International Journal of Computer Networks and Applications (IJCNA)
DOI: 10.22247/ijcna/2025/42 Volume 12, Issue 5, September – October (2025)

RESEARCH ARTICLE

# Twin Delayed Deep Deterministic Policy Gradient Based Routing (TD3PGR) and Security Mechanism for Software-Defined Network (SDN)

J. Delshi Julie
Department of Computer Science, Bishop Ambrose College (Autonomous), Coimbatore, Tamil Nadu, India.
✉ delshibabu80@gmail.com

R. Beulah
Department of Computer Science, PPG College of Arts and Sciences (Affiliated to Bharathiyar University), Coimbatore, Tamil Nadu, India.
beulahr.cas@ppg.edu.in

Received: 01 June 2025 / Revised: 10 August 2025 / Accepted: 26 August 2025 / Published: 30 October 2025

**Abstract – A centralized network management is facilitated by a network architecture named Software Defined Networking (SDN). The Data Plane (DP) and network Control Planes (CP) are separated by SDN.SDN separates CP and DP security issues, and this architectural breakdown presents both interesting possibilities and difficulties. In this study, a TD3PGR on an SDN is suggested. The TD3PGR agent in the suggested approach determines the ideal set of Link Weights (LW) to equalize the network's packet losses (PL) and End-to-End Delay (E2ED) after learning the relationship between network performance and the traffic load (TL) of network switches. To maximise the predicted cumulative Long-Term (LT) reward, a TD3 agent, an Actor-Critic (RL) Reinforcement Learning (ACRL) agent, looks for an optimal policy. The data network of the DP is secured by introducing partially (HE) Homomorphic Encryption (PHE). Complex mathematical operations can be carried out on encrypted data due to this Encryption (E), which keeps the data secure. The agent (AG) uses a secure key created by data network E to retrieve the State (S) and Reward (R) values for the Action (A). The S and R values for the A made to the data network are thus obtained by the agent. As the action, TD3PGR learns the weight for every link. According to the simulation results, in several network topologies, the standard Hop-Count Routing (HCR) and a Traffic Demand (TD)-based RL algorithm performs less than the suggested routing technique.**

**Index Terms – Homomorphic Encryption (HE), Security Mechanism, SDN, Routing Optimization (RO), Deep Reinforcement Learning (DRL), Dynamic Network Management.**

## 1. INTRODUCTION

One contemporary network technology that makes it possible to manage heterogeneous networks effectively is SDN. The increasing need to implement heterogeneous applications with Real-Time (RT) communications requirements cannot be entirely addressed by traditional network design [1]. Conventional networks restrict the flexibility of dynamically configuring the network settings by supporting particular policies that were put in place during the device's build time [2]. Figure 1 shows how the CP and DP components are connected by southbound Application Programming Interfaces (APIs), like OpenFlow [3].

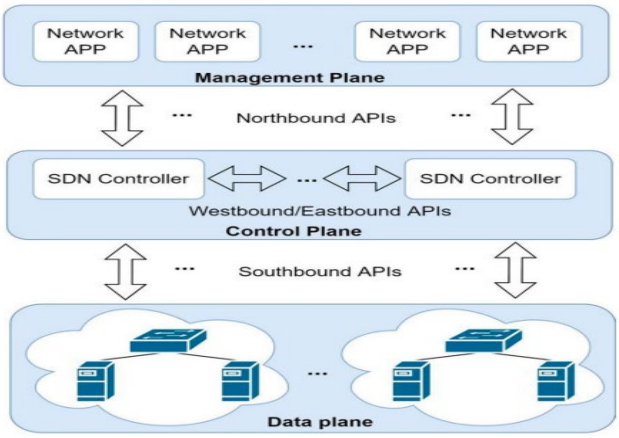



Figure 1 SDN Architecture Diagram [4]

On the other hand, the control or management planes can be used to implement network policies or applications (like firewalls, load balancers, and routers), and the northbound API (like RESTful API [5]) can be used to communicate with the controller. To handle all these apps with a satisfactory Quality of Service (QoS), carrier operators face significant pressure to expand their network Bandwidth (BW). Up until now, over-provisioning network resources has been the standard procedure to guarantee a high QoS. To base capacity

RESEARCH ARTICLE

on estimations of peak traffic load, operators overprovision a network. In the long run, this strategy is not economically justified, despite being simple for networks with predicted peak loads. Open Shortest Path (SP) First (OSPF) [6] is the most widely used Routing Protocol (RP) in a traditional network context. The SP was determined based on the link BW. It might route all network traffic to the same transmission line in a decentralised system. Network congestion (NC) happens when the connection is overloaded. The routing tables are only updated when there is a change in the topology, since OSPF routes packets statically by giving links weights. For route traffic along the SP or distribute it evenly among many transmission lines, Equal-Cost (MP) Multi-Path (ECMP) and OSPF are used [6].

Irrespective of the structure of the network or traffic distribution, these Dynamic Routing (DR) rules provide the ability to route traffic. It is challenging to execute Routing Optimisation (RO) of the whole network's structure and resource management with flexibility, even though this distributed RP offers scalability as it could be employed irrespective of the network scale. The centralized management of packet transmission with a global network are facilitated by the SDN, but it is not simple to create a best RP. Several previous articles define the problem of routing as a limited SP problem [7], nondeterministic Polynomial Time (NP) is typically difficult to determine an optimal solution in nondeterministic Polynomial Time (NP). One common method for resolving the generic multi-commodity flow problem is to study the network topology (NT) as a static framework with changing traffic. Under complex and dynamic traffic, these models are unable to capture ideal network performance [7] adequately.

Even after a new route was trained, NC was still caused by a decentralised system in these studies. Other research has used machine learning (ML) [8-9] to address common routing issues in a traditional network. This research depended on the traditional RP, but SDN and ML [10] are both useful for resolving routing issues. The LSTM [11] and Graph (NN) Neural Network (GNN) [12] are Deep learning (DL) techniques. These methods are effective in prediction of network states and network feature extraction, and it was highlighted by outcomes. Through accurate inference based on input data, DL algorithm models can be trained without the need for intricate assumptions or network environment modelling, and they can produce optimal routing decisions. This means that compared to traditional RP, data-driven DL RP is more suited to various network applications and RO objectives [13].

### 1.1. Problem Statement

Two major obstacles, route optimization and network security, remain despite SDN progress. Inefficient data flows, higher latency, and less-than-ideal network performance are common results when traditional routing methods are unable to adjust to SDN's dynamic nature. When it comes to network security, SDN's adaptability and programmability open the door to several vulnerabilities that may let in attacks like denial-of-service (DoS), routing manipulation, and illegal access. Existing models seldom handle the security issues that come with SDN, even if Deep RL (DRL) has been successful in improving routing choices. Smart routing optimization and real-time security methods are not yet available as integrated solutions, which makes SDN networks inefficient and susceptible to attacks. To address this shortcoming, this study proposes a DRL-TD3PGR model that enhances the efficiency and resilience of SDN systems by routing optimization and the integration of strong security mechanisms.

However, current DL-based intelligent RPs are expensive to deploy and cannot ensure security and resilience in dynamic and complicated network contexts. Thus, the issue about the traditional RP remains unresolved. A Q-Learning (QL) algorithm has been employed in certain research to determine the environment's largest cumulative R, and RL has been utilised for learning novel routes. RL is capable of fully compensating for the drawbacks of traditional routing, as demonstrated in the studies. Risking network performance deterioration for training purposes lowers system reliability since improper routing policies directly raise E2ED and packet loss in a network.

This work presents an optimisation method for SDN called TD3PGR. To lessen the system's E2ED and packet losses, and to identify the ideal combination of LW, a DRL agent is trained. The data network of the DP is secured by introducing partial HE. The suggested RP executes well than both a TD-based RL method and standard HCR in several network topologies, according to an evaluation of its routing performance.

The rest of paper is arranged in this way: Section 2 discusses the literature review, Section 3 proposes the DRL-TD3PGR, the results and discussion are presented in Section 4, and the research is summarized in Section 5.

## 2. LITERATURE REVIEW

A RL-based routing technique for SDN named Q-learning widest-path (WP) Routing Algorithm (Q-WPRA) was suggested by Ke et al. [14]. The broadest transmission path with the highest BW among the source and destination via analysing R function was identified by this algorithm, as it uses RL.

When comparing the Q-WPRA with Dijkstra's algorithm and Dijkstra's WP technique, the Q-WPRA performs well than the conventional methods in detecting broadest transmission path in an SDN environment with variable BW, loss rates (LR), and background traffic, and it was demonstrated by the outcomes.

RESEARCH ARTICLE

ScaleDeep is a scalable DRL-based RP for SDN that enhances routing performance, as suggested by Sun et al. [15]. ScaleDeep exploits DRL and partial control over network nodes. Choose a group of important nodes to serve as driver nodes (DN) in a network. Control theory states that these DN are capable of simulating the entire operation of the network. To alter the routing paths and enhance routing efficiency, the traffic variation on the DN is based on the DRL, as it dynamically adjusts a few LW for a weighted SP algorithm. Various topologies are used for packet-level simulations to verify ScaleDeep's performance.

To address the traffic engineering (TE) task of SDN by throughput (T) and delay (D), Chen et al. [16] created an RL routing method (RLRouting). Experiences are utilized by the RL-Routing, and it will support in resolving this TE issue. It utilizes this way instead of creating an accurate statistical framework. It incorporates extensive network information into its state representation and uses a one-to-many network topology for routing. The upward or downward network performance are optimized by adjusting the R function with T and D of the network. The AG learns a policy that forecasts the underlying network's future behaviour and recommends improved routing routes between switches following the proper training.

To create an effective network control method for TE, Sun et al. [17] suggested combining control theory and DRL technology. A subset of the network's links is chosen and designated as crucial links using ScaleDRL, a concept derived from the pinning control theory. Data regarding traffic distribution is collected by the SDN controller. The data is then employed for optimizing network efficiency via dynamically adjust the set of LW for vital links. This adjustment is attained by DRL method. Then, the application of the weighted SP approach and the dynamic LW facilitates modifying the forwarding paths of the network flows dynamically. In certain network topologies, the ScaleDRL minimized the average E2E transmission delay by up to 39.00%, overtakes the most advanced DRL-based TE method.

In data centre networks based on SDN, a unique RL technique called Deep Q-Learning (DQL) has been developed to automatically create optimal routing paths.This method was suggested by Fu et al. [18]. To satisfy their different needs, DQN are trained for Elephant-Flows (EF) and Mice-Flows (MF) in data centre networks. The low D and low PL rate (PLR) for MF, while for EF, it means achieving high T and low PLR. According to simulation results, the suggested RP can boost the average T of EF while decreasing the average PLR and average delay of MF in contrast to ECMP routing and Selective Randomised Load Balancing (SRL) FlowFit.

