Anomaly Study of Computer Networks Based on Weighted Dynamic Network Representation Learning

Xin Wei*
School of Computer Engineering, Shangqiu University, Shangqiu, Henan, 476000, China

***Abstract*—One of the foremost significant challenges in the continuously increasing technological environment is the requirement to secure the authenticity of data. Network security is a primary method for securing the confidentiality of data throughout communication, one of several types of data security assurance. To secure networks against additional cyberattacks, trustworthy Anomaly Detection (AD) is essential. The drawbacks of conventional AD are gradually increasing as various types of attacks and network changes continually evolve. The researchers of the present study propose a novel approach that incorporates Weighted Long Short-Term Memory (WLSTM) networks with Dynamic Network Representation Learning (DNRL) to address these problems, referring to it as the Weighted Dynamic Network Representation Learning (WDNRL) paradigm. This investigation develops the WLSTM utilizing the Weight of Evidence (WoE), which periodically determines weights to network features in the resulting network model. The WLSTM design functions as the network's coordinator, obtaining data from the recommended model, upgrading the representation, and aggregating the features. The findings showed that the proposed model achieved high accuracy rates of 99.85% for Denial of Service (DoS) attacks and 99.55% for Distributed Denial of Service (DDoS) attacks when evaluated using two datasets, NSL-KDD and CICIDS-2017, compared to different models. Additionally, the simulation's F1-scores, recall rates, and precision are all above average, indicating that it is capable of identifying many network anomalies with minimal false positives (FP).**

***Keywords*—Network security; attacks; weighted dynamic network; anomaly detection; deep learning; LSTM**

I. Introduction

As cyber-attacks become increasingly challenging, network security remains the primary concern for many businesses worldwide. Modern technologies involved in Anomaly Detection (AD) are gradually becoming crucial in this constantly evolving setting, aiming to detect anomalous patterns or behaviors that could indicate a network attack or threaten security [1]. The variable environment of computer network behaviors, the variation of potential anomalies, and the challenge of distinguishing between undetectable and fake activity anomalies are just a few of the significant issues that AD still faces and fails to address [2]. Despite these limitations, the traditional AD has contributed significantly to providing ample security, and the methods employed have ranged from statistical methods to Machine Learning (ML). All such models have their strengths and limitations. Out of the early traditional methods, many have frequently relied on predefined rules or thresholds. Such models, although effective for known attack patterns, have primarily challenged the task of detecting novel and sophisticated threats [3]. This limitation prompted the exploration of advanced Machine Learning (ML) and Deep Learning (DL), which have presented improved adaptability and can also learn to identify attacks based on data patterns without explicit programming.

Even then, such solutions are also not without gaps; the ML and DL that were prominently used in more complex tasks, including popular choices like Generalized Regression Units (GRU) and Long Short-Term Memory (LSTM) [4], have also been faced with faults like such models can be hampered by drawbacks like the high dimensionality of network data, the imbalance between normal and anomalous samples, and also such models need for extensive labelled datasets for training.

Moreover, the static nature of some of these models has significantly limited their effectiveness in certain types of environments where network behavior is constantly changing. To address such limitations, Graph Learning (GL) has been introduced, a novel method that represents network data in the form of graphs, enabling the capture of complex relationships and interactions within the network [5]. This unique GL paradigm encoding enhances the network data model, providing better AR. However, GL's use in network AD is currently in its earliest phases, and several ongoing research projects are attempting to determine methods to maximize its most significant use.

Dynamic Network Representation Learning (DNRL) is an innovation in the domain that addresses AD challenges, particularly for networks that evolve [6]. Employing the physical and time data readily available in network data, the DNRL can operate in an AD and provide a deeper understanding of how networks perform. The synthesis of additional practical and verified DL designs, including GRU + LSTM, has enhanced the design's unique features. The DNRL integrates the parallel data processing abilities of GRU and LSTM with its DNRL limit, thereby rendering it more direct to find how data from networks continues to evolve. Including intricate data connections and temporal dynamics into AD enhances accuracy, making models more adaptable; DNRL's scalability, compared to other ML models, improves the analysis of vast volumes of network data. To circumvent the shortcomings of different approaches in network AD, the model employs a unified method.

Weighted Dynamic Network Representation Learning (WDNRL), the model developed in this research, is based on the factors previously labeled. Dynamic network representation learning for Alzheimer's disease (AD) has been integrated into research employing what is recommended: Weighted Long Short-Term Memory (WLSTM). The WDNRL utilizes the

*Corresponding Author

Weight of Evidence (WoE) enhanced LSTM to dynamically weight the features, allowing the model to adapt to changes in network configurations and interactions. This configuration helped the method to achieve improvement in the accuracy and timeliness of detecting the anomalies by employing a method to process the sequential network data and updating representations that are based on the interactions and temporal context by adjusting the parameters that impact the input features, which are measured using their predictive value the proposed WDNRL accurate AD. Evaluations of the WDNRL model were conducted using two datasets: the NSL-KDD and CICIDS-2017. The model yielded better results for all evaluated metrics compared to other models. The model achieved accuracies of 99.85% and 99.55% for DoS and DDoS attacks, respectively. It had shown precision of up to 98.35% and 96.35%, recall rates of 98.41% and 96.15%, and F1-scores of 97.38% and 97.17%, emphasizing its effectiveness in detecting a standard range of network AD with few False Positives (FP).

The study is organized as follows: Section II presents the literature review for understanding existing works, Section III outlines the methodology of the proposed work, Section IV describes the experiment setup, dataset, and analysis of results, and finally, Section V concludes the work.

II. Literature Review

In [7], the authors provided a work about foundational survey that maps the landscape of AD in the field of dynamic networks. The work categorizes anomalies into four distinct types and presents an analysis of methods standard among numerous approaches, providing an understanding of AD behaviors. They have further elaborated on a two-tiered taxonomy that classifies the studied methods based on different conceptual intuitions and the specific types of anomalies the methods are employed to detect. In [8], the authors have introduced a model called CmaGraph, designed to measure distances between vertices using DL. Implementing a novel combination of detection communities and AD blocks of data, their model focused on enhancing the AD boundaries. Based on the findings of the test examination, which verified the proposed model using multiple real-world data sets, the model they developed performed superior in recognizing dynamic network anomalies.

In [9], the authors developed a novel method termed Content-Aware Anomaly Detection (CAAD) to address the problem of recognizing anomalous connections and nodes from typical ones in a network of nodes. They achieved the layout of the method by integrating DNRL with an algorithm that encodes and decodes data. Convolutional Neural Networks (CNN) were used to build the design of their model. The network was built using structural and content data. The model was experimented with using different datasets, and the proposed CAAD proved superior performance when compared to other existing methods. In [10], the authors have suggested models targeted at exploring AD using a model built on blockchain transactions. The network of their model effectively leveraged the extensive amount of records and the practical application of the graph-like nature of blockchain data. Their research, using their projected model, has attempted to project the application and advantages of ML in the task of detecting anomalous transactions. It also revealed, through experimental analysis, that better performance was achieved using supervised learning techniques.

