning**

| % Distribution /Sample Size | Training | Validation | Testing |
|---|---|---|---|
| | 70% | 15% | 15% |
| 521 | 365 | 78 | 78 |
| 439040 | 307328 | 65856 | 65856 |

Training here was performed by using single and multiple hidden layers. In single layer architecture, we used the tansig transfer function in the hidden layer and the purelin transfer function in the output layer. Alomari et al. (2018) and Quintana et al. (2011) say that this network can ballpark any function with a limited number of discontinuities if the appropriate number of neurons are provided. Figure 7 shows the fitting function for a single hidden layer, and Figure 8 shows the abbreviated layer notation of the single layer network. It shows how the data is received from the input module and moved internally for training, and the trained data is sent to the output layer.

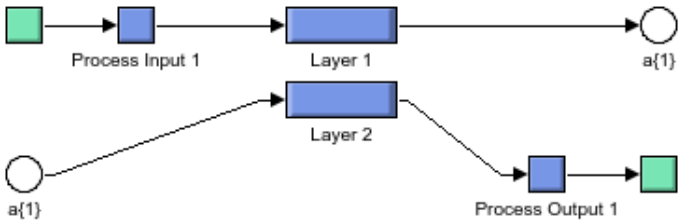


**Figure 7. Fitting function for single-layer network**

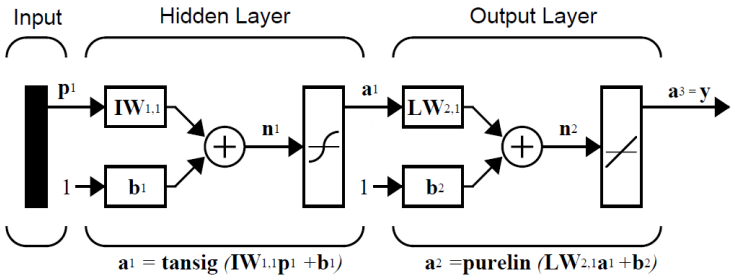


**Figure 8. Abbreviated layer notation in the single-layer network**

Figure 9 shows the detailed notation of multilayer neurons. There is more than one hidden layer in a multilayer network, and each layer plays a different role. The layer that produces output is called the output layer (layer 3), and other layers act as hidden layers (layers 1 and 2). Each layer can use a different transfer function as per the input and output desired. Each layer has a weight matrix W, a bias vector b, and an output vector a. In this multilayer network, R1 denotes input, S1 denotes neurons at the first layer, and S2 denotes neurons at the second layer. Constant value 1 is fed to the biases for each neuron.

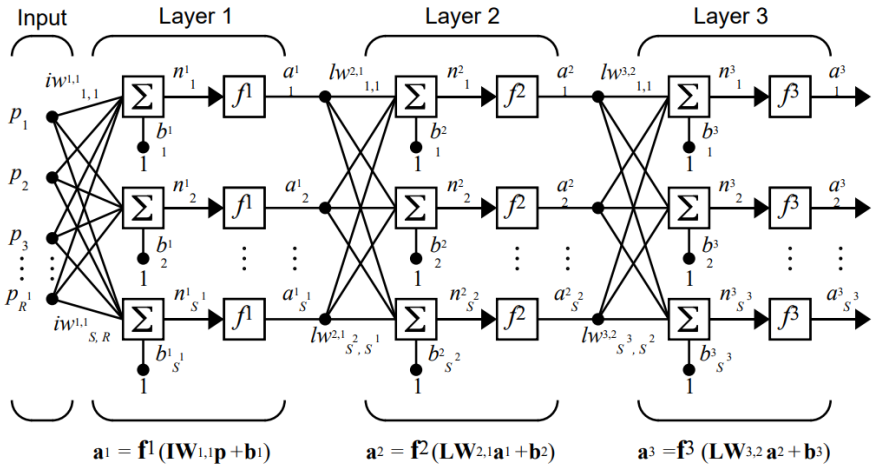


**Figure 9. Layer notation in the three-layer network**

This project uses the schematic structure below in a multilayer neuron network for training. Figure 10 shows input layer has three input parameters, and all layers have at least one input and one output. For example, hidden layer 1 uses 50 neurons, followed by 20 neurons in hidden layer 2. The output layer is using a single neuron. The sum of the output of each layer by applying the respective transfer function is supplied as input to the next layer.

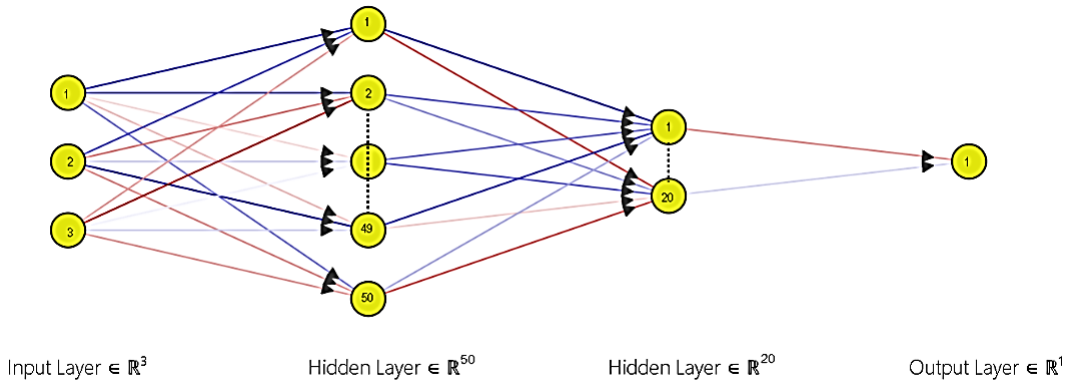


**Figure 10. Schematic structure of the used ANN**

We used data collected in step 6 of the design algorithm section to train the network and performed the training in 5 steps, divided into two parts. In the first part, a single hidden layer, and in the second part, multiple hidden layers were used. Table. 3 shows the training algorithms, hidden layers and number of neurons used in training at each level.

**Table 3. Network training details**

| Training Algorithm | No of Neurons | | | Epoch |
|---|---|---|---|---|
| | Hidden Layer | | Output layer | |
| | logsig | tansig | purelin | |
| Levenberg-Marquardt | | 10 | 1 | 1000 |
| Bayesian Regularization | | 50 | 1 | 1000 |
| Bayesian Regularization | | 100 | 1 | 1000 |
| Levenberg-Marquardt | 10 | 1 | 1 | 1000 |
| Levenberg-Marquardt | 50 | 20 | 1 | 1000 |

The first training was done using the Levenberg-Marquardt algorithm and a single hidden layer with ten neurons. The second training algorithm was changed to Bayesian-Regularization with a single hidden layer, and neurons were increased to 50, whereas 100 neurons were used in the third training keeping other parameters the same. The fourth and fifth training was done using the Levenberg-Marquardt algorithm and multiple hidden layers with neurons verifying between 10-50 on each layer. Output layer used purlin transfer function and a constant 1 neuron.

## TRAINING THE DATA USING ANN

Data collected during the simulation was trained using the methodology described above in the Neural Network Architecture and Training Methodology Used section. As discussed previously, the training setup was designed and divided into five parts, and MATLAB's configuration was done accordingly. In this section, we discuss all the five training methodologies briefly.

Figure 11 shows the actual distribution of the data used for training the network. The dataset was divided into three parts to ensure 'Just Fit' for training testing and validation. Out of the total dataset, 70% was used for training, 15% for validation and the remaining 15% for testing. As shown in Figure 12, we used 1000 epochs for all the five training performed.

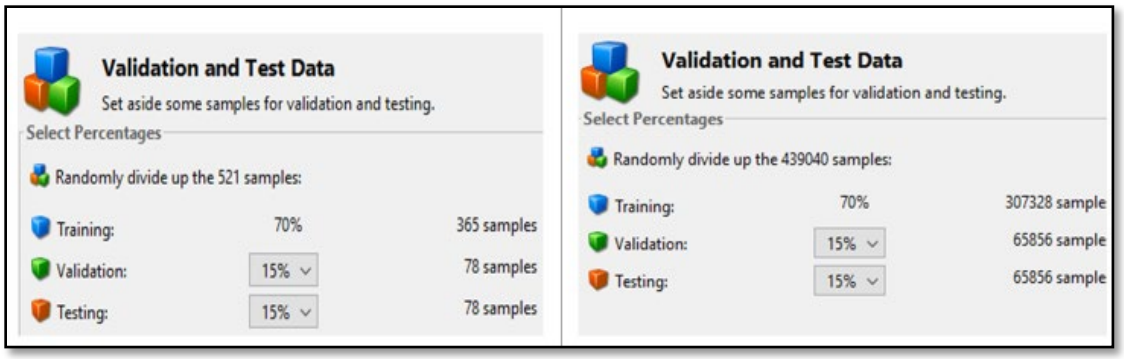


**Figure 11. Data partitioning**

As shown in Figure 12, the training reached 1000 epochs before stopping and was executed until the network error was reduced in the validation samples.

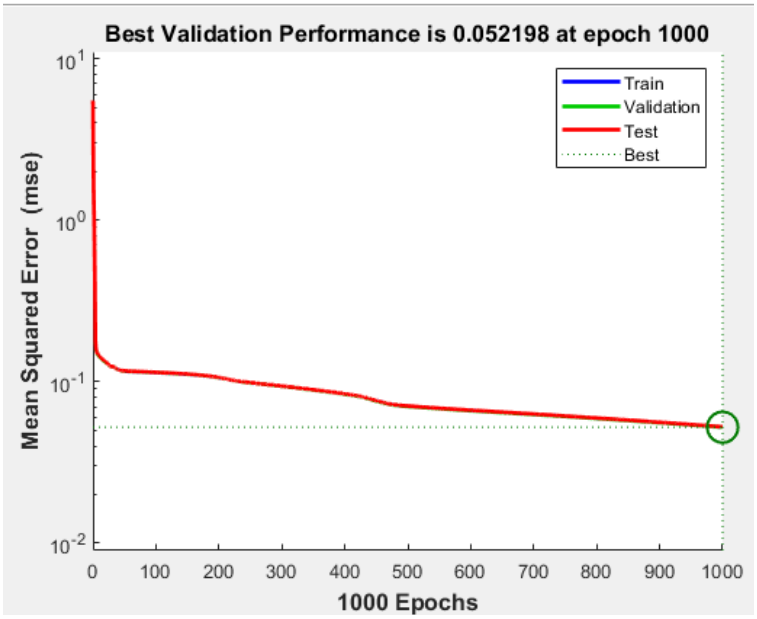


**Figure 12. Network training performance at 1000 Epochs**

## *ITERATION 1*

The first training was done with a minimal data set of 521 samples, and this dataset was divided as per the data partitioning methodology discussed previously. Table 4 shows the neural network characteristics used to train the network. Ten neurons on a single hidden layer were used with tansig as a transfer function. Output layer used purelin as a transfer function. Figure 13 shows the fitting function used in this training.