To accomplish a universal and adaptable RO, Yu et al. [19] suggested a DRL technique for SDN called the Deep Deterministic Policy Gradient Routing Optimisation technique (DROM). By enhancing network performance, including T and delay, through continual black-box optimisation, DROM makes network operation and maintenance easier. When it comes to boosting network performance, such as lowering latency and increasing throughput, DROM offers superior routing configurations and has strong convergence and effectiveness.

The Double DQN (DDQN) was utilized by Xia et al. [20], and suggest a QoS optimization method for the Software-Defined Factory (SDF) heterogeneous networks. First, the optimisation function for Load Balancing (LB) and network D is established, and a QoS optimisation architecture is suggested for SDF heterogeneous networks. To determine an optimal path of multi-source data flows, the SDN controller utilizes DDQN. After this determination, these optimal paths are uniformly issued by the SDN controller. The results of the experiments demonstrated the good convergence and generalizability of the suggested approach. The evaluation is made with network latency, network jitter, and network T. This suggested method improves network LB and routing efficiency and outperforms the current methods.

RL and SDN Intelligent Routing (RSIR) was suggested by Casas-Velasco et al. [21]. In addition to adding a Knowledge Plane to SDN, RSIR specifies an RL-based RP that considers link-state data while making decisions regarding routing. This approach aims to improve network performance by finding the best paths before forwarding, the background, RL's potential, and SDN's global network view are utilized by this algorithm. Optimal pathways can also be determined by considering BW, D, and loss separately or together. The stretch, link T, PL, and D of RSIR is superior than the Dijkstra's algorithm.

The routing pathways of individual data flows in the S-A space is precisely depicted by the RL method with the application of tabular approach, and it was suggested by Rischke et al. [22]. The first RL SDN RP to provide MPR between a specified source (ingress) switch and destination (egress) switch pair is QR-SDN. Because flow paths are directly represented in the QR-SDN S-A space, flow integrity is maintained by QR-SDN.The same routing path is used by the packets of the specific flow in the QR-SDN, but several routes are used by the several flows in the same source-destination switch pair. An SDN emulation testbed is used to implement QR-SDN. Compared to earlier RP, a single source-destination route was determined so easily that the suggested method provides significantly lower flow latency.

A method that re-integrates several resources of the network with various metrics was put out by Liu et al. [23]. For SDN, a resource-recombined state routing architecture is then suggested. A DRL AG installed on an SDN controller dynamically modifies routing decisions based on RT network

RESEARCH ARTICLE

circumstances. For effective traffic distribution, this adaptive routing optimizes resource allocation. DRL-based Routing (DRL-R) is constructed using DQN and DDPG. In terms of robustness, LB, T, and flow completion time, the DRL-R outperforms Open Shortest Path First (OSPF).

For SDN routing optimisation, the African Vulture RO (AVRO) algorithm was suggested by Chen et al. [24]. The AVOA, a Population-Based Metaheuristic (PBMH) intelligent optimisation algorithm with quick convergence speed advantages and global optimisation capability, is the foundation of AVRO. To improve the procedure's view of the structure of the network, first refine the AVOA algorithm's population initialisation technique following the features of the routing problem in the network. To improve the AVOA algorithm's development and produce consistent convergence effects, an optimisation step is then included. At last, conduct comparative tests with the baseline methods, model the network environment, and specify the network optimisation objective. According to the experimental data, the routing algorithm outperforms DRL techniques by 16.90% and conventional routing schemes by 71.8%, indicating that it has superior network awareness. The use of learning techniques, particularly RL approaches, for enhancing routing performance has been the subject of numerous studies.

Cybersecurity in SDN using Hybrid DL Models and a Binary Narwhal Optimiser (CSSDN-HDLBNO) has been suggested by C. Labesh Kumar et al. [25]. An effective and scalable method to protect the SDN environment from growing cyber risks in DDoS assaults is the CSSDN-HDLBNO strategy. In order to standardise the data and scale the features to a constant interval, the min-max normalisation (MMN) method was initially utilized by the suggested method. Feature selection (FS) based on the binary narwhal optimiser (BNO) is carried out in order to further identify the most relevant attributes. An attention mechanism that combines convolutional NN and bidirectional gated recurrent units (CNN-BiGRU-AM) are utilized by the DDoS attack detection (AD) technique. The efficiency of the suggested approach demonstrated a higher precision of 99.40% compared to current frameworks in several assessment metrics.

For SDN, a model transformation-based security policy automatic management framework was introduced by Yunfei Meng et al. [26]. Finding a connected path for every access control rule in the security policy model (SPM) within the DP, translating this path into a system model (SM) of flow entries (FE), and finally producing useful FE based on this model are just a few of the important problems that the framework's functional modules allow it to handle. To confirm the framework's effectiveness and functionality, the author uses the POX controller and Mininet emulator to install it. The simulation findings show that the model can convert SPM into usable FE, detect changes caused by reducing one linked route or altering SPM simultaneously, and maintain the DP, retaining the runtime security attributes indicated by SPM.

For improving SDN–IoT Security, Hazem (Moh'd Said) Hatamleh et al. [27] introduced the Image-Based Authentication and Artificial Intelligence (AI)-Powered Two-Stage Intrusion Detection (ID). The first step in establishing trust between the shadow blockchain nodes and Internet of Things (IoT) devices is an authentication process that uses images. Then, the effectiveness of the metrics was validated by using the Trading-based Evolutionary Game Theory (TEGT) method to authorize user flows. As a further step, we built the attack graph using an IGNN model, which is based on isomorphisms. Then, based on the evaluated detrimental flows' severity, a local risk assessment was carried out. In addition, the fox optimization technique was used to position the multi-controllers in an optimum manner. The built-in blockchain system safely records the produced worldwide routes. Finally, the Dueling DQN (DDQN) algorithm was used by the 2 AG in the multi-controllers to validate and categorize the incoming suspicious flow packets as regular or malicious. This was done by evaluating the operational metrics. The suggested V-Block architecture's scalability, AD ratio, link failure ratio, malicious traffic rate, and AD ratio were assessed using Network Simulator-3.26, among other metrics.

The Privacy-preserving (PP) ID with Homomorphic Encryption (HE) with Deep NN (DNN) for improving security in SDN was covered by Vankamamidi S. Naresh and Ayyappa [28]. Encrypting the dataset, using the encrypted data for training the DNN-based ID framework, and then deploying the model within the SDN architecture are the steps in the suggested methodology. Discoveries show that DNN is suitable for secure applications, on encrypted data, DNN accomplishes great accuracy (87.11%), which is similar to its efficiency on unsecured data (99.99%). Conventional ML models, on the other hand, perform worse on encrypted data than they do on unencrypted data.

Vaishali A. Shirsath et al. [29] proposed SYNTROPY for TCP SYN DDoS attack detection in SDN, utilizing Rényi entropy. The suggested SYNTROPY architecture makes use of Rényi entropy to standardize the assessment of network traffic uncertainty successfully. Unlike Shannon entropy, Rényi entropy enables more accurate detection by allowing the sensitivity to be tuned to various network conditions and attack patterns.It uses a min-max threshold to detect attack patterns correctly and filters traffic based on whether it is innocuous, flash, or suspicious. Thanks to the Ryu Controller, our framework can be easily integrated with SDN systems. The objective of this simulation is to analyze SYNTROPY on the CAIDA UCSD DDoS 2007 Attack Dataset. When compared to standard systems, SYNTROPY consistently

RESEARCH ARTICLE

outperforms them across a range of measures. Included in this are improvements to recall of 10%, a decrease in false negatives of 34%, an increase in true positives of 13%, an improvement in average detection time of 59%, an improvement in average CPU load of 40%, and an increase in F1-Score of 8%. Table 1 shows a summary of existing methods.

Table 1 Summary of Existing Methods

| Author(s) | Method | Results | Limitations |
|---|---|---|---|
| Ke et al. [14] | Q-WPRA: QL-based routing algorithm for SDN. | Better performance than Dijkstra's procedure and Dijkstra's WP in SDN with variable BW and loss rates. | Does not address security concerns in routing optimization. |
| Sun et al. [15] | ScaleDeep: DRL-based routing algorithm with partial control over network nodes. | Improved routing efficiency through dynamic adjustments on driver nodes. | Limited scalability due to partial control and dependency on specific network topologies. |
| Chen et al. [16] | RL-based RP for traffic engineering in SDN (RL-Routing). | Optimizes throughput and delay using RL. | May require a large amount of experience to converge and adapt in real-time scenarios. |
| Sun et al. [17] | ScaleDRL: Combination of DRL and control theory for SDN routing. | Reduced E2E D by up to 39%, outperforming existing DRL-based traffic engineering methods. | Relies on specific network configurations, making it less adaptable in diverse scenarios. |
| Fu et al. [18] | DQL-based method for optimal routing paths in SDN data centers. | Improves T for EF and reduces PLR and delay for MF compared to ECMP and SRL FlowFit. | Limited to data center networks and may not generalize well to other SDN environments. |
| Yu et al. [19] | DROM: Deep Deterministic Policy Gradient-based method for SDN routing optimization. | Enhanced network performance, reducing latency and increasing throughput. | Limited by convergence speed and potential challenges in scaling with larger networks. |
| Xia et al. [20] | DDQN-based QoS optimization for SDF heterogeneous networks. | Improved load balancing and routing efficiency, outperforming traditional methods. | May not generalize well across all network types, particularly non-homogeneous ones. |
| Casas-Velasco et al. [21] | RSIR: RL-based routing algorithm with a Knowledge Plane for SDN. | performs better than Dijkstra's technique by D, PL, link T, and stretch. | May be complex to implement in large-scale networks with various types of data. |
| Rischke et al. [22] | QR-SDN: Traditional tabular RL for SDN routing. | Provides significantly lower flow latency compared to previous methods. | Limited by scalability for more complex SDN environments with larger routing tables. |
| Liu et al. [23] | DRL-based resource-recombined state routing architecture for SDN. | Outperforms OSPF in robustness, load balancing, and flow completion time. | Computationally expensive due to continuous adaptation and resource management. |
| Chen et al. [24] | AVRO: African Vulture Optimization Algorithm for SDN routing. | Outperforms DRL methods by 16.90% and conventional methods by 71.8% in routing performance. | Limited to specific network routing problems and may not scale well across different SDN environments. |
| Kumar et al. [25] | CSSDN-HDLBNO: Hybrid DL with binary narwhal optimizer for SDN cybersecurity. | Achieved 99.4% accuracy (ACC) in DDoS AD in SDN backgrounds. | Limited to DDoS detection and may not address other types of cybersecurity threats in SDN. |
| Meng et al. [26] | Model transformation-based security policy management for | Successfully converts security policy models into flow entries and | May not handle dynamic security threats in real-time as effectively as |

