Hindawi
Complexity
Volume 2021, Article ID 5572881, 10 pages
https://doi.org/10.1155/2021/5572881

Research Article

Selecting the Best Routing Traffic for Packets in LAN via Machine Learning to Achieve the Best Strategy

Bo Zhang and Rongji Liao

School of Information and Communication Engineering, Communication University of China, Beijing, China

Correspondence should be addressed to Bo Zhang; zhangbo2015@cuc.edu.cn

Received 4 February 2021; Revised 19 March 2021; Accepted 31 March 2021; Published 15 April 2021

Academic Editor: M. Irfan Uddin

The application of machine learning touches all activities of human behavior such as computer network and routing packets in LAN. In the field of our research here, emphasis was placed on extracting weights that would affect the speed of the network's response and finding the best path, such as the number of nodes in the path and the congestion on each path, in addition to the cache used for each node. Therefore, the use of these elements in building the neural network is worthy, as is the exploitation of the feed forwarding and the backpropagation in the neural network in order to reach the best prediction for the best path. The goal of the proposed neural network is to minimize the network time delay within the optimization of the packet paths being addressed in this study. The shortest path is considered as the key issue in routing algorithm that can be carried out with real time of path computations. Exploiting the gaps in previous studies, which are represented in the lack of training of the system and the inaccurate prediction as a result of not taking into consideration the hidden layers' feedback, leads to great performance. This study aims to suggest an efficient algorithm that could help in selecting the shortest path to improve the existing methods using weights derived from packet ID and to change neural network iteration simultaneously. In this study, the design of the efficient neural network of appropriate output is discussed in detail including the principles of the network. The findings of the study revealed that exploiting the power of computational system to demonstrate computer simulation is really effective. It is also shown that the system achieved good results when training the neural network system to get 2.4% time delay with 5 nodes in local LAN. Besides, the results showed that the major features of the proposed model will be able to run in real time and are also adaptive to change with path topology.

1. Introduction

Despite the development in communication technology, there are still some obstacles that are basically the main elements of data transmission throughout the network. The most important problems and obstacles are the path of information on the network. The information sent from the source takes multiple paths according to the protocols assigned to the network and the strategies drawn to transfer the data. Given the large number of data and the traffic jam of the data through the network paths, whether wired or wireless, the data takes a different path, but the destination is the same [1]. Multipath creates a problem with data flow times, especially when processing complex real-time data. So, managing this traffic is unavoidable.

There are methods used to solve network congestion through the LAN or within complex networks. Some solutions are in the form of hardware and some in the form of software. The use of machine learning algorithms is one of the most important and successful solutions that are based on learning the system by means of preknown results of experimental data, in order to solve the system in later of real data. Neural network algorithm is worthy for solving such problem same as classification by one of the currently available classifiers.

To increase the accuracy of choosing the path followed in the LAN, we need an algorithm that analyzes the most probabilities and determines the best one. And because of the deep learning algorithms, these algorithms have taken the advantage of, and developed to fit, the features extracted

from the given data. The nature of the features extracted in deep learning is computationally complex because of taking into consideration the outputs of each of the hidden layers and entering them as an input to the other layers, and it is called feed forwarding and backpropagation. The proposed method uses the perfect weights as features, and using powerful and suitable features helps the system predict the best paths within network.

In this study, an overview of the previous studies was presented in the section of the related work, followed by the proposed method for solving the problem, which includes most of the work in this study and the contribution of this study. At the end of the study, a section on conclusions and recommendations for future work was presented in detail.

The competence of deep learning in terms of the enormous quantity of data collected from various fields, from security to home user devices, is shown in Figure 1. Moreover, the neural network implements all of the deep learning techniques to describe an inherited data set, whose architectures are most likely known as deep neural networks.

The word 'deep' refers to many layers that are concealed in deep neural networks, unlike conventional neural networks that are composed of two or three hidden units; it is also noteworthy that over one hundred and fifty hidden layers can be kept by the deep neural network. Furthermore, we indicate the disparity in the categorization of an entity between conventional learning and deep learning. In addition, Figure 2 includes the grouping using the standard approach of learning and a form of deep learning that is implicitly learned from the data collection. The figure illustrates how the learning algorithm combines classical learning and deep learning to classify a target entity [2].

In this paper, an introduction to the routing algorithm is discussed, in addition to its relationship to deep learning. In the section on related work, previous studies related to the topic and solutions that were presented previously are discussed. After previous studies, a group of sections related to deep learning were presented in the form of paragraphs. The contributions to this study are followed by a brief section. The proposed method is discussed in detail. After that, the results are discussed in the Results section. Then, the conclusions section presented lastly contains a conclusion of what has been mentioned in the research.

2. Related Work

Machine learning provides an effective way to find the best solutions in problems that suffer from more than one available solution. In the event that there are several goals that need to be guessed by the best of them, we resort to machine learning in order for the choice to be accurate and effective, as is the case in LAN when the data path has more than one way for the packet to go to reach the target. Different properties are considered by routing algorithm like memory limitation, communication cost, and restriction of the energy. Existing researchers tried to utilize these properties in terms of machine learning. Many algorithms belong to machine learning used in the literature to use data of past traffic for learning machine to predict the

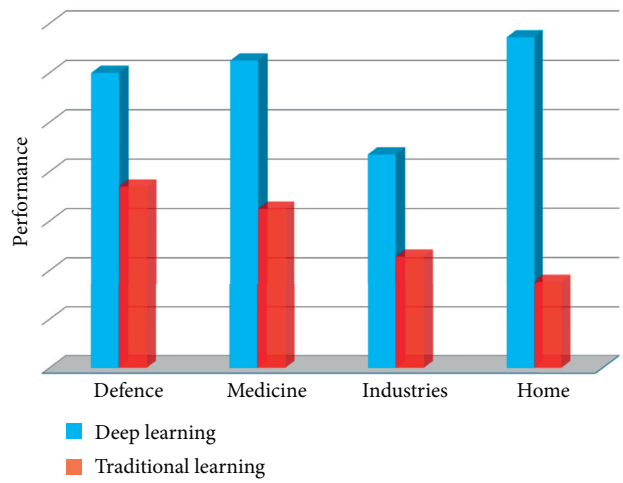

Figure 1: The use of deep learning in different fields of life.

best future routing. Reinforcement Learning (RL) machine learning technique was used in [3, 4] using computer program to generate rules or patterns to maximize network performance to estimate the optimal results from given environment.

