IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 21 July 2022, accepted 21 August 2022, date of publication 6 September 2022, date of current version 14 September 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3204812

RESEARCH ARTICLE

Reinforced Transformer Learning for VSI-DDoS Detection in Edge Clouds

ADIL BIN BHUTTO[1], XUAN SON VU[1], (Member, IEEE), ERIK ELMROTH[1], WEE PENG TAY[2], (Senior Member, IEEE), AND MONOWAR BHUYAN[1], (Member, IEEE)

[1]Department of Computing Science, Umeå University, 901 87 Umeå, Sweden
[2]School of Electrical & Electronics Engineering, Nanyang Technological University, Singapore 639798

Corresponding author: Monowar Bhuyan (monowar@cs.umu.se)

This work was supported in part by the STINT Project funded by the Swedish Foundation for International Cooperation in Research and Higher Education; and in part by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by Knut and Alice Wallenberg Foundation.

ABSTRACT Edge-driven software applications often deployed as online services in the cloud-to-edge continuum lack significant protection for services and infrastructures against emerging cyberattacks. Very-Short Intermittent Distributed Denial of Service (VSI-DDoS) attack is one of the biggest factors for diminishing the Quality of Services (QoS) and Quality of Experiences (QoE) for users on edge. Unlike conventional DDoS attacks, these attacks live for a very short time (on the order of a few milliseconds) in the traffic to deceive users with a legitimate service experience. To provide protection, we propose a novel and efficient approach for detecting VSI-DDoS attacks using reinforced transformer learning that mitigates the tail latency and service availability problems in edge clouds. In the presence of attacks, the users' demand for availing ultra-low latency and high throughput services deployed on the edge, can never be met. Moreover, these attacks send very-short intermittent requests towards the target services that enforce longer delays in users' responses. The assimilation of transformer with deep reinforcement learning accelerates detection performance under adverse conditions by adapting the dynamic and the most discernible patterns of attacks (e.g., multiplicative temporal dependency, attack dynamism). The extensive experiments with testbed and benchmark datasets demonstrate that the proposed approach is suitable, effective, and efficient for detecting VSI-DDoS attacks in edge clouds. The results outperform state-of-the-art methods with $0.9\% - 3.2\%$ higher accuracy in both datasets.

INDEX TERMS Reinforced transformer learning, VSI-DDoS, edge clouds, QoS/QoE, cloud applications.

I. INTRODUCTION

The advent of beyond 5G technology and the era of information-centric decision-making will require deploying multiple edge-driven applications for making day-to-day life more comfortable. For instance, to make mass adoption possible for technologies like augmented and virtual reality, autonomous vehicles, smart cities, tele-healthcare, massive Internet of Things (IoT) devices, home automation, etc. [1], high availability and low-latency service requirements have to be insured. Hence, edge clouds have emerged to mitigate such problems. However, these bring several security issues as nodes are distributed across the edge of the networks for deploying applications closer to users. Besides these, edge computing adds three primary features such as *backbone network alleviation* – by processing data without exchange with distant clouds, *agile service response* – reduces delay in transmission and increased response time, and *robust cloud backup* – extending capability using the distant cloud. Despite the benefit of edge clouds, service providers face several challenges when deploying their services in edge clouds including (but not limited to), e.g., ensuring high availability of services, defense against emerging attacks, and optimal placement of applications among highly distributed geographical nodes.

Even though methods [2], [3], [4] exist to mitigate security holes in the edge clouds, there is still a lack of solutions for multiple problems. These include slow-rate DDoS attacks

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

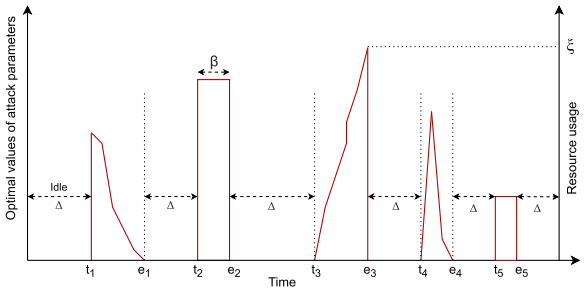


FIGURE 1. Illustration of attack parameter variation in VSI-DDoS attack. β is the degree of HTTP requests concentration from multiple bots, Δ represents an idle period, and $(e_1 - t_1)$ indicates attack intensity.

with various burst times when the attackers target mission-critical services, users' Quality of Experience (QoE), and systems' Quality of Service (QoS) to sabotage performance over the long term instead of a short period. These attacks primarily target the availability of the systems by sending a large amount of traffic to exhaust resources such as CPU, bandwidth, or memory [5]. In these cases, the legitimate users' services are frequently denied or interrupted, resulting in significant financial losses for the service providers. Over time, these attacks become more intelligent, sophisticated and robust through strengthening attack vectors every round and exponentially increasing the attack size, frequency, and randomness. Therefore, defending against such attacks is a priority in academic and industry communities, specifically for edge clouds to meet users' expected latency and throughput. Concurrently, several low-rate DDoS attacks have emerged, which are different from classical volume-based or resource-exhaustion DDoS attacks [6], [7]. The primary goal of such low-rate attacks is to degrade the QoS performed by employing TCP congestion control mechanisms [7]. They are stealthy, and they maintain low-volume, resulting in potential violations of customer service-level agreements (SLAs). Notably, Very-Short Intermittent Distributed Denial of Service (VSI-DDoS) [8] is one of the low-rate attacks and also the biggest threat to services deployed on edge. Typically, it is hard to visualize damages caused by low-rate attacks. However, there are growing concerns about such attacks and their potential implications for violating SLAs associated with the edge cloud, where deployed applications are sensitive to latency and throughput [9]. Moreover, these attacks send short and intelligent legitimate HTTP requests for a short period (often tens of milliseconds) to multiple target services that maximize the response time and reduce users' QoS/QoE in the long run (see Fig. 1). Hence, detecting such attacks at an early stage is more challenging than using classical DDoS detection methods.

Transformer learning primarily applies and performs well in natural language processing (NLP) and computer vision tasks [10], [11]. A key factor to success in these areas is how text, images or videos are represented through representation learning [11]. Transformer models are built based on multi-head attention, which helps analyze time-series data because it considers contextual information (past-future), different representation subspaces, and adapting periodic and nonperiodic patterns. The impressive success of transformers inspires us to use a transformer with reinforcement learning in securing edge systems, which remains unexplored. Primarily, transformer-based reinforcement learning is known to be unstable and inefficient for making downstream applications [12]. The features, including experience replay and multi-head attention, are crucial to adapt dynamic temporal behaviour and discernible patterns in data to induce contextual information in learning to detect VSI-DDoS attacks in edge clouds. Hence, we propose a transformer-based neural model with learnable time representation to detect VSI-DDoS attacks on the edge.

Reinforced transformer learning (RTN) is a learning approach in which a transformer-based model is trained in a reinforcement learning environment. It helps in model training to achieve higher efficacy under multiple settings to detect VSI-DDoS attacks. However, the transformer integrates deep reinforcement learning to employ said features to mitigate emerging service-targeted attacks in edge clouds. This paper makes the following contributions by combining the requirements for low-rate and VSI-DDoS detection with the capability of autonomy in edge clouds.

1) First, we introduce a transformer-based VSI-DDoS detection approach on edge with learnable time representation in its architecture, known as VSI-TN.
2) Second, we introduce a transformer-induced deep reinforcement learning approach known as VSI-RTN to make attack detection efficient and autonomous for edge clouds.
3) Third, transformer integration with deep reinforcement learning makes it possible to prioritize learning on context-driven information (e.g., attack dynamism, temporal dependency) for detecting VSI-DDoS attacks under uncertainty.
4) Lastly, systematic and extensive experimental analyses are carried out with testbed and benchmark datasets while comparing them with state-of-the-art baseline models, including DNN-based models (e.g., Bi-LSTM, LSTM) and Deep Reinforcement Learning model (i.e., DeROL [13]).

Organization. The rest of the paper is structured as follows. Section II discusses prior research on transformers and deep reinforcement learning methods for DDoS detection. The proposed system model is reported in Section III while Section IV presents detailed experimental analysis. Finally, the conclusion and future work are given in Section V.

II. RELATED WORK

Cyber threats in web applications have been rising due to massive-scale services or microservices deployment on edge for multiple domains. Users expect services from service providers with expected QoS when using Internet services. However, service providers can be severely affected by users'

unexpected QoS experience when using services deployed on edge. Typically, users browse multiple web pages together, and if latency reaches several seconds, they move to other service providers. Because users do not like to use underprivileged services for longer, resulting in substantial financial loss for providers. Hence, Google and Amazon[1] are putting much effort into reducing tail latency up to a certain level that reduces users' inconvenience. When service providers identify such behaviour, it ends with new types of attacks, e.g., low-rate attacks. These attacks differ from classical DDoS attacks that paralyze complete links or resources down by sending exponential traffic.

