Svedberg Open
DISSEMINATION OF KNOWLEDGE

International Journal of Artificial Intelligence and Machine Learning

Publisher's Home Page: https://www.svedbergopen.com/

Research Paper Open Access

Intelligent Anomaly Detection in Campus Networks using Lightweight Vector Quantized Conditional Weighted Wasserstein Autoencoder and Optimized Bayesian Temporal Convolutional Network

Ms.D.Sharmila[1] , Dr.S.Uma[2]

[1]Ph.D. Research Scholar, Department of Computer Science, CMS College of Science and Commerce, Coimbatore,
Email: dharmasharmi@gmail.com
[1]Assistant Professor , Department of Computer Science, KG College of Arts and Science, Coimbatore,
Email: dharmasharmi@gmail.com
[2]Professor and Head, Department of Computer Science, Dr.N.G.P Arts and Science College, Coimbatore,
Email: s.umacbe9@gmail.com

Abstract

Campus Network (CN) is a private network that provides centralized management, high-speed connectivity and secured resource sharing in larger private campuses. However, even the CN is vulnerable to security threats such as system failures, data leaks, phishing and unauthorized access due to their open and diverse environments. Recent studies have employed Machine Learning (ML) and Deep Learning (DL) algorithms for detection of complex and unidentified threats in various networks with significant success. Therefore, the DL-based methods are recommended, but the challenges of complexity, inefficient computation, high false rates and generalization to changing patterns must be handled effectively for improved anomaly detection in CN. This paper addresses these limitations by developing a hybrid model called Lightweight Vector Quantized Conditional Weighted Wasserstein Autoencoder and Optimized Bayesian Temporal Convolutional Network (LVQ-CWWAE-OBTCN).In this lightweight model, VQ-CWWAE compresses the high-dimensional traffic data into a latent representation for extracting vital features. It integrates Vector Quantization and Autoencoder with a conditional weighted loss function to extract the infrequent anomalous patterns and Wasserstein distance function for smooth clustering of the latent space representations. These latent representation features are utilized by OBTCN model consisting of Temporal Convolutional Network with the Bayesian network architecture for identifying the anomalies and alerting the CN server. Lyrebird Optimization Algorithm (LOA) is used for hyper-parameter tuning to ensure better training accuracy and computational efficiency.Evaluated on UNSW-NB15, CICIDS 2018, and SIMARGL2021 datasets, the LVQ-CWWAE-OBTCN achieved detection accuracies of 99.53%, 99.81% and 97.1%to quantify uncertainty with confidence intervals for anomaly predictions.

Keywords: Campus Network, Anomaly Detection, Lyrebird Optimization Algorithm, Wasserstein Autoencoder, Vector Quantization, Wasserstein distance regularization, Bayesian Temporal Convolutional Network.

1. Introduction

A Campus Network(CN) is a proprietary local area network (LAN) that spans across multiple buildings within a large campus with limited geographical area such as college, university, information technology parks, large government organizations, industries, manufacturing facilities or similar organizations [1]. This CN is designed to interconnect the devices and systems across the campusarea, enabling seamless communication, centralized management of services and effective resource sharing. While other networks such asWide Area Networks (WAN) and Data Center Networks (DCN) connect larger geographical locations and provide high-performance intra-server communications, they lack centralized management and limited internal traffic management [2]. CN infrastructure is different from the other networks such that it can cover multiple buildings in a large but pre-defined area limits.CN connects end users who are dispersed geographically than in a single LAN, but not

as scattered as they are in WAN. CN provide services to diverse groups in a college or university than the other networks and provide flexible integration of various devices such as laptops,Internet-of-Things (IoT) devices, laboratory equipment and other smart devices. These networks also improve the communication capability between the departments and centralized management servers to enhance the resource sharing and data exchanges [3]. CN increases the high-speed connectivity to ensure quality voice, video or data transmissions among the users which in turn ensures smoother utilization of the collaboration tools for video conference, shared drives and e-learning for student education, collaborative research and administration.

The innovative connection of CAN also causes possibility of security breaches. While the security threats in CN are not as larger or depth as in WAN or DCN, the open and diverse network environment is suitable for system failures, data leaks, phishing and unauthorized access [4]. The malicious attacks such asphishing attacks, denial-of-service (DoS) attacks and insider hacking can causedata breaches and downtime of critical services [5]. Traditional methods, such as rule-based firewalls and signature-based and heuristic-based intrusion detection systems, are used to manage these threats. Still, they lack adaptability and scalability and cannot identify unknown and evolving attack patterns [6].The ML and DL techniques overcome the drawbacks of conventional approachesby using historical data to identify abnormalities and gather hierarchical feature patterns to improve performance. The use of ML and DL techniques identifies the emerging attacks and enhances scalability. However, the ML and DL approaches are not interpretable, have large processing requirements that restrict deployment, cannot manage temporal dependencies in network traffic and struggle to adjust to dynamic network environments.

To address the drawbacks of the conventional ML and DL models, this research creates a hybrid lightweight model called Lightweight Vector Quantized Conditional Weighted Wasserstein Autoencoder and Optimized Bayesian Temporal Convolutional Network (LVQ-CWWAE-OBTCN). The raw traffic data is collected for processing and evaluation.After processing the raw traffic data into structured time series data, important features are extracted for effective data analysis. The LVQ-CWWAE module is used to get the features from the normalized data. The model incorporated vector quantization (VQ) for feature collection and mapping the high-dimensional data into the discrete latent representationsto minimize downstream processing. The model employed Wasserstein Distance Regularization (WDR) to improve the latent space representation and enhance the clustering of anomalous and normal behaviours while the conditional weighted loss (CWL) function prioritizes the infrequent anomaly patterns.The latent features are given to the BTCN classifier model, which uses Bayesian layers to assess prediction uncertainty and produce confidence intervals. The dilated and causal convolutions capture both short-term and long-term dependencies in network traffic. LOA is used to optimize the classifier hyperparameters, such as the dilation rate, kernel size, and Bayesian prior. The approach was created to improve campus networks' anomaly detection precision, effectiveness, and interpretability. The proposed LVQ-CWWAE-OBTCN model is evaluated using benchmark datasets for quantitative analysis. The remainder of the article is organized as: Section 2 describes recent methodologies for security threats in CN networks, Section 3 presents the proposed methodology, the results are discussed in section 4 and the paper is concluded in section 5.

2. Related Works

Awang et al. [7] developed a data mining (DM)method for detectingCN information security vulnerabilities. The data is analysed in an Anaconda environment, and in the learning process the track mapping patterns are used to detect patterns. The data is collected from an intrusion prevention system (IPS) for analysis. The technique identified the highest 388 attacks,on 29 May 2017. However, the increasing size of the CNdevelops complexity.Dimolianis et al. [8]proposed a signature pattern-based traffic classification and mitigation. The signatures patterns are extracted from the network traffic and classified by the multilayer perception (MLP) and random forest (RF). The signature reduction process decreases the malicious signature patterns. The model used a booster dataset and attained 100% true positive and negative rates for RF. However, the RF failed todetect attacks that deviated from the training attack pattern.Lyu et al. [9]developed a hierarchical data structure for detecting domain name system (DNS) attacks. This process includes three levels, host, subnet, and autonomous system (AS), to detect anomalies and behaviours at different levels. The dataset is collected from

university campuses and medium-sized research institutes as DNS traffic data for evaluation. The model obtained an accuracy of 99.76%, TN of 99.81%, TP of 97.21%, and AUC of 99.96%. However, the mixed dataset does not fully reflect realistic network traffic patterns.Liu et al. [10]developed a low-rate DDoS attack detection method (LDDM). The model incorporated a multi-dimensional structure for aggregating and compressing the network flow and a measurement technique based on the Daub 4 wavelet transform. The model was evaluated on five real low-rate DDoS attack datasets. The LDDM attained an accuracy of 97% for Dataset 1, 98% for Dataset 2, 98% for Dataset 3, 98% for Dataset 4 and 95% for Dataset 5. However, this model's boundary between the attack and normal traffic is unclear.