An algorithm called Ant Colony Optimization (ACO) was established, derived from the actual ant's behavior. Pheromone was used as a mediator to communicate path within each other [5–7]. As known, the pheromone is alchemical unsteady substance, which emitted by the ant; then, it can keep track of previous paths and then make decision for the next movement. This algorithm makes visual connection between nodes for building good probability of distribution for prediction routing path.

Another algorithm of machine learning presented by [8, 9] is called Practical Swarm Optimization (PSO). Such algorithm depends on intelligent swarm nature; it is generated by the idea from behavior of bird's flock. This behavior includes how it moves in space searching for exploiting and exploring the shelter and food. PSO contains the number of particles whose name is swarm, and each one provides a solution. Fitness function is used to evaluate verified quality solution. PSO goal is to find optimal position of particle that was evaluated in advance by fitness function. This algorithm is actually used for energy consumption minimization and increasing lifetime of the network [8, 10, 11].

One of the interesting algorithms used in machine learning is Fuzzy Logic (FL), which is a mathematical system used to express the approximation of human reasoning. Intermediate value will establish that, depending on linguistic variables and inference rule protocol that uses FL used perfuming of optimizing route and clustering to get multiple objectives. Its aim is to build cost function based on fuzzy rule to optimize the objectives. In this algorithm, network lifetime of the network increased, and any change in topology can relearned by using generated rule of fuzzy logic [12].

In terms of machine learning, a new technique called Genetic Algorithm (GA) was used widely; it is based on

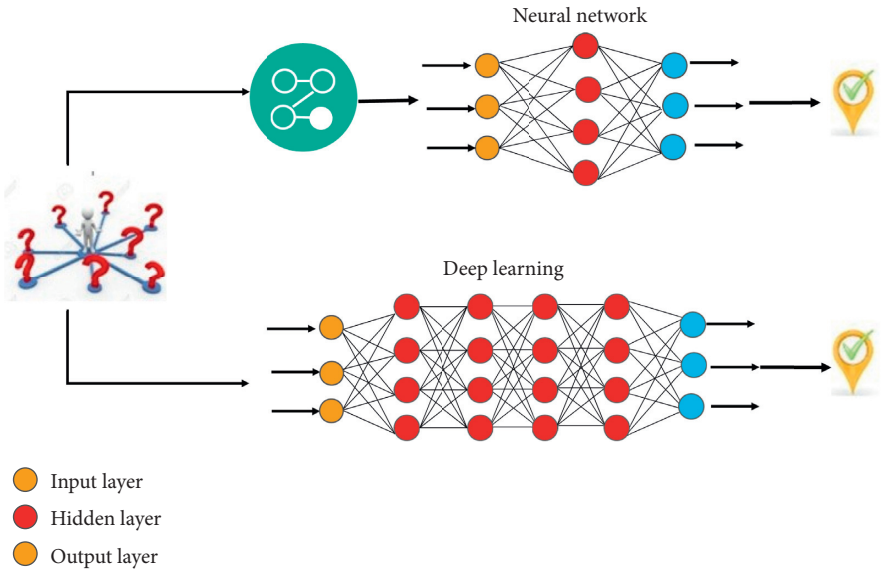

Figure 2: The difference between conventional and deep learning.

evaluation of the neural network that chooses the best population via performing fitness test to the new proposed structure. Chromosome group called population was used to perform solution for specified problem. This algorithm uses value fitness to illustrate the quality of chromosome as problem needed [12]. When chromosome has high fitness value, then this path is the optimal one. The two chromosomes may be inherited from the parent pair of chromosomes to behave differently in the next generation. By this algorithm, we can build clusters in LAN, and for routing aim to create tree of nodes, they will extend the lifetime of LAN.

A perfect algorithm is called Neural Network (NN), which consists of neurons, and the neurons are structured as input, hidden, and output layers. Different models of NN are suggested in the literature, all deferred by layers' connection [13–15]. Different patterns can take these structures and can be limited by the conditions of connections between neurons, learning starts to take the weight of features in different layers, and the pattern of network may change during learning. The testing mode takes effect of prediction status in the final pattern.

Yong et al. developed algorithm based on fuzzy of arc lengths to find shortest path in the network. They solved two main problems: first, determining two edges addition and finding the length of them. Then comparison between two paths by using graded mean integration to find the short one [16]. The method proposed to solve the congestion problem in the network depends on network of best effort. Feed forward of neural network is used to predict the network link, training of using traffic pattern is used to predict the congest path. Shortest path is used as a results of proposed method [17]. Interested algorithm improved that using weight function based on analysis, this algorithm by its result overcomes the shortcomings that appear [18]. A new method is presented to find many shortest paths within rectangular network, and the shortest path takes less time in the network according to travel time [19]. New technique is suggested based on density clustering specifying named Density-Based Spatial Clustering of Applications with Noise (DBSCAN), which is enhanced to Dijkstra's algorithm [20]. A method is introduced based on the shortest path first (SPF) algorithm based on Dijkstra's algorithm that uses the specific short path by searching through the network then building dataset to store these links to show router status [21].

2.1. The Perspective of the Network Expense in Forecasts. The precision of data for network monitoring comes at the expense of increased overhead monitoring (e.g., switch memory, as well as consumed network bandwidth). This stresses the need for both reliable and cost-effective network management schemes [20]. In conventional networks, monitoring applications depend on a predefined collection of tracking probes installed into the firmware/hardware, to reduce their versatility. To capture more diverse monitoring data, https://www.esds.co.in/blog/software-defined-networking-sdn-an-approach-for-flexible-networking/ (SDN) flexible software-based monitoring probes can be executed on demand. Nevertheless, these probes need to run at line rate in many conditions, for instance, tracking traffic volume over a stated switch interface, which is considered so costly over extremely high speed links, and it is hard for it to perform in software [21].