In [11], the authors have suggested a contrastive mechanism for Temporal Representation Learning (TRL) in the field of dynamic networks by designing a method named “Dynamic Network Contrastive Representation Learning” (DNCL). The model's design considers factors such as network topology, node feature data, and network evolution and applies these factors to contrast objective functions, thereby exemplifying the effectiveness of DNCL. In [12], the authors have introduced the Pyramidal Image Anomaly Detector (PIADE), a DL designed to extract image features from input data at multiple scale levels. This representation-based method is distinct from other standard representation models, employing methods that utilize significant similarity and perceptual loss to compare an input image to its reconstructed version effectively. The PIADE’s effectiveness is proven on datasets such as CIFAR10, COIL-100, and MVTec; for all these datasets, the proposed model generated better results. In [13], the authors have attempted to tackle the complex problem of AD in dynamic and multi-attributed network systems through the projected Multi-view Time-Series Hypersphere Learning (MTHL). The model attempted to project the multi-view time-series data into a shared latent subspace and learned a compact hypersphere around the normal samples; this way, it effectively distinguished between normal and abnormal cases. The method used in experiments has proven to have superior performance compared to other baseline methods.

In [14], the authors have suggested a model named DynAD to detect anomalous edges in time-evolving networks. The DynAD was designed with a model that employs a temporal graph convolution network, along with pooling operations, to extract node embeddings. The model further utilizes the GRU for capturing temporal information. Furthermore, the model incorporates an attention mechanism into the network to enhance its ability to detect AD against other baseline methods.

In [15], the authors have concentrated on exploring graph evolution-based prediction by learning spatiotemporal features, which are termed dynamics. Their method involved measuring the affinity scores related to nodes corresponding to the graphs. This way, the model provided a way to have better accuracy in detecting dynamic anomalies, and through this method, their model has addressed the challenge related to sparsity in real-world networks. Their work was experimented with in-network features of public transportation, and the simulation revealed the effectiveness of the proposed model. In [16], the authors presents a model built using a convGRU-based autoencoder for learning the spatial-temporal features of raw network traffic in an unsupervised manner. The application could generate meaningful compressed features and provide an effective means of detecting anomalies based on residual loss. Although the model’s detection capability regarding compressed data was irrelevant, it still enriched AD by exploring an interpretability method and providing a better-compressed representation of the network traffic [17-20].

III. METHODOLOGY

A. WoE and Information Value in Network Anomaly Detection

The Weight of Evidence (WoE) and Information Value (IV) are statistical measures traditionally used in credit scoring and risk assessment to transform definite variables into statistical numerical values and to evaluate the predictive strength of these variables. It is the natural logarithm of the ratio of the proportion of anomalies (positive) to the proportion of normal behavior (negative), as in Eq. (1):

$$WoE = ln(Distribution\ of\ Anomalies = Distribution\ of\ Normal\ Behavior) \quad (1)$$

For any network feature $'x'$, with classes $'x_i'$, WoE is calculated for each type to transform the categorical feature into a continuous scale that reflects the attributes related to AD. IV measures the cumulative predictive strength of a feature by calculating the sum of the differences in the proportions of anomalies and normal behaviors across all types of the feature, each weighted by their respective WoE, as in Eq. (2):

$$IV = \sum_i (Distribution\ of\ Anomalies_i - Distribution\ of\ Normal\ Behavior_i) \times WoE_i \quad (2)$$

The resultant IV values range from 0 to infinity (0 to ∞), out of which the higher values denote a more vital ability to predict network anomalies.

Information Value (IV) can be interpreted as follows:

- 0 to 0.02: Not applicable for prediction.
- 0.02 to 0.1: Weak predictive power.
- 0.1 to 0.3: Medium predictive power.
- 0.3 to 0.5: Strong predictive power.
- More remarkable than 0.5: Suspiciously high (maybe too good to be true or indicate data leakage).

B. Standard LSTM

LSTM is a type of Recurrent Neural Network (RNN) that is designed to overcome the challenges of learning long-term dependencies that were hard to solve using the traditional RNN. The LSTM units retain the data for extended periods due to their inherent structure. The LSTM is applied to fields such as time series analysis, Natural Language Processing (NLP), and network AD [21-25].

Each of the LSTM unit's models is characterized by a series of gates (see Fig. 1) that regulate the flow of data, which are described as follows:

1) Cell state. The Cell state $'C_t'$→ the LSTM unit's long-term memory, which stores relevant information throughout the sequence processing. It is modified by the forget gate and the input gate to add or remove information.

2) Forget gate. The Forget gate $'f_t'$ is used to determine which data to be maintained and which one to be discarded from the cell state. It applies a sigmoid function to the previous output $'h_{t-1}'$ and the current input $'x_t'$ which results in producing a value between 0 and 1 for each number in the cell state $'C_{t-1}'$, as in Eq. (3):

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \quad (3)$$

where,

- W_f, b_f →the weight matrix and bias for the forget gate
- σ→ the sigmoid function.

3) Input gate. The input gate $'i_t'$ is employed to determine which new data is stored in the cell state. Simultaneously, it uses a *tanh* layer to create a vector of new candidate values, $\tilde{C}_t$, that could be added to the state, as in Eq. (4) and Eq. (5):

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \quad (4)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \quad (5)$$

where,

W_i, W_C and b_i, b_C → The weight matrices and biases for the input gate and the candidate values.

4) Cell state update. The old cell state C_{t-1} is updated to that of the new cell state $'C_t'$ by multiplying the old state value by that of the forget gate's output to discard the irrelevant data and perform the task of adding the input gate's output that is being multiplied by the candidate values to add new information, as in Eq. (6):

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \quad (6)$$

5) Output gate. The output gate o_t is used to decide what part of the cell state is used for the output. A *tanh* function filters this output to ensure values stay between -1 and 1, and then it is multiplied by the output value of the sigmoid gate, as in Eq. (7) and Eq. (8):

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \quad (7)$$

$$h_t = o_t * \tanh(C_t) \quad (8)$$

where,

- W_o, b_o → The weight matrix and bias for the output gate
- h_t→ The final output of the LSTM unit at time $'t'$.