Remarkably, the recent VSI-DDoS attacks target primarily applications that offer QoS/QoE sensitive services on edge in contrast to classical DDoS attacks. Mitigation is vital to developing mechanisms to counter such attacks early, but hard to make it in edge clouds. Most existing methods were developed for classical DDoS detection based on machine learning [14], deep learning [15] and deep reinforcement learning [13]. Saied *et al.* [16] present an ANN-based model for detecting high and low-rate DDoS attacks, evaluated only with TCP, UDP, and ICMP protocols. A low-rate DDoS attack detection method for the cyber-physical system is reported in [17] using Deep Convolutional Neural Network (DCNN) and deep Q-network, underperforming for sparse data. Recently, Forough *et al.* [2] reported a VSI-DDoS attack detection mechanism using LSTM-att model but having poor performance and underperforming in adverse conditions. Yeom *et al.* [5] use LSTM (Long Short Term Memory) based model for source-side DoS attack detection. They deploy detection modules on a gateway of a target subnet to detect DoS attacks in advance. Still, this type of deployment is costly, and the involvement of several network service providers and different vendors makes it nearly impossible. Roosmalen *et al.* [18] employ DNN (Deep Neural Network) based supervised detection approach to identify botnets on packet flows. However, it only considers the detection of known botnet anomalies without temporal information. Wu *et al.* [19] introduce a transformer-based approach that utilizes a positional encoding technique to associate sequential information between features and a self-attention mechanism to facilitate network traffic type classification. However, it lacks consideration of temporal information during model training and is assessed only with two benchmark datasets. Yeom *et al.* [20] propose a collaborative source-side DDoS attack detection framework based on LSTM. This approach involves sharing attack detection results amongst source-side networks of multiple regions, making this method expensive and difficult to collaborate with different real-world entities. Moura *et al.* [9] discuss the employment of open-source programmable asset orchestration to defend against faults, congestion, or cyber-attacks in edge cloud systems. Ünal *et al.* [21] propose a multi-anomaly detection model for cyber threat data. Pretrained transformers' variant is used to encode log sequences for learning the structure along with anomaly types. It employs natural language processing to find-out cyber threats from system logs, which cannot be used in real-time detection and mitigation of anomalies. Table 1 gives a comparison amongst existing and our proposed methods.

Many deep reinforcement learning approaches have been developed to detect, protect, and be resilient against cyber threats by utilizing experience replay or feedback mechanisms in multiple domains [22]. However, exploring deep Q-learning combined with a transformer for detecting VSI-DDoS attacks remains an open problem. A high temporal dependency and dynamic behaviour adoption in a short period of time cause VSI-DDoS detection more difficult; also, they bear legitimate behaviour during attacks targeting multiple services for degrading users' QoS/QoE.

III. SYSTEM MODEL

Due to the complex nature of VSI-DDoS attacks (e.g., stealthy, sub-saturating, legitimate utilization of server's resources, varied data patterns in each slot of extreme increase of request), the detection methods [23] overlook attacks before degrading the QoS of web services. For example, the sudden increase of HTTP requests in a short period exceeds the server queue limit and causes a delayed response to legitimate users. Therefore, it's necessary to have a model to capture those patterns to improve the detection performance. The transformer plays a vital role in accomplishing such tasks and is advantageous due to having a self-attention mechanism. Moreover, to employ the features of deep Q-learning combined with transformer, we formulate the VSI-DDoS detection problem as learnable time-representation, experience replay, and dynamic policy update for performing the detection operations early and efficiently.

A. PROBLEM FORMULATION

Given the VSI-DDoS problem, identifying attacks in services deployed among edge servers formulated as a classification task with two classes: *legitimate* and *attack*. However, multiple categories of attacks exist [2], [8] (e.g., VSI-DDoS vertical, VSI-DDoS horizontal, VSI-DDoS application) to manipulate services at different levels of deployed applications. Therefore, without losing generality, we assume that $\mathcal{X}$ and $\mathcal{Y} = \{0, 1\}$ denote an instance space and the set of possible classes with timestamp t, where 0 and 1 encode as *legitimate* and *attack* instances, respectively. Given training data in the form of a finite set of observations:

$$\mathcal{D} = \left\{(\boldsymbol{x}_{n_t}, \boldsymbol{y}_{n_t})\right\}_{n=1}^{N} \subseteq \mathcal{X} \times \mathcal{Y}\,, \tag{1}$$

drawn independently from $\vec{p}(\mathbf{X}, \mathbf{Y})$, i.e., the probability distribution $\vec{p}$ on $\mathcal{X} \times \mathcal{Y}$. The goal of detecting VSI-DDoS attacks is to learn a classifier $\mathbf{h}$, which is a mapping $\mathcal{X} \longrightarrow \mathcal{Y}$ that assigns a label to each instance $\boldsymbol{x}_t \in \mathcal{X}$. Thus, the output of the classifier $\mathbf{h}$ is defined as transformer ($\mathbf{h}_T$) and deep

[1]The tail latency is defined as the latency of a server's 99th percentile response, which is the delay that users experience in the worst case.

TABLE 1. Comparison of existing methods.

Model	Model features for security problems	Application-layer DDoS	Low-rate DDoS	No. of features	Automated feature representation	Datasets		Accuracy
						Testbed	Benchmark	
ML [14]	Inference based on signatures from previous samples of network traffic			20				96.00
LSTM [5]	Source-side DoS attack detection by learning irregular seasonal pattern			3				92.00
ANN [16]	Detect and mitigate known and unknown DDoS attacks in real time environments			–		✓		98.00
DeROL [13]	One shot learning for multiple network attacks detection	✓		–			✓	–
LSTM-Att [2]	Learning from the most important discernible patterns of sequence data	✓	✓	28, 41, 42		✓	✓	89.74
VSI-TN (Ours)	Detect VSI-DDoS adopting dynamic temporal behavior of data	✓	✓	**39, 40, 81**	✓	✓	✓	**98.70**
VSI-RTN (Ours)	Detect VSI-DDoS by priority learning on context-driven information	✓	✓	**39, 40, 81**	✓	✓	✓	**98.43**

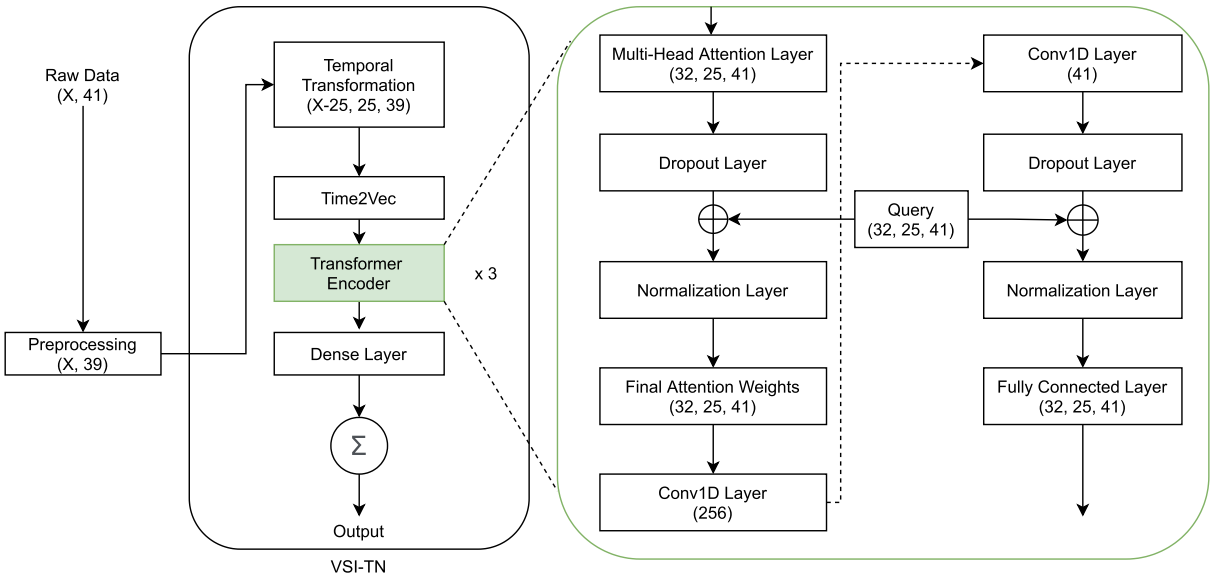


FIGURE 2. Architecture of the proposed VSI-TN - a transformer-based neural model with time-representation layer using Time2Vec [25].

Q-learning with transformer ($\mathbf{h}_{QT}$).

$$\hat{y_t} := \mathbf{h}_T(x_{t_{tr}}) \in \{0, 1\}. \tag{2}$$

$$\hat{y_t} := \mathbf{h}_{QT}\Big(\mathbf{h}_T(x_t), R_p\{x_t, E_d(l_{v_i}, a_{v_j}), M_a(s_t, s_{t'})\}\Big) \in \{0, 1\}. \tag{3}$$

where $\mathbf{h}_T(\cdot)$ is a transformer based model in which the $\mathbf{h}_T$ in Eq(2) receives time transformed $x_{t_{tr}}$ as input, and Eq(3) receives x_t as input. R_p represents deep Q-learning with belief-vector, E_d - will estimate based on the legitimate data l_{v_i} and the attack data a_{v_j} for the current state. M_a computes the similarity between samples at different states and feeds them to the classifier and the policy modules. It is worth noting that the proposed method can detect multi-class VSI-DDoS attacks and overperform existing methods.

B. TRANSFORMER-BASED NEURAL NETWORK

Transformer-based models (e.g., BERT [10]) consist of several encoder and decoder layers with multi-head attention. To solve our problem, we employ the transformer's encoder layer for input data's intensive and compact feature representation. We instantiate and train $\mathbf{h}_T$ for VSI-TN using multi-head attention layers (as shown in Figure 2) inspired from self-attention layer [24]. The input is transformed into three vectors: the query vector q, the key vector k, and the value vector v with dimension $d_q = d_k = d_v = d_{model}$, packed them as K, V, Q. The attention is computed using the following [24].

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{4}$$

The transformer architecture employs one time-embedding layer (time2vec), three encoder layers, and a classification head placed after the last layer for smooth initiation of the training process.