This permits TSF-based means forbidden for traffic forecast. To resolve this issue, two means have newly been examined: interpolation and traffic processing, in addition to the use of characteristics aside from traffic levels for traffic flow forecast. In particular, many flow sampling techniques (deterministic/stochastic, spatial/temporal, etc.) have been recommended within the literature to diminish overhead monitoring. Unfortunately, the latest ML-based method recommended is not decisive and also discloses

contradictory forecast accuracy consequences. In its place, classifiers are applied through Poupart et al. to categorize elephant flows. Finally, at a coarser granularity, classifiers run. Their precision must not be similarly compared with the reliability of the model of regression consecutively on the matching characteristic set. The application of features other than traffic volumes for precise traffic flow forecast remains a hard path for investigation [22].

Moreover, the artificial neural networks are constructed with interrelating components called perceptrons, which are simplified digital models of biological neurons. The networks possess two layers of minimum perceptrons, one for the input layer and the other for the output. By Sandwich one or additional "hidden" layers among the input and the output, one can obtain a "deep" neural network; the bigger the quantity of hidden layers, the deeper the network.

Deep nets can be qualified to choose outlines in data, for instance, patterns on behalf of the pictures of cats or dogs. In fact, training includes utilizing an algorithm to iteratively alter the power of the connections among the perceptrons, ensuring that the network absorbs to associate a specified input (the pixels of a picture) by the right label (cat or dog). When qualified, the deep net must preferably be capable of categorizing an input, which has not been obtained previously [23].

In their wide-ranging structure and role, deep nets need insecurely to compete with brains in which the revised strengths of connections among neurons reflect the learned links. Neuroscientists have regularly pointed out significant limits in that comparison: individual neurons might process information more widely than "dumb" perceptrons do, for instance, when deep nets regularly rely on a type of communication among perceptrons called backpropagation that does not happen in nervous systems. However, for computational neuroscientists, deep nets occasionally appeared like the top accessible choice for modeling parts of the brain [24].

2.2. Price of Mistakes and Comprehensive Documentation. ML, for anomaly identification, without achieving momentum in the market, has received considerable attention in the literature. This is largely due to the high FPR, rendering them inoperative in an operating environment. To investigate false alarms, FPRs costly expend the analyst's time and decrease interest and trust in the IDS [22]. The shortage of accurate documentation on the observed irregularities is also another big problem for anomaly detection techniques. Usually, if there is a divergence from the standard, a flag is raised, and an alert is initiated. Not only is a successful IDS responsible for identifying attacks and network intrusions, but also it has a comprehensive record of historical data gathering and model education and training anomalies.

2.3. Matters of Difficulty. They must use less time and computational power while doing traffic prediction, routing, sorting, and congestion management on intermediate nodes in the network to prevent a reduction in the network capacity. In particular, in resource-constrained networks including IoT, as well as WANETs, this criterion is nontrivial. While performance metrics for ML assessment are extremely well, the sophistication of ML-based methods is difficult to determine a priori. In comparison to conventional algorithms, the complexity of ML algorithms often depends on the size of data and consistency, as well as the performance targets [25].

ML has been effectively implemented in diverse fields of networking over the past two decades. With an emphasis on traffic engineering, performance enhancement, and network security, this survey offers a detailed body of information on the applicability of ML strategies in support of network design and maintenance. In discussing issues relevant to the autonomous operation and maintenance of future networks, the present study discusses the representative literature work and explores and examines the viability, as well as the practicality of the proposed ML solutions [26].

It is recommended that future networks would have to accommodate exponential growth in the amount of data and mobile devices to provide excellent knowledge connectivity and sharing capabilities. The sophistication of traffic engineering functions, such as congestion management, traffic prediction, classification, and routing, as well as vulnerability to faults and security threats, will be exacerbated by the unparalleled size and the extent of ambiguity. Although ML-based solutions have already shown impressive outcomes to solve many challenges in traffic engineering, with the anticipated amount of data, several devices, and applications, their scalability needs to be assessed [27]. Current ML-based fault and security prevention techniques, on the other hand, rely more heavily on single-tenant or partner networks. However, the current ML approaches can be expanded or rearchitected to take into consideration the notion of multitenancy in multilayer networks to improve the fault and security management architecture for future networks.

In this survey, along with some other difficulties and prospects, the current study addresses the above-mentioned problems. To advance that the national government eventually achieve the long-term goal of autonomous networking, the results of the present study are expected to inspire the need for further research [28].

2.4. ML-Driven Networking Architectures. Modern networks contain vast quantities of various data types (e.g., logs, network performance metrics, and traffic flow records) [29]. At 100 Gbps, even with high sampling rates, hundreds of millions of flow records each day can be easily produced by a single large network infrastructure feature. Huge data development has recently led to an exponential improvement in computer hardware and information infrastructure, computing, retrieval, and analytics. This is demonstrated by the advent of huge data centers with tens of thousands of servers and EB computing space, the widespread introduction of large-scale software systems such as Hadoop MapReduce and Apache Spark. Besides, the growing numbers of ML and deep learning libraries, such as

Tensor-Flow, Torch, Caffe, Chainer, Nvidia's CUDA, and MXNet, are built on top of such systems [30]. These libraries are largely open-source and capable of scaling out their workloads on advanced hardware-enabled CPU clusters, including GPUs and TPUs. For the next generation SDN [31], GPUs are expected to be a significant enabler. A much-enhanced packet forwarding capability is also documented for Processor SDN routers. Furthermore, the GPUs on virtual machines can be particularly useful for implementing ML and DL algorithms, as well as acting on the gained expertise to learn different networking scenarios [32]. On the other hand, a cloud-edge ML scheme is more likely to benefit from smaller, resource-constrained, smart networked computers. A cloud-edge ML framework will exploit the cloud's massive computing and memory capacity, stable networks, and huge storage capacities to train and exchange computer-intensive models with edge computers. Moreover, the data collection and analytics that need frequent or near-immediate responsiveness will be done by the edge machines. Eventually, the streamlined ML software applications, such as Caffe2Go and TensorFlow Lite, will allow edge devices to circumvent the cloud and locally create leaner versions [33].