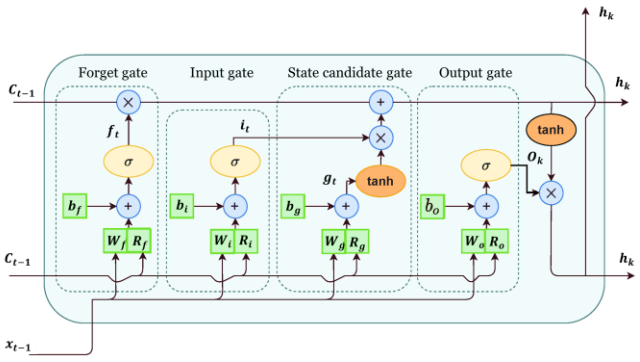

Fig. 1. LSTM

C. Weighted LSTM (WLSTM)

The WLSTM enhances the standard LSTM using a weighting mechanism (Fig. 2). This weighting mechanism-based enhancement is done using the WoE for each of the input features. The WoE using the input feature dynamically adjusts the impact of each input feature based on its predictive value regarding network anomalies. This enhancement enables the LSTM to improve its accuracy for AD in a network environment where features vary significantly over time. The WoE factored input data is fed into the input gate of the LSTM [26-30].

The method of integrating WoE to LSTM is discussed below:

1) Modified input gate. The input gate in WLSTM, which is represented as $'i_t^w'$ that integrates the WoE by weighting the input features $'x_t'$, as in Eq. (9):

$$i_t^w = \sigma\big(W_i \cdot \big[h_{t-1}, \big(x_t \times \mathrm{WoE}(x_t)\big)\big] + b_i\big) \quad (9)$$

Similarly, the candidate values for the cell state update, $\tilde{C}_t^w$, are calculated using the weighted inputs Eq. (10):

$$\tilde{C}_t^w = \tanh\big(W_C \cdot \big[h_{t-1}, \big(x_t \times \mathrm{WoE}(x_t)\big)\big] + b_C\big) \quad (10)$$

2) Cell state update. The cell state update is then modified to adapt to that of the WoE-modified input gate and candidate values, as in Eq. (11):

$$C_t^w = f_t * C_{t-1} + i_t^w * \tilde{C}_t^w \quad (11)$$

3) Output gate and final output. The output gate's function remains unchanged except for the inputs to this gate, which have been modified in the previous steps, and so the same is reflected in the final output, as in Eq. (12):

$$h_t^w = o_t * \tanh(C_t^w) \quad (12)$$

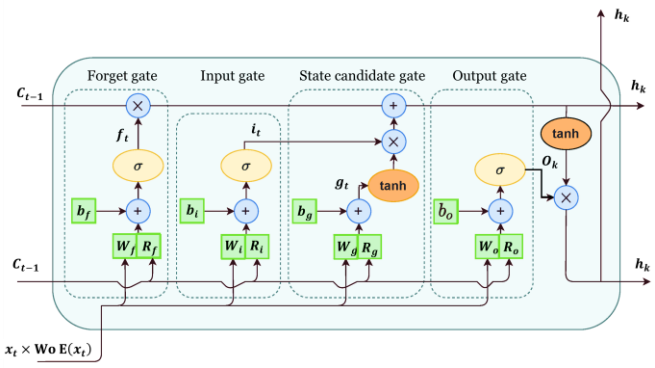

Fig. 2. WLSTM architecture.

D. Problem Definition

Let us consider a network represented by a graph $N = (S, C)$,

where,

- S → The set of nodes related to network devices and endpoints.
- C' → The connections between these nodes. The network behaviour over time is represented as a set of events. (s_a, s_b, τ), in which s_a, s_b denote the nodes.
- τ → The time related to the interaction between both the nodes.

The problem defined here is to do the updation of the DNRL based on the network's evolving network and to represent it in a reduced vector space Z^d to find the anomaly.

In real-world network environments, configurations and interactions change over time [31-35].

The changes in a network may happen due to any of the following actions:

- Addition of a New Node: This represents a device joining the network without initial connections.
- Creation of a New Connection: A new communication link is established between two nodes.
- Integration of a Connected Node: A new node joins and connects to existing nodes.
- Node Removal: The disconnection or failure of a network node.
- Connection Termination: The end of a communication link between nodes.
- Network Topology Changes: Adjustments in the network's structure due to various factors.

The proposed WDNRL (Fig. 3), utilizes the WLSTM to perform dynamic feature weighting and representation learning, thereby enhancing the DNRL. The model utilizes the WLSTM to process weighted features over time, identifying changes in the network's state and facilitating effective AD.

The proposed WDNRL is done using the following process:

1) Feature aggregation. Collecting network features over time and assigning weights based on their predictive value for AD.

2) Representation update. Continuously refining the network's vector selection to reflect its current state, using an optimization that ensures similar entities are represented closely in Z^d.

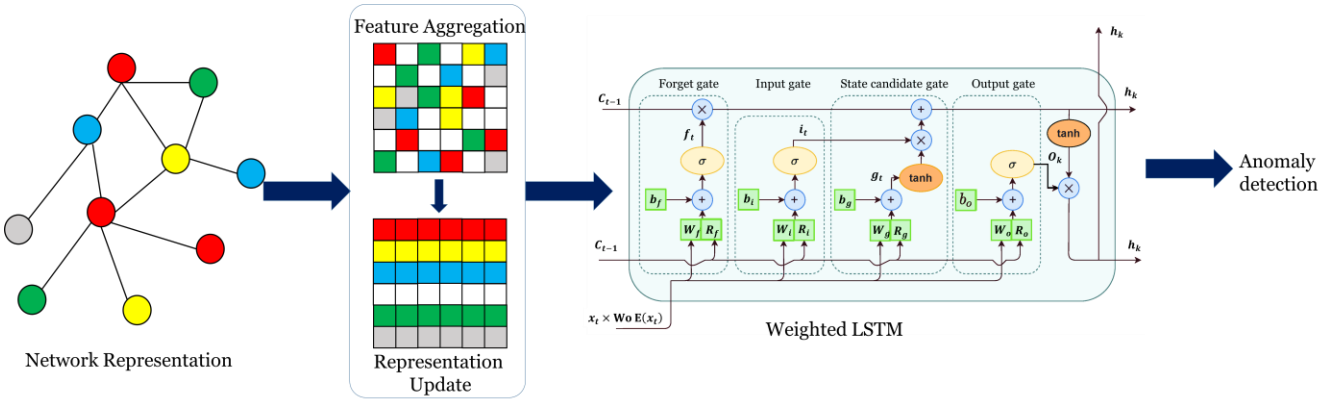

Fig. 3. WDNRL.

E. Aggregating Feature Data for Dynamic Anomaly Detection

The feature data aggregation technique employs predetermined parameters for learning distinct aggregation functions, which are denoted as $AggF^n$ for each layer $'n'$ within the range of 1 to N. These functions collate data from an entity's n-th layer neighbours using a set of weight matrices. W^n for each layer to help the flow of data across different network depths [36-40]. Procedure 1 presents the function of aggregating the feature data.