1) LEARNING TEMPORAL REPRESENTATION

For learning and adopting dynamic temporal behaviour of data, we harmonize the Time2Vec [25] architecture with our

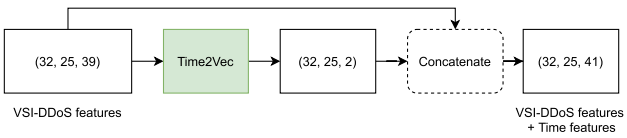


FIGURE 3. Illustration of Time2Vec layer.

proposed approach for model agnostic vector representation of time. This vector representation is expressed as follows.

$$t2v(\tau)_{[i]} = \begin{cases} \omega_i\tau + \varphi_i & i = 0 \\ F(\omega_i\tau + \varphi_i) & 1 \leq i \leq k \end{cases} \quad (5)$$

where $(\omega_i\tau + \varphi_i)$ represents the non-periodic and $F(\omega_i\tau + \varphi_i)$ indicates periodic features of the time vector. As a result, two additional features are obtained from Time2Vec layer followed by concatenation with original input as shown in Figure 3. In each training iteration, the transformer receives 32 sequences with window size 25 and has 39 features (specific to UVSI-DDoS-I and II datasets) per instance for optimal training performance.

C. REINFORCED TRANSFORMER NETWORK

Considering the nature of system states, temporal dependency, and adapting dynamic attack behaviour, we assume that deep Q-learning with a transformer could provide an appropriate solution for detecting VSI-DDoS attacks in edge clouds. Therefore, we propose VSI-RTN blends with VSI-TN to improve the model efficacy as shown in Figure 4, inspired by DeROL [13]. VSI-RTN differs from DeROL: (1) the new *VSIDDoS Attack Classifier* based on VSI-TN model, and (2) the *Rule Base* module. First, the VSI-TN model is designed to efficiently adapt the VSI-DDoS attack's dynamic behaviour. Second, the *Rule Base* module refers to when DRL policy has found doubtful belief-vectors from the classifier module. After assessing the rule-based module, the classifier will be updated to improve the reinforced learning process.

Here, we utilize Deep Q-Network (DQN) as a function approximator that maps from partially observed states to action without storing Q-values. Deep reinforcement learning (DRL) policy composes Long Short Term Memory (LSTM) hidden layers, *tanh* as an activation function, and an output layer with an action handler. The reinforcement learning (RL) agent has three categories of actions.

1) *Automatic classification* (a_p) based on the received belief vector and environment parameters, $a_p \epsilon \{a_0, a_1\}$, a_0 being classified as legitimate and a_1 as attack. The classifier is responsible for producing belief vectors by estimating a distance metric and training the VSI-TN model once it receives a labelled instance from the analyst manager.
2) *Assign the classification task* (a_c) to a *Rule Base* module for further assessment. If no rules are applied or found, the request is queued to wait for new rules from the analyst manager (i.e., the next action).
3) *Delay the classification task* (a_d) if the classifier's output is not satisfactory and a similar classification task is already sent to the *Rule-Base* module, then the RL agent verifies the correct classification of similar task with the *Rule-Base* followed by classifier updation to produce expected accuracy.

VSIDDoS attack classifier (VAC) has three components: a Euclidean distance metric, a memory component, and a transformer model. It estimates similarity scores for a new sample using the distance metric corresponding to each class while the memory component stores for already seen samples. Let S be the recently classified sample stored in the classifier. $S_i \in S$ be a subset of classified samples from legitimate($i = 0$) and attack($i = 1$) class (number of classes, $k = 2$). Similar to [13], for a new sample x, the distance between each class(i) is measured by:

$$d_i(x) = min\left(d_{max}, \min_{z \in S_i} d(z, x)\right) \quad (6)$$

where d_{max} is the maximum distance used, $d(z, x)$ represents an Euclidean distance between samples z and x. Belief-vector $E_d = \{e_0, \cdots, e_i, \cdots, e_d\}$ is expressed in terms of similarity scores and $e_i = (d_{max} - d_i(x))$. The transformer model updates independently whenever DRL policy encounters non-decisive samples, i.e., when it cannot take the automatic classification action a_p.

Reward function validates as correct automatic classification with 0 as a legitimate label and 1 as an attack label. RL agent receives a -2 reward for incorrect classification. Reward for assigning a classification task to a rule-base decreases linearly by a factor of 0.5 depending on the present analyst's load ($L_A(t)$), i.e., $(-0.5) \times L_A(t)$. The reward for delaying a classification task decreases exponentially with each time unit of delay (T_D). The exact reward function is $-2^{T_D/10}$. Accordingly, the Q-value gets updated at each time(t) for every action-state pair as follows.

$$\begin{aligned} Q_{t+1}(s(t), a(t)) = Q_t(s(t), a(t)) + \alpha[r(t+1) \\ + \gamma \max_{a(t+1)} Q_t(s(t+1), a(t+1)) \\ - Q_t(s(t), a(t))] \end{aligned} \quad (7)$$

where $r(t+1)$ is obtained reward after action $a(t)$ in state $s(t)$ for learned value $r(t+1) + \gamma \max_{a(t+1)} Q_t(s(t+1), a(t+1))$. Further, move to the next state-action pair $s(t+1)$ and $a(t+1)$ that maximize Q-values seen in the next state and also minimize the time difference error between the learned value and the current estimated value. Here, the learning rate α assumes close to zero, i.e., $0 < \alpha < 1$, and discounted factor γ to 0.5. Loss function to update Q-values for each training batch is given below [26].

$$\begin{aligned} \sum_t [Q_t(s(t), a(t)) - (r(t+1) + \gamma \max_{a(t+1)} \\ \times Q_t(x(t+1), a(t+1)))]^2 \end{aligned} \quad (8)$$

The DRL policy illustrated in Figure 5 receives the following parameters at time t when sample x enters the system:

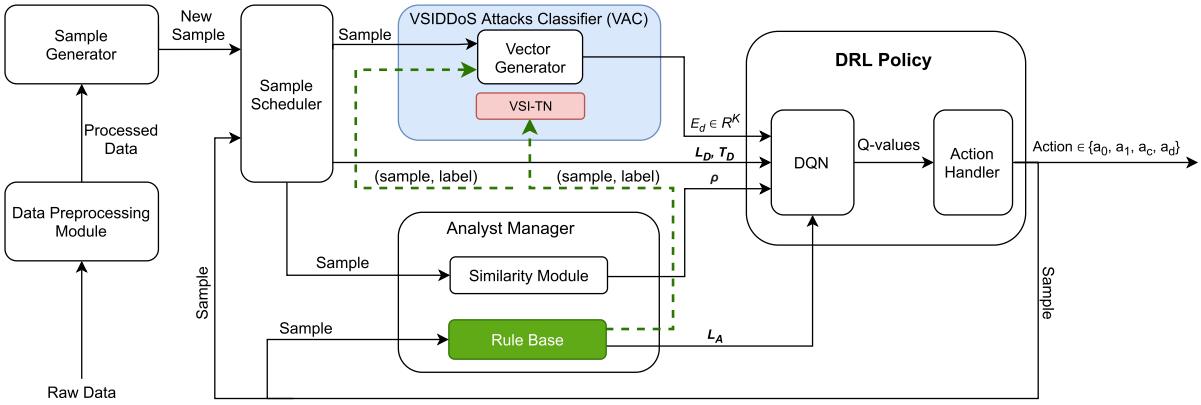


FIGURE 4. Detailed architecture of VSI-RTN, inspired by DeROL [13] where the VSI-RTN uses *Rule Base* to accumulate training data for the *Model Base* to improve real-time model efficacy.

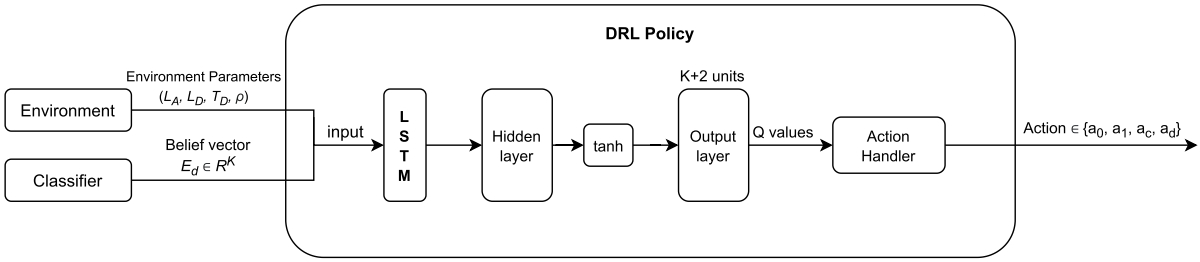


FIGURE 5. Detailed architecture of DRL policy that consists of one LSTM layer, one hidden layer, and an activation layer to learn and optimize for four possible actions $\{a_0, a_1, a_c, a_d\}$.

(i) belief vector $E_d(x) \in \mathbb{R}^K$; (ii) the analyst load $L_A(t)$; (iii) the delayed sample load $L_D(t)$; (iv) classification delay $T_D(x)$, and (v) similarity score $\rho(x)$ reported by Analyst Manager obtain when compares the similarity between x and queued samples for labelling. The input to the DRL policy is expressed as follows:

$$o(t) \triangleq \{E_d(x), L_A(t), L_D(t), T_D(x), \rho(x)\} \tag{9}$$

DRL employs ϵ-greedy policy to explore different actions to maximize Q-value and summarized as follows.

$$a(\epsilon, P_x, P_y) = \begin{cases} a_c & \text{if } P_x < \epsilon \text{ and } P_y < 0.25 \\ a_d & P_x < \epsilon \text{ and } 0.25 \leq P_y < 0.50 \\ a_p & \text{otherwise} \end{cases} \tag{10}$$

where action a_p is *automatic classification*, action a_c is *ask for classification* and action a_d is *delay classification*.