RESEARCH ARTICLE

| | | | |
|---|---|---|---|
| | SDN. | maintains security attributes. | other solutions. |
| Hatamleh et al. [27] | AI-powered two-stage intrusion detection using Image-Based Authentication for SDN-IoT security. | Enhanced attack detection, scalability, and anomaly detection rates. | Complexity of implementation and integration with existing SDN-IoT systems. |
| Naresh and Ayyappa [28] | PP ID using DNN and HE in SDN. | Achieved high ACC (87.11%) on encrypted data, with negligible performance variances. | Requires substantial encryption and decryption overhead, potentially slowing real-time detection. |
| Shirsath et al. [29] | SYNTROPY: DDoS AD using Rényi entropy in SDN. | Significant improvements in detection accuracy, detection time, and CPU load reduction. | May not detect all attack types equally well and could be limited by specific attack patterns. |

However, there remains an issue where the efficiency of the network data deteriorates because of exploration issues during the learning procedure, and the more intricate the traffic features, the more time-consuming it is for the learning process to converge. Existing methods often fail to handle the dual issues of security vulnerabilities and dynamic network traffic management, even if SDN and routing optimization have made great strides. DRL and TD3PGR are only two of the methods that have shown promise in routing optimization. However, very few of these algorithms have included strong security measures to counteract attacks as they happen. Furthermore, current approaches fail to make full use of DRL's adaptive learning capabilities when it comes to efficient and safe routing in SDN settings. By integrating TD3PGR-based routing with a sophisticated security mechanism, the suggested DRL-TD3PGR architecture guarantees top-notch performance and resistance to network assaults, thereby resolving this gap. This combined method improves QoS and meets the increasing need for intelligent and autonomous network management in SDN systems.

## 3. PROPOSED METHODOLOGY

An SDN is used to suggest TD3PGR. In the recommended approach, the TD3PGR agent determines the ideal set of LW to balance the network's packet losses and E2ED after learning the link among the efficiency of the network and its switches TL. To maximise the projected cumulative LT reward, an ACRL agent, which is a TD3 agent, looks for the optimal policy. The DN of the DP is secured by introducing a partially HE. Using a secure key generated through the data network E, the agent retrieves the S and R values for the A. The overall procedure of the suggested framework is given in Figure 2.

### 3.1. System Model (SM)

Examine a backbone structure that contains switches with SDN capabilities. Many servers or clients are among the traffic sources connected to the edge switches. Each edge switch is taken to be the backbone network's departure or arrival point in this case. Consider that the SDN-based network contains N switches. The notation $V = [v_1, \dots, v_N]^T$ represents each switch. $\mu_n$ represents the $n^{\text{th}}$ switch's service rate. The traffic routing within a single autonomous system is the main topic here.

The expression for a communication network $G$ is $G = (V, E)$. Then, the set of links e between the switches is denoted as $E$.

If $v_i, v_j$, are any 2 switches $V$ $(i \neq j)$, it assumes that there exists at least one path forwarding (PF) $p_{i,j} = \{e_1, e_2, \dots, e_{|p|}\}$ that can send the traffic from $v_i$to $v_j$.

Then, $p_{i,j}^* = \{e_1, e_2, \dots, e_{|p^*|}\}$ denotes the SP among the paths. In this case, the WL value $e_m$is represented by $w_m(0 \leq m \leq |E|)$, and the SP is determined using a weighted SP method on G.

The expression $p_{i,j}^* = \left\{v_1^{i,j}, v_2^{i,j}, \dots., v_{|p|}^{i,j}\right\}$ is another way to describe this path.In this case, the $l^{\text{th}}$ switch, when sent over the shortest path, is $v_l^{i,j}$. $v_i$ and $v_j$are equivalent to $v_1^{i,j}$and $v_{|p^*|}^{i,j}$, correspondingly. Since $M_t$ is the flow count in the network at time *t*, let $f_t^k$ $(0 \leq \leq M_t)$ represent the kth traffic flow at *t*.

In this context, a flow is a pair of a source and a destination. As the flow passes via switches along the route, delays occur in both data transmission (DT) and data processing. Based on the estimated SP, $f_t^k$ is routed through specific switches in the network routing system.

Assume that the Poisson arrivals have exponentially distributed service times and a rate of $\lambda_t^k$. In this case, the data processing and DT delay can be represented as M/M/1/K queues, and every switch has a restricted system dimension. Next, the following Equation (1) can be used to determine the anticipated delay in $v_n$ at *t*:

$$E[d_n(t)] = \frac{E[N_n(t)]}{\lambda_n(t)\left(1-P_b^n(t)\right)} \quad (1)$$

RESEARCH ARTICLE

Here, the total arrival rate (AR) into $v_n$ at $t$ is denoted by $\lambda_n(t)$.Due to a buffer overflow in $v_n$, a packet is lost at $t$, and its probability is denoted as $P_b^n(t)$. The queue occupation represents the switch's packet count at t, $N_n(t)$. In this case, $P_b^n(t)$can be found in Equation (2):

$$P_b^n(t) = \frac{(1-\rho_n(t))(\rho_n(t))^{K_n}}{(1-\rho_n(t))^{K_n+1}} \quad (2)$$

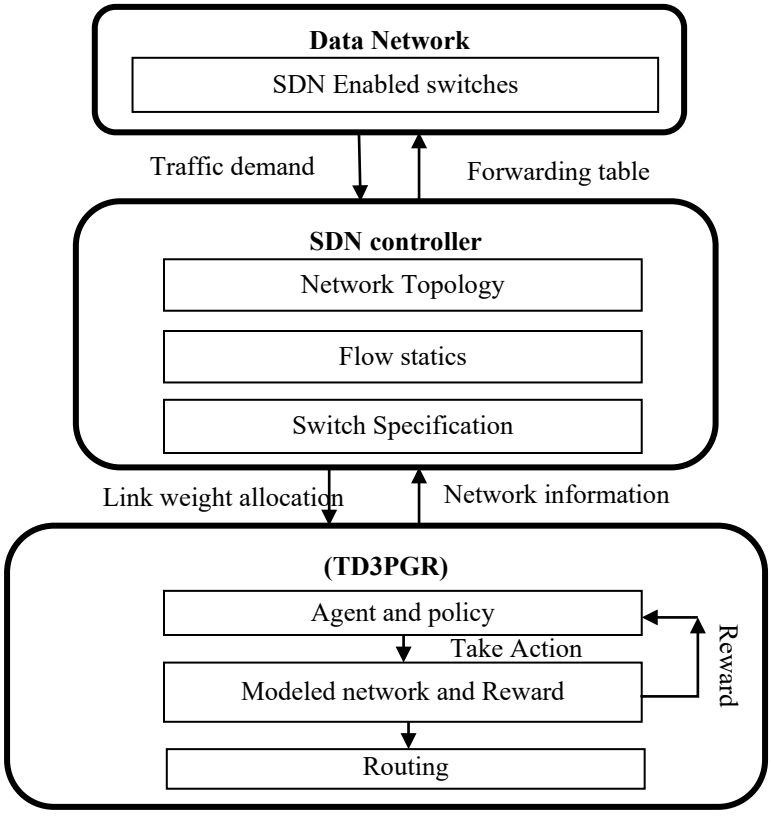



Figure 2 Overall Procedure of the Suggested Framework

The traffic intensity at switch $n$ is represented by $\rho_n(t) = \lambda_n(t)/\mu_n$, while the switch $n$ 's overall system capacity is represented by $K_n$. Additionally, the following Equation (3) is a way to determine the expected queue occupation:

$$E[N_n(t)] = \begin{cases} \frac{\rho_n(t)}{1-\rho_n(t)} - \frac{(K_n+1)(\rho_n(t))^{K_n+1}}{1-(\rho_n(t))^{K_n+1}} & if\ \rho_n(t) < 1 \\ \frac{K_n}{2} & if\ \rho_n(t) = 1 \end{cases} \quad (3)$$

The following Equation (4) is the E2ED of $f_t^k$ sent over $p_{i,j}^*$, with a consideration that the link's propagation delay is extremely small [30].

$$D_{e2e}^k(t) = \sum_{n \in path(p_{i,j}^*)} E[d_n(t)] \quad (4)$$

Here, the collection of switches on the forwarding path from $v_i$to $v_j$ is represented by path $path(p_{i,j}^*)$. It is thus possible to determine the average E2ED of network flows in the following Equation (5):

$$D_{e2e}^{avg}(t) = \frac{1}{M_t}\sum_{k=1}^{M_t} D_{e2e}^k(t) \quad (5)$$

Additionally, the following techniques can be used to ascertain the expected network loss traffic at t as well as the expected loss (EL) traffic of switch n, and are given in Equations (6) and (7)

$$E[L_n(t)] = \lambda_n(t)P_b^n(t) \quad (6)$$

$$E[L_{tot}(t)] = \sum_{k=1}^{N} \lambda_n(t)P_b^n(t) \quad (7)$$

The efficiency of the network is maintained, and the neural model can be trained effectively through the application of a sophisticated network modeling-based technique.

It should be mentioned that this work is easily adaptable to MP Routing (MPR), it is predicated on Single-Path (SP) networking, similar to OSPF. For the same set of LW, the data network (DN) has varying S and R depending on the RA.

But, in the framework of RL, the RP is regarded as a component of the environment, and so that the learning is re-

RESEARCH ARTICLE

performed effectively, the RL agent may react to various backgrounds.

3.2. MDP Framework for LW Allocation

From one switch to the next in line, the Aggregated Traffic Volume Matrix (ATVM) shows the traffic rate.

With $t_{i,j}$ being the quantity of traffic from the i[th] switch to the j[th] switch, let $T = [t_{i,j}]_{N,N}$ represent the ATVM of the DN. The TD of data flows and their network routing patterns determine the network's ATVM for a certain network design. Because it depicts both the network's link utilisation and the connectivity architecture among its switches. The agent can comprehend the interdependency between the links due to this representation of network S.

During the training phase of the suggested modeling-based technique, there is no communication between the DRL agent and the SDN controllers. This suggests that the agent fails to assess the overall traffic volume at switches throughout the training phase. Based on the RP and the LW of the previous A, the DN model and the routing path decision are made. These are used to calculate ATVM.