Procedure 1: Aggregation of Feature Information

Step 1. **Input:** Graph $N = (S, C)$; initial features $\{x_s, \forall s \in S\}$; depth N; weight matrices $W^n, \forall n \in \{1,2,\dots,N\}$; non-linear activation σ; aggregation functions $\text{AggF}^n, \forall n \in \{1,2,\dots,N\}$; neighborhood sampling sets $N^n, \forall n \in \{1,2,\dots,N\}$.

Step 2. **Output:** Vector selection as z_s for all $s \in S$.

Step 3. **Initialize** $h_s^0 = x_s$ for each entity $s \in S$.

Step 4. **For Each** layer $n = 1$ to N :

Step 5. **For Each** entity $\in S$

Step 6. **Aggregate** features $h_{N(s)}^n = \text{AggF}^n(\{h_u^{n-1}, \forall u \in N^n(s)\})$.

Step 7. **Update** entity feature $h_s^n = \sigma\left(W^n \cdot \text{concat}\left(h_s^{n-1}, h_{N(s)}^n\right)\right)$.

Step 8. **Normalize** h_s^n to unit length for each entity $s \in S$.

Step 9. **Assign** $z_s = h_s^N$ for each entity $s \in S$.

The input for the aggregation procedure comprises the network graph $N = (S, C)$ alongside the feature data for the entities within the network, denoted as $x_s, \forall s \in S$. A sampling strategy is deployed to find and select neighboring entities across 1 to N layers surrounding an assumed entity. Subsequently, the features of these selected neighboring entities are compiled. Further, a pooling approach, $AggF_{pool}^n$, is employed for aggregation, defined as the element-wise maximum $\max\left(\left\{\sigma\left(W_{\text{pool}} \cdot h_{u_i}^n + b\right), \forall u_i \in N(s)\right\}\right)$, where max represents the element-wise maximum operation, and $'\sigma'$ is a non-linear activation function. The current selection of an entity, h_s^{n-1} along with the pooled vector of neighborhood features, $h_{N(s)}^n$, are then processed through a fully connected layer employing the non-linear activation function $'\sigma'$, which effectively updates the entity's selection. The outcome is the enhanced vector selection of entity $'s'$.

F. Updating Network Selection for AD

The recommended method provides a model for refreshing entity models based on inputs collected in response to network dynamics. This method enables objects to rapidly modify their models to reflect the evolving state of the network by emphasizing the incorporation of recent changes to the network. The method of updating the web model is explained in Procedure 2.

Procedure 2: Network Selection Update Using WLSTM

Input: Change indicator (flag); vector selection $z_{s_a}(\tau)$ and $z_{s_b}(\tau)$; prior selection state $r_s(\tau-)$; time interval $\Delta\tau$; decay function g; weight matrices W_1, W_2; activation function $\text{act}(\cdot)$.

Output: Updated selection state $r_s(\tau)$.

Step 1. Formulate the entity impact embedding $e(\tau) = \text{act}\left(W_1 \cdot z_{s_a}(\tau) + W_2 \cdot z_{s_b}(\tau) + \epsilon\right)$, capturing the essence of recent interactions.

Step 2. Utilize $g(\Delta\tau)$ to modify the selection, highlighting the significance of temporal proximity to the last update.

Step 3. Employ the WLSTM×to refresh the entity's selection as $r_s(\tau)$, considering the impact embedding $e(\tau)$, the preceding selection state $r_s(\tau-)$, and the change flag.

In this model, the network changes are perceived as sequences which are denoted by $(s_a, s_b, \tau,$ flag $)$, where s_a and s_b represent the entities involved, $'\tau'$ indicates the time of change, and the flag defines network expansion ($flag = 1$) or contraction (flag $= -1$). The WLSTM is employed to handle temporal dynamics, thereby enabling the model to adapt to changes in network dynamics [41-45]. To establish consistency among closely situated entities, an optimization technique is used to minimize the dissimilarity, which is defined as a loss function, as in Eq. (13):

$$J(z_s) = -\text{Log}\left(\sigma\left(z_s^T z_{s'}\right)\right) - Q \cdot \mathbb{E}_{s_n \sim P_n(s)}\left[\text{Log}\left(\sigma\left(-z_s^T z_{s_n}\right)\right)\right] \quad (13)$$

where,

- $z_s \rightarrow$ The embedding of an entity impacted by network changes
- $Q \rightarrow$ The count of negative samples drawn

- $P_n(s)$ d→ A distribution for negative sampling, s' is a neighbouring entity near
- s, s_n → Entities sampled from the negative sampling distribution.

IV. Experimental Setup

The experiments were conducted using a server equipped with an Intel(R) Core(TM) i5-10210U CPU at 2.11 GHz, running Windows 10 and utilizing PyTorch for model implementation.

A. Dataset for Experimentation

For the aim of testing the success rate of the proposed approach, the present research utilizes two distinct datasets: NSL-KDD and CICIDS-2017.

1) NSL-KDD. A collection of four distinct types of cyberattacks has been included in the NSL-KDD dataset. These are Denial of Service (DoS), User-to-Root (U2R), Remote-to-Local (R2L), and Probing (Probe). The dataset also includes data on normal network activities. The dataset comprises 41 features for all samples and includes an ID for each example, indicating whether it represents a typical instance or a malicious attack type.

2) CICIDS-2017. The CICIDS-2017 dataset captures 79 network flow features. The dataset comprises various connection logs, including SSH, email, HTTP, and FTP, collected from 25 users on different operating systems.

The following Tables I and II present the type of attacks contained in both datasets:

TABLE I Attack Types in NSL-KDD Dataset

Dataset	Split	U2R	DoS	R2L	Probe	Normal
NSL-KDD	**Train**	52	45,927	995	11,656	67,343
	Test	67	7458	2887	2422	9710

TABLE II Attack Types in the CICIDS-2017 Dataset

Dataset	Split	DDoS	FTP-Patator	PortScan	SQL Injection	Benign
CICIDS-2017	**Train**	112,901	6997	140,043	19	72,7397
	Test	25388	1574	31,492	4	163,572

The WDNRL was compared against other baseline models, such as RNN, LSTM, and DNRL + LSTM [46-50]. The proposed model was trained using the hyperparameters shown in Table III for the above two datasets:

TABLE III Hyperparameter for Training

Hyperparameter	Specific Value
Learning Rate	$1e{-}3$
Number of Epochs	200
Batch Size	6464
Weight Decay	$1e{-}4$
LSTM Hidden Units	256
Sequence Length	30
Learning Rate Scheduler	Step Decay
Dropout Rate	0.2
Negative Sampling Rate	10
Embedding Dimension	128