During the learning process, we heuristically decide the ϵ value to 0.05 for optimal DRL policy with exploration and exploitation of varied actions. P_x and P_y are random probabilities associated with each decision. During testing, ϵ is set to 0, which results in exploiting known actions to maximize the Q-value. Here, the DRL policy acts as an offline policy learner since it learns from taking different actions $\{a_c, a_d, a_b\}$ as described in Eq(10). The Q-value function $Q_t(s(t), a(t))$ is learnt *independently* from previous policy, i.e., greedy policy. The steps to explain how VSI-RTN works is given in *Algorithm 1*.

Algorithm 1 VSI-RTN

Require: New Sample from Sample Generator module
Ensure: Trained DQN and VSI-TN model

for iteration $i = \{1, 2, \ldots, R\}$ **do**
  for new sample $m = \{1, 2, \ldots, n\}$ **do**
    add m to sample scheduler (S_{ch})
    while S_{ch} has pending samples **do**
      get a sample $s \in S_{ch}$
      form $o(t)$ following **Eq. (9)**
      generate an estimation of the Q-values $Q(o(t))$ for available actions, $a \in \{a_p, a_c, a_d\}$ by the DQN
      take action a_t according to policy ϕ given $Q(o(t))$
      if action a is a_c **then**
        send S_{ch} to Analyst Manager for correct labelling, VAC's updation and training of VSI-TN
        send S_{ch} to Sample Scheduler for further classification attempt
      end if
      obtain reward $r(t)$
    end while
  end for
  if training phase **then**
    train DQN using loss **Eq. (8)**
  end if
end for

IV. EXPERIMENTAL EVALUATION

This section evaluates the VSI-TN and VSI-RTN models for assessing the efficacy of VSI-DDoS detection by conducting extensive experiments using testbed and benchmark datasets. We explain our testbed setup, data collection, and benchmark datasets following data pre-processing. We also explain window-based time transformation, which is only used for the VSI-TN model. Afterwards, we assess the VSI-TN and the VSI-RTN models to adopt different learning dynamics with and without DRL settings. Finally, we compare and analyse the performance with state-of-the-art methods.

A. DATASETS

We conducted experiments with four real-world datasets, including two testbeds and two benchmark datasets.

1) TESTBED SETUP AND DATA COLLECTION

Testbed setup and data collection is designed and developed by following similar settings available in [27]. We configure an edge server with an n-tier web application benchmark *RUBiS*[2](i.e., web server, an application server, and a DB server) to assess our proposed VSI-DDoS detection models. The 3-tier architecture is followed and deployed on the edge cloud illustrated in Figure 6. *Web application server* deployed as an independent instance with the same virtual specification and offered services using RUBiS. The imitation of *legitimate users* were made using the workload generator RUBBoS.[3] The *Apache Bench* in collaboration with LOIC [4] was used to create bots for injecting VSI-DDoS attacks towards deployed services. We collect data by considering two main scenarios with and without VSI-DDoS attacks simulating on and off periods across multiple periods using *R-RMON* tool. The R-RMON is a remote resource monitoring tool developed by us to monitor systems, application resources, and services. The scenarios are known as UVSI-DDoS-I and UVSI-DDoS-II, which are explained below.

- *UVSI-DDoS-I*: This scenario is designed to inject vertical VSI-DDoS attacks by targeting deployed web services using synchronized bots. We consider $\beta = 40$ milliseconds as common burst time for each interval with 5000 HTTP requests. Here, we scaled the attack vertically to increase the intensity of attacks in each interval to degrade the QoS of legitimate users.
- *UVSI-DDoS-II*: In this scenario, we set $\beta = 100$ milliseconds with 2000 HTTP requests for each interval. We scaled the attack horizontally with multiple bots to impact longer burst time with the same number of requests that degrade the QoS of legitimate users.

Further, we set $\Delta = 2$ seconds and data collected for 2 hours for each scenario from three levels in the testbed, including physical, virtual, and applications. It is worth mentioning that we employed normal load generator Locust[5] with 1000 users to ensure having enough legitimate users and experience QoS degradation. We chose fixed and random starting points of attacks for both scenarios.

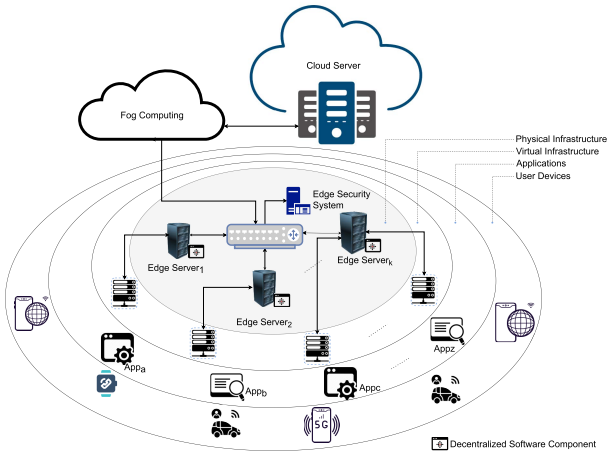


FIGURE 6. Illustration of UVSI-DDoS testbed architecture.

2) BENCHMARK DATASETS

We used two benchmark datasets for evaluating our proposed methods, namely, CIC-DDoS2019 [28] and UNSW-NB15 [29] due to the non-availability of benchmark datasets for VSI-DDoS. CIC-DDoS2019 is a realistic dataset that collects through background traffic using B-Profile System [30]. The dataset simulates and collects the behaviour of multiple users with protocols such as HTTP, SSH and having 18 variations of DDoS attacks for training and testing. UNSW-NB15 dataset is composed of real-time scenarios that combine modern normal activities and synthetic contemporary attack behaviour, collected through IXIA PerfectStrom[6] tool.

B. DATA PROCESSING

Before feeding data into the learning models, we preprocess both testbed and benchmark datasets by converting, normalization, filling missing values, extracting relevant features, and time transformation.

For both UVSI-DDoS datasets, we extract relevant features and processes across the training and testing set. The UVSI-DDoS-I dataset consists of 38847 attacks and 105040 legitimate instances with 39 features. The UVSI-DDoS-II dataset contains 37875 attacks and 27620 legitimate instances with 39 features.

On the other hand, CIC-DDoS2019 has 87 features across training and testing data and has 226437 attack and 112731 legitimate instances. UNSW-NB15 dataset consists of 9 attacks and 49 extracted features. Details of datasets are also summarized in Table 2.

[2]https://github.com/uillianluiz/RUBiS
[3]https://github.com/michaelmior/RUBBoS
[4]https://github.com/NewEraCracker/LOIC
[5]https://locust.io/
[6]https://research.unsw.edu.au/projects/unsw-nb15-dataset

TABLE 2. Details of testbed and benchmark datasets.

Datasets	Instances(#*)	Attacks(#)	Features(#)
UVSI-DDoS-I	143887	38847	39
UVSI-DDoS-II	65495	37875	39
UNSW-NB15	257673	164673	40
CICDDoS2019	339168	226437	81

∗ Number of.

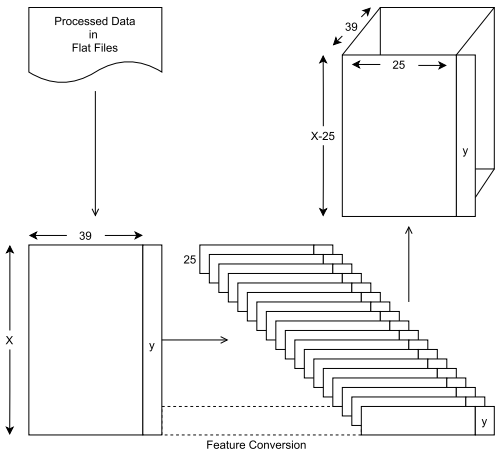


FIGURE 7. Illustration of window-based time transformation.

1) WINDOW-BASED TIME TRANSFORMATION

After obtaining relevant features and standardization, we employ the window-based time transformation to embed temporal information for both datasets. For x number of data instances, the matrix looks like $x \times 39$ with 39 features. Each instance contains a label(y) at the end. This matrix is sampled with a continuous window size of 25 resulting in $x - 25$ instances of block size $(25, 39)$. The assigned label for each block belongs to 25th element (the last one in the block); this way, the model can learn both short and long-term temporal patterns. Finally, a three-dimensional matrix of size $(x - 25, 25, 39)$ is obtained and fed as input to learning models. The steps to explain how window-based time transformation works are given in *Algorithm 2*.

Algorithm 2 Time Transformation

Require: Data instances with temporal order
Ensure: Transformed instances based on window size (w)
1: l = number of instances in X
2: w = window size (i.e., $w = 25$)
3: **for** iteration $i = \{1, 2, \ldots, l - w\}$ **do**
4: $\quad i_{th}$ element of X'_k for $k = \{i_{th}, \ldots, (i + w - 1)_{th}\}$ (k instances from X)
5: $\quad i_{th}$ label in X' = label for $(i + w)_{th}$ instance in X
6: **end for**
7: return X'

C. RESULTS AND ANALYSIS

The evaluation metrics include Area Under the Receiver Operating Characteristic Curve (AUC) [31], precision, recall, and accuracy to establish the model's capability for detecting VSI-DDoS attacks. The occurrences of attack class are rarer than the legitimate class, leading to class imbalance problems and vice versa depending on the time data were collected. Hence, we employ AUC as a validation measure to alleviate this problem.