The convolutional neural network is the most common DNN. It integrates the transfer learning and data sets and incorporates the two-dimensional convolution layers, eliminating the need to remove the manual element. By extracting the characteristics directly from the images, the coevolutionary neural network functions. During the program's training on the set of the images, the required characteristics are not pretrained but taught in parallel. The DNN is, therefore, more compatible and sufficient for the detailed classification process [30].

NN is among the most efficient and commonly used algorithms in the so-called deep learning subfield of machine learning. Additionally, the neural networks can look like a black box at first glance; an input layer brings the data into the "hidden layers," and the information given by the output layer can been seen after a magic trick. However, knowing what the secret layers are doing is the main step in integrating and maximizing the neural network [34].

In a nutshell, a NN is characterized as a computer system composed of several basic but highly interrelated nodes or elements, named 'neurons,' which are arranged into layers that use dynamic state responses to external inputs in order to process information. As can be revealed below, this algorithm is incredibly useful in detecting the patterns that are too hard for the computer to be manually extracted and trained to know [35]. In the form of this design, an input layer, which has one neuron for every feature existing in the input data, is added to the neural network by patterns and transmitted to one or even more hidden units participating in the network. Such units are named 'hidden' only because they do not represent the output or input layer. It is in the hidden layers where all computation ultimately takes place through a series of connections defined by biases and prejudices (commonly referred to as W and B): the input is collected, and the neuron calculates a weighted sum that also adds the bias and according to the outcome and a preset activation function (sigmoid, σ is the most popular one, but it is almost no longer used, and there are bets). The neuron, then, transmits the data downstream to several other associated neurons in a system called "forward transfer." The last secret layer is connected to the output layer at the end of this phase, which has 1 neuron with each potential desired output [23]. The fundamental structure of NN is shown in Figure 3.

Wi: Weight of the corresponding relation. Note: When calculating the numbers of layers present within the network, the input layer is not included [33].

2.5. Liquid Model. During LSM, the liquid is created of a big amount of spiking LIF model neurons, situated in a virtual three-dimensional column. The liquid owns two significant roles in the arrangement of time-series data. Primarily, its fading memory is in charge for gathering and fitting the input signal over time [36]. Every one of the neurons in the liquid preserves the particular state, which provides the liquid a durable fading memory. The action within the network besides the real firing of the neurons possible also lasts for a while after the signal is over, which can be observed as an additional type of memory. Then, in the liquid of an LSM, the various input signals are divided, permitting the readout to categorize them. Such a separation is imagined to occur through maximizing the dimensionality of the signal [35].

For instance, if the input signal owns 20 input channels, this is converted into 135 ($3 \times 3 \times 15$) signals, in addition to the conditions of neurons in the liquid. For each pair of input signal and liquid neuron, there exists a specific chance of being linked, for instance, 30%. The links among the neurons are assigned in a stochastic method. All neurons in a liquid will completely link to the readout roles [35].

2.6. Contribution and Motivation. We aim to suggest an efficient algorithm that could help in selecting the shortest path to improve the existing methods using weights derived from packets ID and to change Neural network iteration simultaneously. In this study, extracted new features will be approved with the weight of each feature, so that the system can be trained on labeled data in advance with many iteration of neural network.

The proposed contribution depends on the use of significant factors that have effect on routing traffic and learning the network during train bifurcated weights. In conclusion, this study focuses on two main things, namely, the network topology and the way of teaching machine by neural network. Six issues are used in training as follows:

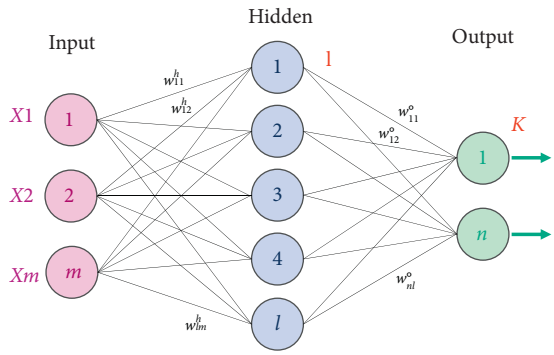

Figure 3: A standard-layer Neural Network's fundamental structure.

$$\begin{aligned} W1 &= \text{sort path},\\ W2 &= \text{load of traffic},\\ W3 &= \text{acknowledge feedback},\\ W4 &= \text{no. of nodes in each path},\\ W5 &= \text{buffering packets in each node},\\ W6 &= \text{delay packet header processing}. \end{aligned} \tag{1}$$

3. Methods

This section will present the proposed method in this study, which is inspired essentially by the model pruning [37]. Therefore, the proposed method aims to improve the structure of both LAN topology and many of the hidden layers inside the neural network system. The output channels decide the best path, which is considered as the input data to the hidden layer's group, as well as the set of information gathered from input layer as packet ID:

$$W_n = \{W_1, W_2, \ldots, W_6\}. \tag{2}$$

These sets of weight derived from input layer consist of all the information needed and already stored in packet ID. The control gate is learned to get optimal routing decision for the LAN by using the output result that feeds it back from the learning channel. Moreover, the learning strategy takes the weight factor from each weight and compares it with other factors for each traffic step, which means a set of input factors:

$$\text{vector}_{\text{in}} = [\,\text{data}_1, \text{data}_2, \ldots, \text{data}_n], \tag{3}$$

It is used only for the first time, then proceeds with neural network NN, and is also updated interactively with layer beyond. However, the final prediction remains unchanged until influenced by the effective weight taken from input channel or output feedback information. Each sample is considered as a small network needed for subdecision for learning one stage.

3.1. Control Gate. This section will identify node on sub-LAN of one iteration. The routing path that is distilled from subnetwork may degenerate severely; for this, it has to present the control gate of the value λ; this associates with every output layer in the channel. During the process, the control gate will change, and thus, it needs to keep it for comparison during learning. Moreover, λ_K can be applied for multiple process or can be called the k-th layer. The results obtained from this control gate information value are used to fix the nodes in each layer; so each layer produces routing node, which is linked to the routing paths. However, the critical data to optimize the path can be identified as $\wedge = \{\lambda_1, \lambda_2, \ldots, \lambda_K\}$ for K layers on control gate.