B. Evaluation Metrics

The evaluation of the model is done using the following metrics:

- Accuracy: It measures the model's ability in correct AD, as in Eq. (14):

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \tag{14}$$

- Precision: It measures the proportion of accurate AD traffic to that of the total classified as attacks in the dataset, as in Eq. (15):

$$\text{Precision} = \frac{TP}{TP+FP} \tag{15}$$

- Recall: It calculates the fraction of correctly predicted attack traffic over the actual attack instances, as in Eq. (16):

$$\text{Recall} = \frac{TP}{TP+FN} \tag{16}$$

- F1-score: The F1-score is the harmonic mean of precision and recall, as in Eq. (17):

$$\text{F1-score} = \frac{2\times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{17}$$

C. Accuracy Analysis

In the NSL-KDD (Table IV and Fig. 4), the proposed WDNRL achieved an impressive accuracy of 99.85% for Denial of Service (DoS) attacks, surpassing the scores of the other models. Next, the DNRL with LSTM follows closely at 98.7%. For the probing attacks, the WDNRL scored 98.76% accuracy. Remote-to-Local (R2L) and User-to-Root (U2R) attacks are challenging types of attacks to detect. For these attacks, the proposed model achieved accuracies of 87.21% and 78.57%. The improvement of LSTM achieved using WoE is reflected in the results, showing a notable 10% decrease in U2R attacks.

For the CICIDS-2017 (Table IV and Fig. 4), the WDNRL proves a clear edge over all other models, achieving accuracies of 99.55% and 98.46% for detecting Distributed Denial of Service (DDoS) and FTP-based attacks. In detecting PortScan (PS) and SQL injection attacks, the WDNRL achieves accuracies of 94% and 86.95%, which is a significant improvement over the immediately following DNRL with LSTM, which achieved only 67.5% and 79% [51-55].

It is also worth noting that the proposed model achieves the highest score in classifying normal traffic, with a perfect score of 100 and 99.7% for the NSL-KDD and CICIDS-2017.

TABLE IV ACCURACY PERFORMANCE RESULTS

Models	NSL-KDD					CICIDS-2017				
	DoS	Probing	R2L	U2R	Normal	DDoS	FTP	PS	SQL	Normal
RNN	96.04	93.63	80.33	64.28	97.3	94.44	94.82	53.13	76.71	93.7
LSTM	97.08	94.28	79.35	61.73	98.6	93.77	93.44	74.59	77.68	94.1
DNRL + LSTM	98.77	97.68	85.16	68.54	99.76	96.26	95.36	67.57	79.09	97.5
WDNRL (Proposed)	99.85	98.76	87.21	78.57	100	99.55	98.46	94	86.95	99.7

TABLE V PRECISION PERFORMANCE RESULTS

Models	NSL-KDD					CICIDS-2017				
	DoS	Probing	R2L	U2R	Normal	DDoS	FTP	PS	SQL	Normal
RNN	96.3	96.21	84.98	60	95.26	94.03	89.59	70.47	72.47	91.36
LSTM	96.56	94.21	78.32	62.67	96.5	93.62	91.95	79.47	73.96	92.56
DNRL + LSTM	96.6	94.85	77.37	70.91	97.1	93.86	92.97	64.81	77.11	94.28
WDNRL (Proposed)	98.35	96.29	91.03	79.61	99	96.35	94.61	98.78	83.92	98.23

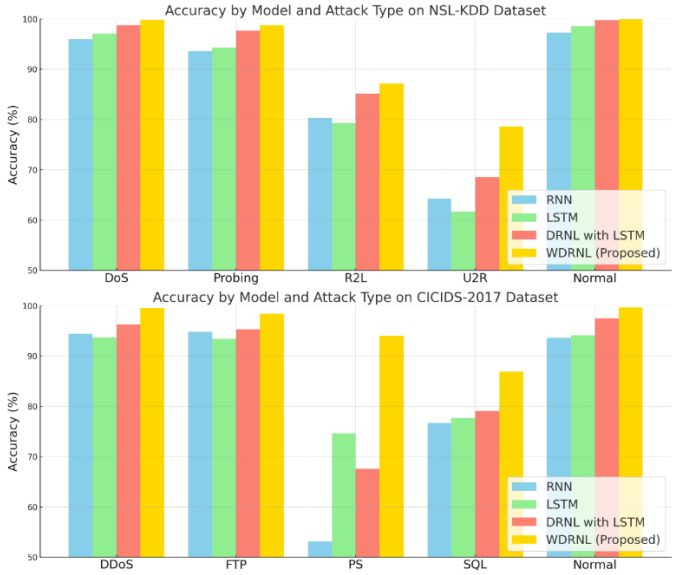

Fig. 4. Accuracy analysis for the NSL-KDD and CICIDS-2017

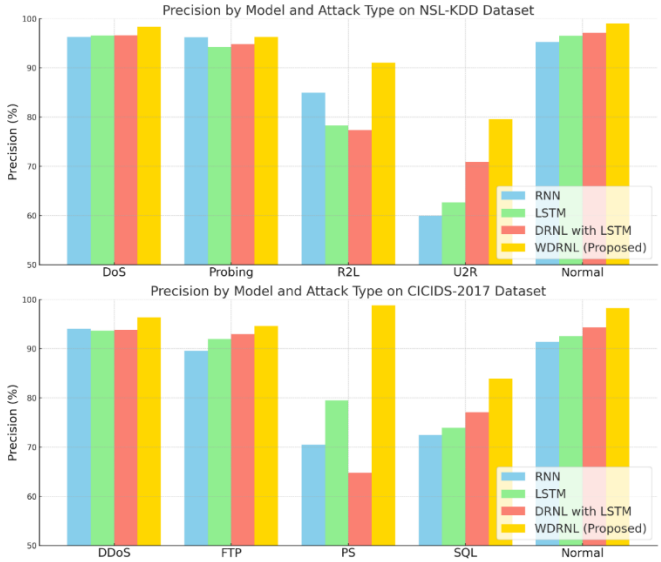

Fig. 5. Precision analysis for the NSL-KDD and CICIDS-2017

D. Precision Analysis

Table V and Fig. 5 present the analysis of the precision achieved by the proposed model and the other compared models. For the NSL-KDD, the proposed model achieves a precision of 98.35% for DoS attacks and 96.29% for Probing attacks; these high scores demonstrate its effectiveness in accurately identifying threat types with minimal error. Further, the model shows 91.03% and 79.61% precision scores for the R2L and U2R attacks [56-60]. When compared to the CICIDS again, the proposed model showed better performance across all attacks. Specifically, the WDNRL achieved a precision of 96.35% in detecting DDoS attacks and 94.61% for FTP-based anomalies. Additionally, the precision in identifying PortScan and SQL injection attacks reached scores of 98.78% and 83.92%, respectively. From both datasets, the proposed model showed a much higher score than the other compared models [61-70]. Further, the model's performance in detecting normal traffic was also higher, with 99% and 98.3% precision for the NSL-KDD and CICIDS.