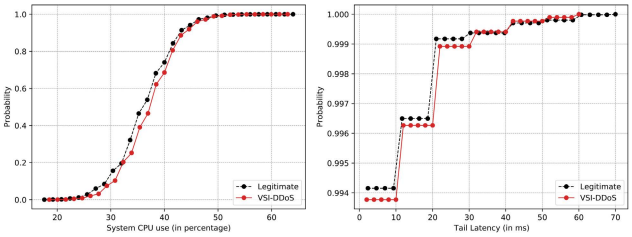


FIGURE 8. Cumulative density analysis for UVSI-DDoS-I testbed data in presence and absence of attacks.

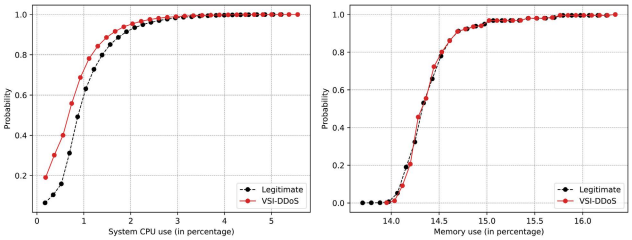
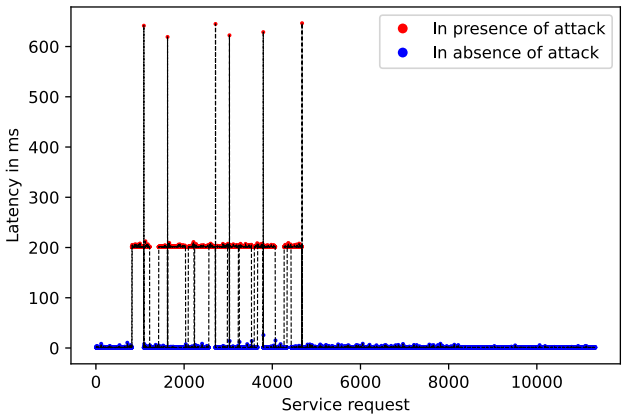


FIGURE 9. Cumulative density analysis for UVSI-DDoS-II testbed data in presence and absence of attacks.

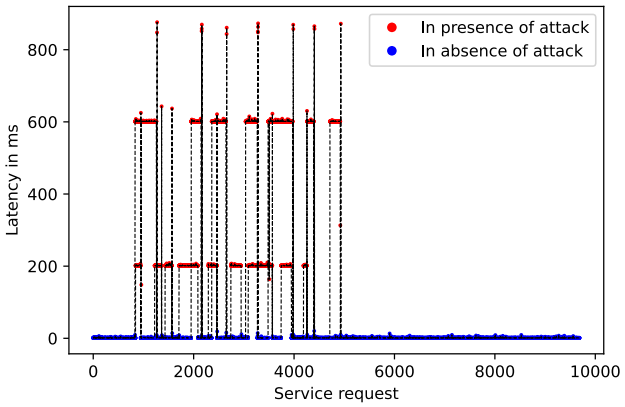
We begin our experiments with the characterization of data using cumulative density analysis for CPU utilization and tail latency in the UVSI-DDoS-I testbed dataset as shown in Figure 8. Figure 9 shows the same for CPU utilization and memory usage in the UVSI-DDoS-II testbed dataset. The cumulative difference between legitimate and attack is very close (seen in the Figures), increasing detection difficulty. Figure 10 shows the latency variation of HTTP requests in the presence and absence of VSI-DDoS attacks within the UVSI-DDoS-I dataset. Under normal traffic conditions, latency remains very close to 0. However, during the attack period, it peaks between 200 ms to 800 ms. We used the Keras library with the Tensorflow backend for implementing the proposed VSI-TN and VSI-RTN models.

1) VSI-TN

We begin with the UVSI-DDoS-I dataset for assessing models with time-representation layers that achieve significant model performance in detecting VSI-DDoS and iterate for other datasets. The hyper-parameters of each model are tuned with a grid search mechanism to obtain optimal model performance. Based on these, we achieve the best results with a sliding window size of 25 instances, 12 attention heads, 10 epochs, and a batch size of 32 for the VSI-TN model. The dropout value sets 0.1 and employs a global average pooling for the encoder layer to prevent model overfitting. ADAM [32] optimizer was used for our experiments with



(a) Latency plot of train set 1 and 2



(b) Latency plot of train set 3 and 4

FIGURE 10. Latency analysis for UVSI-DDoS-I testbed in presence and absence of attacks.

TABLE 3. Hyperparameters of *VSI-TN* for UVSI-DDoS-I dataset. In case of UVSI-DDoS-II dataset size of query, key and value were reduced to 128 and number of attention heads was set to 6; reduced size of the neural net eliminated over fitting issue caused by relatively small sized UVSI-DDoS-II dataset.

Parameter Name	Symbol	Value
Size of Query	Q_d	256
Size of Key	K_d	256
Size of Value	V_d	256
Attention Heads	H_T	12
Learning Rate	α	0.001
Batch Size	B_T	32
Total Epoch	epochs	10
Validation Split	validation_split	0.2

'binary-crossentropy' as loss function and sigmoid as activation function to obtain an accurate and stable model. The validation split 0.2 achieves optimal model accuracy. The model's hyperparameters are given in Table 3.

2) VSI-RTN

Transformer-based reinforcement learning assesses with limited labelled data to alleviate the model stability problem. Limited labelled data from a security system is typical because we need expert efforts to label them. After training them on different data sizes to confirm the superior learning

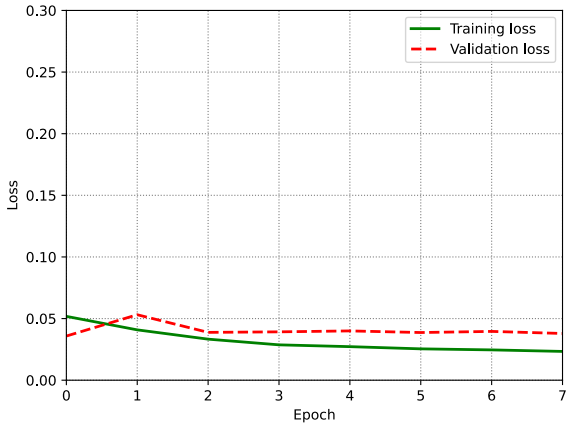
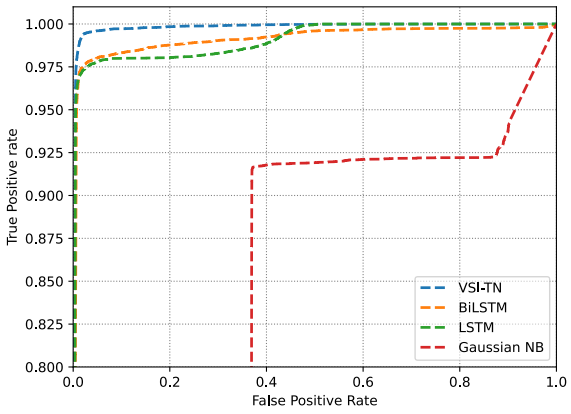
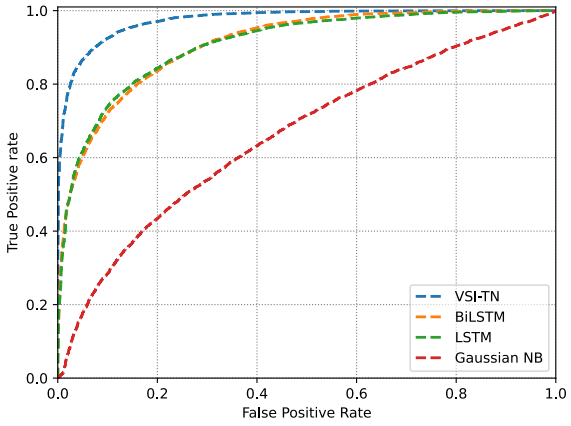


FIGURE 11. Learning behaviour analysis - training VSI-TN.



(a) UVSI-DDoS-I



(b) UVSI-DDoS-II

FIGURE 12. ROC analysis - VSI-TN with baseline models.

behaviour expected from reinforcement settings, we evaluate both VSI-TN and VSI-RTN models. These data sizes include 10%, 30%, 50%, 80% and 100% of both UVSI-DDoS-I and UVSI-DDoS-II datasets that maintain temporal consistency. VSI-RTN verifies this with learning stability under variable data size and data imbalance ratios (see Figure 16). We observe that VSI-TN does not achieve stable performance

TABLE 4. Hyperparameters of *VSI-RTN*.

Parameter Name	Symbol	Value
Training Iterations	T_n	2000
Sample Size	S_n	20
LSTM Units in DQN	L_T	200
Learning Rate	α	0.001
Discount Factor	γ	0.5
Epsilon Value	ϵ	0.05
Size of Q, K, V	d_q, d_k, d_v	156
Attention Heads	H_T	12
Batch Size	B_T	8

TABLE 5. Comparison among baseline models with proposed methods using UVSI-DDoS-I and UVSI-DDoS-II datasets.

	UVSI-DDoS-I		UVSI-DDoS-II	
Model Name	AUC	ACC	AUC	ACC
Bi-LSTM	98.26	98.67	90.88	85.17
LSTM	97.78	98.37	90.81	82.43
NB	96.03	97.04	66.20	60.01
VSI-TN (Ours)	98.62	**98.7**	91.18	**91.44**
VSI-RTN (Ours)	**99.25**	98.43	**95.88**	90.26

with varied data size and data imbalance ratios as given in Table 6.