In the suggested method, each switch in the real network notices the aggregated traffic volume value. This value is the same as the value calculated for the deployed link weight allocation (WA) and TD in the fine modeling-based learning background.

The DRL Agent (AG) can use this consistency to train a neural model that can be applied to the DN and use the clear RP without compromising the efficiency of the DN.

These crucial S-R pairs are also preserved in the off-policy-based DRL agent's (replay buffer -B).

The procedure of experience replay can be utilized.

3.3. S AND A SPACE

The observation for every switch $v_n$ at t can be articulated as $o^n(t)$ using Equation (8).

$$o_t^n = [t, K_n, N_n(t), \lambda_n(t), L_n(t), \rho_n(t), d_n(t)] \quad (8)$$

Here, the time step is denoted as $t$. The switch's index is denoted by n. The flow index is represented by k. The number of switches is N. The flow count is $M_k$. The n[th] switch is $v_n$. The nth switch's service rate is denoted by $\mu_n$. The kth traffic flow at t is denoted by $f_t^k$.

The PL probability in $v_n$ at t is $P_b^n(t)$. The overall system capacity of $v_n$ is denoted by $K_n$. The Queue occupation at t is $N_n(t)$. The total AR into $v_n$ at t is denoted by $\lambda_n(t)$. The EL of traffic $v_n$ at t is denoted by $L_n(t)$. The utilisation of $v_n$ at t is denoted by $\rho_n(t)$, while the expected delay in $v_n$at t is denoted by $d_n(t)$.

These observations are taken from the agent's modelled network in the suggested technique, rather than from measurements on all switches in the DP. The SDN controller reports network information, including switch specifications and network topology, which is used to build the modelled system. The network S at t, represented as $S_t$ the SP is obtained as follows: At each t, the SP of flows in the modelled network is determined based on the LW supplied by the previous A.

$$S_t = \{s_t^{i,j} | i, j \in V\} \quad (9)$$

$$s_t^{i,j} = \min\left(1, \frac{1}{\mu_{max}} \sum_{k=1}^{M_t} \lambda_t^{k(i)} x_{ij}^k(t)\right) \quad (10)$$

In Equations (9) and (10), $\lambda_t^{k(i)} = \prod_{l \in path(src(k) \to i)} \left(1 - P_b^l(t)\right)$.The binary indication that indicates whether the connection from $v_i$ to $v_j$ is part of the path that the kth flow is transmitted via, following the SP, is represented by $\lambda_t^k$ and $x_{ij}^k(t)$. The switches with the maximum service rate are denoted by $\mu_{max}$, and the component range of state is adjusted to [0, 1] using the $min(.)$function. Consider that the component $t_{i,j}$ of ATVM normalised by $\mu_{max}$corresponds to the state $s_t^{i,j}$. In the meantime, the DRL agent decides how to distribute the link weights based on the $s_t$ at each t. The following Equation (11) is the definition of the action space:

$$a_t = a_t^1 \times a_t^2 \times \ldots \times a_t^{|E|} \quad (11)$$

Here, the weight value given to the m[th] link at t is indicated by $a_t^m (0 \leq m \leq |E|)$. Each weight's range is denoted by $a_t^m \in [w_{min}, w_{max}]$. The symbols represent the link weight's minimum and maximum values $w_{min}$ and $w_{max}$. Since the weighted SP algorithm determines packet forwarding, the DRL agent's action reflects the LW, and link WA indicates the routing policy of the network. Figure 3 provides a straightforward illustration of how to use a modeling-based environment to compute the S and the associated delay and loss.

Without impacting the DN. There are 2 TD on the network, which consists of six switches. Following its routing policy, the SDN controller controls PF in the DN and provides the learning agent with data network information. Without changing the DN, the DRL agent trains the Neural Network (NN) for a specific network design. The modelled network is used to compute the DRL data, which includes the network state, average E2ED (A E2ED), and expected total packet loss.

$\mathbb{P}_{sa}: (s, a) \to s_{t+1}$ is a representation of the probability that activity at will cause the network routing system to change from $s_t$ to $s_{t+1}$. The possibility of choosing an action at $a_t$ and $s_t$ given by policy $\pi$ can be used to calculate the transition probability $\mathbb{P}_{sa}$ because the WA to every link can be changed

RESEARCH ARTICLE

dynamically. The suggested routing system aims to accomplish two primary objectives: (2) reducing the rise in PL at switches, and (3) minimizing the AE2ED of flows. The PL reward $r_p(t)$ and delay performance reward $r_d(t)$ should be defined as follows at t in equations (12) and (13):

$$r_d(\mathrm{t}) = 1 - \frac{D_{e2e}^{avg}(t)}{\sum_{n \in path(p_{max})} \frac{K_n}{\mu_n}} \quad (12)$$

$$r_p(\mathrm{t}) = 1 - \frac{L_{tot}(t)}{\sum_{n=1}^{N} \lambda_n(t)} \quad (13)$$

The path with the greatest hop count is presumed to be $p_{max}$. R. represents the reward function The routing algorithm's use of LW for lessening the AE2ED while considering bottleneck packet loss is indicated by the value that R returns. Here, R is defined when an action is performed in the $s_t$.

$$R(s_t, a_t) = \alpha r_d(t) + (1 - \alpha) r_p(t) \quad (14)$$

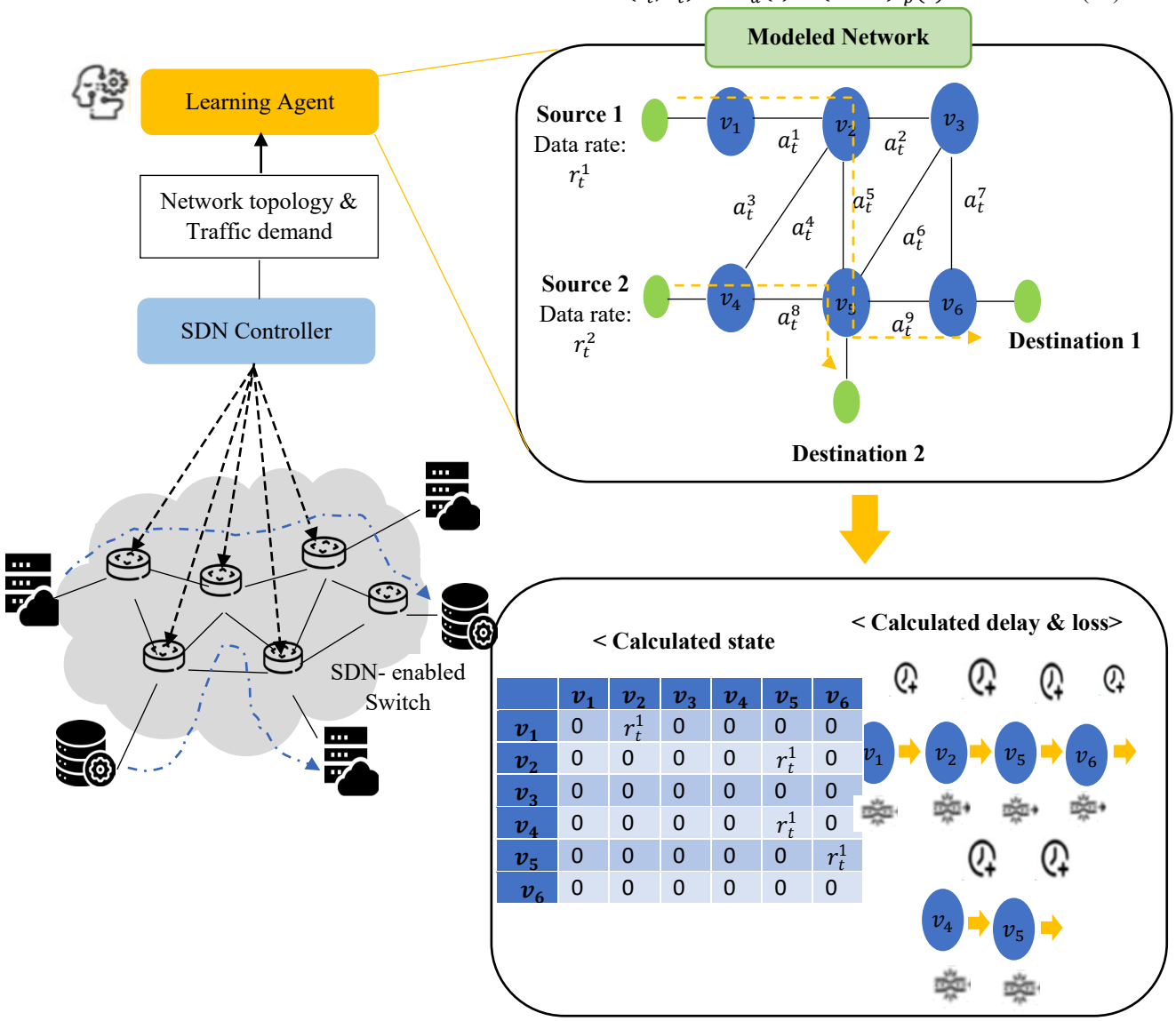



Figure 3 Basic Instance of Modeling-Based Procedure [31]

To balance the weights of network packet loss (PL) and delay, α is utilized as the weight factor and $R(s_t, a_t)$ has a range of [0, 1]. R seeks to minimise a weighted linear mixture of PL and network delay performance metrics, as indicated in Equation (14).

However, R can be set in several ways, such as maximising T or minimising maximum link utilisation, based on the necessities of the network procedure. Define the total predicted discounted reward under policyπ in the following manner to account for the influence of present actions on future R:

RESEARCH ARTICLE

$R_t^{\pi} = R(s_t, a_t) + \sum_{i=1}^{\infty} \gamma^i . R(s_{t+i}, a_{t+i})$ (15)

In Equation (15), from $t + 1$ to the infinite t, the discount factor $\gamma \in [0, 1]$ establishes the significance of future R. Stated otherwise, the total of the discounted R from $t + 1$ to the infinite time step and the present reward at t are represented as $R_t^{\pi}$.

Adaptive Discount Factor ($\gamma$) using Entropy: The core idea is to adjust the discount factor (γ) based on the policy's entropy. If the entropy is high, indicating high uncertainty or instability, the discount factor can be decreased. This encourages the agent to focus on immediate rewards and explore more, potentially finding a better policy. If the entropy is low, indicating stability and good learning, the discount factor can be increased. This allows the agent to consider future rewards and potentially find optimal long-term strategies.