**Table 4. Neural network characteristics iteration one**

| | |
|---|---|
| Training Function | Levenberg-Marquardt |
| Performance Function | MSE |
| Hidden Layers | 1 |
| Transfer function | tansig |

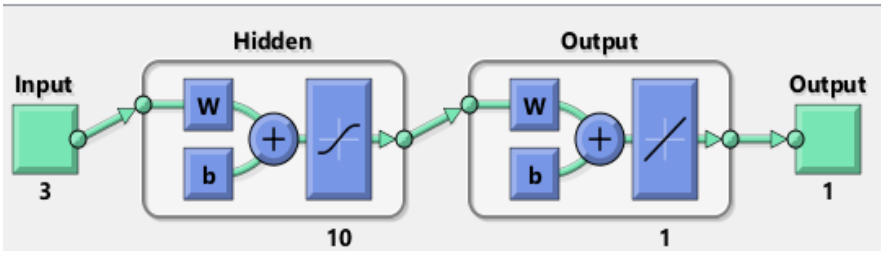


**Figure 13. Fitting function of iteration one**

Training time consumed during this iteration was minimal and took less than 10 mins. However, the R-value obtained from this training was approximately 0.59, which could not predict the exact results, and most of the values lie outside the index line. The training, validation, and test plots are shown in Figure 14.

Figure 14 shows the correlation value for the training sample (R=0.58898), validation samples (R=0.56437), testing samples (R=0.67383), and fitting co-relation of the prediction model as R=0.59301.

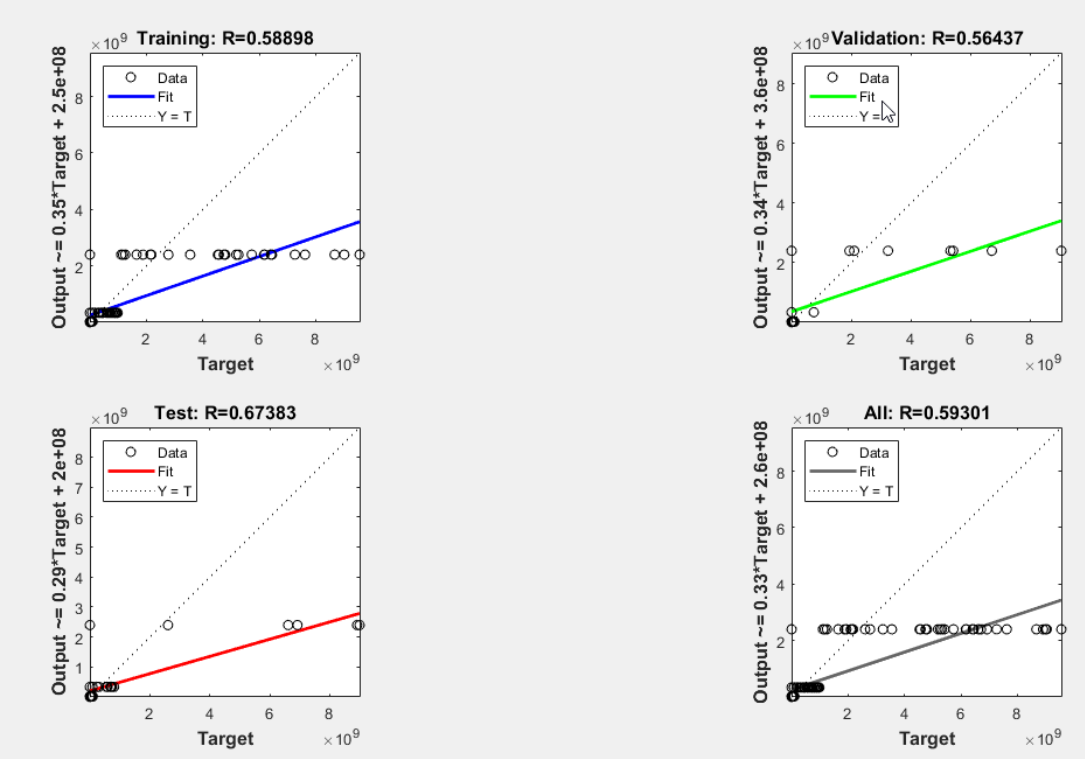


**Figure 14. Training one output**

Results obtained from this training were not optimal. Therefore, we decided to perform a second round with an increased dataset and a more significant number of neurons.

## *ITERATION 2*

The second training used a data set of 10,000 samples, and the dataset was divided as per the data partitioning methodology discussed previously. Table 5 shows the neural network characteristics used

to train the network. Fifty neurons on a single hidden layer were used with tansig as a transfer function. Output layer used purelin as a transfer function. Thus, the fitting function would remain the same as shown in Figure 13 except for the number of neurons on the hidden layer, i.e., 50 neurons in this training.

**Table 5. Neural network characteristics iteration 2**

| | |
|---|---|
| Training Function | Bayesian Regularisation |
| Performance Function | MSE |
| Layers | 2 |
| Transfer function | tansig |
| Input Sample | 10000 (training=7000, Validation=1500, testing=1500) |

Training time consumed during this iteration was a bit more than training one and took around 20 mins. R-value obtained from this training was approximately 0.67, which could not predict the exact results, and most of the values lie outside the index line. The training, validation, and test plots are shown in Figure 15.

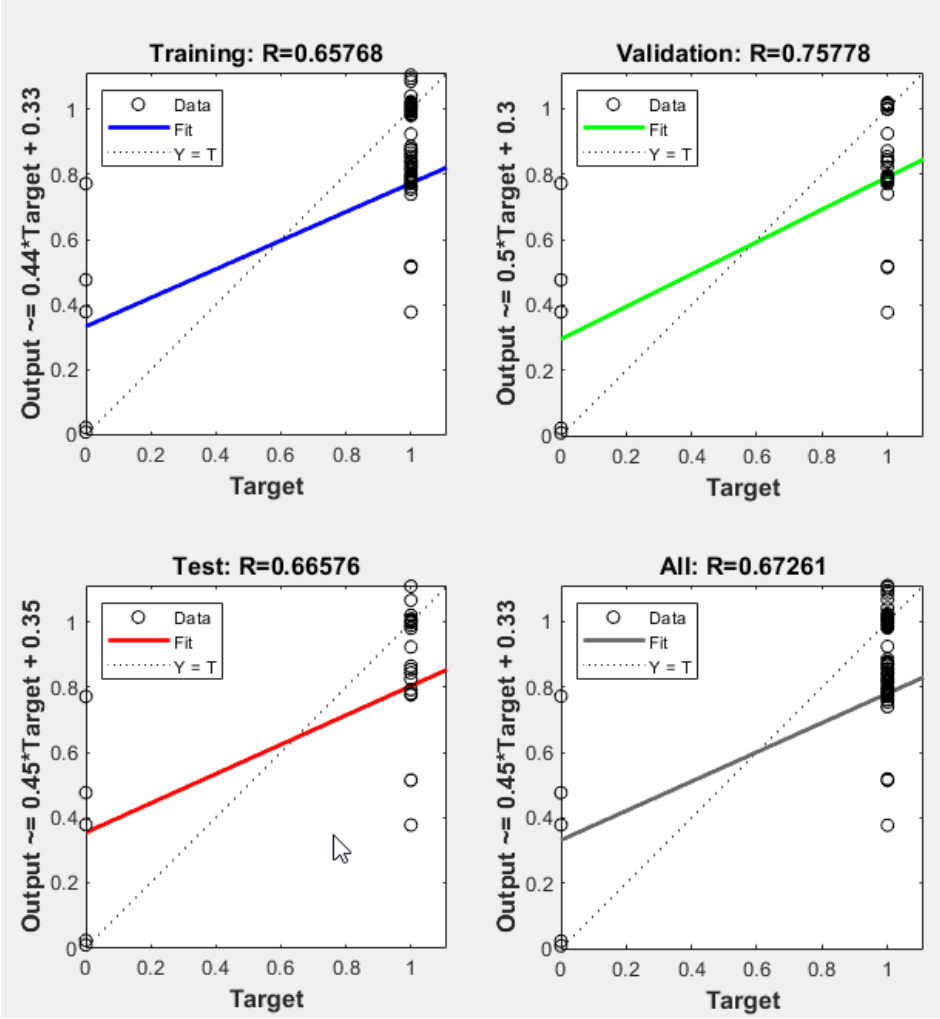


**Figure 15. Training two output**

Figure 15 shows the correlation value for the training sample (R=0.65768), validation samples (R=0.75778), testing samples (R=0.66576), and fitting co-relation of the prediction model as R=0.67261.

Results obtained from this training were also not optimal. Therefore, we decided to perform a third round with an increased dataset and a more significant number of neurons.

## *ITERATION 3*

After observing the results of the first two iterations, we decided to increase the size of the dataset. Therefore, the third training was done with a data set of 439,040 samples, and this dataset was divided as per the data partitioning methodology discussed previously. Table 6 shows the neural network characteristics used to train the network. One hundred neurons on a single hidden layer were used with tansig as a transfer function. Output layer used purelin as a transfer function. Figure 16 shows the fitting function used in this training.

**Table 6. Neural network characteristics iteration 3**

| | |
|---|---|
| Training Function | Bayesian Regularisation |
| Performance Function | MSE |
| Layers | 2 |
| Transfer function | tansig |

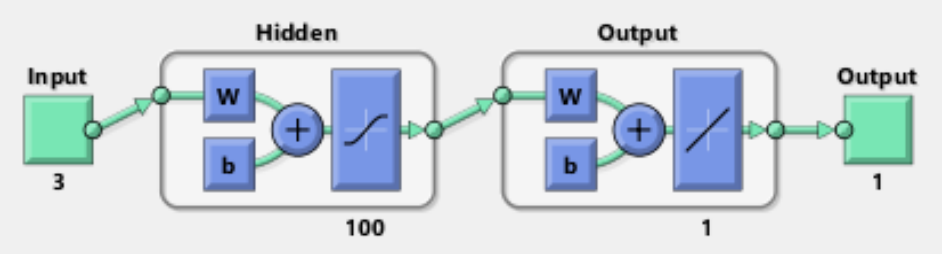


**Figure 16. Fitting function of iteration three**

Training time consumed during this iteration was optimal and took around 1 hour. R-value obtained from this training was approximately 0.88, which is bear to exact value 1. However, there was an immense deviation in the values and could not predict the exact results in real-time as most values lie outside the index line. The training, validation, and test plots are shown in Figure 17.