Lent et al. [11] proposed a gated recurrent unit (GRU) and fuzzy logic for network anomaly detection and mitigation. The data are applied to the GRU, and six NNs are trained to detect flow dimensions. The fuzzy membership function compares each detection result with actual traffic for classification. The model used two datasets for evaluation, the initial dataset with emulated traffic data created through the data communication and networking research and the CICDDoS2019 dataset. For dataset 1, the model attained a 99% precision, 98% recall and F1-score. For CICDDoS2019, the model achieved a 92% precision, 96% recall, and 94% F1-score. However, the threshold for detection may not be optimal for all attack types. AL-Ghamdi et al. [12] presented an AI-based cybersecurity intrusion detection algorithm for higher education institutions (AICID-HEI). The model used the improved differential evolution algorithm (IDEA) to select the features and bidirectional long short-term memory (BiLSTM) for prediction, and it was optimized using the Adam optimizer.The model was computed on a KDDCup99 dataset and attained 99.01%sensitivity, 99.54%specificity, accuracy of 99.01%, 98.82% F1-score, and kappa of 98.47%. Yet, the model increased computational cost due to BiLSTM and Adam optimizer.Raza et al. [13]developed a class probability random forest (CPRF) for detection. The CPRF model used the CICIDS2017 dataset to predict the class probabilities and develop a feature set. The developed feature set is applied to the logistic regression, RF, Gaussian naïve Bayes, and decision tree for detection. The RF model attained 99% accuracy, and precision, 100% F1 score, and recall. However, the generated feature set is not universally best for all types of network attacks. Rong et al. [14]presented an N-gram combined character-based deep network (NCBDN) model that incorporated a domain generation algorithm (DGA) and gated recurrent neural network (GRNN) for detection. The CNN model was used for malicious domain name detection, and the NCBDN model was used for intrusion detection. The model was computed using the two datasets and obtained98.73% accuracy and 98.91% recall for ISCX2012 dataset and 96.91% accuracy and 99.55%recall for the CICIDS2017 datasets. However, the model struggles to collect threat data in large-scale network traffic.

Li et al. [15] presented anindex grading criteria for network security prevention based on exposure and uncertainty using the Markov-network security model. The model uses normalized autocorrelation coefficients of order weight to learn the CN path security and student privacy prevention. For intrusion detection systems, the model attained security of 51% and 63 widest usage. However, the network conditions and security threats can vary significantly, making the model complex.Qazi et al. [16] developed a hybrid DL-based network intrusion detection system (HDLNIDS). This HDLNIDS method uses CNN to extract the local features and RNN for classification. The model was evaluated on the CICIDS 2018 dataset and attained a precision of 98.63%, recall of 99.14%, F1-score of 99.03%, and accuracy of 98.90%. However, the model faces difficulties with zero-day attacks.Shaorong et al. [17] proposed an intrusion prediction algorithm for designing the network security protection system. The proposed intrusion detection system accurately identifies the non-compliant, abnormal, and unauthorized content in the system and configuration. The model was evaluated on a cloud database and attained a server response time of less than 0.6s when there is no attack and less than 0.6s when starting the IDS system. The proposed method faces scalability issues in the growing campus network. Doriguzzi-Corin et al. [18] proposed a model using P4 programmable data planes P4DDLe for centralized intrusion prediction. In the P4DDLe algorithm, the LUCID's parser is utilized to extract the packet-level features, and it is employedto the control plan for building the array and feature normalization. The algorithm was simulated using CICDDoS2019 and attained an average flow length of 12.5% and a DDoS flow percentage of 73.82%. However, the model prediction accuracy was reduced when working in an unpredictable traffic environment. Kumar et al. [19] presented a deep residual convolutional neural network (DCRNN) for intrusion prediction. The binary grasshopper optimization model is applied for feature selection. The chosen features are applied to the DCRNN

model for classification. Then, the improved gazelle optimization algorithm (IGOA) was used to optimize the DCRNN hyperparameters to enhance the detection accuracy. The model was evaluated on three datasets and attained accuracies of 99.16% for UNSW-NB-15, 99.56% for CISDDoS2019, and 99.37% for CICIDS2017. However, the proposed model faces challenges with overfitting and sparsity issues.

Chen et al. [20] presented a network intrusion detection algorithm based on feature segmentation and cascaded random forest (FS-CRF). The features are segmented using the FS module, and the CRF model classifies the segmented features. The algorithm simulated on CICIDS2018 and attained a request matching rate of 93% and security defense effectiveness of 96.5%. However, the model did not support an extensive, scalable resource search mechanism. Mallikarjun et al. [21]developed an ML-based network intrusion and anomaly detection system (NIADS). Based on previous studies, the dataset was generated using multiple private campuses' 5G networks. The generated dataset is applied to the LSTM and autoencoders for anomaly detection. The LSTM model attained a97% precision, 95% recall, and 96% F1-score. However, the LSTM models are still vulnerable to zero-day attacks. Suethanuwong et al. [22] proposed a gateway-controlled address resolution protocol (GC-ARP) for cache poisoning attack prevention in router-on-a-stick (RoaS) networks using MikroTik networking tools. The incoming ARP requests from the host in all VLAN are handled by applying the GCA model. This ensures hosts gets legitimate ARP responses from the router. This process prevents ARPfrom spoofing packets from malicious attackers. The model attained an average processing time of 145ms. However, the model needs more computational resources.Lin et al. [23] proposed an edge-based GraphSAGE with residual connections called E-GRACL. Based on the edge-enhanced GraphSAGE, the model uses an improved graph encoder, which contains an attention and gating mechanism for extracting the features. Then, the graph contrastive learning method was used to enhance the classification performance. The proposed model uses NFBoTIoT, NFToNIoT-v2, and CSECICIDS2018 datasets for evaluation. The model attained accuracies of 0.99 for NFBoTIoT, 0.96 for NFToNIoT-v2, and 0.99 for CSECICIDS2018. However, this method includes multiple techniques which can be computationally overhead.Bamber et al. [24] presented a CNN-LSTM for intrusion prediction. The features are selected using a decision tree classifier. The features arethen applied to the CNN-LSTM for classification. The model was computed on the NSL-KDD and attained an accuracy of 95%, recall of 89%, and F1-score of 94%. However, the complexity of the model leads to delays in processing.

3. Methodology

The proposed LVQ-CWWAE-OBTCN model is developed for anomaly detection in campus networks by combining feature representation techniques, sequential analysis, and uncertainty estimation.

3.1. Network design

The network design includes the system to detect and respond to data stream (DS) threats that originate and pass through the campus network. Packet capture tools, including Wireshark and tcpdump, intercept data packets traversing the network to monitor and collect real-time data from the campus network. These tools capture raw networks with the payloads, and these packets serve as the input. This tool captures inbound and outbound traffic, including HTTP, HTTPS, and FTP protocols. The detection process includes information flow data, content analysis of network packet patterns, and response to abnormal behavior methods. The model also uses switches with port mirroring to replicate traffic to sensors without affecting network performance. The collected data is utilized to make judgments and decisions. The model collected detailed code, information, and alerts about abnormalities to remove hidden threats.