### 3.4. Data Security Using Partially Homomorphic Encryption (PHE)

A PHE method specifically supports additive homomorphic operations. For data E, it generates 2 types of keys: a public key (n, g). Here the modulus is denoted by n. g, the basis for encryption (E). The private decryption (D) key is secret, and it is denoted as (λ, μ). The formulation $n = p \times q, \lambda(n) = lcm\,(p - 1, q - 1)$yielded random integers g. In a modular multiplicative formulation, the parameters include n, μ, and two arbitrary huge prime values, p and q that the method accepts as input for the initial setup.

$\mu = \left(L.\left(g^{\lambda}.modn^2\right)\right)^{-1} modn$ (16)

In Equation (16), the quotient of (x − 1) separated by n is represented by the function L(x). Using the following Equation (17), the E stage creates a ciphertext ($c$):

$c = g^m . r^n (modn^2)$ (17)

Concerning the requirement gcd (r, n) = 1, the value r is chosen at random for every message m, here 0 < r < n and $r \in Z_{n^2}^*$. The message m = Decrypt (c; λ, μ) is the result of D. It suggests utilising a private key (λ, μ) to introduce a c that must be D, $\in Z_{n^2}^*$

The following is how the D procedure is executed:

$m = \frac{L.(c^{\lambda} modn^2)}{L.(g^{\lambda} modn^2)} . modn = L.\left(c^{\lambda} modn^2\right).\mu \times modn$ (18)

The following equations (18), (19), (20), and (21) define the + and multiplication operations when two $c$ are considered using the E technique shown below:

$c_1 = g^{m_1} . r_1^n (modn^2)$ (19)

$c_2 = g^{m_2} . r_2^n (modn^2)$ (20)

$E(m_1).E(m_2) = \left(g^{m_1} . r_1^n (modn^2)\right).\left(g^{m_2} . r_2^n (modn^2)\right) = g^{m_1+m_2} . (r_1 . r_2)^n (modn^2)$ (21)

To preserve information availability, confidentiality, and integrity, it is used to improve data security during the authentication process.

### 3.5. Network Routing Algorithm Using TD3PGR

The suggested method minimises packet loss and network delays for PF with balanced link WA by using a DRL agent for neural model training, to determine an optimal action-selection policy. DQN and Distributed Priority Queue (DPQ) are combined in the RL approach known as DDPG. The critic and the actor are its two constituents. The actor is a function that creates a continuously changing action given the current state as input. A Q-value is the critic network's function. This network generates an estimated Q-value after receiving the current S and A as input.

The critic provides the actor with information about how fantastic the action is and how it should be tweaked, but the actor chooses which action should be performed. A PG technique is used in the construction of actor learning. By calculating the value function and adjusting its parameters using the temporal difference approach, the critics assess the A that the actor produced.

Equation (22), updates the actor using the DPG algorithm.

$\alpha = \mu(s|\theta_{\mu}) + N$ (22)

The symbol for random noise function is denoted by $N$. Recent work formulates exponential smoothing for modernizing the factors of the critic $\theta_Q$ and actor $\theta_{\mu}$ [32]

$\theta_{\mu'} = \tau\theta_{\mu} + (1 - \tau)\theta_{\mu'}(actor)$ (23)

$\theta_{Q'} = \tau\theta_Q + (1 - \tau)\theta_{Q'}(\text{critic})$ (24)

In Equations (23) and (24), τ is the smoothing factor. The critic network is used to generate the Bellman equation and the A value estimation [33] and is given in Equation (25).

$Q'(s, a) = E[r(s, a) + \gamma Q'(s', a')]$ (25)

For updating the critic factors, the loss function (LF) is minimized by TD error and it also uses $y = r + \gamma Q'(s', a')$, as specified in Equation (26) [34] when the $\gamma \ll 1$.

$LF = \frac{1}{M}\sum_{i=1}^{M}\left(y_i - Q(s_i, a_i)\right)^2$ (26)

To maximise the expected discounted reward, the PG represented by Equation (27) is utilised.

$\nabla_{\theta_{\mu}} J \approx \frac{1}{M}\sum_{i=1}^{M}\left[\nabla_a Q(s, a)|_{s=s_i, a=\mu(s_i|\theta_{\mu})} \nabla_a \mu(s|\theta_{\mu})|_{s_i}\right]$ (27)

The DDPG method is enhanced and expanded upon by the TD3 algorithm, which is an online, off-policy, model-free RL method. The DDPG algorithm is similar to how TD3 is

RESEARCH ARTICLE

computed. During training, this overestimation will be detrimental to the policy update. To address these issues, DQL and DDQN techniques have been employed. These methods update the actor's selection and separate Q-values using two networks. Equations (28–29) [35] define double Q-value networks, which the DQL uses to determine the next state number.

$$y_1 = r + \gamma Q_{\theta_1'}\left(s', \mu'\left(s'|\theta_{\mu'}\right)\right) \quad (28)$$

$$y_2 = r + \gamma Q_{\theta_2'}\left(s', \mu'\left(s'|\theta_{\mu'}\right)\right) \quad (29)$$

Equation (30) is the expression for the TD error:

$$y = r + \gamma_{i=1,2}^{min} Q_i'(s', a') \quad (30)$$

Here, the critic index is denoted by $i$. The Q-value must be smoothed to avoid overfitting. Clipped normal distribution noise is employed to do this, and the modified target's outcome is expressed in equations (31) and (32),

$$y = r + \gamma Q\left(s', \mu'\left(s'|\theta_{\mu'} + \varepsilon\right)\right) \quad (31)$$

$$\varepsilon \sim clip(N(0, \sigma), -c, c) \quad (32)$$

To choose the best course of action, the agent gathers information on the state at each t. The RL AG provides input on the network's convergence and yields a negative R. To direct the AG to execute actions that maximise the values, the R is essential. Equation (33) expresses the new R.

$$RF = -|e| - 0.01|u| \quad (33)$$

Here, the RL agent's control action is denoted as $u$. Equation (33) yields the highest reward and the lowest error when the sign is negative. Additionally, the OB and A sizes of the background are obtained, and an environment interface object is produced. The network's latency and packet loss have been decreased by using this R. Moreover, the R can produce quick convergence, excellent performance, and little computation with fewer delays and packet loss as the agent continuously modifies its parameters. TD3 uses an actor representation to decide what to do next.

To create this actor, the first step is to build a DNN that gets the A output and observation (OB) inputs.TD3 uses an actor representation to decide what to do next. Equation (34), which defines a NN with a single Fully Connected (FC) layer including delays and packet loss, is used to represent network routing.

$$u = [\int edt\, e] * \left[K_i K_p\right]^T \quad (34)$$

Here, the NN weights' absolute values are defined by $K_p$ and $K_i$. The long-term R from A and OB are estimated via the TD3 agent estimates using two critic cost function descriptions. To create the critics, a DNN is constructed with two inputs, OB and A, and a single output.

First, randomly generated weights and beginning data are used to initialise the NN model in the DRL AG. The M/M/1/K queues-based network model is constructed utilizing the ND from the SDN controllers. The SDN controller's reported traffic demand and network topology data, along with the S and R data computed in the network system model, are the sources of the first observation. The DRL agent uses randomly generated synthetic traffic to train the NN. Thus, using the previously gathered network characteristics that have been seen in the target DN can greatly enhance learning performance. Algorithm 1 presents the step-by-step process of TD3PGR for network routing.

Input: Empty replay buffer $\mathcal{D}$, initial policy parameters $\theta$, and Q-Function parameters $\theta_1, \theta_2$

Output: Optimal path p

1. Set the target parameters to main parameters $\theta_{targ} \leftarrow \theta$, $\theta_{targ,1} \leftarrow \theta_1, \theta_{targ,2} \leftarrow \theta_2$
2. Repeat
3. Notice state s and select action a
4. Perform a in the environment
5. Notice the next state $s'$, reward r, and path p to indicate whether $s'$ is terminal
6. Store $(s, a, r, s', p)$ in replay buffer $\mathcal{D}$
7. if $s'$is terminal, reset environment state
8. if its time to update then
9. for j in range do
10. randomly sample a batch of transitions $, M - \{(s, a, r, s'p)\}$ from $\mathcal{D}$
11. Calculate target actions $a's' = clip\left(\mu\theta_{targ}(s') + clip(\epsilon, -c, c), a_{low}, a_{high}\right), \epsilon \sim \mathcal{N}(0, \sigma)$
12. Calculate targets $y = r + \gamma Q\left(s', \mu'\left(s'|\theta_{\mu'} + \varepsilon\right)\right)$
13. Update Q fu Equation (27) can be used to update Q functions by one gradient descent step
14. if j mod policy_delay=0 then
15. Update Q functions by one step of gradient descent by $\nabla_\theta \frac{1}{|M|} \sum_{s \in B} Q_{\theta_1}\left(s, \theta_\mu(s)\right)$
16. Update the target networks $\theta_{targ} \leftarrow \rho\theta_{targ} + (1 - \rho)\theta$
17. end if
18. end for
19. end if

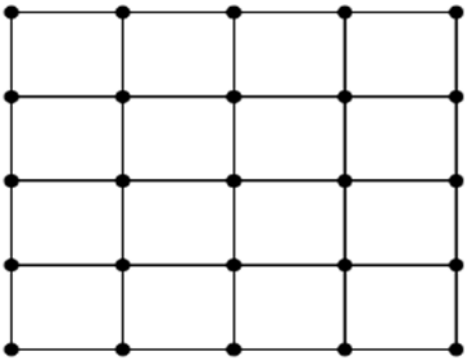
RESEARCH ARTICLE

20. Until convergence and optimal path is determined

Algorithm 1 TD3PGR

## 4. RESULTS AND DISCUSSION

The performance of the suggested RP and current protocols in an SDN-based network is assessed in this section using the simulation outcomes. A stable-baseline model for the DDPG method and MATLABR2021b for network topology generation were used in the simulations. Numerical results are obtained from the runs on a server equipped with an Intel i9-10940X CPU operating at 3.3 GHz, an NVIDIA Quadro RTX6000, 24 GB of RAM, and CUDA 11.1. Examined two types of network topologies: a GEANT topology with 40 nodes and 61 full-duplex (FD) links, as illustrated in Figures 4(a) and 4(b), and a grid topology (GT) with 25 switches set up in a 5x5 grid pattern with 40 FD links.



(a) GT Using 25 (SWT) Switches

(b) Geant Topology With 40 SWT

Figure 4 Performance Evaluation with Network Topologies

For topologies, the service rate and system capacities are set at 3,000 packets per second (8pps) and 10,000 packets per second, respectively. A Poisson process determines the arrival rate of the kth flow entering the structure with average $\lambda^k$. In the range of 10 to 300 packets/second, $\lambda^k$is determined by a uniform random distribution. Source and destination switch pairs were chosen at random, and the $\lambda^k$was altered at each iteration to account for network flow unpredictability. Link weights are chosen between 1 and 5, and the weight factor $\alpha$ was adjusted to 0:9. Dijkstra's method, a popular weighted SP algorithm, was utilised to determine each flow's SP. Table 2 contains a list of the simulation parameters (SP).