E. Recall Analysis

Table VI and Fig. 6 analyze the performance of the proposed model in comparison with the other models in terms of the Recall metric. For the NSL-KDD, the WDNRL's recall performance was the highest, capturing all attacks without missing them mainly. For the DoS and Probing attacks [71-75], the proposed model achieved scores of 98.41% and 96.38%. The recall scores for the Remote to Local (R2L) and User to Root (U2R) attacks were 85.36% and 62.9%, respectively. The only model that could come close is the DNRL + LSTM [76-82]; the rest of the models performed too poorly compared to the proposed model. For CICIDS-2017, a similar trend was observed, with the proposed model exhibiting a higher recall score across all attack cases, recording 96.15% recall in detecting DDoS, 95.13% for FTP-based attacks, 78.32% for PortScan, and 84.25% for SQL injection attacks. For both datasets, the model achieved recall rates of 98.1% and 98.2% for normal cases in the NSL-KDD and CICIDS-2017.

TABLE VI RECALL PERFORMANCE RESULTS

Models	NSL-KDD					CICIDS-2017				
	DoS	**Probing**	**R2L**	**U2R**	**Normal**	**DDoS**	**FTP**	**PS**	**SQL**	**Normal**
RNN	96.96	91.27	65.93	31.2	92.5	89.88	89.03	58.61	66.51	90.78
LSTM	95.75	92.67	54.17	34.13	93.71	92.44	91.02	40.32	51.3	91.37
DNRL with LSTM	95.44	93.31	73.21	31.2	94.4	93.77	93.09	45.98	52.74	93.28
WDNRL (Proposed)	98.41	96.38	85.36	62.9	98.1	96.15	95.13	78.32	84.25	98.28

Fig. 6. Recall analysis for the NSL-KDD and CICIDS-2017

F. F1-Score Analysis

Table VII and Fig. 7 present the results of F1-score metrics from the NSL-KDD and CICIDS-2017. On the NSL-KDD dataset, the WDNRL, as seen in the earlier metrics, was the higher-performing model, achieving F1-scores of 97.38% and 97.34% for DoS and Probing attacks. For R2L and U2R attacks, it scored 91.2% and 78%. The next performing model was DNRL + LSTM, which scored 94.6%, 95.2%, 84.2%, and 61% for DoS, Probing, R2L, and U2R. When compared to the CICIDS-2017, the proposed model again demonstrated higher F1-score values, achieving 97.17% and 96.16% for DDoS and FTP, respectively, as well as 88.1% and 91.38% for PortScan (PS) and SQL injection attacks. In comparison to classifying normal traffic, the proposed model achieved an impressive score of 99.2% for NSL-KDD and CICIDS-2017.

Upon comparing all the metrics, it is evident that the proposed model outperformed the others, achieving the highest score across all metrics for both datasets. The next model that follows the proposed model is the DNRL+ LSTM for all metrics in both datasets. Of the other two models, the RNN and its variant, LSTM, exhibit similar performance, with LSTM holding a slight edge in certain metrics for specific types of attacks.

TABLE VII F1 PERFORMANCE RESULTS

Models	**NSL-KDD**					**CICIDS-2017**				
	DoS	**Probing**	**R2L**	**U2R**	**Normal**	**DDoS**	**FTP**	**PS**	**SQL**	**Normal**
RNN	92.16	92.43	64.05	45	91.13	90.47	89.31	49.91	71.58	90.28
LSTM	91.02	91.07	75.23	43.33	92.98	92.99	90.35	69.62	61.93	91.92
DNRL with LSTM	94.63	95.23	84.26	61.06	95.5	96.82	95.03	83.49	82.64	97.28
WDNRL (Proposed)	97.38	97.34	91.2	78	99.2	97.17	96.16	88.1	91.38	99.21

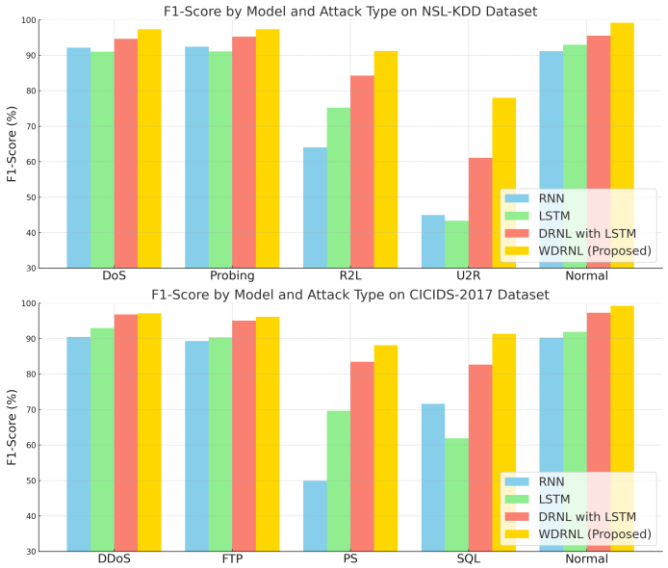

Fig. 7. Recall analysis for the NSL-KDD and CICIDS-2017.

V. CONCLUSION AND FUTURE WORK

This study aims to adapt the DNRL for use in AD applications within networks. The study proposed designing a better AD for dynamic network instances. The idea was cropped with the introduction and progress in the field of DNR. The study attempted to optimize the standard DNRL by incorporating a weighted concept to aggregate features of higher importance to the network AD. To achieve this, the proposed study introduced a WoE-optimized LSTM that computes the WoE over the possible AD against normal traffic for the input data. Using the WLSTM, the WDNRL effectively navigates the evolving landscape of network features. The effectiveness of the proposed model was evaluated using the NSL-KDD and CICIDS-2017 datasets, with metrics including accuracy, precision, recall, and F1-score. It is observed that, for all metrics, the proposed model outperforms the other baseline models for all attack instances.

Future work would focus on enhancing the model by integrating more advanced DL and scaling the model to other AD modalities.

REFERENCES

[1] Dunn Cavelty, M., & Wenger, A. (2020). Cyber security meets security politics: Complex technology, fragmented politics, and networked science. Contemporary Security Policy, 41(1), 5-32.

[2] Singh, J. P. (2022). Mitigating Challenges in Cloud Anomaly Detection Using an Integrated Deep Neural Network-SVM Classifier Model. Sage Science Review of Applied Machine Learning, 5(1), 39-49.