Table 4 explains the hyperparameters for the VSI-RTN model. We perform our experiments with a random selection of 20 samples per training iteration in the reinforced learning process by maintaining the ratio of 15 : 5, where 15 samples were legitimate, and 5 samples were attacks.

The proposed approach carries extensive experiments with UVSI-DDoS testbed and CIC-DDoS datasets. Figure 11 shows training and validation loss curves for VSI-TN on the UVSI-DDoS dataset Scenario-I. Training loss can keep decreasing and eventually infuse. In contrast, validation loss fluctuates during the initial epochs and eventually immerses as well. Table 5 shows performance of the proposed models (i.e., VSI-TN and VSI-RTN) using both UVSI-DDoS dataset scenarios. We observe that our proposed models outperform baseline models with 0.9% to 3.2% more AUC score using the UVSI-DDoS dataset. The proposed models achieve superior performance by automatically uncovering the relations of attack patterns with temporal information via multi-head self-attentions. Moreover, with experience replay and the adaptation of new attacks via the reinforced learning process, the model could evolve and better detect future attacks. In particular, in the later experiment presented in Figure 16, the detection performance remains high across different attack scenarios and high variants of *unique attacks* coming into the systems.

a: LEARNING DYNAMICS OF VSI-RTN

To assess the learning stability and emulate real-time applications behaviour deployed on edge, we carried out extensive experiments to observe the performance correlation among different ratios of data for VSI-TN model within the reinforced learning process, i.e., VSI-RTN model. Figure 13

TABLE 6. Comparison across training data size for VSI-TN.

	UVSI-DDoS-I		UVSI-DDoS-II	
Data	AUC	ACC	AUC	ACC
10%	96.63	97.21	53.57	46.45
30%	95.04	96.42	73.72	72.80
50%	98.08	98.4	50.19	42.37
80%	97.14	97.67	89.82	90.24
100%	96.7	97.61	91.18	91.44

shows the decreasing losses on UVSI-DDoS-I data and Figure 14 shows accumulated rewards along training iterations for two models. One is the proposed VSI-RTN model, and another baseline model named RNN-RL, which uses LSTM instead of transformer in reinforcement settings. We have reported three different runs for loss and rewards for the VSI-RTN model, all showing a similar trend. Compared to the baseline model RNN-RL, VSI-RTN can achieve higher rewards in less amount of iterations. In terms of loss, the baseline model suffers several spikes during training. As shown in Figure 16, the proposed model achieves stable performance even fed varying amounts of data over time to the models. In Figure 16, *Unique Normal RL* and *Unique Attack RL* refer to normalized total unique normal and attack instances seen by DRL policy while training. *Unique Normal* and *Unique Attack* show the amount of unique normal and attack instances that were sent back to the analyst manager and, in turn, fed to the VSI-TN component for independent training. As we can see, there is a steady growth in the AUC score as we increase the data size. Also, we observe that increased data size leads to a steady increase of unique data received by the DRL policy. Unique data sent for training the VSI-TN that originally decreases and eventually remains the same without compromising performance. It illustrates that VSI-TN in reinforcement settings can be trained with fewer instances to make expected decisions for the model. This training strategy can eliminate learning instability and data imbalance problems and train the VSI-TN with only relevant training examples. The resultant model is cost-effective and efficient under those constraints.

Figure 15a shows that introducing a high penalty for delay classification implies maximum effect by progressing model training with lower delay in UVSI-DDoS-I data due to high non-similarity within data instances. The typical case is the fluctuation in performance at the beginning for wrong classifications; eventually, the model minimizes them. Throughout the training, correct classification dominates over requests for labelling from the rule base. Figure 15b shows the results of the same experiment on UVSI-DDoS-II data, where the training performance of the model fluctuates during initial steps and eventually minimizes the wrong classification. In addition, classification is requested from the analyst manager in case of previously unobserved data due to low confidence in the classifier by the RL agent. During this process, if similar to already sent data arrives, then the classification task is delayed because of the analyst manager's classification.

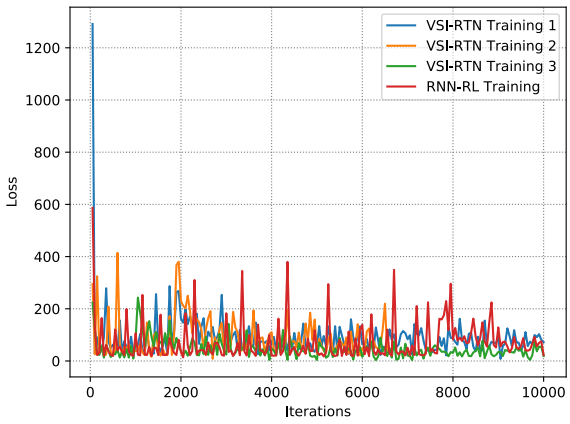


FIGURE 13. Convergence behaviors of VSI-RTN for illustrating model stability. The figure is best viewed in color.

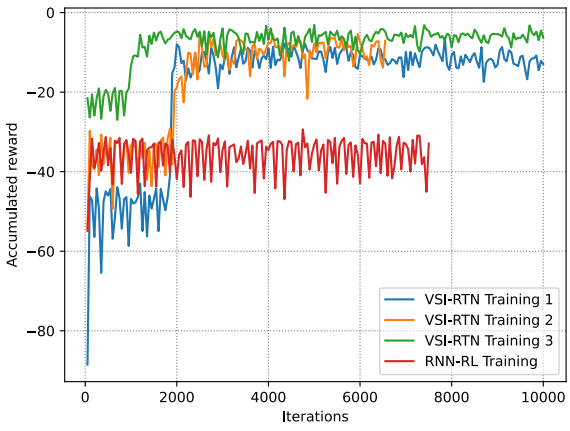


FIGURE 14. Accumulated rewards during training in VSI-RTN to illustrate model stability. The figure is best viewed in color. Here, RNN-RL is a baseline solution using LSTM with the same reinforced learning settings as VSI-TN.

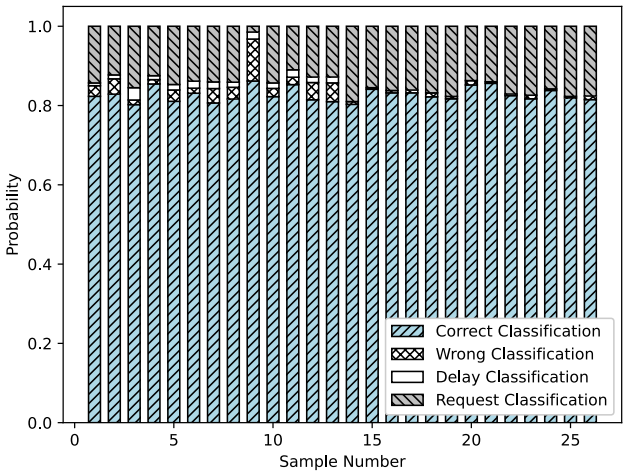
TABLE 7. Overall performance of our proposed models in comparison to other works using CIC-DDoS dataset.

Model(s)	Test set	AUC	ACC	Precision	Recall
DDoSNet [15]	0.12%	-	-	99	99
GRU-SDN [33]	-	99.74	-	99.83	99.79
LSTM (baseline)	3.44%	61.46	99.2	99.2	99.9
Bi-LSTM (baseline)	3.44%	50	98.97	98.97	1
VSI-TN (Ours)	3.44%	-	99.69	99.8	99.88
VSI-RTN (Ours)	3.44%	98.56	99.8	-	-
VSI-RTN (Ours)	100%	98.96	98.74	1	98.73

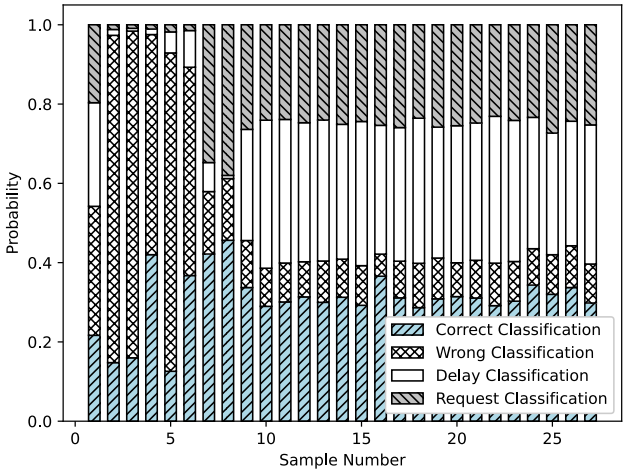
In such a way, the model does not need to repeatedly send a request for classification to the analyst manager. Instead, the model can learn efficiently and use the same manual labour.