Table 2 Simulation Parameters (SP)

| SP | Grid | GEANT |
|---|---|---|
| Switches Count | 25 | 40 |
| Links Count | 40 | 61 |
| Flows Count | 150 | 150 |
| $\alpha$ | 0.9 | 0.9 |
| $w_{min}$ | 1 | 1 |
| $w_{max}$ | 5 | 5 |

The actor network and critic network employ two FC Hidden Layers (HL) of 400 and 300 units, to carry out the DDPG-based algorithm.

NNs are trained using the Adam optimiser (AO). 0:99 was used as the $\boldsymbol{\gamma}$. To gather transitions before learning started, the mini batch's batch size(H) was fixed at 100, and learning started after 100 steps. Learning rates (LR) are set to $10^{-5}$ for the AO, actor, and critic networks. Table 3 describes the DDPG algorithm's hyperparameters (HP) values.

Table 3 HP Values Utilized for the DDPG Procedure

| HP | Value |
|---|---|
| $\gamma$ | 0.99 |
| $B$ | 50,000 |
| H | 100 |
| Critic LR | 0.00001 |
| Actor LR | 0.00001 |

The total of the delays noticed at a series of intermediate nodes on the way to the destination is the E2ED of every packet. A fixed component, such as transmission and propagation delays, and a variable component, like processing and queueing delays at the nodes, makes up each delay. The equation (35) is an expression for the Packet Delivery Ratio (PDR):

$$R_{Success} = \frac{P_{succ}}{P_{all}} \times 100\% \quad (35)$$

Here, the number of packets that the router has successfully dispatched is denoted as $P_{succ}$, $P_{all}$ shows the total number of packets that have passed through the router, and $R_{Success}$ is

RESEARCH ARTICLE

the ratio of successful packets. The following equation (36) is an expression for the packet loss ratio (PLR):

$$R_{drop} = \frac{P_{drop}}{P_{all}} \times 100\% \quad (36)$$

where $R_{drop}$represents the ratio of packet loss, $P_{drop}$denotes the packet count dropped by the router, $P_{all}$this indicates the total packets transmitted through the router.

The amount of data that flows across a network in a specific amount of time is called T. Megabits per second (Mbps) or bits per second (bps) are the common units of measurement.

4.1. Average Network Performance Based on the Number of Iterations in GT

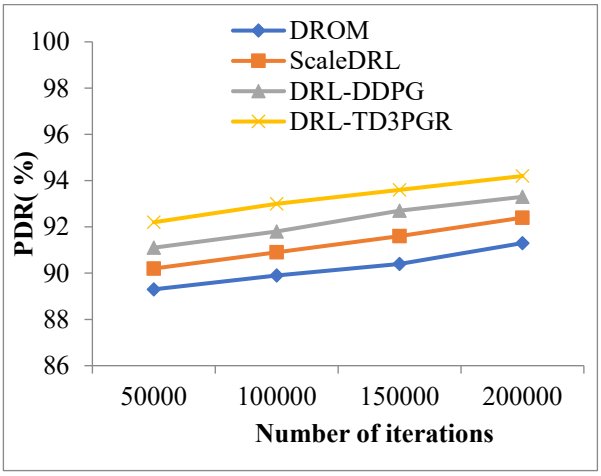



Figure 5 Packet Delivery Ratio (PDR) vs. Routing Protocols

Figure 5 shows that the proposed system has the highest PDR of 94.20%. Other methods, such as DROM, ScaleDRL, and DRL-DDPG, have the lowest PDR of 91.30%, 92.40%, and 93.30% for 200000 iterations.

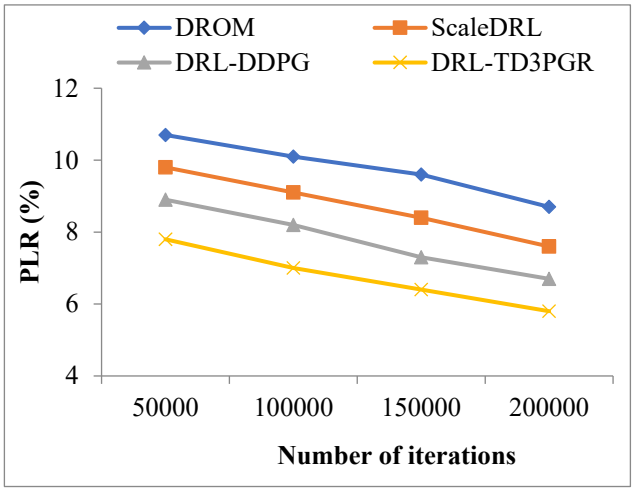



Figure 6 Packet Loss Ratio (PLR) vs. Routing protocols

Figure 6 shows that the proposed system has the lowest PLR of 5.80%. Other methods, such as DROM, ScaleDRL, and DRL-DDPG, have the highest PLR of 8.70%, 7.60%, and 6.70% for 200000 iterations.

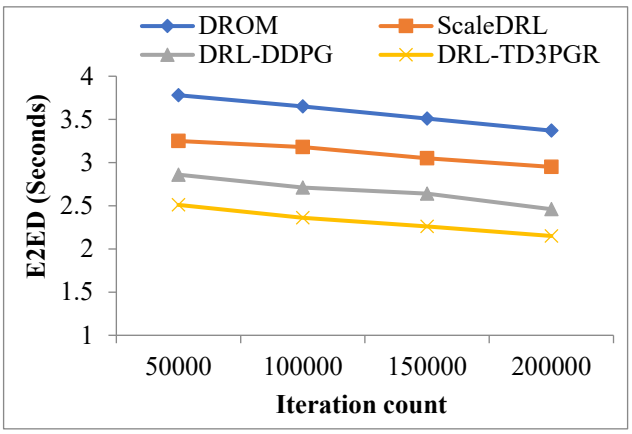



Figure 7 E2ED vs. Routing Protocols

Figure 7 shows that the proposed system has the lowest delay of 2.15 seconds, whereas other methods, such as DROM, ScaleDRL, and DRL-DDPG, have the highest delays of 3.37 seconds, 2.95 seconds, and 2.46 seconds, respectively, for 200000 iterations.

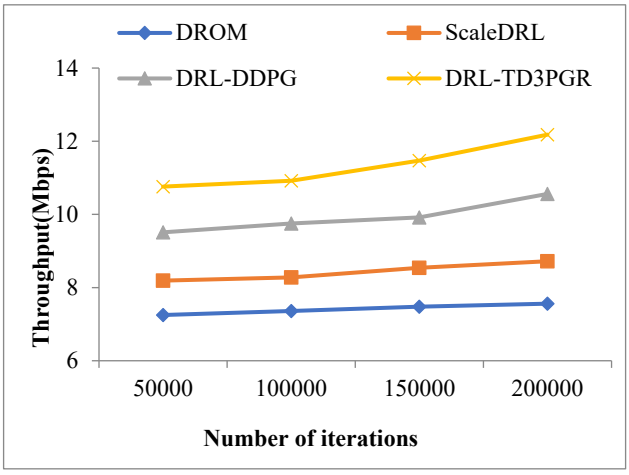



Figure 8 Throughput vs. Routing Protocols

Figure 8 shows that the proposed system has the highest throughput of 12.18 Mbps, whereas other methods, such as DROM, ScaleDRL, and DRL-DDPG, have the lowest throughput of 7.56 Mbps, 8.72 Mbps, and 10.56 Mbps for 200000 iterations.

4.2. Average Network Performance Based on Iterations of the GEANT Topology

Figure 9 shows that the proposed system has the highest PDR of 94.70%. Other methods, such as DROM, ScaleDRL, and

RESEARCH ARTICLE

DRL-DDPG, have the lowest PDR of 92.20%, 92.90%, and 93.60% for 200000 iterations.

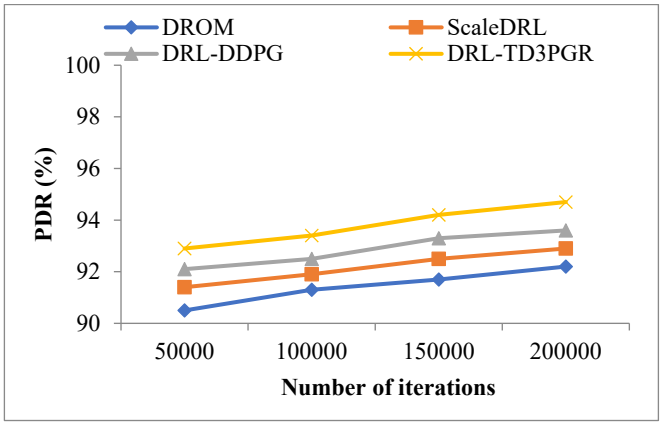



Figure 9 Packet Delivery Ratio (PDR) vs. Routing Protocols

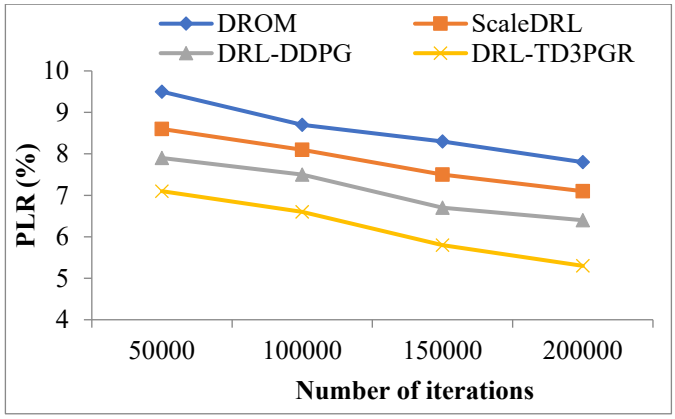



Figure 10 Packet Loss Ratio (PLR) vs. Routing Protocols

Figure 10 shows that the proposed system has the lowest PLR of 5.30%. Other methods, such as DROM, ScaleDRL, and DRL-DDPG, have the highest PLR of 7.80%, 7.10%, and 6.40% for 200000 iterations.