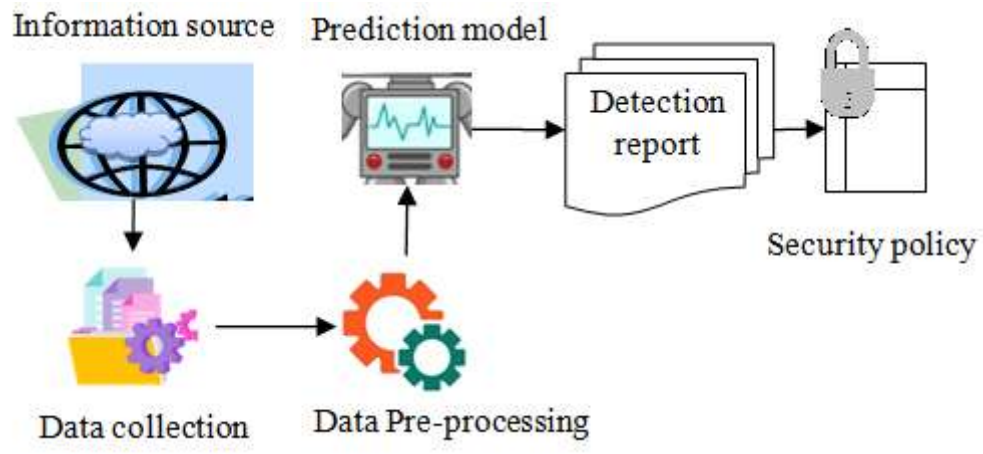

Figure 1: Network design for attack detection

3.2. Dataset Description

UNSW-NB15dataset was generated by the Australian Centre Researchers for Cyber Security (ACCS) lab at the University of New South Wales (UNSW), which includes 100GB network traffic data monitored by the TCP-Dump tool. The dataset contains 2,540,044 real-time samples and network traffic types, including UDP, TCP, HTTP, and ICMP. This dataset comprises source and destination information and each packet's time duration. It contains 56,000 of training data and 37,000 of testing data. This dataset provides information regarding nine cyber-attack instances, including fuzzers, backdoors, Analysis, Exploits, Denial of service (DoS), generic, reconnaissance, shellcode, and worms.

CICIDS-2018 Dataset was generated by the Canadian Institute for Cybersecurity Lab, which consists of seven attack types. The attacks are force, heartbleed, intrusion, botnet, DoS, web attacks, and network infiltration. This infrastructure used to evaluate attacks includes 50 machines, and the victim organization consists of 420 machines, 5 departments, and 30 servers. The dataset comprises 16,232,943network traffic records and system logs and 80 features extracted from the recorded traffic using CICFlowMeter-V3.

SIMARGL2021 Dataset was collected from the SIMARGL project, supported by the European Union through the Horizon program in partnership with the Romanian education network (RoEduNet). It includes authentic network traffic data from real-world traffic analysis. The data structure is similar to the Netflow, which includes 31,433,875 total records, 3 labels, and 29 features. The CISCO protocol was developed to capture and monitor the network flows.

Table 1: Dataset characteristics

Dataset name	Attack types	Features	Samples
UNSW-NB15	DoS, DDoS, probe, U2R, and R2L.	66	2,540,044
CICIDS-2018	Force, heartbleed, Brute, web attacks, DoS, DDoS, botnet, web attacks, and network infiltration.	80	16,232,943
SIMARGL2021	RUDY, Slowloris, FIN Scan, UDP scan, and XMAS scan.	44	31,433,875

The proposed lightweight and interpretable model ensures applicability with minimal computational overhead. The model includes converting the raw network traffic data into a structured format, feature extraction using LVQ-CWWAE to extract the relevant latent representations, and the OBTCN classifier for anomaly detection. The decision and alert system indicates the high and lower confidence scores for triggering the alarms.Figure 2 depicts the flowchart of the LVQ-CWWAE-OBTCN model.

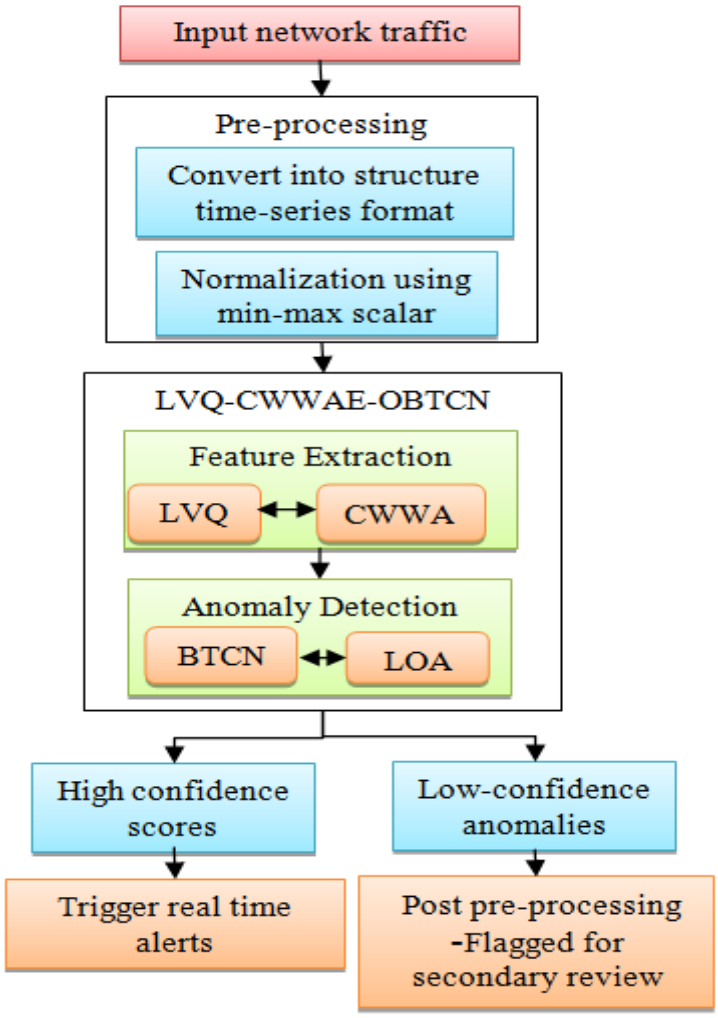

Figure 2: Flowchart of the proposed LVQ-CWWAE-OBTCN model

3.3. Pre-processing

The collected dataset is pre-processed using cleaning, transforming into a structured time-series format, and normalizing to enhance the data quality and suitability for analysis. The data cleaning process includes data imputation. In this process, the null values are replaced with the mean, median, and mode values. If any imputation introduces bias, the rows and columns are dropped. Then, the duplicate values in the data are identified and eliminated to avoid redundancy. The errors, irrelevant information, and inconsistencies in the dataset are removed. Then, this cleaned dataset is converted into a structured format using reshaped and redefined methods based on the data attributes. The packet sniffers capture the raw traffic data, often broken down into discrete time windows to create time series data. Then, the different network features are extracted for each time window. Then, these extracted features are normalized using the min-max scalar method. This method transforms the data into a consistent format by scaling them between 0 and 1 to prevent the features from dominating during the training. The pre-processing steps enhance the data quality to increase the prediction results.

3.4. Feature Extraction using LVQ-CWWAE

The pre-processed data are used in the LVQ-CWWAE model to extract the optimal features, compress and encode meaningful latent features, and distinguish normal and abnormal patterns. The LVQ-CWWAE model connects the viewpoints of the WS distance, generative models, and deep discrete representation learning. The LVQ model is used for real settings of discrete representation learning. The pre-processed training

dataset $TD = \{x_1, \ldots, x_N\} \subset \mathbb{R}^{n_x}$, here, N refers to the number of data points, where each point is a vector $\mathbb{R}^{n_x}$.The model learns the set of codeword $CW = \{c_k\}_{k=1}^{K} \subset \mathbb{R}^{k \times n_z}$, where K indicates the number of codewords that are in the latent space(n_z). Each data example maps through the encoder to a sequence of codewords(M) and these codewords collect the important characteristics in the latent space(Z). The discrete probabilistic representation of codewords is established as,

$$\mathbb{P}_{c,\pi^m} = \sum_{k=1}^{K} \pi_k^m \delta_{c_k}, m = 1, \ldots, M \quad (1)$$

Here,δ_{c_k}refers to the Dirac delta function centred at codeword ck,π_k^m refers to the weight assigned to k^{th} codeword for m^{th} distribution. Here,$\sum_{k=1}^{K} \pi_k^m = 1$ indicates the weights π_m and forms a probability distribution $(\pi_1^m, \pi_2^m, \ldots, \pi_k^m)$ over k codewords and the weights live in $(K-1)$-simplex is defined as$\Delta_{K-1} = \{\alpha \geq 0 : \|\alpha\|_1 = 1\}$, here all weights are non-negative and sum to 1.