Figure 17 shows the correlation value for the training sample (R=0.88148), validation (R=0.88161), testing samples (R=0.88105), and fitting co-relation of the prediction model as R=0.88143.

Results obtained from this training were also not optimal because of the deviation in plots, and most values lie outside the index line. Therefore, we decided to perform the fourth round of testing with the same number of datasets and increase the number of hidden layers and the transfer function of the first hidden layer to logsig.

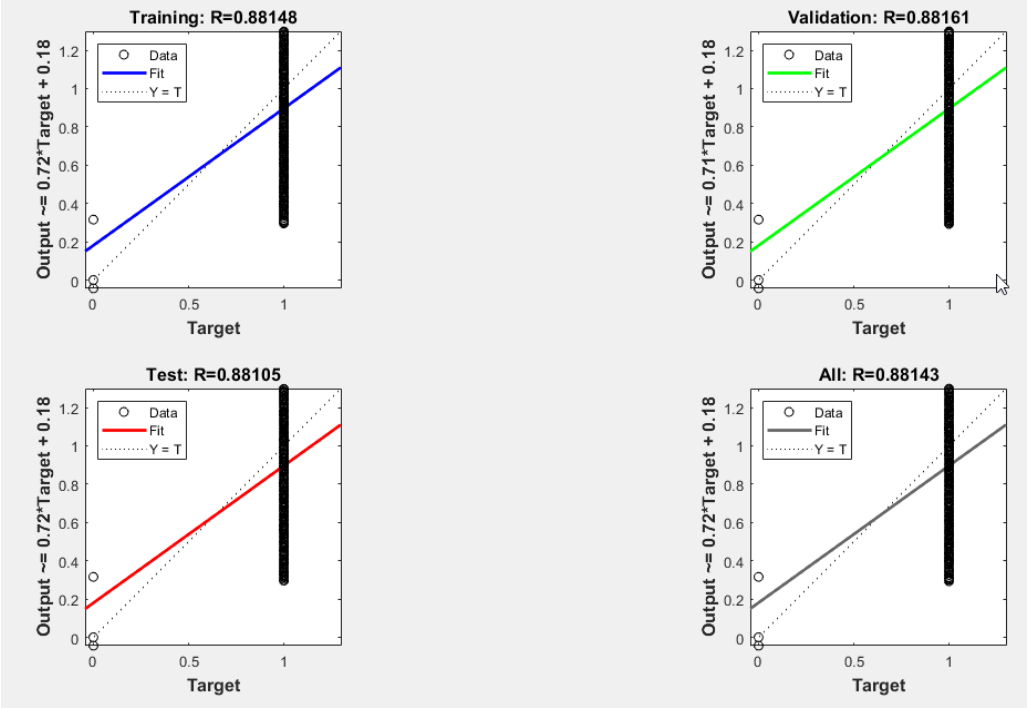


**Figure 17. Training three output**

## *ITERATION 4*

After observing the results of the first three iterations, we decided to change the training strategy. In all the previous iterations, a linear transfer function tansig was impacting the results. Therefore, we decided to use two hidden layers and logsig as a transfer function on the first hidden layer. The dataset and its data partitioning methodology remained the same as used in Iteration 3. Table 7 shows the neural network characteristics used to train the network. Figure 18 shows the fitting function used in this training. Ten neurons were used in the first hidden layer and one neuron on the second hidden layer with logsig and tansig as a transfer function. Output layer used purelin as a transfer function.

**Table 7. Neural network characteristics iteration 4**

| | |
|---|---|
| Training Function | Levenberg-Marquardt |
| Performance Function | MSE |
| Layers | 3 |
| Transfer function | logsig/tansig |

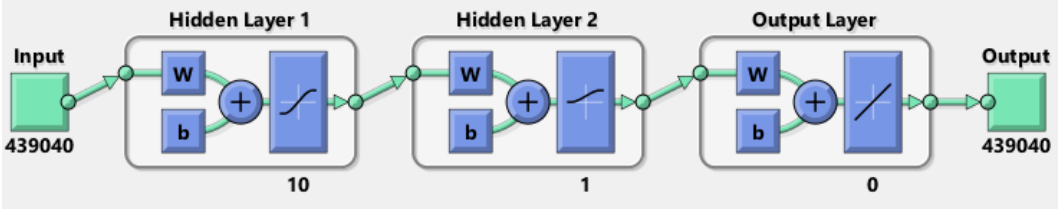


**Figure 18. Fitting function of iteration four**

Training time consumed during this iteration was optimal and took around 3 hours. R-value obtained was approximately 0.998, and the deviation was as per expectations. It could predict near to exact results in real-time as most values lie on the index line. The training, validation, and test plots are shown in Figure 19.

Figure 19 shows the correlation value for the training sample (R=0.99864), validation (R=0.99883), testing samples (R=0.99887) and fitting co-relation of the prediction model as R=0.9987.

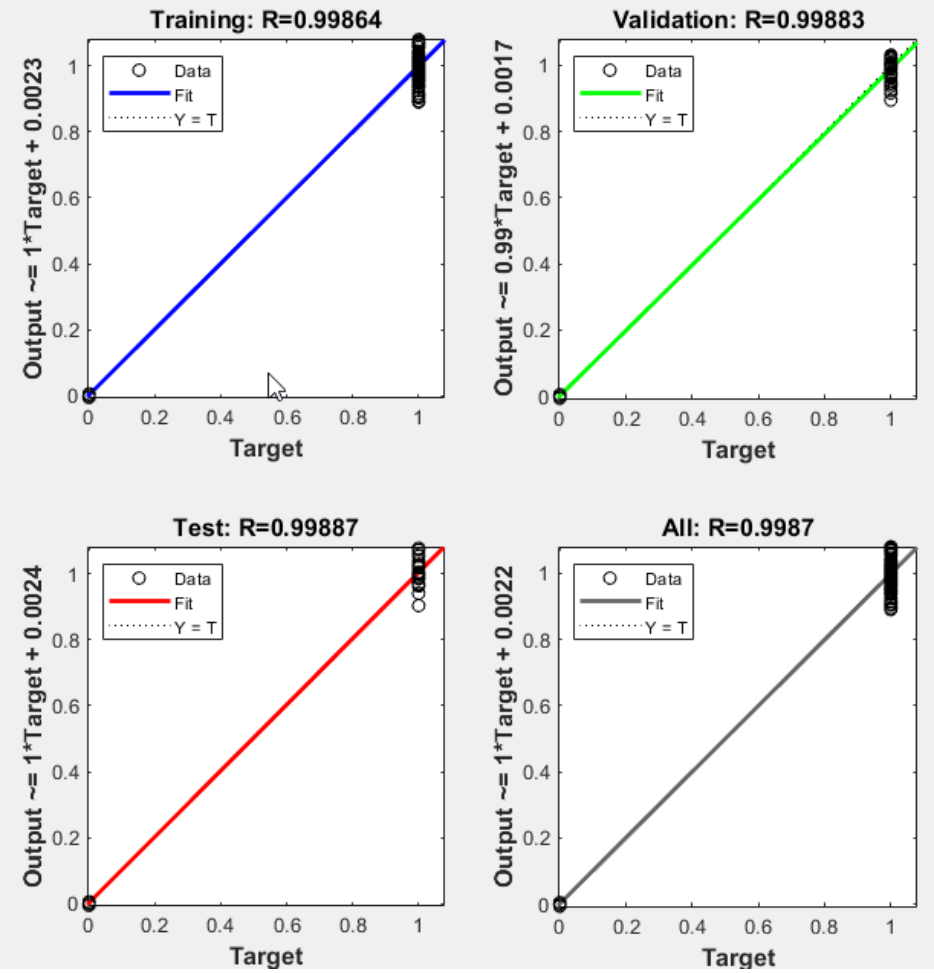


**Figure 19. Training four output**

The results obtained from this training were satisfactory because of very minimal deviation in plots, and most values lie around the index line. Therefore, we decided to perform one more round of testing with the same fitting function; however, the number of neurons was increased.

## *Iteration 5*

A new strategy was used in iteration four of our training dataset, and it gave promising results. However, the next round of testing was performed to check for any further improvement. Two hidden layers were used in this training and logsig as a transfer function on the first hidden layer. The dataset was kept the same as 439,040 samples, and it was divided as per the data partitioning methodology discussed previously. Table 7 shows the neural network characteristics used to train the network. Figure 20 shows the fitting function used in this training. Fifty neurons were used in the first hidden

layer and twenty neurons on the second hidden layer with logsig and tansig as a transfer function. Transfer function on output layer remained purelin.

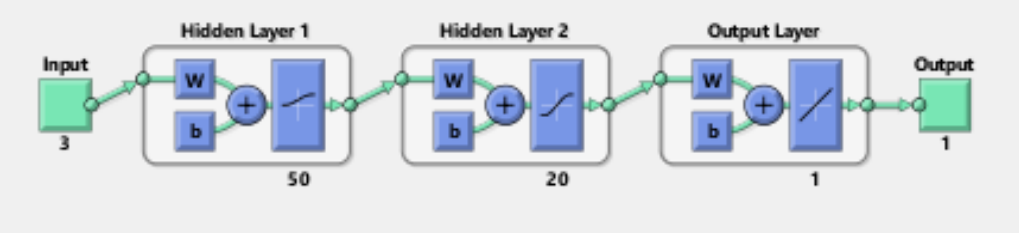


**Figure 20. Fitting function of iteration five**

Training time consumed during this iteration was optimal and took around 8 hours. R-value obtained was accurate to 0.999, and the deviation was as per expectations. The obtained results could predict the near to exact results in real-time as most values line on the index line. The training, validation, and test plots are shown in Figure 21.

Figure 21 shows the correlation value for the training sample (R=0.99971), validation (R=0.9971), testing samples (R=0.9971) and fitting co-relation of the prediction model as R=0.9971.