For the critical data of routing paths can consider that the λ's has to satisfy two conditions: first, when λ 's is not negative, and λ must amplify or suppress output channels. Second, when the case with negative value on λ means the output activations with producing subnetwork, and drastically, the distribution changes with unexpected influence through interpretation of such model. λ should be close to zero, such as the existing models [38] and expected results.

3.2. Subnetwork-Guided Routing. The efficiency of finding control gate is used in the pertained subnetwork sign as $f_\theta(.)$ for input packets ID, and it then develops the Subnetwork Guided Routing (SGR) method that inspired knowledge of the existing method [31] for the adaptive knowledge of such models to subnetwork of routing path. In this regard, the optimization objective related to control gates ($\wedge$) can be defined as

$$\min_{\Lambda} \gamma\left(f_\theta(x),\ f_\theta(x;\Lambda)\right) + \omega \sum_k \left|\lambda_k\right|_1. \tag{4}$$

Denote $\lambda_k \geq 0$, $k = 1, 2, \ldots, K$. γ is the loss of prediction probability, and $f_\theta(x) = [p_1, p_2, \ldots, p_m]$ are the previous probability, while the new probability becomes $f_\theta(x;\Lambda) = [q_1, q_2, \ldots, q_m]$, since$\gamma = \sum_i^m -p_i \log q_i$ where m considers category number, and ω is the balance parameter.

Because the prediction can start from any stage and can process any data, there is no need for the ground truth to deal with it, because it is the training of gathering information. By this stage, the path for subnetwork and gathering information can be predicted again to feed the system taking into account the new process.

3.3. Representation of Paths Routing. Control gates optimization is considered as

$\Lambda^* = \{\lambda_1^*, \lambda_2^*, \ldots, \lambda_K^*\}$, and then to obtain the critical path selection, the threshold (ν) can be used to determine the nodes that can be generated accordingly and controlled by binary mask. The coefficient here is represented as weights, and weights actually play an important role. The representation changes during the training system, and weights, when changed, may be fed to the neural system again, and it subsequently may be suitable with other weights to specify the appropriate path. Changing weights are still in process until they obtain the final result.

It is perhaps crucial to clarify the various types of neurons that can be used in the proposed network in order to do that.

Perceptron is the first type of neuron that will be described. While their use today has decayed, learning how they act can give us a clear sense of how more advanced neurons function [39].

By converting a set of binary parameters to a one binary output, a percentage performs a function to train a classifier model and can also be used during the supervised learning [40]. The perceptron consists of the following steps in this context:

(1) Calculating all inputs through the weights w, matrix multiplication that shows that it is important for the output to be related inputs.

(2) Combining them as a sum weight: $WJ{\cdot}Xj$, $WJ{\cdot}xj$,

(3) In other words, implementing the activation function to decide when sum weighted is bigger than the limit value, whether the limit is equal to a bias, and refer 1 or below and refer 0 as the actual output [41].

In the following terms, it can also perform the formula of function:

$$t(x) = \begin{cases} 0, & \text{if } W \cdot X + P < 0, \\ 1, & \text{if } W \cdot X + P \geq 0, \end{cases} \tag{5}$$

where b is considered prejudice, which is equal to the threshold, and $W \cdot X$ is considered as w's dot product, a part vector of the scales, and x has become an input image.

The hidden layer plays the main role in the neural network system due to the fact that the variable weights take an effect in this layer, and it is controlled when training is finished until the desired results are obtained. The shortest path is calculated by the delay time taken from packet leaving the source till reaching the destination as shown in Figure 4.

Moreover, the delay time can be calculated as follows:

$$T^* = \sum_{i=1}^{L} D_t(f_t) = \sum_{i=1}^{L} \frac{f_t}{C_t - f_t}, \tag{6}$$

where f_t is the average flow on link l in information node/s and C_t is the capacity of link or path l in information node/s. Then, the routing problem can be summarized in the formula

$$\text{NC} = \{1, 2, \ldots, K\} K \text{ of } (\text{S} - \text{D}). \tag{7}$$

S-D is the pair inside the network and corresponding to the K can obtain the set of paths given as

$$\begin{aligned} P_1 &= 1, 2, \ldots, n_1; \\ P_2 &= n_1 + 1, n_1 + 2, \ldots; \\ P_K &= n_{K-1} + 1, \ldots, n_K, \end{aligned} \tag{8}$$

where P_i is considered as the set path directed to route i, and n_K is the total path number of K nodes.

Then, the function of traffic flow can be assumed as $f(n)$ (node/s) with path n, and the vector of paths can be obtained as input to the neural process as

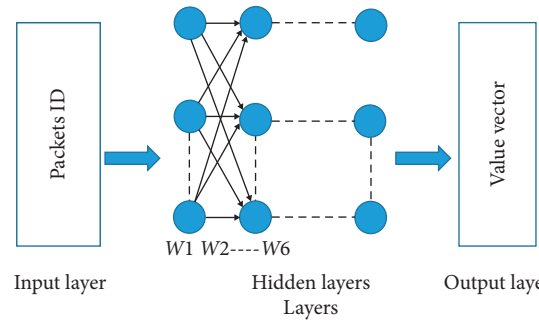

Figure 4: Packets strategy within neural network.

$$f = [f(1), f(2), \ldots, f(n_K)]^T, \tag{9}$$

and the external traffic can be $\gamma(i) = (\text{node}/s)$, where $i \in\in NC$.

Then, the link flow can be depicted as follows:

$$f_l = \sum_{n=1}^{n_K} f(n)\delta(n, l), \tag{10}$$

where

$$\delta(n, l) \begin{cases} 1, & \text{if path } n \text{ contain link } l, \\ 0, & \text{otherwise.} \end{cases} \tag{11}$$

For the whole traffic, it can be illustrated as

$$F(f) = \sum_{i=1}^{L} D_i(f_i) = \sum_{i=1}^{L} \frac{\sum_{n=1}^{nk} f(n)\delta(n, l)}{C_l - \sum_{n=1}^{nk} f(n)\delta(n, l)}. \tag{12}$$

So the path vector f is determined by

$$\gamma(i) = \sum_{n \in P_i} f(n), \quad \forall_i \in \text{NC}, f(n) \geq 0, \quad \forall_n \in P_1 \bigcup P_2 \bigcup \ldots P_K. \tag{13}$$

This can be considered as the link capacity such as holding track flow within certain time.