The set of each joint distribution over a sequence of codewords M is denoted as$\Gamma = \Gamma(\mathbb{P}_{c,\pi^1,\ldots,} \mathbb{P}_{c,\pi^M})$, and the marginal distribution of Γ corresponds to$(\mathbb{P}_{c,\pi^1,\ldots,} \mathbb{P}_{c,\pi^M})$ admitting to all joint distributions. The joint distribution (Γ) captures the possible ways to jointly distribute M codewords and preserves the marginal probabilities$(\mathbb{P}_{c,\pi^M})$. Here, $\pi = [\pi^1, \ldots, \pi^M]$refers to a set of all weights vectors that contain the probabilities for assigning each data point. From the generative viewpoint, the decoder function $F_d: \mathrm{Z}^M \longrightarrow X$ used for the learning process, and the CW codebook and weights (π) to minimize the codewords is formulated as,

$$CM = \min_{C,\pi} \min_{\gamma \in \Gamma} \min_{F^d} W_{d_x}(D\#\gamma, \mathbb{P}_x) \quad (2)$$

Here, the empirical data distribution is represented as $\mathbb{P}_x = \frac{1}{N} \sum_{n-1}^{N} \delta_{xn}$ and the cost metric on data space is represented as d_x. The joint distribution $\gamma \in \Gamma$ used as a distribution over sequences of M codewords in c^M to the data space and considered the codebook data distortion as $W_{d_x}(F_d\#\gamma, \mathbb{P}_x)$. Then, the D decoder is learned subsequently to minimize the codebook-data distortion given γ. Then, the codebook, π, and γ are adjusted to reduce the best codebook data distortion. This process reduces the redundancy and ensures more structured latent representations.

The conditional weighted loss function applies adaptive weights to prioritize rare anomaly patterns. The different source domain's similarities are quantified by the model to the target domain to decrease the discrepancy in the conditional distribution to help the model. This proposed approach transforms the source data into an normal distribution using Wasserstein transforms. The Wasserstein distance between the P_s and P_n distributions are formulated as,

$$W(P_s, P_n) = \inf_{\gamma \in \pi(P_s, P_n)} E_{(x,y)\sim\gamma}[\|x - y\|] \quad (3)$$

Here, the joint probability distribution is defined asγ, and the set ofa joint probability distribution is$\Pi(P_s, P_n)$. The r^{th} source domain weight is considered as $W_r$and the large value indicates that a target domain and source domain have higher similarities. The class-conditional maximum mean distance (MMD) is performed to determine W_r and the r^{th} source domain between the target dissimilarity to minimize the conditional distribution discrepancy between various domains. It is formulated as,

$$V_k = \frac{1}{N} \sum_{n=1}^{N} \left\| \frac{\sum_{i=1}^{m_t} \hat{y}_{i,n} D_t(x_i^t)}{\sum_{i=1}^{mt} \hat{y}_{i,n}^t} - \frac{1}{m_{s_r}^n} \sum_{k=1}^{m_{s_r}^n} D_{s_r}(X_{i,n}^{s_r}) \right\|^2 \quad (4)$$

Here, the number of fault conditions denoted as N, $i^{th}$the sample under the fault condition is $X_{i,n}^{s_r}$ in the r^{th} source domain, and x_i^t denotes the i^{th}sample in the target domain. Under fault condition $(n)$the number of samples in the r^{th} source domain is $m_{s_r}^n$, and the number of samples in the target domain is $mt\hat{y}_{i,n}^t$ represents the probability in fault condition n.

Then, the received dissimilarity is used to calculate the corresponding source domain weight. Here, the higher dissimilarity indicates a smaller weight. This weight is determined as,

$$W_k = \frac{1}{r-1}\left(\sum_{i=1}^{r}\frac{\exp(V_i)}{1+\exp(V_i)} - \frac{\exp(V_r)}{1+\exp(V_r)}\right) \quad (5)$$

$$W_k = \frac{1}{r-1}\sum_{\substack{i=1 \\ i\neq r}}^{r}\frac{\exp(V_i)}{1+\exp(V_i)} \quad (6)$$

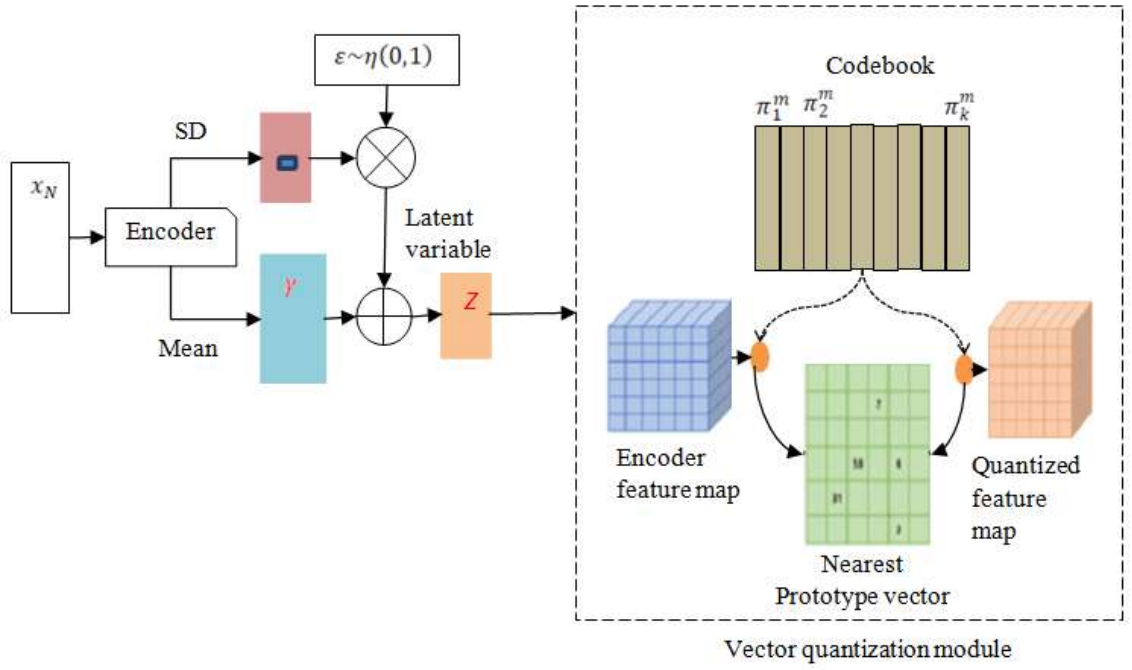

Figure 3: Process of VQ-CWWAE model

Figure 3 illustrates the process of the VQ-CWWAE model. The VQ-CWWAE integrates vector quantization with the Wasserstein Autoencoder framework, achieving efficient discrete latent representations. The pipeline processes input data by an encoder that generates two outputs. These outputs describe the latent space distribution of the input.A codebook containing prototype vectors defines possible quantized representations.The extracted features from the encoder are organized by the encoder feature map.The quantized feature map is constructed by matching each feature in the encoder map to its closest prototype vector in the codebook using a nearest-neighbor search.This process bridges the continuous latent space and discrete representations, enabling the model to capture structured, interpretable features in the latent space.