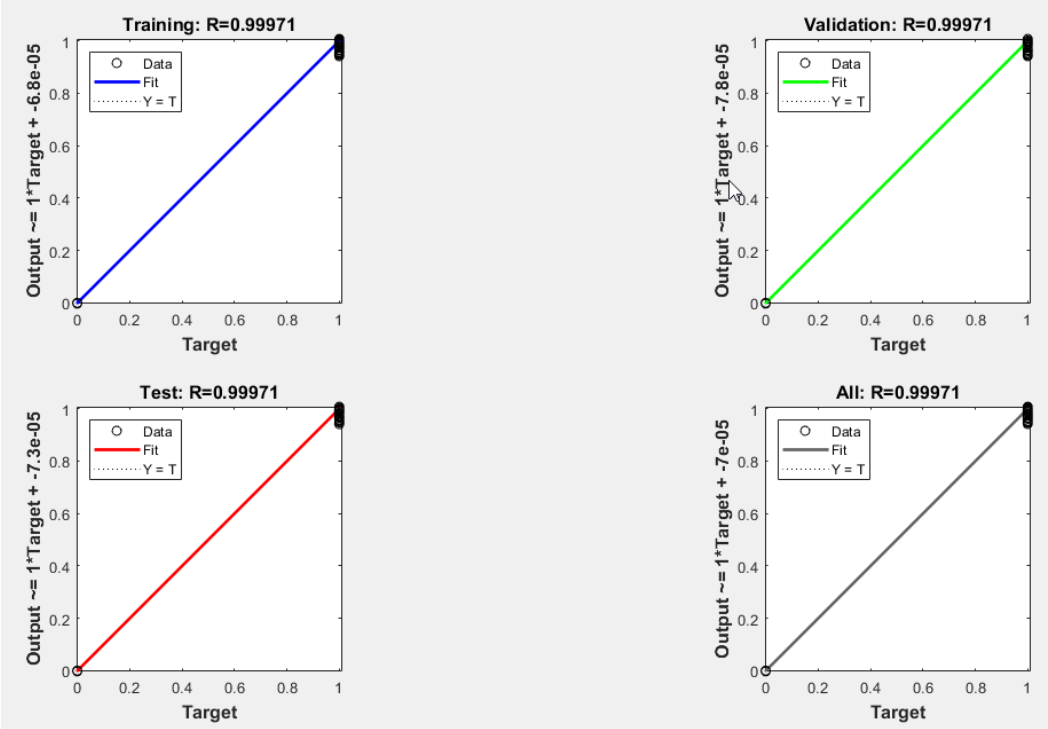


**Figure 21. Training five output**

The results obtained from this training were satisfactory and matched the results observed in iteration 4 of the training. A very minimal deviation in plots and most values lie around the index line. Thus, a minor improvement is seen in the results even though we increased the number of neurons multiple times. Figure 22 shows the abbreviated layer notation in the three-layer network used in Iteration five. It shows the number of neurons used in each layer.

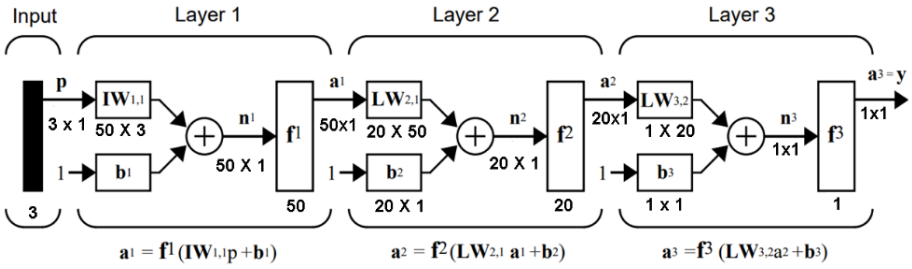


**Figure 22. Abbreviated layer notation used in iteration five**

Neurons in the first hidden were increased from 10 to 50 and from 1 to 20 on the second layer. Hence, we can conclude that no further training is required for this data set. However, if we decide to train the network further, this model would be considered 'Over-fit' as the neurons start memorising the values and giving incorrect results in real scenarios.

## DATA PROCESSING AND SIMULATION

We tested the application by simulating the data after training the dataset as per the strategy used in the last section. Testing was performed twice, with two datasets, i.e., 1,000 and 10,000, one with security and attack mode enabled and the other as disabled. The time instance of every set remains the same as 180 seconds. This simulation process remains the same as performed in our previous tests. The main point of difference is that neural networks trained data to detect and disable suspicious clients. During this simulation, we captured only the required and necessary artefacts to prove the correct functioning of the simulation. The simulation was first performed with 1,000 clients to test the correct functioning of the application. The second iteration was performed to check the application's behaviour when encumbered with ten times the number of clients.

The application configuration was changed from a generic IoT ecosystem to a healthcare ecosystem to match its unique requirements.

Some of the changes made in the application are as follows:

- New slices were added to cater for the need for healthcare. Names of the slices are mentioned in Figure 26.
- Mobility pattern percentage was reconfigured as per the healthcare system.
- A new slice as a backup for critical care services ensures uninterrupted services in attack or network failure.
- Training for the data and simulation was done only for one type of attack.
- All the other changes are handled manually.

Security and attack mode was enabled for two iterations. Figure 23 shows that the enabled mode using which we performed the simulation. Enabling both modes helps us test the durability and robustness of the application. Figure 24 shows that the client (71 and 83) using the slice named 'e-smart devices' are connected and requested the required bandwidth to perform their operation. Both these clients are successfully connected and are performing their operations.

```
attack_mode: True
level_one: True
level_two: True
security: True
```

**Figure 23. Status of security and attack mode**

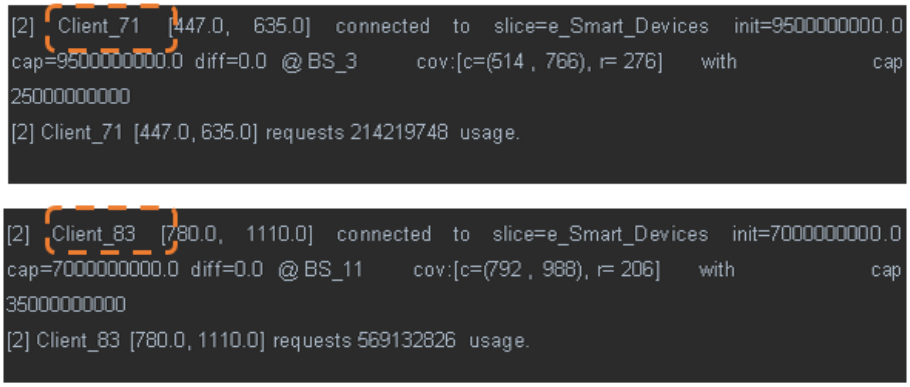


**Figure 24. Logs for connected client**

Figure 25 shows the final output of the simulation. In this simulation, we have performed all the use cases discussed in the section, "Healthcare use cases". Figure 25 shows the output of each of the use cases marked as 1, 2, 3 and 4. An explanation of each use case is given in the next section.

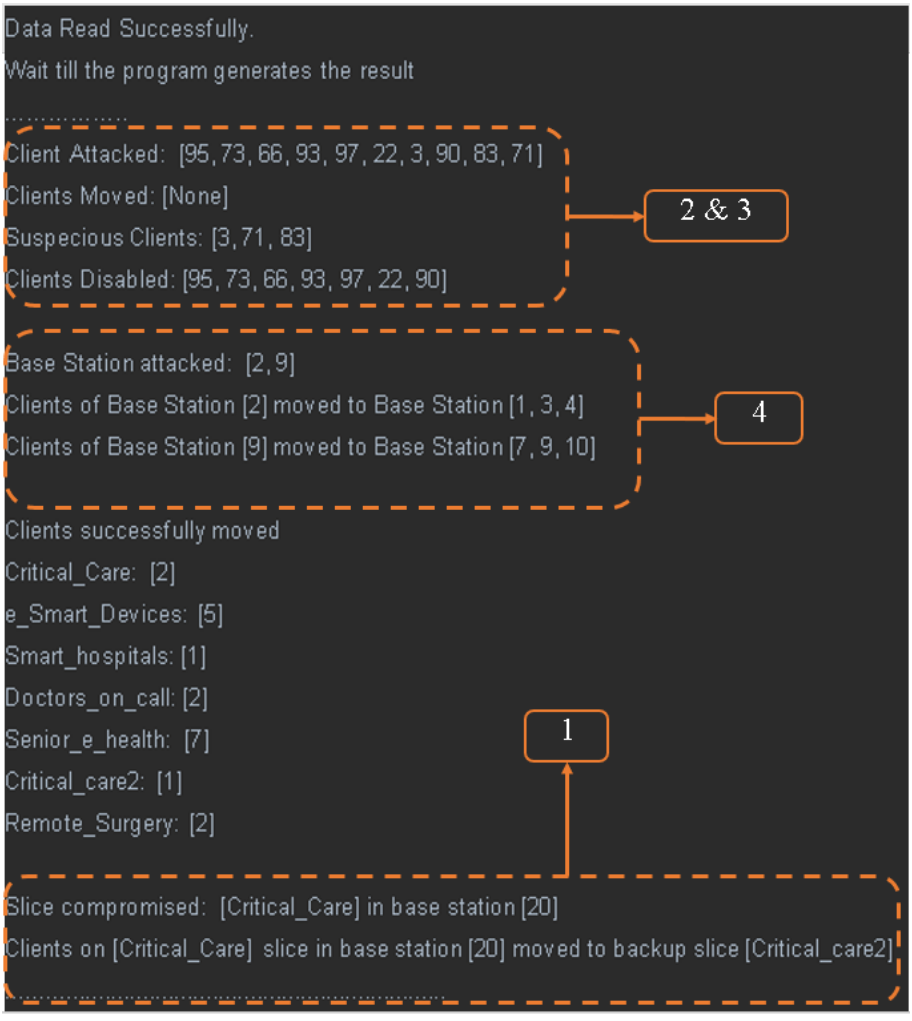


**Figure 25. The final output of the simulation**

### *Case 1. Life-Critical Services*

If we refer to point one of Figure 25, one module related to critical service was compromised. The following line shows that the clients attached to it are immediately migrated to the backup slice reserved for critical services or the slice reserved on another station. This use case is mission-critical for healthcare service as it is life-impacting.

### *Case 2. Non-Critical Services*

In point two of Figure 25, some of the clients were detected as attacked. These clients were categorised as suspicious and disabled based on the learning by neural network training. The output values of the client, which lies near zero, are marked as disabled because they are considered compromised devices, whereas the client whose value lies between zero and one are marked as suspicious. Clients with value one continue to enjoy uninterrupted services. The alert is generated for the clients considered attacked, and the administrator is informed about both cases. The rest of the decision is left to the discretion of administrators to investigate and decide.

### *Case 3. Suspicious Devices*

The suspicious devices can use either critical or non-critical services. If the device uses a non-critical service, it is either disabled or marked as suspicious, and the application sends the changes to the administrator. If any client uses a critical care service and is suspicious, the device is marked as suspicious, and an alert is sent to the administrator. The critical care service device is not disabled immediately to avoid any harm caused by false alerts.