[3] Liu, Q., Hagenmeyer, V., & Keller, H. B. (2021). A review of rule learning-based intrusion detection systems and their prospects in smart grids. IEEE Access, 9, 57542-57564.

[4] Sengupta, S., Basak, S., Saikia, P., Paul, S., Tsalavoutis, V., Atiah, F., ... & Peters, A. (2020). A review of deep learning with particular emphasis on architectures, applications, and recent trends. Knowledge-Based Systems, 194, 105596.

[5] Wang, S., Hu, L., Wang, Y., He, X., Sheng, Q. Z., Orgun, M. A., ... & Yu, P. S. (2021). Graph learning based recommender systems: A review. arXiv preprint arXiv:2105.06339.

[6] Chikwendu, I. A., Zhang, X., Agyemang, I. O., Adjei-Mensah, I., Chima, U. C., & Ejiyi, C. J. (2023). A comprehensive survey on deep graph representation learning methods. Journal of Artificial Intelligence Research, 78, 287-356.

[7] Ranshous, S., Shen, S., Koutra, D., Harenberg, S., Saloutos's, C., & Samatova, N. F. (2015). Anomaly detection in dynamic networks: a survey. Wiley Interdisciplinary Reviews: Computational Statistics, 7(3), 223-247.

[8] Lin, W., Bao, X., & Li, M. J. (2021). Cmagraph: A triblocks anomaly detection method in dynamic graph using evolutionary community representation learning. In Artificial Neural Networks and Machine Learning–ICANN 2021: 30th International Conference on Artificial Neural Networks, Bratislava, Slovakia, September 14–17, 2021, Proceedings, Part I 30 (pp. 105-116). Springer International Publishing.

[9] Li, Z., Jin, X., Zhuang, C., & Sun, Z. (2020). Content-Aware Anomaly Detection with Network Representation Learning. In Algorithms and Architectures for Parallel Processing: 20th International Conference, ICA3PP 2020, New York City, NY, USA, October 2–4, 2020, Proceedings, Part III 20 (pp. 50-64). Springer International Publishing.

[10] Martin, K., Rahouti, M., Ayyash, M., & Alsmadi, I. (2022). Anomaly detection in blockchain using network representation and machine learning. Security and Privacy, 5(2), e192.

[11] Jiao, P., Chen, H., Tang, H., Bao, Q., Zhang, L., Zhao, Z., & Wu, H. (2024). Contrastive representation learning on dynamic networks. Neural Networks, 106240.

[12] Mishra, P., Piciarelli, C., & Foresti, G. L. (2020). A neural network for image anomaly detection with deep pyramidal representations and dynamic routing. International Journal of Neural Systems, 30(10), 2050060.

[13] Teng, X., Lin, Y. R., & Wen, X. (2017, November). Anomaly detection in dynamic networks using multi-view time-series hypersphere learning. In Proceedings of the 2017 ACM Conference on Information and Knowledge Management (pp. 827-836).

[14] Zhu, D., Ma, Y., & Liu, Y. (2020, December). A flexible attentive temporal graph networks for anomaly detection in dynamic networks. In 2020 IEEE 19th International Conference on Trust, Security, and Privacy in Computing and Communications (TrustCom) (pp. 870-875). IEEE.

[15] Lee, J., Bae, H., & Yoon, S. (2020). Anomaly detection by learning dynamics from a graph. IEEE Access, 8, 64356-64365.

[16] Kopp, F. (2022). Representation learning for content-sensitive anomaly detection in industrial networks. arXiv preprint arXiv:2205.08953.

[17] Thirimanne, S. P., Jayawardana, L., Yasakethu, L., Liyanaarachchi, P., & Hewage, C. (2022). Deep neural network-based real-time intrusion detection system. SN Computer Science, 3(2), 145.

[18] Yang, L., Li, J., Yin, L., Sun, Z., Zhao, Y., & Li, Z. (2020). Real-time intrusion detection in wireless network: A deep learning-based intelligent mechanism. Ieee Access, 8, 170128-170139.

[19] Rokade, M. D., & Sharma, Y. K. (2021, March). MLIDS: a machine learning approach for intrusion detection for real-time network dataset. In 2021 International Conference on Emerging Smart Computing and Informatics (ESCI) (pp. 533-536). IEEE.

[20] Liu, J., Zhang, W., Ma, T., Tang, Z., Xie, Y., Gui, W., & Niyoyita, J. P. (2020). Toward security monitoring of industrial cyber-physical systems via hierarchically distributed intrusion detection. Expert Systems with Applications, 158, 113578.

[21] Dutt, I., Borah, S., & Maitra, I. K. (2020). Immune system based intrusion detection system (IS-IDS): A proposed model. IEEE Access, 8, 34929-34941.

[22] Rose, J. R., Swann, M., Grammatikakis, K. P., Koufos, I., Bendiab, G., Shiaeles, S., & Kolokotronis, N. (2022). IDERES: Intrusion detection and response system using machine learning and attack graphs. Journal of Systems Architecture, 131, 102722.

[23] Hossain, M. D., Inoue, H., Ochiai, H., Fall, D., & Kadobayashi, Y. (2020). LSTM-based intrusion detection system for in-vehicle can bus communications. Ieee Access, 8, 185489-185502.

[24] Hnamte, V., Nhung-Nguyen, H., Hussain, J., & Hwa-Kim, Y. (2023). A novel two-stage deep learning model for network intrusion detection: LSTM-AE. IEEE Access.

[25] Fu, Z. (2022). Computer network intrusion anomaly detection with recurrent neural network. Mobile Information Systems, 2022.

[26] Alferaidi, A., Yadav, K., Alharbi, Y., Razmjooy, N., Viriyasitavat, W., Gulati, K., ... & Dhiman, G. (2022). Distributed deep CNN-LSTM model for intrusion detection method in IoT-based vehicles. Mathematical Problems in Engineering, 2022.

[27] Sridhar Panneerselvam et al., (2024), Federated learning-based fire detection method using local MobileNet, Scientific Reports, vol. 14, No. 30388, pp. 1-24,

[28] Asir Chandra Shinoo Robert Vincent and Sudhakar Sengan, (2024). Edge computing-based ensemble learning model for health care decision systems, Sci Rep 14, 26997.

[29] Asir Chandra Shinoo, Robert Vincent and Sudhakar Sengan, Effective clinical decision support implementation using a multi-filter and wrapper optimisation model for Internet of Things based healthcare data. Sci Rep 14, 21820.

[30] Gulista Khan et al. (2024) Energy-Efficient Routing Algorithm for Optimizing Network Performance in Underwater Data Transmission Using Gray Wolf Optimization Algorithm, Journal of Sensors, Vol. 2024, No. 2288527, 1-15.