D. COMPARISON WITH EXISTING METHODS

To verify our proposed models compared to state-of-the-art methods, we consider the CIC-DDoS2019 and UNSW-NB15 datasets for the non-availability of the VSI-DDoS benchmark datasets. Table 7 shows the performance of our proposed models, where VSI-TN and VSI-RTN outperform existing



(a) UVSI-DDoS-I



(b) UVSI-DDoS-II

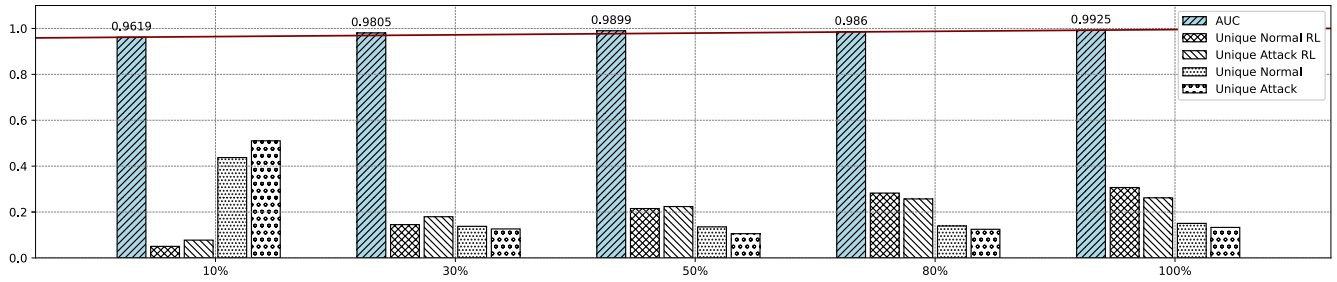
FIGURE 15. Probabilities of actions for each sample during training with UVSI-DDoS-I and UVSI-DDoS-II datasets.

TABLE 8. Overall performance of VSI-TN in comparison to other works using UNSW-NB15 dataset.

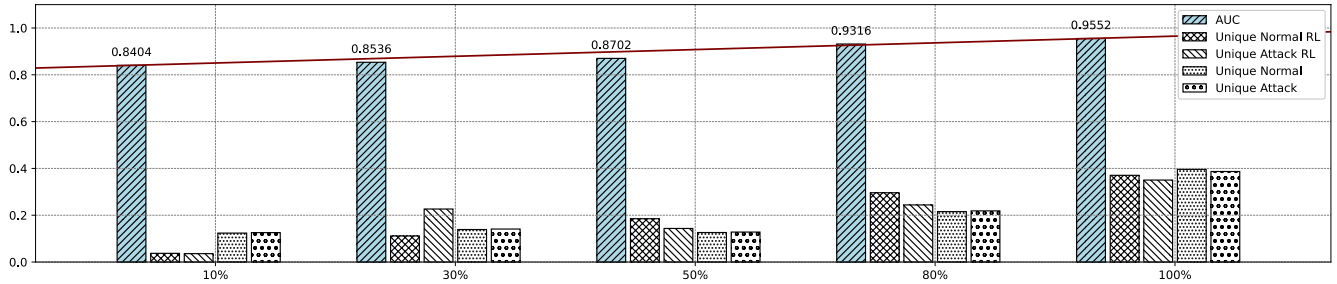
Exp-Name	AUC	ACC	Precision	Recall
LSTM-Att [2]	96.64	96.6	97.57	95.9
LSTM (baseline)	96.82	85.68	79.38	**99.95**
Bi-LSTM (baseline)	96.67	93.13	96.15	93.66
VSI-TN (Ours)	**98.19**	**98.26**	**97.99**	98.87

works. Our models differ from existing methods: (i) experiments made for a maximum amount of test sets 19 million instances, (ii) developed transformer-induced multi-task deep reinforcement learning, (iii) dynamic adoption of attack behaviour, and (iv) mitigate service availability and QoS/QoE problems in edge clouds, outperforms in compare to existing methods such as DNN-based baseline models, DDoSNet [15], and GRU-SDN [33].

Results on UNSW-NB15 dataset are reported in Table 8. LSTM-Att [2] offers AUC of 96.64% and 96.6% accuracy, whereas baseline LSTM shows 96.82% AUC and



(a) UVSI-DDoS-I



(b) UVSI-DDoS-II

FIGURE 16. **Performance variants of VSI-TN in the VSI-RTN pipeline in correlation with different amounts of data received in the reinforced learning process. *Unique Normal RL* and *Unique Attack RL* represent total unique normal and attack instances seen by the DRL Policy of the VSI-RTN model while training. *Unique Normal* and *Unique Attack* indicate the number of unique normal and attack instances seen by the VSI-TN component of VSI-RTN. The counts are normalized across all the percentages for every type of count to scale between 0 and 1.**

TABLE 9. **Training and testing time analysis for the proposed and the baseline models.**

	Training-time (sec)				Testing-time (per instance, μsec)			
Dataset	LSTM	BiLSTM	**VSI-TN** (Ours)	**VSI-RTN** (Ours)	LSTM	BiLSTM	**VSI-TN** (Ours)	**VSI-RTN** (Ours)
UVSI-DDoS-I	2263.2	3444.3	2527.3	1316.51	44.15	60.58	76.86	79.23
UVSI-DDoS-II	591.8	1057.8	454.8	437.9	16.83	37.63	20.49	7.95
UNSW-NB15	2890.7	4887.2	4165.4	1341.55	63.04	62.47	67.01	79.84
CIC-DDoS2019	6143.2	9875.4	9272.8	1358.16	66.78	64.60	68.75	83.64

lower accuracy of 85.68% compared to LSTM-Att [2]. Our proposed model achieves higher accuracy of 98.26% with an improvement of 1.66% to 12.57% compared to other methods.

Training time and testing time analysis across our proposed and other deep learning-based methods is given in Table 9. From the table, once data size increases, the training time for all methods is also growing, but testing time per instance remains closer or similar. In the case of UVSI-DDoS-I data, VSI-TN has a testing time of 76.86μSec slightly higher than LSTM with 44.15μSec and BiLSTM with 60.58μSec. For UVSI-DDoS-II data, VSI-TN has a testing time of 20.49μSec less than BiLSTM with 37.63μSec and slightly higher than LSTM with 16.83μSec. In addition, UNSW-NB15 testing times for VSI-TN, BiLSTM and LSTM are pretty close to one another with 67.01μSec, 62.47μSec and 63.04μSec, respectively. Similarly, in the case of CIC-DDoS2019, VSI- TN requires 68.75μSec per instance, BiLSTM requires 64.60μSec, and LSTM requires 66.78μSec, respectively. For the VSI-RTN model, training times are for 10000 iterations in each dataset, requires 1316.51 *Sec* for UVSI-DDoS-I, 437.9 *Sec* for UVSI-DDoS-II, 1341.55 *Sec* for UNSW-NB15, and 1358.16 *Sec* for CIC-DDoS2019, respectively. Testing time per instance for VSI-RTN remains relatively close to one other despite increased data size. This analysis shows that the proposed VSI-DDoS detection models perform well on the microsecond scale, implying that models can improve service availability by controlling these attacks on the edge at a very early stage.

The ROC curve of VSI-TN, along with other baseline methods for UVSI-DDoS-I and UVSI-DDoS-II test data, are shown in Figure 12. ROC curve shows the model's ability to differentiate between the target classes in True Positive Rate (TPR) and False Positive Rate (FPR). The proposed VSI-TN outperforms BiLSTM, LSTM, and Gaussian NB, which also reflects from Area Under the Curve (AUC) score reported in Table 5. As a result, VSI-TN achieves stable learning ability, adapts to dynamic and temporal data behaviour, and manages data imbalance problems when detecting VSI-DDoS attacks in edge clouds.

V. CONCLUSION AND FUTURE WORK

This paper demonstrated that VSI-DDoS attacks primarily target time-sensitive services deployed on edge to degrade

users' QoS/QoE. Hence, we developed a reinforced transformer learning-based approach to detect such attacks that mitigate the problems of service non-availability and dirty users' QoS/QoE experience in edge clouds. The integration of transformer and deep reinforcement learning makes the model more intelligent and effective as it uses an encoding layer for compact feature representation of raw data. Our proposed model has multiple features, such as adopting dynamic attack behaviour, learning stability, and a rule-base for smoother decisions, outperforming testbed and benchmark datasets under adverse conditions. Multihead attention of transformer-based models helps to analyze contextual information in time-series data, resulting in better attack detection capability. Our comprehensive experimental evaluation shows that the proposed approach outperforms state-of-the-art methods and ensures model stability, efficiency, and robustness for detecting VSI-DDoS attacks at an early stage. Moreover, the time analysis shows the feasibility of using our proposed model in the early detection of VSI-DDoS attacks in edge clouds with testing time in microseconds.

An extension of this work is undergoing by deploying diverse mission-critical edge applications in 6G testbed to handle users' QoS/QoE problems.

REFERENCES

[1] Y. Wang, W. Wang, J. Zhang, J. Jiang, and K. Chen, "Bridging the edge-cloud barrier for real-time advanced vision analytics," in *Proc. 11th USENIX Workshop Hot Topics Cloud Comput.* Renton, WA, USA: USENIX Assoc., Jul. 2019, pp. 1–7. [Online]. Available: https://www.usenix.org/conference/hotcloud19/presentation/wang

[2] J. Forough, M. Bhuyan, and E. Elmroth, "Detection of VSI-DDoS attacks on the edge: A sequential modeling approach," in *Proc. 16th Int. Conf. Availability, Rel. Secur.*, Aug. 2021, pp. 1–10.

[3] Z. Tian, C. Luo, J. Qiu, X. Du, and M. Guizani, "A distributed deep learning system for web attack detection on edge devices," *IEEE Trans. Ind. Informat.*, vol. 16, no. 3, pp. 1963–1971, Mar. 2020.

[4] D. V. Medhane, A. K. Sangaiah, M. S. Hossain, G. Muhammad, and J. Wang, "Blockchain-enabled distributed security framework for next-generation IoT: An edge cloud and software-defined network-integrated approach," *IEEE Internet Things J.*, vol. 7, no. 7, pp. 6143–6149, Jul. 2020.