With the proposed method, the Neural Network (NN) can handle the topology of LAN network and any change that occurs. During training, the failure happened many times until obtaining the desired path that achieves the best rout. This failure leads to learning the system and set it as 0 path to change direction as shown in Figure 5.

Changing the topology of the network works simultaneously with changing function vector value f, and, therefore, the strategy is taken by each node (in case of the link failure) as follows:

(i) Step one: certain node tells the hidden vector about neighboring nodes

(ii) Step two: hidden vector is updated by aggregates node in LAN

(iii) Step three: repeat the process again from step one till the hidden vector cannot change (predefined)

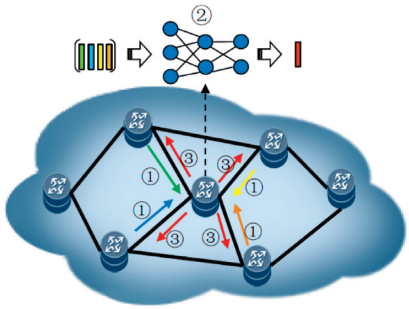

Figure 5: Using NN to control LAN paths.

The training system includes running the program with known results in advance to make the system save the experience as shown in Figure 6.

During training, the network packets ID always change and are automatically updated in the hidden vector shown in Figure 7. This process allows to change the topology of the network that will decide routing of the packets and which path is the best at this time. It is not necessary that one path has less nodes and that short nodes are the best, because they may have jam traffic under certain demand and do not become worthy.

Many applications require the use of machine learning in local networks, as is the case in electrical power distribution stations and smart home applications. The speed of response and reducing the time of transferring data are important things that must be bypassed in modern systems. So, the proposed method is helpful in this regard.

4. Results

Figure 1 displays the theoretical adaptive beamforming (NAB) model of the neural network. For each time frame k, a tiny window of M waveform samples is taken from the C channel inputs for every channel c, denoted as

$$x1(k)[t], x2(k)[t], \ldots, xC(k), \quad [t]\text{for}\{1, \ldots, M\}. \tag{14}$$

A filter-and-sum beamformer with a finite impulse response (FIR) can be described as

$$\begin{aligned} y[t] &= CX - 1, \quad c = 0, \\ &NX - 1, \quad n = 0, \\ &hc[n]xc[t - n - c], \end{aligned} \tag{15}$$

where $hc[n]$ is the n-th tap of the microphones c-associated filter, $xc[t]$ is the signal received by the microphone at time t, xc is the guiding delay caused by the microphone receiving a signal to match it with the other channels of the series, and $y[t]$ is the transfer function. N is the filter's duration. Equation (3) optimization infrastructure needs an estimation of the steering delay $\bar{c}$, typically achieved from a different localization model. By optimizing signal-level targets, the filter coefficients are also collected. In the NAB model, by explicitly minimizing a cross-entropy or sequence loss function, the filter coefficients can be approximated along with the AM parameters.

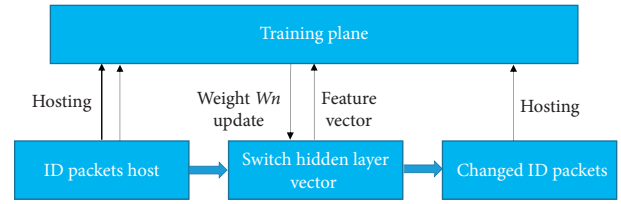

Figure 6: The system within the proposed method.

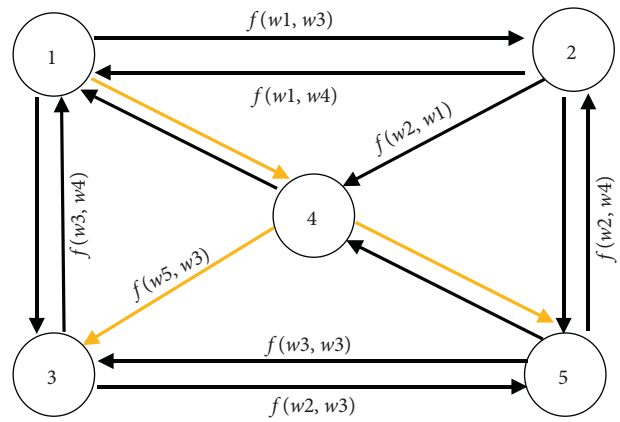

Figure 7: Different paths according to different weights in the neural system.

$$\begin{aligned} y(k)[t] &= CX - 1, \quad c = 0, \\ &NX - 1, \quad n = 0, \\ &hc(k)[n]xc(k)[t - n]. \end{aligned} \tag{16}$$

Traffic is the big issue in the results of the study, and the delay time controls this traffic. The condition of using one path related to minimum delay can be obtained from the training system. The neural network system must be able to adapt any topology of the network, because topology could change during the period of running the system, and paths can be expected according to the variable or weights derived from the neural results as illustrated in Figure 7.

Figure 7 shows the changing of topology and paths during learning the system, as well as correspondence with changing the weight. When the sender is node no. 1, and destination is node no. 5, then there are many paths that can take an effect: from node no. 1 to node no. 2; no. 5 or node no. 1 to no. 4; no. 5 or node no. 1 to no. 3, and then no. 5 . . ., etc.; the path from no. 1 to no. 4 and no. 5 is the best, but actually, in practice, it is totally different. The selection routing has a complex procedure to find the best, as mentioned before based on the weight factors. From this point, it can be said that the efficiency of using machine learning is in solving problems of this kind; the system can be trained using five nodes, and after some iteration, good results can be obtained due to many possible choices allowed as shown in Table 1.