This model uses the CNN to build the feature encoder and decoder for feature extraction. The feature encodes all domains into the latent feature representation space, and the encoded source distributions are near to the Gaussian prior distribution based on the Wasserstein distance penalty constraints. The l_e distributed dependency loss in all encoded source feature distribution to the Gaussian prior distribution calculation is formulated as,

$$l_e = \sum_{r=1}^{r}\left[\frac{1}{n^r}\sum_{x^{s_r}\epsilon x^{s_r}}(E(x^{s_r}) - g)\right] \quad (7)$$

Here, the total number of source domains is r, the number of r^{th} source domain sample is n^r, g is the Gaussian prior distribution, and $E(.)$ represents the encoding process. The l_d distribution discrepancy loss between the two sets of reconstructed data is determined as,

$$l_d = \frac{1}{n^t}\sum_{x^t \epsilon X^T}\left\|D\left(E(x^t)\right) - D(g)\right\|_1 \quad (8)$$

Here, n^t is the number of target domain samples, and $D(.)$ is the decoding process. During the training process, the encoder and decoder share the details on the parameters. During the training, the Gaussian prior distribution is approached by the encoded and target feature, progressively reducing the distribution discrepancy and distributed features. The model makes the encoded and target feature similar by using the encoder and the feature discriminator training. The source and target domains of the latent feature distribution are extracted using the former, and it is used to detect the latent feature distribution sources. From this training process, the most approximate latent feature distributions are extracted using a feature encoder, making the feature discriminator unable to detect the latent feature distribution sources effectively. This process is formulated as,

$$l_{e,DI} = \sum_{r=1}^{r}\frac{W_r}{n^r}\sum_{i=1}^{n^r} l_s\left[DI\left(E\left(x_i^{s_r}\right)\right), z_j^{s_r}\right] + \frac{1}{n^t}\sum_{i=1}^{n^t} l_s\left(DI\left(E(x_i^t)\right), z_i^t\right) \quad (9)$$

Here, W_r represents the weight of r^{th} source domain, $l_s[.]$ represents the squared loss, and the discriminative process is $DI(.)$. The domain labels are $z_j^{s_r}$ and z_i^t corresponding to $x_i^{s_r}$and x_i^t. The classifier uses the corresponding labels to detect the source domain's fault conditions effectively, and the target domain pseudo labels are obtained for weight calculation and identification results. The identification loss is formulated as,

$$l_c = \sum_{r=1}^{r}\frac{W_r}{n^r}\sum_{i=1}^{n^r} l_c\left[C\left(E\left(x_i^{s_r}\right)\right), y_i^{s_r}\right] \quad (10)$$

Here, $l_c[.]$ is the cross-entropy loss, $C(.)$ is the prediction process, and $y_i^{s_r}$ is the $x_i^{s_r}$ Corresponding label. The decoder calculated the reconstruction loss to ensure anomaly detection effectiveness. The feature extractor and classifier are minimized in this process, and the total loss is maximized. The total loss of the proposed model is formulated as,

$$l_{total}\left(\theta_e, \theta_d, \theta_{DI}, \theta_c\right) = l_c(\theta_e, \theta_c) + \alpha\, l_e(\theta_e) + \beta l_d(\theta_d) - \lambda l_{e,DI}\left(\theta_e, \theta_{DI}\right) \quad (11)$$

Here, the balancing factors are α, β and λ, and the parameters of the feature encoder, decoder, discriminator, and classifier are $\theta_E, \theta_D, \theta_{Di}$, and θ_c.

The collected latent features are applied to the BTCN classifier for anomaly detection in sequential data by leveraging Bayesian principles and temporal convolutional network. This model combines probabilistic modelling with causal and dilated convolutions to efficiently capture network traffic data's short- and long-term dependencies. The Bayesian layers model weights as probability distributions, which enables uncertainty estimation in prediction and uses weights as distributions that are representedas $W \sim p(W)$. During training, weights are sampled to account for uncertainty,resulting in confidence interval predictions. For an input X, the output Y of the Bayesian layer is computed as

$$Y = f(X, W) \quad (12)$$

Here, $W \sim p(W|D)$ represents the sampled from the posterior distribution of the weights $p(W|D)$, D represents the given training data.

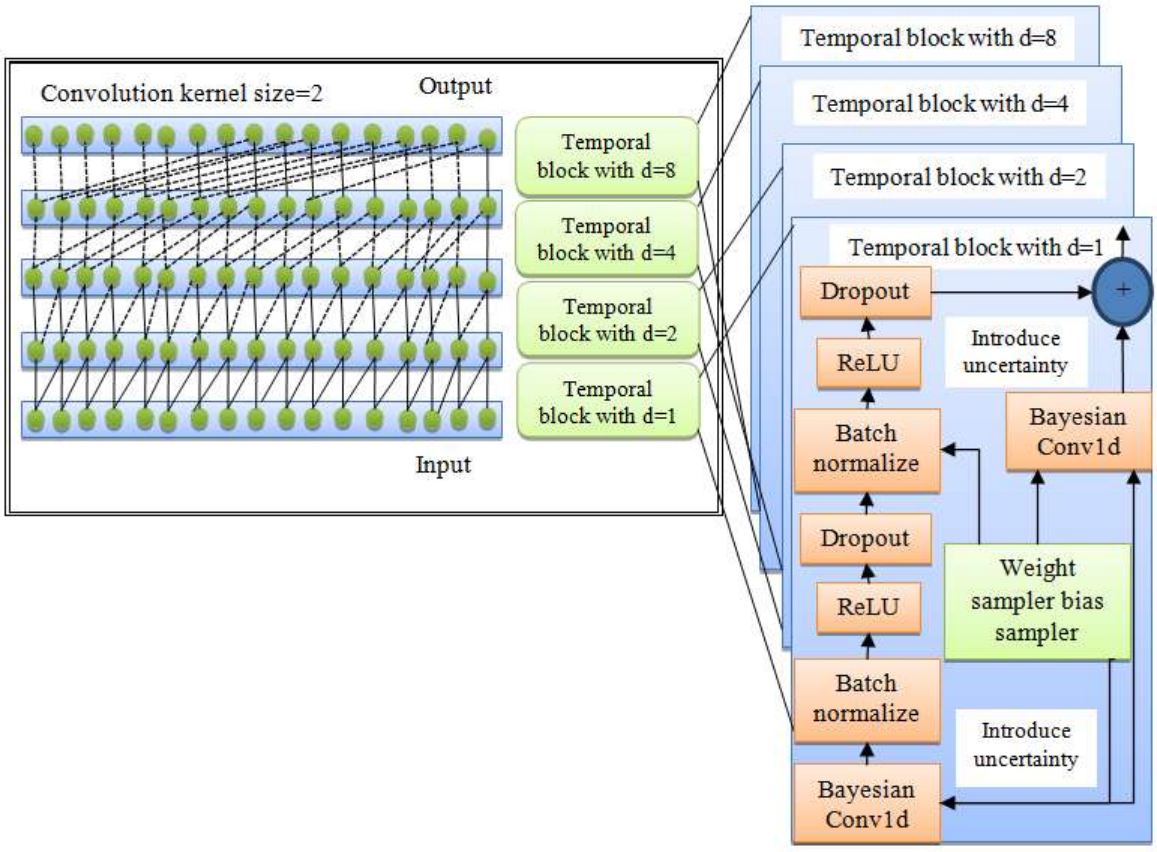

Figure 4: Architecture of BTCN model

Figure 4 depicts the architecture of the BTCN model that integrates the convolutional and temporal blocks with Bayesian uncertainty. The model processes the initial data and passes through the convolutional kernel. The convolutional operation extracts the features and provides the multiple temporal blocks for multi-scale temporal analysis. Each temporal block is assigned a specific dilation factor (d) to capture patterns at different time resolutions.The temporal blocks perform a sequence of operations, including dropout layers to prevent overfitting, ReLU activations for introducing non-linearity, and batch normalization to stabilize the training process.Bayesian convolutional layers are incorporated at multiple stages to introduce uncertainty quantification into the model. These Bayesian layers allow the network to effectively model probabilistic behaviours and handle data uncertainty.