[5] S. Yeom, C. Choi, and K. Kim, "Source-side DoS attack detection with LSTM and seasonality embedding," in *Proc. 36th Annu. ACM Symp. Appl. Comput.*, Mar. 2021, pp. 1130–1137.

[6] R. Rasti, M. Murthy, N. Weaver, and V. Paxson, "Temporal lensing and its application in pulsing denial-of-service attacks," in *Proc. IEEE Symp. Secur. Privacy*, May 2015, pp. 187–198.

[7] M. H. Bhuyan, H. J. Kashyap, D. K. Bhattacharyya, and J. K. Kalita, "Detecting distributed denial of service attacks: Methods, tools and future directions," *Comput. J.*, vol. 57, no. 4, pp. 537–556, 2013.

[8] H. Shan, Q. Wang, and Q. Yan, "Very short intermittent DDoS attacks in an unsaturated system," in *Proc. Int. Conf. Secur. Privacy Commun. Syst.* Cham, Switzerland: Springer, 2017, pp. 45–66.

[9] J. Moura and D. Hutchison, "Resilience enhancement at edge cloud systems," *IEEE Access*, vol. 10, pp. 45190–45206, 2022.

[10] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," 2018, *arXiv:1810.04805*.

[11] G. Zerveas, S. Jayaraman, D. Patel, A. Bhamidipaty, and C. Eickhoff, "A transformer-based framework for multivariate time series representation learning," in *Proc. 27th ACM SIGKDD Conf. Knowl. Discovery Data Mining*, Aug. 2021, pp. 2114–2124.

[12] E. Parisotto and R. Salakhutdinov, "Efficient transformers in reinforcement learning using actor-learner distillation," 2021, *arXiv:2104.01655*.

[13] A. Puzanov and K. Cohen, "Deep reinforcement one-shot learning for artificially intelligent classification systems," 2018, *arXiv:1808.01527*.

[14] F. S. D. L. Filho, F. A. F. Silveira, A. de Medeiros Brito Junior, G. Vargas-Solar, and L. F. Silveira, "Smart detection: An online approach for DoS/DDoS attack detection using machine learning," *Secur. Commun. Netw.*, vol. 2019, pp. 1–15, Oct. 2019.

[15] S. U. Rehman, M. Khaliq, S. I. Imtiaz, A. Rasool, M. Shafiq, A. R. Javed, Z. Jalil, and A. K. Bashir, "DIDDOS: An approach for detection and identification of distributed denial of service (DDoS) cyberattacks using gated recurrent units (GRU)," *Future Gener. Comput. Syst.*, vol. 118, pp. 453–466, May 2021.

[16] A. Saied, R. E. Overill, and T. Radzik, "Detection of known and unknown DDoS attacks using artificial neural networks," *Neurocomputing*, vol. 172, pp. 385–393, Jan. 2016.

[17] Z. Liu, X. Yin, and Y. Hu, "CPSS LR-DDoS detection and defense in edge computing utilizing DCNN Q-Learning," *IEEE Access*, vol. 8, pp. 42120–42130, 2020.

[18] J. van Roosmalen, H. Vranken, and M. van Eekelen, "Applying deep learning on packet flows for botnet detection," in *Proc. 33rd Annu. ACM Symp. Appl. Comput.*, Apr. 2018, pp. 1629–1636.

[19] Z. Wu, H. Zhang, P. Wang, and Z. Sun, "RTIDS: A robust transformer-based approach for intrusion detection system," *IEEE Access*, vol. 10, pp. 64375–64387, 2022.

[20] S. Yeom, C. Choi, and K. Kim, "LSTM-based collaborative source-side DDoS attack detection," *IEEE Access*, vol. 10, pp. 44033–44045, 2022.

[21] U. Unal and H. Dag, "AnomalyAdapters: Parameter-efficient multi-anomaly task detection," *IEEE Access*, vol. 10, pp. 5635–5646, 2022.

[22] Y. Huang, L. Huang, and Q. Zhu, "Reinforcement learning for feedback-enabled cyber resilience," 2021, *arXiv:2107.00783*.

[23] J. Park, D. Nyang, and A. Mohaisen, "Timing is almost everything: Realistic evaluation of the very short intermittent DDoS attacks," in *Proc. 16th Annu. Conf. Privacy, Secur. Trust (PST)*, Aug. 2018, pp. 1–10.

[24] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2017, *arXiv:1706.03762*.

[25] S. M. Kazemi, R. Goel, S. Eghbali, J. Ramanan, J. Sahota, S. Thakur, S. Wu, C. Smyth, P. Poupart, and M. Brubaker, "Time2Vec: Learning a vector representation of time," 2019, *arXiv:1907.05321*.

[26] M. Woodward and C. Finn, "Active one-shot learning," 2017, *arXiv:1702.06559*.

[27] K. Landfors, "Detection and resolution of VSI-DDoS attacks for containerized clouds," M.S. thesis, Dept. Comput. Sci., Umeå Univ., Umeå, Sweden, 2019.

[28] (2021). *DDoS Evaluation Dataset (CIC-DDoS2019)*. Accessed: Jun. 1, 2021. [Online]. Available: https://www.unb.ca/cic/datasets/ddos-2019.html

[29] N. Moustafa and J. Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," in *Proc. Mil. Commun. Inf. Syst. Conf. (MilCIS)*, Nov. 2015, pp. 1–6.

[30] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *Proc. 4th Int. Conf. Inf. Syst. Secur. Privacy*, vol. 1, Jan. 2018, pp. 108–116.

[31] D.-T. Le, X.-S. Vu, N.-D. To, H.-Q. Nguyen, T.-T. Nguyen, L. Le, A.-T. Nguyen, M.-D. Hoang, N. Le, H. Nguyen, and H. D. Nguyen, "ReINTEL: A multimodal data challenge for responsible information identification on social network sites," 2020*arXiv:2012.08895*.

[32] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.

[33] M. V. O. Assis, L. F. Carvalho, J. Lloret, and M. L. Proença, "A GRU deep learning system against attacks in software defined networks," *J. Netw. Comput. Appl.*, vol. 177, Mar. 2021, Art. no. 102942. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1084804520304008

ADIL BIN BHUTTO received the bachelor's degree in computer science and engineering from Tezpur University, Assam, India, in 2021.

The topic of his bachelor's thesis was detection of VSI-DDoS attack using attention models in edge clouds. He is currently a Visiting Researcher at the Autonomous Distributed Systems Laboratory, Umeå University, Sweden. His research interests include machine learning, distributed systems, and cyber security.

XUAN SON VU (Member, IEEE) received the B.S. degree in information system from the University of Engineering and Technology, VNU, Hanoi, Vietnam, in 2011, the M.Sc. degree from Kyungpook National University, South Korea, in 2014, and the Ph.D. degree in computer science from Umeå University, Sweden, in 2020, with focus in privacy-preserving machine learning with big data.

Before joining Umeå University, from 2015 to 2016, he was a full-time member of the UKP Laboratory, TU Darmstadt, Germany. He is currently a Postdoctoral Fellow at Umeå University, where he focuses on research in robust machine learning. His research interests include knowledge—both acquiring knowledge from multimodal data, and using structured knowledge to power downstream applications. For the former, he has worked on ontology, information retrieval and natural language processing. For the latter, multimodal knowledge learning is one of the topics he enjoys researching the most. His awards and honors include the third place for best paper awards at CICLING 2019, the Best Student Paper Award, and the Best Inquisitive Mind Award at CICLING 2018. He also received a Best Paper Award at the 40th Conference of the Korea Information Processing Society, in 2014.

ERIK ELMROTH is currently a Professor in computing science at Umeå University. He has been the Head and the Deputy Head of the Department of Computing Science for 13 years and the Deputy Director of the National Supercomputer Center for another 13 years. He has established the Umeå University research on distributed systems (http://www.cloudresearch.org). His experience from management and executive groups in large-scale research projects include highlights, such as the 550 m euro Wallenberg AI, Autonomous Systems and Software Program and the Strategic Research Area eSSENCE. He has been a member of the Swedish Research Council's Committee for Research Infrastructure and the Chair of its expert panel on eScience, and the Chair of Board of the Swedish National Infrastructure for Computing. He has developed two international research strategies for the Nordic Council of Ministers. His international experiences include a year at NERSC, Lawrence Berkeley National Laboratory, University of California at Berkeley, Berkeley, and one semester at the Massachusetts Institute of Technology (MIT), Cambridge, MA, USA. He is a Lifetime Member of the Swedish Royal Academy of Engineering Sciences and the Vice Chair of its Division for Information Technology.

WEE PENG TAY (Senior Member, IEEE) received the B.S. degree in electrical engineering and mathematics and the M.S. degree in electrical engineering from Stanford University, Stanford, CA, USA, in 2002, and the Ph.D. degree in electrical engineering and computer science from the Massachusetts Institute of Technology, Cambridge, MA, USA, in 2008. He is currently an Associate Professor with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore.

MONOWAR BHUYAN (Member, IEEE) received the Ph.D. degree in computer science and engineering from Tezpur University, Assam, India, in 2014.

He has been an Assistant Professor with the Department of Computing Science, Umeå University, Sweden, since January 2020, and one of the research group leaders at Autonomous Distributed Systems Laboratory. Before this, he worked with the Nara Institute of Science and Technology, Japan, Umeå University, Assam Kaziranga University, India, and Tezpur University, India, from January 2009 to December 2019, in different levels from a Junior Researcher to an Associate Professor. He has published over 50 papers in the leading international journals and conference proceedings and has written a book with Springer. His experience to lead/co-lead research projects is over 20 MSEK, including national and EU grants. His research interests include machine learning, anomaly detection, security and privacy, and distributed systems.