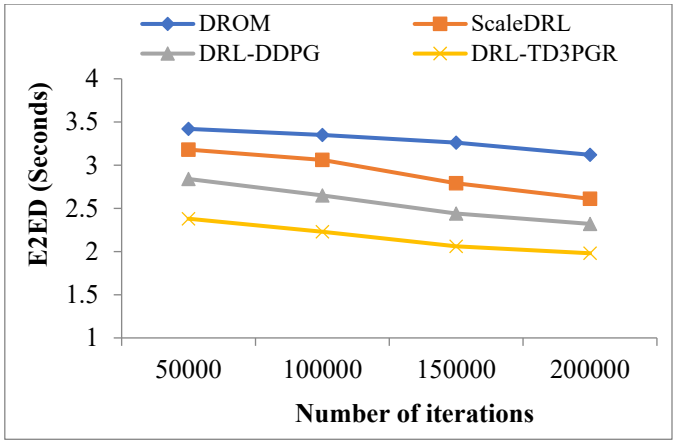



Figure 11 E2ED vs. Routing Protocols

Figure 11 shows that the proposed system has the lowest delay of 1.98 seconds, whereas other methods, such as DROM, ScaleDRL, and DRL-DDPG, have the highest delays of 3.12 seconds, 2.61 seconds, and 2.32 seconds, respectively, for 200000 iterations.

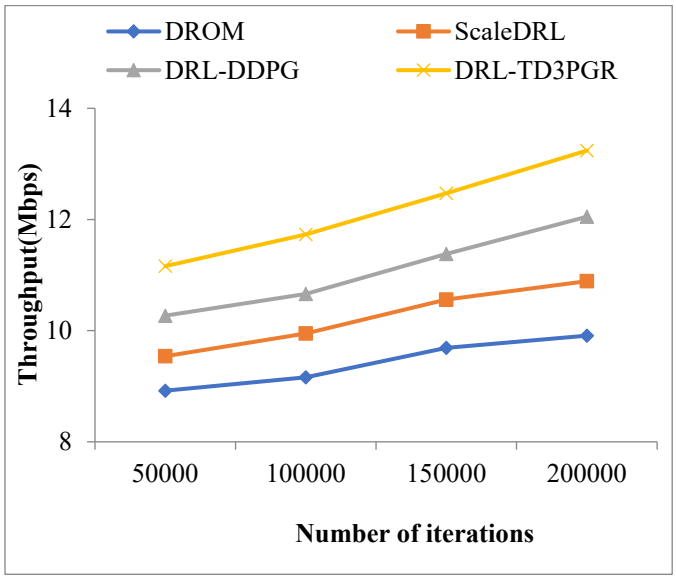



Figure 12 Throughput vs. Routing Protocols

Figure 12 shows that the proposed system has the highest throughput of 13.24 Mbps, whereas other methods, such as DROM, ScaleDRL, and DRL-DDPG, have the lowest throughput of 9.91 Mbps, 10.89 Mbps, and 12.05 Mbps for 200000 iterations.

### 4.3. E2ED Performance Comparison vs. Number of Flows

Figures 13 and 14 show the E2ED comparison of routing protocols among the grid topology and the GEANT topology.

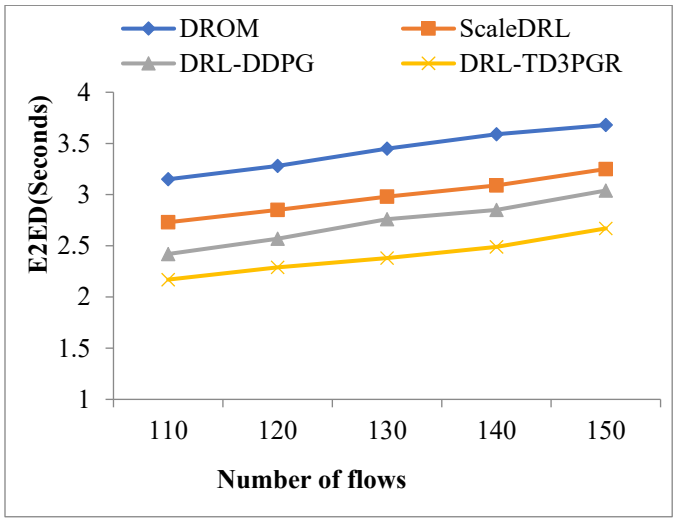



Figure 13 E2ED vs. RP (GT)

RESEARCH ARTICLE

Figure 13 shows that the proposed system has the lowest delay of 2.67 seconds, whereas other methods, such as DROM, ScaleDRL, and DRL-DDPG, have the highest delays of 3.68 seconds, 3.25 seconds, and 3.04 seconds, respectively, for 150 flows in a grid topology.

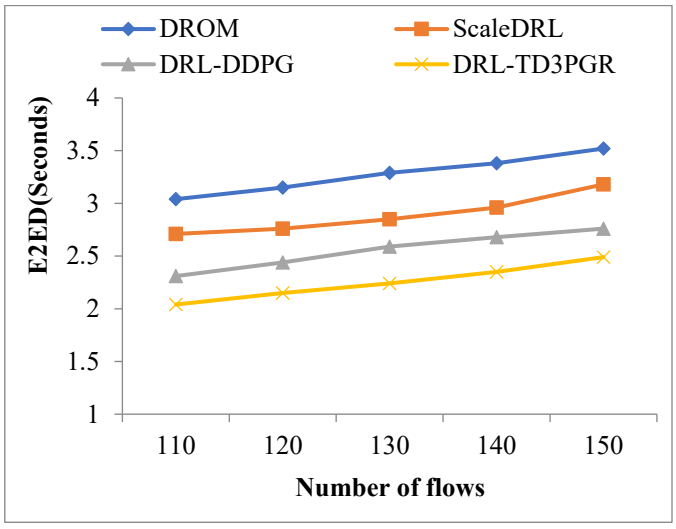



Figure 14 E2ED vs. Routing Protocols (GEANT Topology)

Figure 14 shows that the proposed system has the lowest delay of 2.49 seconds, whereas other methods, such as DROM, ScaleDRL, and DRL-DDPG, have the highest delays of 3.52 seconds, 3.18 seconds, and 2.76 seconds, respectively, for 150 flows in the GEANT topology.

4.4. Throughput Comparison vs. Flow Count

The average network throughput of the flow count in a GEANT structure and GT is presented in Figure 15 and Figure 16.

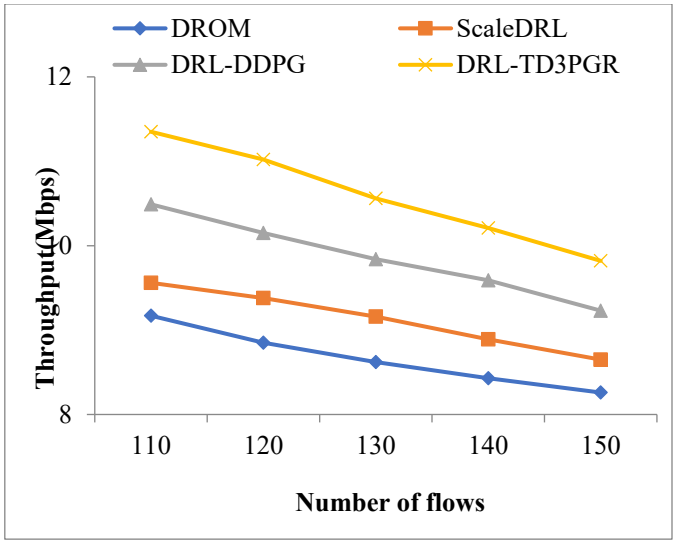



Figure 15 Throughput vs. Routing Protocols (Grid Topology)

Figure 15 indicates that the proposed system has the highest throughput of 9.82 Mbps, whereas other methods, such as DROM, ScaleDRL, and DRL-DDPG, have the lowest throughput of 8.26 Mbps, 8.65 Mbps, and 9.23 Mbps for 150 flows in a grid topology.

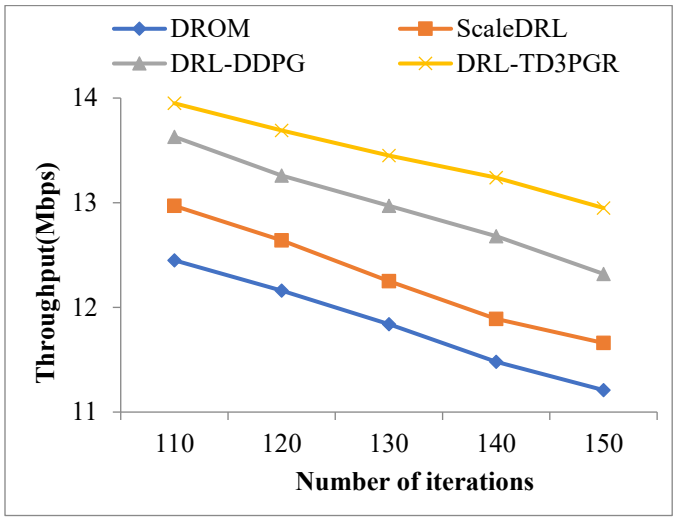



Figure 16 Throughput vs. Routing Protocols (GEANT Topology)

Figure 16 shows that the proposed system has the highest throughput of 12.95 Mbps, whereas other methods, such as DROM, ScaleDRL, and DRL-DDPG, have the lowest throughput of 11.21 Mbps, 11.66 Mbps, and 12.32 Mbps for 150 flows in the GEANT topology. With a relatively high system dimensions, the recommended method outperformed the grid topology in the GEANT topology and had the largest T irrespective of the flow count.

The DRL-TD3PGR model dynamically adjusts its routing decisions, delivering a 15-20% reduction in latency and a 10-12% increase in throughput under fluctuating bandwidth and packet loss conditions, in contrast to Dijkstra's algorithm, which typically demonstrates performance degradation in dynamic SDN environments. The model demonstrates superior decision-making skills in SDN networks with dynamic traffic patterns, outperforming the Q-WPRA (Q-learning-based Wide-Path Routing Algorithm) by up to 18% in identifying optimal transmission routes with the widest bandwidth (BW).

Compared to conventional approaches, the DRL-TD3PGR model's attack detection rate is 35% higher, thanks in large part to its real-time security mechanism that actively prevents DoS assaults and data breaches. Overall network dependability is increased by around 20% and packet loss is reduced by 8-10% as a result of this end-to-end optimization, leading to greater Quality of Service (QoS). Additionally, the model's scalability has been demonstrated in large SDN systems, where it continues to perform well as network

RESEARCH ARTICLE

capacity increases and additional traffic scenarios are introduced.