### *Case 4. Base Station*

Section 4 of Figure 24 shows the action taken in case any base station is compromised. As mentioned earlier, this is a rare scenario, but the application should be programmed to handle such scenarios. Figure 25 shows that base stations 2 and 9 were compromised, and the clients using the services of these base stations were moved to different base stations where empty slots were available.

Figure 26 and Table 8 show the output generated from the simulation performed. As discussed earlier in this section, we have performed four iterations with two datasets of 1,000 and 10,000 each. The first two iterations tested the application's functionality, and subsequent two iterations with the dataset of 10,000 were performed to test the robustness of the application.

**Table 8. Output values of simulation**

| Iteration | Security | Clients | TALR | TC | TuB(Gbps) | TALR | CR | BC | HC |
|---|---|---|---|---|---|---|---|---|---|
| 1 | On | 1,000 | 0.84 | 840 | 51.057 | 0.08 | 0.87 | 0.0773 | 0.0013 |
| 2 | Off | 1,000 | 0.9 | 900 | 56.936 | 0.07 | 0.94 | 0.1063 | 0.0013 |
| 3 | On | 10,000 | 0.71 | 7100 | 389.367 | 0.53 | 0.81 | 0.2366 | 0.0058 |
| 4 | Off | 10,000 | 0.72 | 7200 | 415.111 | 0.53 | 0.84 | 0.2651 | 0.0071 |

From Table 8, we can make the following observations:

- The average load ratio(TALR) increased when the security mode is disabled.
- Total connected users(TC) decreased with Security mode 'ON'.
- Bandwidth usage(TuB) increased without security mode.
- Handover (HC) also increased without security mode.

These results are expected and were the basis of enabling and enhancing the security in our solution. As mentioned in the methodology section, we have restricted the bandwidth to avoid its misuse, resulting in the desired output.

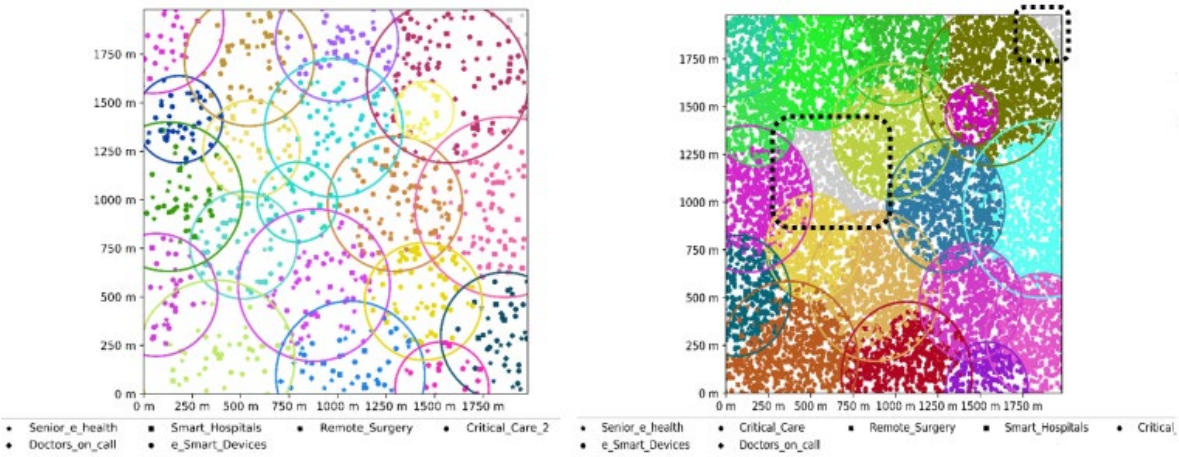


**Figure 26. Output for iterations 2 and 3**

The output in Figure 26 the results obtained out of iterations 2 and 3. Iteration 2 was performed with 1,000 devices and security mode disabled, whereas iteration 3 was performed with 10,000 devices with security mode enabled. Both the graphs in Figure 26 show the location and slices of all the 1,000 and 10,000 devices. Two slices highlighted in the dotted squares show the slices disabled after an attack.

Tests ensure that all the application modules are working fine with no adverse impact. The load was increased from 1,000 to 10,000 with both the security and attack mode 'ON' and one module enabled with neural network learning. Another motive to conduct these four iterations was to check the legitimacy of the results and establish a relation among all the test scenarios. The relation among all can be established from the data presented in Table 8.

## COMPARATIVE ANALYSIS OF RESULTS

The main objective of the simulation performed in this paper was to implement security using Machine learning in IoT based healthcare domain. A neural network was used to train the data set collected during different phases of the simulation. The design and training methodology discussed in the previous section of this paper explains the procedure and the changes made into the application to incorporate the enhancement. There are two types of results that we are comparing in this section:

- Results of training using a neural network on the dataset.
- Results of application after incorporating machine learning using a neural network.

We performed multiple iterations for obtaining the best results out of both tests. This section discusses and validates the results obtained during the simulation.

### *DATASET TRAINING ANALYSIS*

The dataset training was done using the MATLAB Artificial Neural Network module in five iterations. Figure 27 shows the results obtained from all five iterations. Training of dataset started with a dataset size of ~500, which was later increased to ~400,000 to obtain the best and optimum results. It is clear from Figure 27 that the results obtained from iterations 4 and 5 are best where R-value exceeded 0.99. However, an R-value of 1 is considered perfect, but it might not be possible to get the exact value for multiple reasons. In these scenarios where sometimes output can be assumptive, a

perfect R-value is considered 'Over-fit'. Training iterations 1-3 were performed using a single hidden layer and tansig as the transfer function. Tansig is good at handling linear regression functions, whereas we expect the output value between 0 to 1. Therefore, iterations 4 and 5 were conducted using two hidden layers and the transfer function of the first hidden layer was changed to logsig.

The change was visible in iterations 4 and 5, where we obtained almost perfect results. Figure 27 shows the function used in these iterations. Again, a device value 0 is blocked, and a device with the value of 1 is considered an appropriate device.

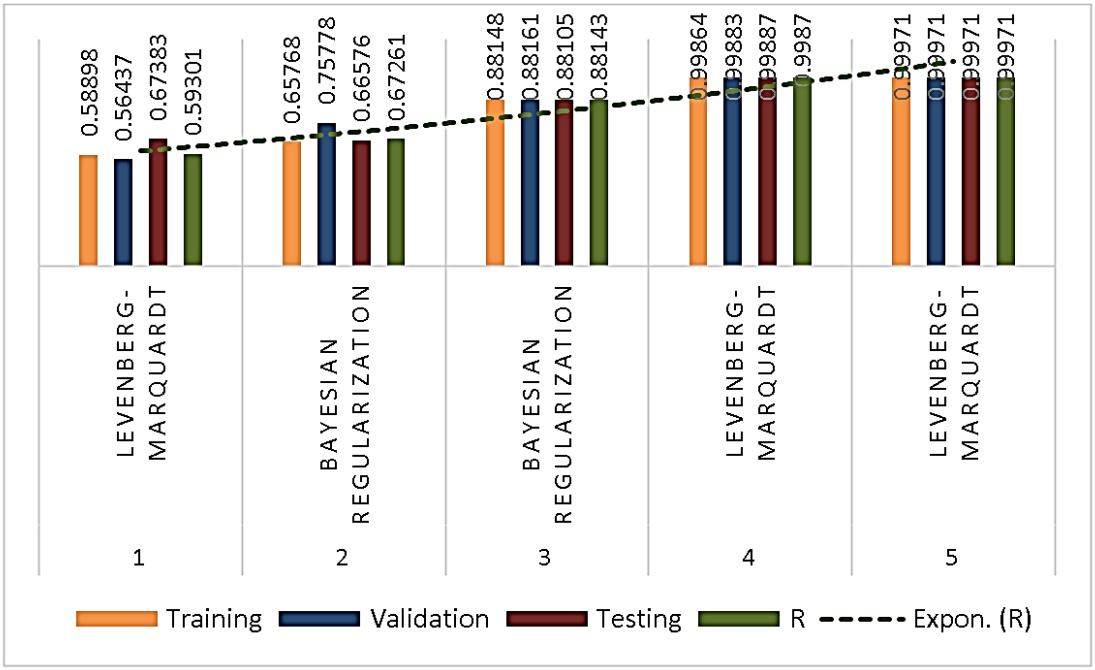


**Figure 27. Training result**

Figure 28-A shows the deviation in values obtained by results generated from iteration 5 of the training. This iteration is considered the 'Just-fit' case out of all the five pieces of training conducted. Figure 28-B also shows that there are chances of false-positive and true-positive cases. The same is visible in the error histogram of Figure 28-C. The error value is 3.14E-05 (0.0000314), which almost lies near 0. All these readings show many chances of error because of the deviation, as shown in Figure 28-A. Such deviations are handled by sending alerts to administrators using the alert management module of the IoT layer architecture. Therefore, as discussed in Figure 4, the alert and administrator module becomes essential to handle such scenarios.

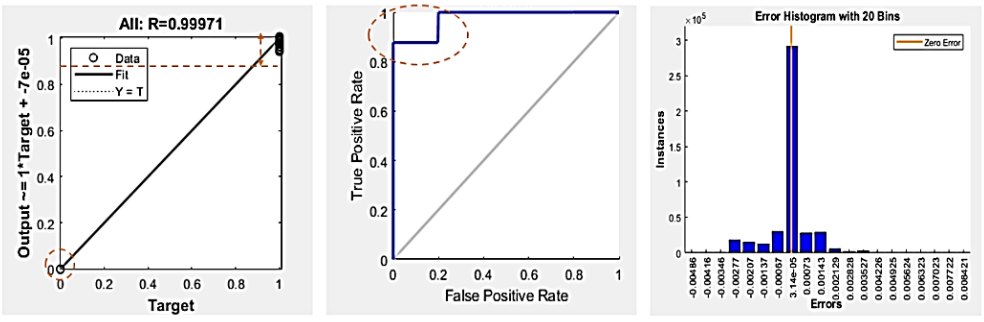


**Figure 28. Training analysis**

In Figure 29, the first two diagonal cells show the number and percentage of correct classifications by the trained network. For example, 156,800 bandwidths are correctly classified as blocked. It corresponds to 35.7% of all datasets. Similarly, 279,627 cases are correctly classified as ok to proceed, corresponding to 63.7% of all datasets.