The Bayesian model introduces uncertainty quantification by modelling the parameters and for the given data (D) the posterior distribution of the parameter (θ) is formulated as

$$p(\theta|D) \propto \mathrm{p}(D|\theta)p(\theta) \tag{13}$$

Here, $\mathrm{p}(D|\theta)$ refers to the likelihood and $p(\theta)$ refers to the prior. The loss function in the Bayesian layers includes a regularization term to approximate the posterior, and it is given as

$$L = L_{task} + KL\big(q(W)||p(W)\big) \tag{14}$$

Here, L_{task}refers to the specific loss, KL refers to the Kullback-Leibler divergence between the approximate posterior $q(W)$and prior $p(W)$. This regularization ensures uncertainty in predictions and provides confidence intervals for anomalies.

The TCN refers to the convolutional architectures that process sequential data using causal (temporal) convolutions residual and dilation connections. The causal convolutions ensure that predictions at a time step t depend only on the previous information. for the input sequence $X = [x_1, x_2, \dots, x_T]$, the output at the t, and it is formulated as

$$y_t = \sum_{i=0}^{k-1} W_i x_{t-i} \quad (15)$$

Here, k refers to the kernel size and W_i refers to the convolution filter weights.

The dilated convolution expands the network's receptive field, which allows the network to model long-range dependencies efficiently. The output y_t of a TCN layer is computed as

$$y_t = \sigma\left(\sum_{i=0}^{k-1} W_i x_{t-i} + b\right) \quad (16)$$

Here, x_{t-i} represents input values at time $t - i$, W_i refers to the learnable convolutional weights, b refers to the bias term, and σ refers to the activation function.

The BTCN model extends TCN by incorporating Bayesian principles, which model uncertainties in weight and bias. Each weight W_i in the TCN is determined as a random variable with the distribution $p(W_i)$, denotes as $W_i \sim \eta\left(\mu_i, \sigma_i^2\right)$, here $\mu_i \; and \; \sigma_i^2$ refers to the mean and variance of the distribution. The predictive distribution for an output y is given as

$$p(y|x, D) = \int p(y|x, \theta) p(\theta|D) d\theta \quad (17)$$

The integral function is intractable in deep networks, so approximation functions like variational inference are used to approximate the posterior $p(\theta|D)$ and the variational inference is defined as

$$KL\left(q(\theta)||p(\theta|D)\right) = \mathbb{E}_{q(\theta)}[\log q(\theta) - \log p(D, \theta)] \quad (18)$$

Here, $q(\theta)$ refers to the variational distribution, and the BTCN incorporates a Bayesian loss function to regularize uncertainty and is formulated as

$$L = \mathbb{E}_{q(\theta)}[-\log p(D, \theta)] + KL\left(q(\theta)||p(\theta|D)\right) \quad (19)$$

The model's hyperparameters, such as dilation rate, kernel size, and Bayesian prior distributions,are optimized using LOA to maximize performance by exploring the hyperparameter combinations and selecting the best-performing configuration.It automatically balances the accuracy and computational efficiency. The LOA is considered a population-based metaheuristic model [25]. This algorithm uses the iteration-based process based on the bird's searching power of its member in the problem-solving space. Each lyrebird is considered an LOA individual, and the decision variable values are calculated based on its position in the problem-solving space. The population of the model is formed by using LOA members $PM = [PM_1, PM_t, \dots, PM_N]$. The LOA individual's initial position in the problem-solving space is initialized randomly using this formula,

$$PM_{t,d} = Ib_d + r.(ub_d - Ib_d) \quad (20)$$

Here, the PM represents the LOA population matrix, PM_t is t^{th} population matrix, $PM_{t,d}$ is the decision variable of t^{th} population matrix, N represents the number of individuals, m is the number of decision variables, the random number is r in the interval of [0,1], and the lower bound and upper bound of the d^{th} decision variable is Ib_d and ub_d. The objective function is predicted in general based on the LOA individual, and this function is rewritten based on the hyperparameters, and it is expressed as,

$$F(x) = \alpha_1 . DR + \alpha_2 . KS + \alpha_3 . BD \quad (21)$$

Here,$\alpha_1, \alpha_2\ and\ \alpha_3$is a random vector, DR is the dilation rate, KS is kernel size, and BD is Bayesian distributions.The dilation rate is given as two, the kernel size is three, and the Objective Priors versus Bayesian distribution is provided in the model. In each iteration, the position of the population is updated based on the lyrebird strategy for sensing danger. The population's updating process includes two phases, escaping and hiding, based on the individual's decision. In this algorithm, the decision-making process selects any one of the phases during a dangerous situation, and the update process is evaluated using this equation,

$$PM_{t+1} = \begin{cases} E_s, r_p \leq 0.5 \\ H_s,\ else \end{cases} \quad (22)$$

Here, r_p represents the random number from the [0, 1] interval, E_s is the escaping strategy and H_s represents the hiding strategy. In the initial phase, the population individual's position is updated based on the escape strategy of lyrebirds from the dangerous position to the safe region. In LOA, the position of other population individuals with an optimal objective function value is considered a safe region for each individual. For each LOA individual, the set of safe regions is calculated using this formula,

$$SR_i = \{PM_k, OF_k < OF_i\ and\ k\ \epsilon\ \{1, 2, \dots, N\}\}, t = 1, 2, \dots, N \quad (23)$$

Here, SR_t represents the set of safe regions for the t^{th} individual and PM_k is the $k^{th}$the row of PM matrix. Then, each LOA individual's new position is determined based on the lyrebird displacement modeling, and it is formulated as,

$$PM_{t,j}^{p1} = PM_{t,j} + r_{t,j} . (SSR_{t,j} - I_{t,j} . PM_{t,j}) \quad (24)$$

Here, SSR_i represents the selected safe region for t^{th} individual, $SSR_{t,j}$ is the selected safe region fort^{th} individual'sj^{th} dimension, PM_t^{p1} represents the first new position determined for t^{th} individual relied on the LOA model escaping strategy and $I_{t,j}$ is the identity matrix. Then, if the objective function value is enhanced, this previous position is swapped with the new position of the corresponding individual based on,

$$PM(t+1) = \begin{cases} PM_t^{p1},\ OF_t^{p1} \leq OF_{t,} \\ PM_t \qquad\qquad else, \end{cases} \quad (25)$$

Here, OF_t^{p1} represents the objective function value for a first new position. In the second phase, the population's individual position is updated based on the hiding strategy of the lyrebird in the safe region. This algorithm modulates the movement of lyrebird t towards the closest suitable region for hiding.For each LOA individual, the new position was calculated by using,

$$PM_{t,j}^{p2} = PM_{t,j} + (1 - 2r_{i,j}) . \frac{ub_j - lb_j}{t} \quad (26)$$

This previous position is swapped with the corresponding new position individual if it enhances the objective function value based on the equation,

$$PM(t+1) = \begin{cases} PM_t^{p2},\ OF_t^{p2} \leq OF_{t,} \\ PM_t \qquad\qquad else, \end{cases} \quad (27)$$

Here, the PM_t^{p2} represents the second new position determined for the t^{th} lyrebird based on the LOAhiding strategy, and OF_t^{p2} is the objective function value for a second new position. The LOA model enhances the prediction results of the BTCN through its efficient optimization process.

The OBTCN model analyses the sequential patterns and detects the anomalies with the linked confidence score. The confidence intervals are taken and are separated into the high and low confidence scores for the post-processing and alerting system. Figure 5 depicts the post-processing and alerting model.