## 5. CONCLUSION AND FUTURE WORK

TD3PGR, which combines DPQ and DQN, is presented in this study. To reflect the features of the DP's DN, the network model in the suggested system is built using DN, such as NT and switch specifications, supplied by the SDN controller of the CP. A DNN is used to build the TD3PGR actor, which uses observation inputs and action outputs. An NN is used to represent network routing by combining one FC layer. To determine the optimal action-selection strategy for PF with balanced link WA, the TD3PGR agent trains a neural model, hence minimizing PL and delay in the network. For data security in the DN layer that supports a partially homomorphic method, the Paillier encryption technique is introduced. The agent can train the NN without degrading the data network's performance by employing offline learning with modelled networks. The suggested routing technique can enhance network performance by decreasing E2ED, increasing PDR, decreasing PLR, and achieving the greatest T, according to simulation results on two distinct network topologies. In subsequent research, DL techniques and nature-inspired approaches will be used to optimise routing. It might be more feasible to optimise SDN routing using heuristic techniques. To obtain nearly optimum solutions, metaheuristics, which are an extension of heuristics, offer an algorithm framework that is independent of the problem and may iteratively enhance the solution quality of a given fitness function.

## REFERENCES

[1] J. Ali, R.H. Jhaveri, M. Alswailim, and B.H. Roh, 2023. "ESCALB: An effective slave controller allocation-based load balancing scheme for multi-domain SDN-enabled-IoT networks," Journal of King Saud University-Computer and Information Sciences, vol. 35, no. 6, pp. 101566.

[2] J. Chen, X. Huang, Y. Wang, H. Zhang, C. Liao, X. Xie, X. Li, and W. Xiao, 2023. "ASTPPO: A proximal policy optimization algorithm based on the attention mechanism and spatio–temporal correlation for routing optimization in software-defined networking," Peer-to-Peer Networking and Applications, vol. 16, no. 5, pp. 2039-2057.

[3] Arun, M., Barik, D., Chandran, S. S., Praveenkumar, S., & Tudu, K. (2025). Economic, policy, social, and regulatory aspects of AI-driven smart buildings. Journal of building engineering, 99, 111666.

[4] J.Chen, W.Xiao, H.Zhang, J. Zuo, and X.Li, 2024. "Dynamic routing optimization in software-defined networking based on a metaheuristic algorithm," Journal of Cloud Computing, vol.13, no.1, pp.1-18.

[5] A. Cummaudo, R. Vasa, J. Grundy, and M. Abdelrazek, 2020. "Requirements of API documentation: a case study into computer vision services," IEEE Transactions on Software Engineering, vol. 48, no. 6, pp. 2010-2027.

[6] N.C. Luong, D.T. Hoang, S. Gong, D. Niyato, P. Wang, Y.C. Liang, and D.I. Kim, 2019. "Applications of deep reinforcement learning in communications and networking: A survey," IEEE communications surveys & tutorials, vol. 21, no. 4, pp. 3133-3174.

[7] G. Trimponias, Y. Xiao, X. Wu, H. Xu, and Y. Geng, 2019. "Node-constrained traffic engineering: Theory and applications," IEEE/ACM Transactions on Networking, vol. 27, no. 4, pp. 1344-1358.

[8] Abubeker, K. M., Prajitha, C., Rinesh, S., Arun, M., & Senthil Kumar, A. P. (2025). Internet of Things and LoRaWAN-assisted real-time indoor air quality monitoring and automation system for enhancing passenger safety. International Journal of Low-Carbon Technologies, 20, 1427-1439.

[9] Z.M. Fadlullah, F. Tang, B. Mao, N. Kato, O. Akashi, T. Inoue, and K. Mizutani, 2017. "State-of-the-art deep learning: Evolving machine intelligence toward tomorrow's intelligent network traffic control systems," IEEE Communications Surveys & Tutorials, vol. 19, no. 4, pp. 2432-2455.

[10] Arun, M., Barik, D., Othman, N. A., Praveenkumar, S., & Tudu, K. (2025). Investigating the performance of AI-driven smart building systems through advanced deep learning model analysis. Energy Reports, 13, 5885-5899.

[11] Y. Yu, X. Si, C. Hu, and J. Zhang, 2019. "A review of recurrent neural networks: LSTM cells and network architectures," Neural computation, vol. 31, no. 7, pp. 1235-1270.

[12] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P.S. Yu, 2020. "A comprehensive survey on graph neural networks," IEEE transactions on neural networks and learning systems, vol. 32, no. 1, pp. 4-24.

[13] S. Yang, C. Tan, D.Ø. Madsen, H. Xiang, Y. Li, I. Khan, and B.J. Choi, 2022. "Comparative analysis of routing schemes based on machine learning," Mobile Information Systems, vol. 2022, no. 1, pp. 4560072.

[14] C.H. Ke, Y.H. Tu, and Y.W. Ma, 2023. "A reinforcement learning approach for widest path routing in software-defined networks," ICT Express, vol. 9, no. 5, pp. 882-889.

[15] P. Sun, Z. Guo, J. Li, Y. Xu, J. Lan, and Y. Hu, 2021. "Enabling scalable routing in software-defined networks with deep reinforcement learning on critical nodes," IEEE/ACM Transactions on Networking, vol. 30, no. 2, pp. 629-640.

[16] Y.R. Chen, A. Rezapour, W.G. Tzeng, and S.C. Tsai, 2020. "RL-routing: An SDN routing algorithm based on deep reinforcement learning," IEEE Transactions on Network Science and Engineering, vol. 7, no. 4, pp. 3185-3199.

[17] P. Sun, Z. Guo, J. Lan, J. Li, Y. Hu, and T. Baker, 2021. "ScaleDRL: A scalable deep reinforcement learning approach for traffic engineering in SDN with pinning control," Computer Networks, vol. 190, pp. 107891.

[18] Q. Fu, E. Sun, K. Meng, M. Li, and Y. Zhang, 2020. "Deep Q-learning for routing schemes in SDN-based data center networks," IEEE Access, vol. 8, pp. 103491-103499.

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

[20] D. Xia, J. Wan, P. Xu, and J. Tan, 2022. "Deep reinforcement learning-based QoS optimization for software-defined factory heterogeneous networks," IEEE Transactions on Network and Service Management, vol. 19, no. 4, pp. 4058-4068.

[21] D.M. Casas-Velasco, O.M.C. Rendon, and N.L. da Fonseca, 2020. "Intelligent routing based on reinforcement learning for software-defined networking," IEEE Transactions on Network and Service Management, vol. 18, no. 1, pp. 870-881.

[22] J. Rischke, P. Sossalla, H. Salah, F.H. Fitzek, and M. Reisslein, 2020. "QR-SDN: Towards reinforcement learning states, actions, and rewards for direct flow routing in software-defined networks," IEEE Access, vol. 8, pp. 174773-174791.

[23] W.X. Liu, J. Cai, Q.C. Chen, and Y. Wang, 2021. "DRL-R: Deep reinforcement learning approach for intelligent routing in software-defined data-center networks," Journal of Network and Computer Applications, vol. 177, pp. 102865.

[24] J. Chen, W. Xiao, H. Zhang, J. Zuo, and X. Li, 2024. "Dynamic routing optimization in software-defined networking based on a metaheuristic algorithm," Journal of Cloud Computing, vol. 13, no. 1, pp. 41.

[25] Kumar, C. L., Betam, S., Pustokhin, D., Laxmi Lydia, E., Bala, K., Aluvalu, R., & Panigrahi, B. S. (2025). Metaparameter optimized

RESEARCH ARTICLE

hybrid deep learning model for next generation cybersecurity in software defined networking environment. Scientific Reports, 15(1), 14166.
[26] Meng, Y., Ke, C., & Huang, Z. (2024). A model transformation based security policy automatic management framework for software-defined networking. Computers & Security, 142, 103850.
[27] Hatamleh, H., Alnaser, A. A. M. A. A., Saloum, S. S., Sharadqeh, A., &Alkasassbeh, J. S. (2025). Pictureguard: Enhancing software-defined networking–internet of things security with novel image-based authentication and artificial intelligence-powered two-stage intrusion detection. Technologies, 13(2), 55.
[28] Naresh, V. S., & Ayyappa, D. (2025). Enhancing security in software defined networks: Privacy-preserving intrusion detection with Homomorphic Encryption. Journal of Information Security and Applications, 92, 104084.
[29] Shirsath, V. A., Chandane, M. M., Lal, C., & Conti, M. (2024). SYNTROPY: TCP SYN DDoS attack detection for Software Defined Network based on Rényi entropy. Computer networks, 244, 110327.
[30] T.T. Nguyen, N.D. Nguyen, and S. Nahavandi, 2020. "Deep reinforcement learning for multiagent systems: A review of challenges, solutions, and applications," IEEE transactions on cybernetics, vol. 50, no. 9, pp. 3826-3839.
[31] G. Kim, Y. Kim, and H. Lim, 2022. "Deep reinforcement learning-based routing on software-defined networks," IEEE Access, vol. 10, pp. 18121-18133.
[32] M. Nicola, C.I. Nicola, and D. Selișteanu, 2022. "Improvement of the control of a grid connected photovoltaic system based on synergetic and sliding mode controllers using a reinforcement learning deep deterministic policy gradient agent," Energies, vol. 15, no. 7, pp. 2392.
[33] S. Dankwa, and W. Zheng, 2019. "Twin-delayed ddpg: A deep reinforcement learning technique to model a continuous movement of an intelligent robot agent," In Proceedings of the 3rd international conference on vision, image and signal processing, pp. 1-5.
[34] N.A. Mosali, S.S. Shamsudin, O. Alfandi, R. Omar, and N. Al-Fadhali, 2022. "Twin delayed deep deterministic policy gradient-based target tracking for unmanned aerial vehicle with achievement rewarding and multistage training," IEEE Access, vol. 10, pp. 23545-23559.
[35] T.T. Nguyen, N.D. Nguyen, and S. Nahavandi, 2020. "Deep reinforcement learning for multiagent systems: A review of challenges, solutions, and applications," IEEE transactions on cybernetics, vol. 50, no. 9, pp. 3826-3839.

Authors

**Mrs. J. Delshi Julie** is an Assistant Professor in the Department of Computer Science at Bishop Ambrose College (Autonomous), Coimbatore – 641045.

**Dr. R. Beulah** is an Assistant Professor in the Department of Computer Science at PPG College of Arts and Sciences (Affiliated to Bharathiar University), Coimbatore.

**How to cite this article:**

J. Delshi Julie, R. Beulah, "Twin Delayed Deep Deterministic Policy Gradient Based Routing (TD3PGR) and Security Mechanism for Software-Defined Network (SDN)", International Journal of Computer Networks and Applications (IJCNA), 12(5), PP: 703-719, 2025, DOI: 10.22247/ijcna/2025/42.