Table 1 illustrates how the system trains network of five nodes and the probability of solution. Normal paths can be calculated easily, but when running, the network with the

Table 1: The resulting NN system within different criteria.

Topology	Path	Node seq.	Weight seq.	Time delay (%)
1	1	1, 4, 5	1, 3, 2, 4, 5	2.6
	2	1, 2, 5	2, 1, 3, 4, 5	2.7
	3	1, 3, 5	2, 1, 4, 3, 1	3.2
	4	1, 4, 3	2, 3, 1, 4, 2	3.5
	5	1, 4, 2	3, 2, 3, 4	3.6
	6	1, 3, 2	3, 1, 4, 5	3.5
	7	1, 2, 4, 5	2, 4, 3, 5, 1	3.1
	8	1, 2, 4, 3, 5	2, 3, 1, 4, 3	2.4
	9	1, 4, 3, 5	2, 4, 3, 2, 3	2.9
	10	2, 1, 3	1, 3, 2, 3, 1	3.1
	11	2, 4, 3	4, 2, 3, 1, 1	3.4
	12	2, 5, 3	1, 2, 1, 4	2.4
	13	2, 1, 4, 3	2, 1, 4, 3, 5	2.8
	14	2, 5, 4, 1	1, 3, 2, 5, 3	3.1
2	15	2, 5, 3, 4	2, 4, 1, 3, 2	2.8
	15	2, 1, 4, 3, 5	2, 1, 4, 3, 1	2.4
	17	2, 5, 3, 1	2, 3, 1, 4, 5	2.4
	18	2, 5, 4, 3, 1	2, 1, 4, 5, 2	2.4
	19	2, 4, 1, 3, 5	2, 3, 4, 5, 1	2.5
	20	2, 1, 4, 3, 5	4, 1, 5, 3, 5	2.4
3	21	3, 1, 2	4, 2, 3, 2, 2	2.8
	22	3, 4, 2	5, 3, 4, 5, 2	2.6

congestion of packets has to control them and get the best rout and time delay. The above table shows many iterations with neural network and using weights, which are $W = \{W_1, W_2, \ldots, W_6\}$. We notice that the rows labeled with yellow color obtain the best time delay that makes the neural system work in the best performance with the best path, and it is not necessary that there should be less number of nodes in this path.

5. Conclusion

As a conclusion, it can be said that the neural networks and corresponding LAN topologies are changed based on the proposed weights derived from packets ID. Training neural network system on LAN also gives the facility of changing the rout between sender and receiver to find the best path of achieving short time. Besides, exploiting the power of computer and software is represented by deep learning algorithms such as neural network, which gives the system preference to decide the shortest path during communication. Neural network is adopted as an algorithm due to its flexibility and ability to change the strategy according to the real-time input of the system. This study revealed that the best path does not necessarily gain short distance or a less number of nodes located. However, the best one is achieved when avoiding the congestion traffic with minimum time delay.

Data Availability

All data, models, and code generated or used during the study are included within this article.

Conflicts of Interest

The authors confirm that there are no conflicts of interest regarding the publication of this study.

Acknowledgments

This research was supported by the Fundamental Research Funds for the Central Universities (CUC2019B072).

References

[1] P. Sun, Y. Hu, J. Lan, L. Tian, and M. Chen, "TIDE: time-relevant deep reinforcement learning for routing optimization," *Future Generation Computer Systems*, vol. 99, pp. 401–409, 2019.

[2] M. Shafiq, Z. Tian, A. K. Bashir, A. Jolfaei, and X. Yu, "Data mining and machine learning methods for sustainable smart cities traffic classification: a survey," *Sustainable Cities and Society*, vol. 60, Article ID 102177, 2020.

[3] D. Praveen Kumar, T. Amgoth, and C. S. R. Annavarapu, "Machine learning algorithms for wireless sensor networks: a survey," *Information Fusion*, vol. 49, pp. 1–25, 2019.

[4] R. M. Yas, "Permuting convergence overcoming of genetic algorithm using arnold cat," *Map International Journal of Science and Research*, vol. 6, no. 5, 2017.

[5] E. Kurtuluş, A. R. Yıldız, S. M. Sait, and S. Bureerat, "A novel hybrid Harris hawks-simulated annealing algorithm and RBF-based metamodel for design optimization of highway guardrails," *Materials Testing*, vol. 62, no. 3, pp. 251–260, 2020.

[6] R. M. Yas and H. H.. Soukaena, "Unequal clustering and scheduling in wireless sensor network using advance genetic algorithm," *Journal of Physics: Conference Series*, vol. 1530, no. 1, 2020.

[7] B. Zhou, Q. Song, Z. Zhao, and T. Liu, "A reinforcement learning scheme for the equilibrium of the in-vehicle route choice problem based on congestion game," *Applied Mathematics and Computation*, vol. 371, Article ID 124895, 2020.

[8] M. Michalis, L. Changhe, and Y. Shengxiang, "A survey of swarm intelligence for dynamic optimization: Algorithms and applications," *Swarm and Evolutionary Computation*, vol. 33, pp. 1–17, 2017.

[9] M. B. Imad and M. Y. Ruwaida, "Design and implementing a software tool to ensure undeadlock state by perfect distribution of resources' instances among competing processes," *Journal of Advanced Computer Science & Technology*, vol. 4, no. 2, pp. 237–247, 2015.

[10] G. Turabee, M. R. Khowja, P. Giangrande et al., "The role of neural networks in predicting the thermal life of electrical machines," *IEEE Access*, vol. 8, pp. 40283–40297, 2020.

[11] T. S. Ahmed, F. Nidaa, and G. H. Amaal, "Using daub achy wavelet for shot boundary detection," *IOSR Journal of Computer Engineering (IOSR-JCE)*, vol. 16, no. 6, pp. 66–70, 2014.

[12] H. Bhaumik, M. Chakraborty, S. Bhattacharyya, and S. Chakraborty, "Detection of gradual transition in videos," *Intelligent Analysis of Multimedia Information*, IGI Global, Hershey, PA, USA, pp. 282–318, 2017.