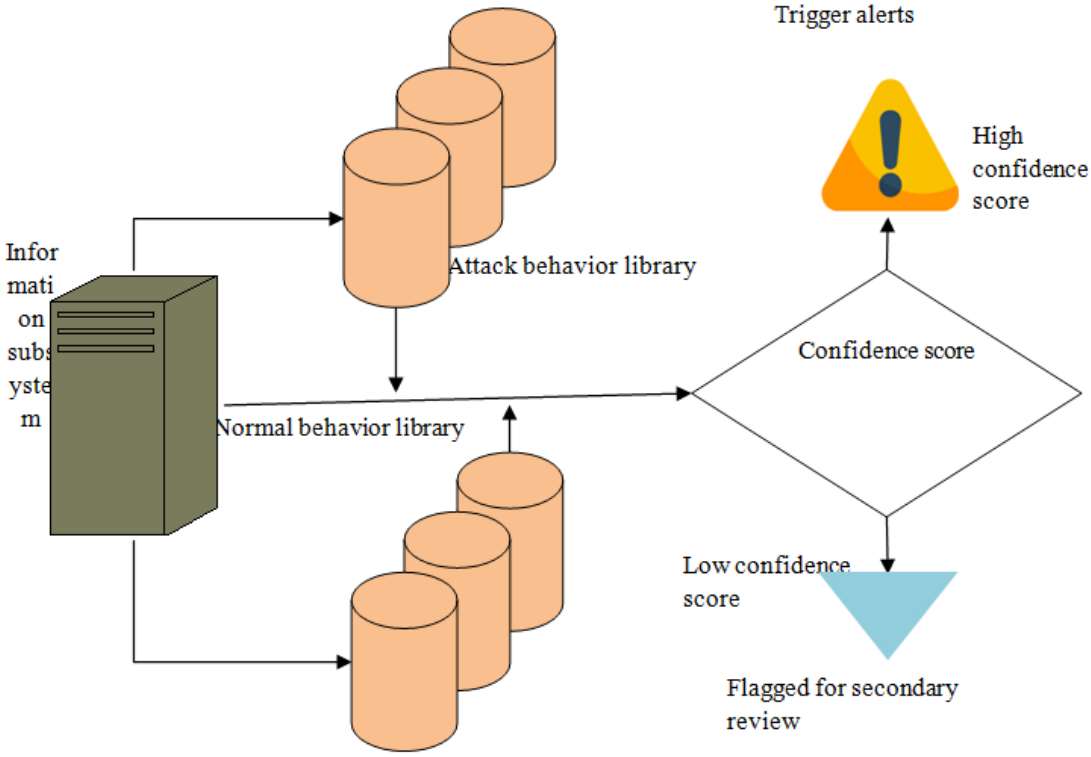

Figure 5: Post-processing and alerting model

The subsystem processes and analyses the captured data and provides the analyzed data into the libraries. The library stores the attack patterns and signatures, which are used to identify abnormal activities. An analysis unit evaluates the activity and produces a confidence score based on matches with the behavior libraries.The activity is divided based on the confidence score, and high confidence triggers an alert for the network security team. Low confidence flags the activity for a secondary review.This post-processing model ensures accurate classification of threats, minimizes false alarms and provides a balance by automating high-confidence threat detection.

4. Results and Discussion

The LVQ-CWWAE-OBTCN detection model is implemented using Python language on a PyCharm tool, with the Intel Core i5 processor system configuration, Windows 10 OS with 8GB RAM and 512GB SSD.The algorithm was computed based on accuracy, precision, recall, F1 score, and detection processing time (s). The algorithm's performance is simulated on four datasets: UNSW NB15, CICIDS 2018, and SIMARGL2021. Table 1 compares the LVQ-CWWAE-OBTCN with the existing algorithms for the UNSW NB15 dataset.

Table 2: Comparison of the LVQ-CWWAE-OBTCN with the existing model for the UNSW NB15dataset

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)	PT (s)
DCRNN	99.16	93	89.1	91	2847.7534
CNN	96.12	94.26	96.87	95.5	1456.5637
LSTM	97.23	96.45	97.57	97.16	1812.1264
LVQ-CWWAE-OBTCN	99.53	99.57	99.41	99.49	1112.7199

Table 1 exhibits the comparative analysis between existing models and the proposed LVQ-CWWAE-OBTCN model based on the UNSWNB15 dataset. The proposed model provided higher accuracy improved by 0.37%,

3.41% and 2.3%; the precision enhanced by 6.57%, 5.31%, and 3.12%; the recall improved by 10.31%, 2.54%, and 2.25%, the F1-score enhanced by 8.49%, 4.49%, and 2.33%, than DCRNN, CNN, and LSTM models, respectively. The processing time decreased by 155.927%, 30.90%, and 62.85% compared toDCRNN, CNN, and LSTM models. The LVQ-CWWAE-OBTCNmodel demonstrated higher performance among various metrics, including accuracy for classification in low execution time. LVQ-CWWAE-OBTCNmodel also effectively indicated the classification instances compared to the other existing models.

The models such as HDLNIDS, FS-CRF, E-GRACL, CNN, and LSTM are compared with the proposed LVQ-CWWAE-OBTCN model for the CICIDS2018 dataset.

Table 3: Comparison of the LVQ-CWWAE-OBTCN with the existing model for the CICIDS 2018dataset

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)	PT (s)
HDLNIDS	98.90	98.63	99.14	98.82	3575.1896
FS-CRF	96.4	97.65	97.49	97.62	4784.4832
E-GRACL	99	87	98.45	92.3	3834.2865
CNN	93.29	94.87	92.12	93.56	4213.6743
LSTM	94.32	95.48	93.31	94.44	3964.7389
LVQ-CWWAE-OBTCN	99.81	99.84	99.66	99.75	3379.1678

Table 2 compares the LVQ-CWWAE-OBTCN with the existing model for the CICIDS 2018 dataset.The model provided higher accuracy improved by 0.91%, 3.41%, 0.81%, 6.52%, and 5.49%, the precision enhanced by 1.21%, 2.19%, 12.84%, 4.97%, and 4.36%, the recall improved by 0.52%, 2.17%, 1.21%, 7.54% and 6.35%, F1 score is enhanced by 0.93%, 2.13%, 7.45%, 6.19%, and 5.31%, than HDLNIDS, FS-CRF, E-GRACL, CNN, and LSTM models, respectively. The processing time is decreased by 5.80%, 41.58%, 13.46%, 24.69%, and 17.32% than HDLNIDS, FS-CRF, E-GRACL, CNN, and LSTM models. The LVQ-CWWAE-OBTCN model demonstrated higher performance among various metrics, including accuracy for classification in low execution time. The LVQ-CWWAE-OBTCN model also effectively indicated the classification instances compared to the other existing models.

The models such as CNN, LSTM, BiLSTM and TCN are compared with the proposed LVQ-CWWAE-OBTCN model for the SIMARGL2021 dataset. The algorithm is tested using the same dataset and settings for accurate analysis and performance assessment.

Table 4: Comparison of the LVQ-CWWAE-OBTCN with the existing model for the SIMARGL2021 dataset

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)	PT (S)
CNN	93.45	93.12	94.6	93.66	19012.1738
LSTM	95.78	95.68	94.51	95.22	12173.3653
BiLSTM	96.12	95.73	95.23	95.74	14115.5412
TCN	96.52	93.01	93.64	93.38	17412.2754
LVQ-CWWAE-OBTCN	97.10	97.89	95.51	96.58	10753.1238

Table 4 compares the LVQ-CWWAE-OBTCN with the existing model for the SIMARGL2021 dataset. The model provided higher accuracy improved by 3.65%, 1.42%, 0.98%, and 0.58%, the precision enhanced by 4.77%, 2.21%, 2.16%, and 4.88%, the recall improved by 0.91%, 1%, 0.28%, and 1.87%, and the F1-score enhanced by 2.92%, 1.36%, 0.84%, and 3.2%, than CNN, LSTM, BiLSTM, and TCN models, respectively. The processing time decreased by 76.80%, 13.20%, 31.26%, and 61.92% compared to CNN, LSTM, BiLSTM, and TCN models. The LVQ-CWWAE-OBTCN model demonstrated higher performance among various metrics, including accuracy for classification in low execution time.