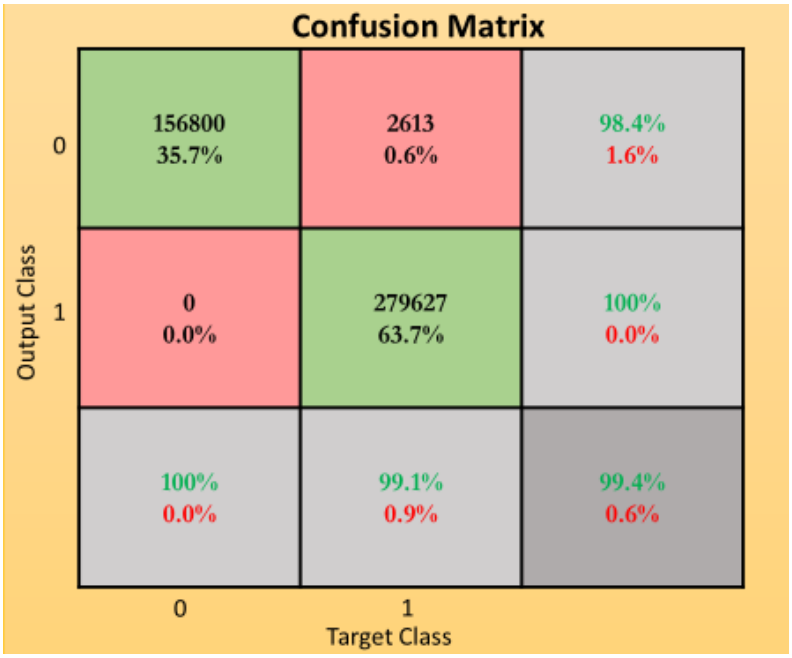


**Figure 29. Confusion matrix**

Of the datasets, 2,613 are incorrectly classified as blocked, corresponding to 0.6% of all the datasets. Similarly, 0 of the bandwidth data are incorrectly classified as allowed, corresponding to 0% of all data. The result obtained is perfect because none of the suspicious clients is allowed to use services. Administrators manually handle such cases.

- For output class 0, out of 159,413 bandwidth predictions, 98.4% are correct, and 1.6% are wrong.
- For output class 1, out of 279,627 client bandwidth predictions, 100% are correct, and 0% are wrong.
- For target class 0, out of 156,800 cases, 100% correctly predicted, and 0% predicted as blocked.
- For target class 1, out of 282,240 cases, 99.1% were correctly classified as allowed to use services, and 0.9% were classified as suspicious.

Overall, 99.4% of the predictions are correct, and 0.6% are wrong. 0.6% seems to be a meagre figure, but if we consider many customers using the services, then 0.6% can impact a large number of clients. Therefore, a large number can, in turn, increase the load on administrators and agents handling such cases.

## *Result Analysis of Application Simulation*

The application simulation was performed to ensure that results obtained after the inclusion of the neural network module are optimal and that there is no adverse impact on the application.

As shown in Figure 30, four iterations of application simulations were performed to match the results. The first round of simulation was done with 1,000 clients, followed by 10,000 clients. Each

round was simulated with and without a security feature. Time instance was constant at 180 seconds. All the relations among the data generated were analysed as correct. We need to note that both the security and attack mode in simulation number 1 and 3 was enabled, whereas in iteration number 2 and 4, both these modes were disabled. These simulations were performed to establish the relationship and confirm that the output generated is correct. Table. 9 shows the definitions of the data captured during simulation.

**Table 9. Definition of values**

| Definition | Total connected users | Total used bandwidth | Average slice load ratio | Average slice client count | Coverage Ratio | Handover Count | Block Count |
|---|---|---|---|---|---|---|---|
| Symbol | TC | TuB | TALR | TACC | CR | HC | BC |

Total connected users (TC): We can see that while the total connected users average was around 90% when the simulation was tried with 1,000 clients, it was reduced to around 70% in the case of 10,000 clients. The drop in the number of connected users is as expected due to resource limitations, and as the active number of users was high, it becomes hard to get new connections.

Total used bandwidth (TuB): TuB is the bandwidth allocated to the total connected users (TC). The TuB is directly proportional to the TC. Therefore, as the number of clients increases, the bandwidth allocated also increases and vice versa. The same is clear from Figure 30, where TuB increases with the total number of clients.

Handover Count (HC): HC is the count of client handover among different base stations. Handover happens with the movement of a device from one location to other, which is done on the base station at the movement site. HC is also directly proportional to the TC, more connected clients means more handover.

Most of the other terminologies are self-explanatory.

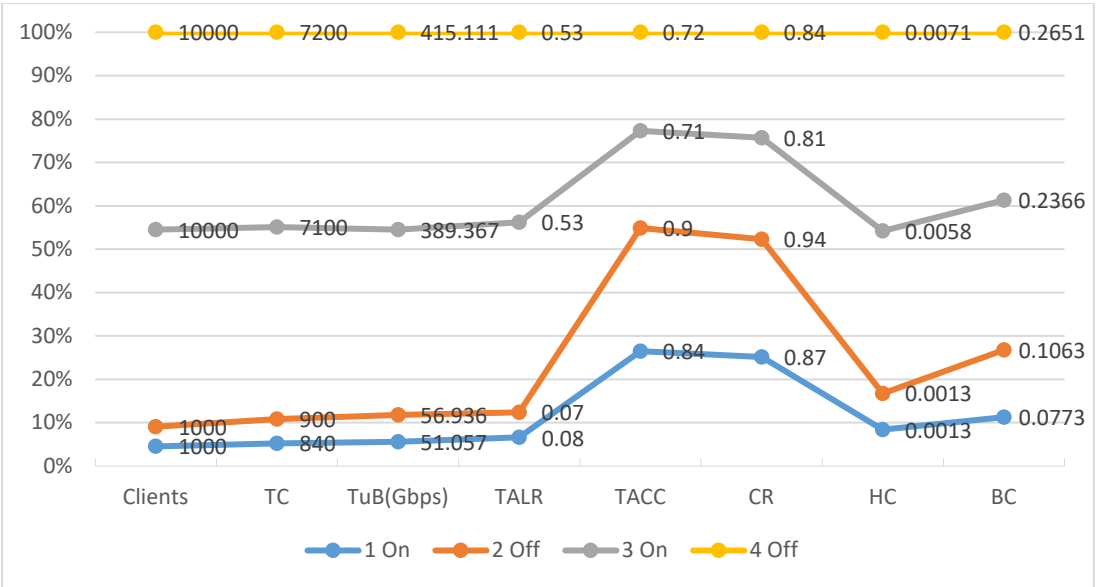


**Figure 30. Simulation results**

Figure 30 shows that the results generated in all four simulations are correct and expected. The results are following the real-world scenarios. There is no significant impact on the application's performance by introducing ANN in the bandwidth module of the application. It can be analysed that the application worked as expected during all four iterations of the simulations.

Figure 25 shows that the four case studies studied by enabling security and attack mode worked fine. Devices were marked as suspicious, and few others were blocked. Impact on the critical care services was handled promptly, and impacted devices were moved to the backup slice to ensure seamless services. If the base station is impacted, all the resources belonging to it are migrated to other free base stations.

Hence, we can conclude that the relation among all the tests performed is established. The security feature of the application helps detect suspicious devices and blocks them if required. All the required snapshots are copied here to prove them and establish the facts.

## *Research Analysis and Discussion*

This paper has amplified and implemented our previous research to include machine learning for detecting and disabling the devices, termed as suspicious. Here, we have implemented the solution to the healthcare domain, which also shows the flexibility that it can be easily moulded and implemented to any other domain. We have selected healthcare because, due to the increasing impact of COVID, this sector is being stress-tested for the last couple of years and requires some cost-effective security solutions. Most of the solutions discussed in the literature review section propose a partial solution for any one domain or only provides a data privacy solution. Our research has designed a machine learning solution and implemented it using network slicing technology. We have already explained the technical aspects of the results, and the experiments were done in several iterations in the previous section. The four use cases are unique and are applied only to the healthcare care domain.

As discussed previously, Makani and Reddy (2018) proposed a machine learning solution that acts as the first layer of defence that works below the application layer. P. et al. (2021) suggests a solution that detects DoS/DDoS on the IoT ecosystem. In contrast, our machine learning solution provides multi-layers of defence that can work on network-layer and application layers. While working on the network layer, it can detect attacks on or by IoT devices. Furthermore, it can help isolate attacks on/by the device using critical and non-critical services on the application layer. Such capability is missing in most of the other solutions discussed previously.

The resource used was very optimal under the test conditions, and we were able to execute all the simulations on a typical desktop machine. As mentioned in the Future Work section, a test scenario created under the production scenario would explain the exact details. The additional resource required in this experiment was used for the machine learning module, and automation was used to handle four use cases.

The two other solutions discussed previously and proposed by Hamza et al. (2020) and Luo et al. (2018) present healthcare data privacy solutions. However, the methodology used by both these papers is different where the first solution used encryption and the latter developed a cloud-based solution for secure transfer of medical records. Our solution proposes an end-to-end security solution for each layer, and the proposed architecture also caters to the whole healthcare ecosystem. Furthermore, the administrator and alert management module proposed in the layered architecture is an added feature. The design of this solution is modular, and anyone can easily configure it to suit the needs of their ecosystem, which is missing in most of the solutions discussed previously.

The solution discussed by Jimenez and Torres (2015) provides a monitoring and alerting solution which can be integrated with our solution to ensure data privacy and security. Moreover, our solution prioritises the patients based on the criticality in case any attack scenario occurs.

We have discussed a few of the existing problems and solutions provided in the previous researches in our literature review section. The different gaps identified in the literature review varied differently. Most of the solutions failed to handle the life saving critical services separately and provided encryption or user authentication and authorisation solutions. These points address the security issue to some extent but are not an end-to-end solution. We didn't find any related research that can cater to all the four use cases in a single solution cost-effectively using machine learning. The design methodology explains every aspect of the solution in a detailed way showing the application and layer architecture. There are many IDS solutions currently available in an open market; few of them are discussed here. Individually looking, the solutions are amicable and provide the best market solution, but the gaps pop up when combined with healthcare. Many of the prevalent gaps can get filled by using this solution. Moreover, the flexibility of this solution is an added advantage, and it can be moulded and configured easily as per the applicable use case in most of the IoT ecosystems.

Hence, we can easily conclude that the solution presented in this research is unique in many aspects to most of the prevalent solutions in many ways discussed above and very much comparable to the other similar solutions available in the market. The main benefit of this solution is the ability to handle security in many ways and domains and its pliability to get easily adapted. The main goal of this research was to provide a cost-effective solution that can be easily implemented and used worldwide. We think it has been fulfilled to some extent; more work is required to automate this solution fully, and all the modules enabled with artificial intelligence.