[13] N. Sabor and M. Abo-Zahhad, "A comprehensive survey of intelligent-based hierarchical routing protocols for wireless

sensor networks," *Nature Inspired Computing for Wireless Sensor Networks*, pp. 197–257, 2020.

[14] M. Javier, M. Ignacio, J. D Ramon et al., "Artificial intelligence (AI) methods in optical networks," *A Comprehensive Survey Optical Switching and Networking*, vol. 28, pp. 43–57, 2018.

[15] Y. Liu, J. Zhang, W. Li, Q. Wu, and P. Li, "Load balancing oriented predictive routing algorithm for data center networks," *Future Internet*, vol. 13, no. 2, p. 54, 2021.

[16] A. Kumar and N. Sachdeva, "Multimodal cyberbullying detection using capsule network with dynamic routing and deep convolutional neural network," *Multimedia Systems*, pp. 1–10, 2021.

[17] R. M. Yas and H. H.. Soukaena, "A survey on enhancing wire/wireless routing protocol using machine learning algorithms," *IOP Conference Series: Materials Science and Engineering*, vol. 870, no. 1, 2020.

[18] Q. Guo, Z. Zhang, and Y. Xu, "Path-planning of automated guided vehicle based on improved Dijkstra algorithm," in *Proceedings of the IEEE 2017 29th Chinese Control and Decision Conference (CCDC)*, pp. 7138–7143, Chongqing, China, May 2017.

[19] Z. Yang, D. H. Lee, and K. Hyun, "Dijkstra's-DBSCAN: fast, accurate, and routable density based clustering of traffic incidents on large road network transportation research record: national academy of sciences," *Transportation Research Board*, vol. 2672, no. 45, 2018.

[20] H. Nazmul, H. P. Akter, A. Tarek, and EmranulHaque, "Network route minimization using time based interface control," *International Journal on Recent and Innovation Trends in Computing and Communication*, vol. 6, no. 2, pp. 166–176, 2018.

[21] J. Xie, F. R. Yu, T. Huang et al., "A survey of machine learning techniques applied to software defined networking (SDN): research issues and challenges," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 1, pp. 393–430, 2019.

[22] F. Liang, C. Shen, and F. Wu, "An iterative BP-CNN architecture for channel decoding," *IEEE Journal of Selected Topics in Signal Processing*, vol. 12, no. 1, pp. 144–159, 2018.

[23] G.-E. Zaharia, T.-A.-I. Şoşea, R.-I. Ciobanu, and C. Dobre, "Machine Learning-Based traffic offloading in fog networks," *Simulation Modelling Practice and Theory*, vol. 101, p. 102045, 2020.

[24] C. Yu, J. Lan, Z. Guo, and Y. Hu, "DROM: optimizing the routing in software-defined networks with deep reinforcement learning," *IEEE Access*, vol. 6, pp. 64533–64539, 2018.

[25] C. H. Martin and M. W. Mahoney, "Predicting trends in the quality of state-of-the-art neural networks without access to training or testing data," 2020, https://arxiv.org/abs/2002.06716.

[26] P. V. Klaine, M. A. Imran, O. Onireti, and R. D. Souza, "A survey of machine learning techniques applied to self-organizing cellular networks," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, pp. 2392–2431, 2017.

[27] V. Mnih, K. Kavukcuoglu, D. Silver et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, p. 529, 2015.

[28] C. H. Martin and M. W. Mahoney, "Statistical mechanics methods for discovering knowledge from modern production quality neural networks," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 3239-3240, Anchorage, AK, USA, August 2019.

[29] M. Mahoney and C. Martin, "Traditional and heavy tailed self-regularization in neural network models," in *Proceedings of the International Conference on Machine Learning*, pp. 4284–4293, Long Beach, CA, USA, June 2019.

[30] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," *ICISSP*, pp. 108–116, 2018.

[31] Y. Sun, M. Peng, Y. Zhou, Y. Huang, and S. Mao, "Application of machine learning in wireless networks: key techniques and open issues," *IEEE Communications Surveys Tutorials*, vol. 21, no. 4, 2019.

[32] Y. Yamamoto, T. Leleu, S. Ganguli, and H. Mabuchi, "Coherent ising machines—quantum optics and neural network perspectives," 2020, https://arxiv.org/abs/2006.05649.

[33] S. Bubeck, "Convex optimization: algorithms and complexity," *Now Foundations and Trends*, 2015.

[34] A. Saxe, S. Nelli, and C. Summerfield, "If deep learning is the answer, what is the question?" *Nature Reviews Neuroscience*, pp. 1–13, 2020.

[35] R. Doshi, N. Apthorpe, and N. Feamster, "Machine learning DDoS detection for the consumer internet of things devices," in *Proceedings of the IEEE Security and Privacy Workshops (SPW)*, pp. 29–35, San Francisco, CA, USA, May 2018.

[36] A. Mehta, J. Kaur Sandhu, and L. Sapra, "Machine learning in wireless sensor networks: a retrospective," in *Proceedings of the 2020 Sixth International Conference on Parallel, Distributed and Grid Computing (PDGC)*, IEEE, Solan, India, November 2020.

[37] A. Karami and N. Derakhshanfard, "RPRTD: routing protocol based on remaining time to encounter nodes with destination node in delay tolerant network using artificial neural network," *Peer-to-Peer Networking and Applications*, vol. 1–17, 2020.

[38] Di Wu and G. Gary Wang, "Causal artificial neural network and its applications in engineering design," *Engineering Applications of Artificial Intelligence*, vol. 97, no. 2021, p. 104089.

[39] M. Saravanan and P. Ganeshkumar, "Routing using reinforcement learning in vehicular ad hoc networks," *Computational Intelligence*, vol. 36, no. 2, pp. 682–697, 2020.

[40] E. Ba¸stug, M. Bennis, M. Médard, and M. Debbah, "Towards˘ interconnected virtual reality: opportunities, challenges and enablers," *IEEE Communications Magazine*, vol. 55, no. 6, pp. 110–117, June 2017.

[41] T. O'Shea, K. Karra, and T. C. Clancy, "Learning approximate neural estimators for wireless channel state information," in *Proceedings of the IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, Tokyo, Japan, September 2017.