5. Conclusion

The paper presented LVQ-CWWAE-OBTCN model that integrated feature extraction, probabilistic modeling, and temporal analysis and addressed the limitations of existing methodologies. The LVQ-CWWAE-OBTCN model capacity handles high-dimensional data and efficiently isolates rare anomaly patterns,demonstrating a robust design for campus networks' dynamic and evolving threat. The hybrid model preserves the anomaly-specific complexities, captures the sequential dependencies, and quantifies predictive uncertainties, enhancing detection accuracy and interpretability.The datasets, such as UNSW-NB15, CICIDS 2018, and SIMARGL2021,demonstrate the model's superior performance metrics. This LVQ-CWWAE-OBTCN model is used for evolving security demands of large-scale and complex network infrastructures, enhancing its capabilities to address zero-day attacks and more complex threat landscapes.

References

1. Ali, M. N. B., Hossain, M. E., &Parvez, M. M. (2015). Design and implementation of a secure campus network. *International Journal of Emerging Technology and Advanced Engineering*, *5*(7), 370-374.
2. Olanrewaju, O. M. (2018). The Modeling and Simulation of Wireless Campus Network. *International Journal of Computer Science & Information Security*, *16*(9).
3. Shen, N., Yu, B., Huang, M., & Xu, H. (2021). *Campus Network Architectures and Technologies*. CRC Press.
4. Zheng, R., Ma, H., Wang, Q., Fu, J., & Jiang, Z. (2021). Assessing the security of campus networks: the case of seven universities. *Sensors*, *21*(1), 306.
5. Chu, H., Lan, H., Xu, J., & Sun, X. F. (2022). Analysis of Campus Network Security. *Journal of New Media*, *4*(4).
6. Naagas, M. A., MiqueJr, E. L., Palaoag, T. D., & Cruz, J. D. (2018). Defense-through-deception network security model: Securing university campus network from DOS/DDOS attack. *Bulletin of Electrical Engineering and Informatics*, *7*(4), 593-600.
7. Awang, N., Samy, G. N., Hassan, N. H., Maarop, N., Magalingam, P., &Kamaruddin, N. (2020, May). Identification of information security threats using data mining approach in campus network. In *Journal of Physics: Conference Series* (Vol. 1551, No. 1, p. 012006). IOP Publishing.
8. Dimolianis, M., Pavlidis, A., &Maglaris, V. (2021). Signature-based traffic classification and mitigation for DDoS attacks using programmable network data planes. *IEEE Access*, *9*, 113061-113076.
9. Lyu, M., Gharakheili, H. H., Russell, C., &Sivaraman, V. (2021). Hierarchical anomaly-based detection of distributed DNS attacks on enterprise networks. *IEEE Transactions on Network and Service Management*, *18*(1), 1031-1048.
10. Liu, X., Ren, J., He, H., Wang, Q., & Song, C. (2021). Low-rate DDoS attacks detection method using data compression and behavior divergence measurement. *Computers & Security*, *100*, 102107.
11. Lent, D. M. B., Novaes, M. P., Carvalho, L. F., Lloret, J., Rodrigues, J. J., &Proença, M. L. (2022). A gated recurrent unit deep learning model to detect and mitigate distributed denial of service and portscan attacks. *IEEE Access*, *10*, 73229-73242.
12. AL-Ghamdi, A. S., Ragab, M., &Sabir, M. F. S. (2022). Enhanced artificial intelligence-based cybersecurity intrusion detection for higher education institutions. *Computers, Materials & Continua*, *72*(2).
13. Raza, A., Munir, K., Almutairi, M. S., &Sehar, R. (2023). Novel class probability features for optimizing network attack detection with machine learning. *IEEE Access*.
14. Rong, Q., & Zhao, G. (2023). Campus Network Intrusion Detection Based on Gated Recurrent Neural Network and Domain Generation Algorithm. *International Journal of Advanced Computer Science and Applications*, *14*(8).
15. Li, D. (2023). Exploring the path of network security and student privacy protection in smart campus based on Markov model. *Applied Mathematics and Nonlinear Sciences*, *8*(1), 2793-2806.
16. Qazi, E. U. H., Faheem, M. H., & Zia, T. (2023). HDLNIDS: hybrid deep-learning-based network intrusion detection system. *Applied Sciences*, *13*(8), 4921.
17. Shaorong, W., & Guiling, L. (2023). Research on campus network security protection system framework based on cloud data and intrusion detection algorithm. *Soft Computing*, *27*(10), 6835-6844.
18. Doriguzzi-Corin, R., Knob, L. A. D., Mendozzi, L., Siracusa, D., &Savi, M. (2024). Introducing packet-level analysis in programmable data planes to advance Network Intrusion Detection. *Computer Networks*, *239*, 110162.
19. Kumar, G. S. C., Kumar, R. K., Kumar, K. P. V., Sai, N. R., &Brahmaiah, M. (2024). Deep residual convolutional neural network: an efficient technique for intrusion detection system. *Expert Systems with Applications*, *238*, 121912.
20. Chen, Z. (2024). Campus Network Security Intrusion Detection Based on Feature Segmentation and Deep Learning. *Journal of Cyber Security and Mobility*, 775-802.

21. Mallikarjun, S. B., Mudraje, A., Maddali, J., &Schotten, H. D. (2024, May). Machine Learning Based Anomaly and Intrusion Detection to mitigate DoS and DDoS attacks in Private Campus Networks. In *Mobilkommunikation; 28. ITG-Fachtagung* (pp. 19-24). VDE.
22. Suethanuwong, E. (2025). An Effective Prevention Approach against ARP Cache Poisoning Attacks in MikroTik-based Networks. *ECTI Transactions on Computer and Information Technology (ECTI-CIT)*, *19*(1), 1-12.
23. Lin, L., Zhong, Q., Qiu, J., & Liang, Z. (2025). E-GRACL: an IoT intrusion detection system based on graph neural networks. *The Journal of Supercomputing*, *81*(1), 42.
24. Bamber, S. S., Katkuri, A. V. R., Sharma, S., &Angurala, M. (2025). A hybrid CNN-LSTM approach for intelligent cyber intrusion detection system. *Computers & Security*, *148*, 104146.
25. Dehghani, M., Bektemyssova, G., Montazeri, Z., Shaikemelev, G., Malik, O. P., &Dhiman, G. (2023). Lyrebird optimization algorithm: a new bio-inspired metaheuristic algorithm for solving optimization problems. *Biomimetics*, *8*(6), 507.
26. A.Surendar (2025). A Comparative Evaluation of a Machine Learning-Based Probability Algorithm for Early Diagnosis of Diabetes. *Journal of Computational Medicine and Informatics, 1*(1), 1-9.
27. Sweetha S, & Dhivya K. (2025). Smart Head Band for Insomnia Monitoring with Brainwave Modulation and Guided Meditation Support Signal. *National Journal of Signal and Image Processing*, *2*(1), 8-14.
28. Alejandro G. Martínez. (2026). Design and FPGA Prototyping of a Scalable VLSI Architecture for High-Performance Signal Processing. *Journal of Integrated VLSI and Signal Processing*, 41–49.
29. Gaurav Tamrakar. (2025). Metasurface-Enabled Beam-Steering Antenna Architecture for Millimeter-Wave and Terahertz Wireless Networks. *National Journal of Antennas and Wireless Communication Systems*, *1*(1), 10–18.