## CONCLUSION AND FUTURE WORK

This research aimed to develop a secure and cost-effective solution that can be deployed in the healthcare ecosystem. A 5G slicing system was selected to achieve this objective, and the machine learning module was introduced to make the solution more robust and intelligent. With the penetration of numerous IoT devices in the healthcare domain, achieving this objective was much more manageable. This paper proposed an IoT-enabled healthcare ecosystem that included almost all healthcare components like ambulances, hospitals, and many health monitoring devices. The IoT layered architecture discussed in the design and methodology section provides a high-level view of the application design and the placement of its different modules. Alert management, log analysis, intrusion detection, and administrators module are the most crucial application parts. These modules manage the whole functioning of the application because devices at all layers are directly or indirectly connected and coordinated by these modules only. Additionally, the modules are managing all the essential operations and functioning of the application.

The healthcare ecosystem has always been the centre stage for any economy throughout the world. A country is considered developed, developing or underdeveloped based on the healthcare infra and services available. Because of the nature of data generated, privacy is one of the most important factors when considering healthcare. Privacy of any data is directly related to its security when at various stages of its life, i.e., data in transit, data in process, data at rest, and while being destroyed. This research aims to provide a security solution for the healthcare ecosystem to provide seamless services while detecting and eliminating suspicious devices in the network and ensuring zero impact for critical care units. As mentioned earlier, this research is an advancement of our previous research, where a generic security solution using knowledge-based IDS was implemented.

To conclude that the application's performance and behaviour remain the same, even after including a new module, the simulation was performed using 1,000 and 10,000 devices respectively, with and without security mode. Again, the results have proved that there was no significant impact on the performance of the applications. We selected the four most relevant use cases in our simulation, one of which was for critical care services. Handling critical care services was crucial because any impact on life-saving services nullifies the whole purpose of this prolonged exercise. However, if implemented, this solution can work as a boon in the current scenario where COVID impacts the whole

world, especially for the underdeveloped and developing countries whose healthcare system has almost collapsed.

Analysis of the trained dataset shows that the training performed with the neural network using two hidden layers and logsig as the transfer function on the first hidden layer provided the best results. This combination was performed using iteration numbers four and five. The number of neurons on both the layers was increased multiple times in iteration five. However, the data training accuracy was increased marginally from 0.9987 to 0.99971. The result was not as expected in iterations one, two, and three, where we used a single hidden layer and tansig transfer function. The result obtained concludes that using multiple hidden layers and a non-linear transfer function logsig made a huge difference. This simulation is a subset of the more extensive simulation performed by us on the 5G network sliced network to enhance its security. The module trained with the neural network can be applied to make the network more intelligent and robust. This enhancement helps detect the probable and suspicious devices and isolate them before any harm is caused on the network. The solution works both as an intrusion detection and prevention system by detecting and blocking them from using network resources. We can discuss the system's accuracy because 1.6% of devices were incorrectly identified and requires manual administrators intervention and decision to allow or block them. However, we have handled this scenario for life-saving critical services by not blocking them in any scenario and passing the message to administrators.

Here, we would again like to repeat the statement, "Imaginations and technology enhancements have no limit". We have enhanced our research from testing ten devices with implementing security to 10,000 devices and included neural networks to train the network and detect suspicious devices. As a result, practitioners can efficiently use this solution as an affordable answer to the costly security solutions available in the commercial market and deploy it over a sliced network. However, there is always a scope for improvement, and telecom is the liveliest example of this. Telecom technology has grown from 1G to 5G and is currently developing 6G technology. Jain et al. (2021) have discussed using artificial intelligence to enhance the solution, whereas our paper accomplishes it to some extent by implemented ML using ANN. Hereon, we can enhance this application to a new level where new security modules like authentication and authorisation mechanism, and data encryption can be included.

Furthermore, machine learning and artificial intelligence can be extended to other parts of this application as well. For example, in the literature review section of this paper, we have reviewed some of the encryption mechanisms designed specially to ensure the privacy of healthcare records. In addition, we can include authentication modules to make it more robust and secure at all levels. Finally, we can say that further prospects can be achieved by deploying and testing it in a live environment by having artificial intelligence enabled in the remaining modules.

## REFERENCES

Adams, C. (2005). Impersonation attack. In H. C. A. van Tilborg (Ed.). *Encyclopedia of cryptography and security*. Springer. https://doi.org/10.1007/0-387-23483-7_196

Alomari, M. H., Younis, O., & Hayajneh, S. M. A. (2018). A predictive model for solar photovoltaic power using the Levenberg-Marquardt and Bayesian Regularisation algorithms and real-time weather data. *International Journal of Advanced Computer Science and Applications*, *9*(1): 347-353 https://doi.org/10.14569/IJACSA.2018.090148

American Nurses Association (ANA). (2017). *Defining staffing: Workforce management patient classification and security system.* https://www.nursingworld.org/~497e37/globalassets/practiceandpolicy/work-environment/nurse-staffing/website-staffing-and-acuity-systems-pdf-final_2017.pdf

Andrews, J. G., Buzzi, S., Choi, W., Hanly, S. V., Lozano, A., Soong, A. C. K., & Zhang, J. C. (2014). What will 5G be? *IEEE Journal on Selected Areas in Communications*, *32*(6), 1065-1082. https://doi.org/10.1109/JSAC.2014.2328098

Ansari, A., Mohaghegh, SD., Shahnam, M., & Dietiker, J-F. (2019). Modeling average pressure and volume fraction of a fluidised bed using data-driven smart proxy. *Fluids*, *4*(3): 123. https://doi.org/10.3390/fluids4030123

Barakabitze, A. A., Ahmad, A., Mijumbi, R., & Hines, A. (2020). 5G network slicing using SDN and NFV: A survey of taxonomy, architectures and future challenges. *Computer Networks*, *167*, 106984. https://doi.org/10.1016/j.comnet.2019.106984

Barakabitze, A. A., Barman, N., Ahmad, A., Zadtootaghaj, S., Sun, L., Martini, M., & Atzori, L. (2020). QoE management of multimedia streaming services in future networks: A tutorial and survey. *IEEE Communications Surveys & Tutorials*, *22*(1), 526-565. https://doi.org/10.1109/COMST.2019.2958784

Casas, P., Fiadino, P., & D'Alconzo, A. (2016, April). Machine-learning based approaches for anomaly detection and classification in cellular networks. *8th Traffic Monitoring and Analysis Workshop, Louvain La Neuve, Belgium*. http://dl.ifip.org/db/conf/tma/tma2016/tma2016-final50.pdf

Centers for Disease Control and Prevention (CDC). (2018). *Health Insurance Portability and Accountability Act of 1996 (HIPAA)*. https://www.cdc.gov/phlp/publications/topic/hipaa.html

Chacko, A., & Hayajneh, T. (2018). Security and privacy issues with IoT in healthcare. *Pervasive Health and Technology, 18*(14): e2. https://doi.org/10.4108/eai.13-7-2018.155079

Chandra-Sekaran, A. K., Dheenathayalan, P., Weisser, P., Kunze, C., & Stork, W. (2009). Empirical analysis and ranging using environment and mobility adaptive RSSI filter for patient localisation during disaster management. *Fifth International Conference on Networking and Services*, 276-281. https://doi.org/10.1109/ICNS.2009.63

Chawla, S., & Thamilarasu, G. (2018). Security as a service: Real-time intrusion detection in internet of things. *Proceedings of the Fifth Cybersecurity Symposium* (Article 12). Association for Computing Machinery. https://doi.org/10.1145/3212687.3212872

Condoluci, M., Sardis F., & Mahmoodi T. (2016). Softwarization and virtualisation in 5G networks for smart cities. In B. Mandler, J. Marquez-Barja, M. E. M. Campista, D. Cagáňová, H. Chaouchi, S. Zeadally, M. Badra, S. Giordano, M. Fazio, A. Somov, R-L. Vieriu (Eds.), *Internet of Things. IoT Infrastructures. IoT360 2015. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, Vol 169*. Springer. https://doi.org/10.1007/978-3-319-47063-4_16

Cui, M., Wang, J., & Yue, M. (2019). Machine learning-based anomaly detection for load forecasting under cyberattacks. *IEEE Transactions on Smart Grid*, *10*(5), 5724-5734. https://doi.org/10.1109/TSG.2018.2890809

D'Souza, M., Wark, T., & Ros, M. (2008, December). Wireless localisation network for patient tracking. *2008 International Conference on Intelligent Sensors, Sensor Networks and Information Processing*, *Sydney, Australia,* 79-84. https://doi.org/10.1109/ISSNIP.2008.4761966

Dey, S. K., & Rahman, M. M. (2019). Effects of machine learning approach in flow-based anomaly detection on software-defined networking. *Symmetry*, *12*(1), 7. https://doi.org/10.3390/sym12010007

Freitas, D., Lopes, G. L., & Morgado-Dias F. (2021). A neural network-based approach for approximating arbitrary roots of polynomials. *Mathematics*, *9*(4), 317. https://doi.org/10.3390/math9040317

Garoosiha, H., Ahmadi, J., & Bayat, H., & Formisano, A. (2019). The assessment of Levenberg-Marquardt and Bayesian Framework training algorithm for prediction of concrete shrinkage by the artificial neural network. *Cogent Engineering*, *6*(1). https://doi.org/10.1080/23311916.2019.1609179

Geekflare (2020, August 21). *12 Open source Internet of Things (IoT) platforms and tools.* https://geekflare.com/iot-platform-tools/

Github (2020). *SliceSim: A simulation suite for network slicing in 5G networks.* https://github.com/cerob/slicesim

Gloss, K. (2020, August 14). Healthcare IoT security risks and what to do about them. *IoT Agenda.* https://internetofthingsagenda.techtarget.com/feature/Healthcare-IoT-security-issues-Risks-and-what-to-do-about-them

Goldberg, R. P. (1974). Survey of virtual machine research. *Computer*, *7*(6), 34-45. https://doi.org/10.1109/MC.1974.6323581

Hamza, R., Yan, Z., Muhammad, K., Bellavista, P., & Titouna, F. (2020). A privacy-preserving cryptosystem for IoT E-healthcare. *Information Sciences*, *527*, 493-510. https://doi.org/10.1016/j.ins.2019.01.070

HCPRO (2019, September 10). Patient classification systems to coordinate patient care. *Nurse leader Insider*. http://www.hcpro.com/NRS-279130-975/From-the-staff-development-bookshelf-Patient-classification-systems-to-coordinate-patient-care.html

Hodo E., Bellekens, X., Hamilton, A., Dubouilh, P., Iorkyase, E., Tachtatzis, C., & Atkinson, R. (2016, May). Threat analysis of IoT networks using artificial neural network intrusion detection system. *2016 International Symposium on Networks, Computers and Communications (ISNCC)*, *Yasmine Hammamet, Tunisia*, 1-6. https://doi.org/10.1109/ISNCC.2016.7746067

Hu, Q., (2019). *Notes of backpropagation & Levenberg–Marquardt & Bayesian regularisation algorithm*. https://blog.tonyhao.xyz/2019-06-06/Notes_BP_LM_BR

Huch, F., Golagha, M., Petrovska, A., & Krauss, A. (2018, March). Machine learning-based run-time anomaly detection in software systems: An industrial evaluation. *2018 IEEE Workshop on Machine Learning Techniques for Software Quality Evaluation, Campobasso, Italy,* 13-18. https://doi.org/10.1109/MALTESQUE.2018.8368453

Ibrahim, A., Tarik, T., Konstantinos, S., Ksentini, A., & Flinck, H. (2018). Network slicing and softwarization: A survey on principles, enabling technologies, and solutions. *IEEE Communications Surveys & Tutorials*, *20*(3), 2429-2453. https://doi.org/10.1109/COMST.2018.2815638

Jain, A., & Singh, T. (2020). Security challenges and solutions of IoT ecosystem. In M. Tuba, S. Akashe, & A. Joshi (Eds.), *Proceedings of the Information and Communication Technology for Sustainable Development Conference*, *Vol. 933 (pp.* 259-270). Springer. https://doi.org/10.1007/978-981-13-7166-0_25

Jain, A., Singh, T., & Sharma, S. K. (2018). Threats paradigm in IoT ecosystem. *2018 7th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*, pp. 1-7, IEEE. https://doi.org/10.1109/ICRITO.2018.8748558

Jain, A., Singh, T., Sharma, S. K., & Prajapati, V. (2021). Implementing security in IoT ecosystem using 5G network slicing and pattern matched intrusion detection system: A simulation study. *Interdisciplinary Journal of Information, Knowledge, and Management*, *16*, 1-38. https://doi.org/10.28945/4675

Jain, A., Singh, T., Sharma, S. K., Marwaha, R., & Bardia, R. (2020). Securing IoT endpoints using network slicing on 5G network. *Solid State Technology*, *63*(5), 689-705. http://solidstatetechnology.us/index.php/JSST/article/view/1615

Jimenez, F., & Torres, R. (2015, November). Building an IoT-aware healthcare monitoring system. *34th International Conference of the Chilean Computer Science Society, Santiago, Chile,* 1-4. https://doi.org/10.1109/SCCC.2015.7416592

Lam, J., & Abbas, R. (2020). Machine learning based anomaly detection for 5G networks. *ArXiv*. http://arxiv.org/abs/2003.03474

Luo, E., Bhuiyan, M. Z. A., Wang, G., Rahman, M. A., Wu, J., & Atiquzzaman, M. (2018). Privacyprotector: Privacy-protected patient data collection in IoT-based healthcare systems. *IEEE Communications Magazine*, vol. 56, no. 2, pp. 163-168, Feb. 2018. https://doi.org/10.1109/MCOM.2018.1700364

Makani, R., & Reddy, B. V. R. (2018). Taxonomy of machine leaning based anomaly detection and its suitability. *Procedia Computer Science*, *132*, 1842-1849. https://doi.org/10.1016/j.procs.2018.05.133

Maple, C. (2017). Security and privacy in the internet of things. *Journal of Cyber Policy*, *2*(2), 155-184. https://doi.org/10.1080/23738871.2017.1366536

Mavrogiorgou, A., Kiourtis, A., Perakis, K., Pitsios, S., & Kyriazis, D. (2019). IoT in healthcare: Achieving interoperability of high-quality data acquired by IoT medical devices. *Sensors*, *19*(9), 1978. https://doi.org/10.3390/s19091978

Mitchell, R., & Chen, I. (2014). A survey of intrusion detection in wireless network applications. *Computer Communications*, 42, 1-23. https://doi.org/10.1016/j.comcom.2014.01.012

Mitrokotsa, A., Komninos, N., & Douligeris, C. (2007, July). Intrusion detection with neural networks and watermarking techniques for MANET. *IEEE International Conference on Pervasive Services*, *Istanbul, Turkey,* 118-127, https://doi.org/10.1109/PERSER.2007.4283901

Moosavi, S. R., Gia, T. N., Rahmani, A., Nigussie, E., Virtanen, S., Isoaho, J., & Tenhunen, H. (2015). SEA: A secure and efficient authentication and authorisation architecture for IoT-based healthcare using smart gateways. *Procedia Computer Science*, *52*, 452-459. https://doi.org/10.1016/j.procs.2015.05.013

Nanda, S., & Tzi-cker, C. (2005). *A survey on virtualisation technologies*. https://rtcl.eecs.umich.edu/papers/publications/2011/TR179.pdf

Nguyen, Q. H., Ly, H-B., Nguyen, T-A., Phan, V-H., Nguyen, L. K., & Tran, V. Q. (2021). Investigation of ANN architecture for predicting shear strength of fiber reinforcement bars concrete beams. *PLoS ONE*, *16*(4), e0247391. https://doi.org/10.1371/journal.pone.0247391

Nursing Guide. (2015). *The patient classification system*. https://www.nursingguide.ph/category-career-guides/the-patient-classification-system

Occhiuzzi, C., Vallese, C., Amendola, S., Manzari, S., & Marrocco, G. (2014). Night-care: A passive RFID system for remote monitoring and control of overnight living environment. *Procedia Computer Science*, *32*, 190-197. https://doi.org/10.1016/j.procs.2014.05.414

Open Networking Foundation (ONF) (2018). https://www.opennetworking.org/

P., J., Shareena, J., Ramdas, A., & P., H. A. (2021). Intrusion detection system for IOT botnet attacks using deep learning. *SN Computer Science 2*, 205. https://doi.org/10.1007/s42979-021-00516-9

Pat Research. (n.d.). *Top 27 artificial neural network software*. https://www.predictiveanalyticstoday.com/top-artificial-neural-network-software/

Perroca, M. G. (2009). Patients classification systems – the Brazilian experience. *Health Management*, *4*(1). https://healthmanagement.org/c/it/issuearticle/patients-classification-systems-the-brazilian-experience

Quintana, G., Rudolf, T., Ciurana, J., & Brecher, C. (2011). Surface roughness prediction through internal kernel information and external accelerometers using artificial neural networks. *Journal of Mechanical Science and Technology 25*, 2877-2886. https://doi.org/10.1007/s12206-011-0806-0

Redondi, A., Chirico, M., Borsani, L., Cesana, M., & Tagliasacchi, M. (2013). An integrated system based on wireless sensor networks for patient monitoring, localisation, and tracking. *Ad Hoc Networks*, *11*(1), 39-53. https://doi.org/10.1016/j.adhoc.2012.04.006

Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural Networks*, 61, 85-117. https://doi.org/10.1016/j.neunet.2014.09.003

Selvaraj, S., & Sundaravaradhan, S. (2020). Challenges and opportunities in IoT healthcare systems: A systematic review. *SN Applied Sciences 2*, 139. https://doi.org/10.1007/s42452-019-1925-y

Tao, H., Bhuiyan, M. Z. A., Abdalla, A. N., Hassan, M. M., Zain, J. M., & Hayajneh, T. (2019). Secured data collection with hardware-based ciphers for IoT-based healthcare. *IEEE Internet of Things Journal*, *6*(1), 410-420. https://doi.org/10.1109/JIOT.2018.2854714

Wong, B. K., Lai, S. V., & Lam, J. (2000). A bibliography of neural network business applications research: 1994-1998. *Computers & Operations Research*, *27*(11-12), 1045-1076. https://doi.org/10.1016/S0305-0548(99)00142-2

Younghwa, A. (2012). Security analysis and enhancements of an effective biometric-based remote user authentication scheme using smart cards. *BioMed Research International*, *2012*, Article 519723. https://doi.org/10.1155/2012/519723

Zolanvari, M., Teixeira, M. A., Gupta, L., Khan, K. M., & Jain, R. (2019). Machine learning-based network vulnerability analysis of industrial internet of things. *IEEE Internet of Things Journal*, *6*(4), 6822-6834. https://doi.org/10.1109/JIOT.2019.2912022

## AUTHORS

**Anshul Jain** received his Master's in Network Technology and Management from Amity University, Uttar Pradesh, India. He is currently pursuing his PhD from Amity Institute of Information Technology, Amity University, U.P., India.

He has a broad work experience of over 14 years in different multinational organizations in information security, integration, maintaining 5G and other Telecom software like VAS (VMS/SMSC/Prepaid Recharge/Mobile Money), IoT, Radio Frequency, Mobile Financial Services, VoIP. His interest and research areas include Information Security, the Internet of Things, Blockchain, Artificial Intelligence. Currently, he is working as a Technical Manager in Amdocs.

**Tanya Singh** received her Bachelor's degree in Electronics from MIT, Dr. Babasaheb Ambedkar University, India, Master's in Information Technology from the School of Information Technology, Guru Gobind Singh Indraprastha University, New Delhi, India, and her PhD in IT and Engineering from Banasthali University, India. Prof. (Dr.) Tanya Singh is currently working as Dy. Director (Academics), Amity University, Uttar Pradesh, India.

She has emerged as a Technical Evangelist for Networking and Cyber Security with a comprehensive experience of over 20 years in Academia, Research, Planning, and Development in Education and Operational roles. She aims to teach and learn new technologies, assist and develop the knowledge amongst students and instructors by encouraging critical thinking, implementation, and converting technological ideas into innovation. She has more than 40 research papers in reputed journals with Thompson ISI, Scopus.

**Satyendra Kumar Sharma** received his Bachelor's in Science in 1985 from CCS University, India, and a second Bachelor's in Computer Science in 1990 from Marathwada University, Aurangabad, India. He completed his Master's in Computer Science in 1993 from Roorkee University, India, and MBA in 2013 from Sikkim Manipal University, India. He got his first PhD in Operational Research in 2011 from CCS University, India, and his second PhD in IT from J R N Rajasthan Vidyapeeth (Deemed) University, India.

He is an academically enriched Educationist with rich academic and administrative experience, offering over an overall experience of 25+ years in various Engineering Colleges. He is an accomplished trainer and a mentor with a record of creative scholastic achievements. He has vast experience in education and research, having authored five technical books, around 100 research papers in different Journals and International and National Conferences.