[31] Sudhakar Sengan et al. 2023, Improved LSTM-Based Anomaly Detection Model with Cybertwin Deep Learning to Detect Cutting-Edge Cybersecurity Attacks, Human-centric Computing and Information Sciences, Vol. 13, No. 55.

[32] Varatharaj Myilsamy et al., (2023). State-of-Health Prediction for Li-ion Batteries for Efficient Battery Management System Using Hybrid Machine Learning Model, Journal of Electrical Engineering & Technology.

[33] Indumathi Nallathambi et al. (2023). Impact of Fireworks Industry Safety Measures and Prevention Management System on Human Error Mitigation Using a Machine Learning Approach, Sensors, 23 (9), 4365.

[34] Vinothini Arumugham et al. (2023) An Artificial-Intelligence-Based Renewable Energy Prediction Program for Demand-Side Management in Smart Grids, Sustainability, Vol. 15, No. 6, 5453.

[35] Rasheed Abdulkader et al. (2023), Soft Computing in Smart Grid with Decentralized Generation and Renewable Energy Storage System Planning, Energies, Vol. 16, No. 6, 2655.

[36] Parkavi Krishnamoorthy et al. (2023). Effective Scheduling of Multi-Load Automated Guided Vehicle in Spinning Mill: A Case Study, IEEE Access.

[37] Arodh Lal Karn et al. (2023). IoT Based Smart Framework Monitoring System for Power Station, Computers, Materials & Continua, Vol. 74, No. 3, pp. 6019-6037.

[38] Arul Rajagopalan et al. 2022, Oscar Danilo Montoya, Walid El-Shafai, Mostafa M. Fouda, and Moustafa H. Aly, Modernized Planning of Smart Grid Based on Distributed Power Generations and Energy Storage Systems Using Soft Computing Methods, Energies, Vol. 15, No. 23, 8889.

[39] Eman S. Sabry et al. (2022), Sketch-Based Retrieval Approach Using Artificial Intelligence Algorithms for Deep Vision Feature Extraction, Axioms, Vol. 11, No. 12, pp. 663.

[40] Arodh Lal Karn et al. (2022). Measuring the Determining Factors of Financial Development of Commercial Banks in Selected SAARC Countries, Journal of Database Management, Vol. 33, No. 1, 2022 pp. 1–21.

[41] Prabu Selvam et al. (2022) A Transformer-Based Framework for Scene Text Recognition, IEEE Access, Vol. 10, pp. 100895-100910.

[42] Arodh Lal Karn et al. (2022) Fuzzy and SVM-Based Classification Model to Classify Spectral Objects in Sloan Digital Sky, IEEE Access, Vol. 10, pp. 101276-101291.

[43] Tribhuwan Kumar et al. (2022). Fuzzy Logic and Machine Learning-Enabled Recommendation System to Predict Suitable Academic Program for Students, Mathematical Problems in Engineering, vol. 2022, No. 5298468, pp. 1-7.

[44] Roy Setiawan et al. (2022). IoT Based Virtual E-Learning System for Sustainable Development of Smart Cities, Journal of Grid Computing, Vo. 20, No. 24.

[45] S. Priyadarsini et al. (2022), Automatic Liver Tumor Segmentation in CT Modalities Using MAT-ACM, Computer Systems Science and Engineering, vol. 43, no. 3, pp. 1057–1068, 2022.

[46] S. Priyadarsini et al. (2022). Classification of Liver Tumors from Computed Tomography Using NRSVM, Intelligent Automation & Soft Computing, Vol. 33, No. 3, 2022, pp. 1517-1530.

[47] Thirumoorthy Palanisamy et al. (2022) Improved Energy Based Multi-Sensor Object Detection in Wireless Sensor Networks, Intelligent Automation & Soft Computing, Vol. 33, No. 1, pp.227-244.

[48] Omaima Bamasaq et al. (2022). Syed Hamid Hassan, Distance Matrix and Markov Chain Based Sensor Localization in WSN, Computers, Materials & Continua, Vol. 71, No. 2, pp. 4051-4068,

[49] Sudhakar Sengan et al. (2021), Real-Time Automatic Investigation of Indian Roadway Animals by 3D Reconstruction Detection Using Deep Learning for R-3D-YOLOv3 Image Classification and Filtering, Electronics, Vol. 10, No. 24, 3079.

[50] Abolfazl Mehbodniya et al. (2021). Fetal health classification from cardiotocographic data using machine learning, Expert Systems.

[51] Sudhakar Sengan et al. (2021). A Secure Recommendation System for Providing Context-Aware Physical Activity Classification for Users, Security and Communication Networks, Vol. 2021, No. 4136909, pp. 1-15.

[52] Vasanthi Raghupathy et al. (2022). Interactive Middleware Services for Heterogeneous Systems, Computer Systems Science and Engineering, Vol. 41, No. 3, pp. 1241-1253.

[53] Abolfazl Mehbodniya et al. (2022). Proportional Fairness Based Energy Efficient Routing in Wireless Sensor Network, Computer Systems Science and Engineering, Vol. 41, No. 3, pp. 1071-1082.

[54] D. Stalin David et al. (2022), Enhanced Detection of Glaucoma on Ensemble Convolutional Neural Network for Clinical Informatics, Computers, Materials & Continua, Vol. 70, No. 2, pp. 2563-2579, .

[55] D. Stalin David et al. (2022). Cloud Security Service for Identifying Unauthorized User Behaviour, CMC-Computers, Materials & Continua, Vol. 70, No. 2. pp. 2581-2600.

[56] Ngangbam Phalguni Singh et al. 2021. Investigation on characteristics of Monte Carlo model of single electron transistor using Orthodox Theory, Sustainable Energy Technologies and Assessments, Vol. 48, 101601.

[57] K. Rajakumari et al., (2022), Fuzzy Based Ant Colony Optimization Scheduling in Cloud Computing, Computer Systems Science and Engineering, Vol. 40, No. 2, pp. 581-592.

[58] Arodh Lal Karn et al. (2021), An integrated approach for sustainable development of wastewater treatment and management system using IoT in smart cities, Soft Computing.

[59] R. Nithya et al., (2022). Edwin Hernan Ramirez-Asis, Priya Velayutham, V. Subramaniyaswamy, Sudhakar Sengan, An Optimized Fuzzy Based Ant Colony Algorithm for 5G-MANET, Computers, Materials & Continua, Vol. 70, No. 1, pp: 1069–1087.

[60] Razia Sulthana Abdul Kareem et al., (2021). Multilabel land cover aerial image classification using convolutional neural networks, Arabian Journal of Geosciences, Vol. 14, No. 1681.

[61] Roy Setiawan et al., Utilizing Index-Based Periodic High Utility Mining to Study Frequent Itemsets, Springer, Arabian Journal for Science and Engineering, 2021.

[62] K. Muthumayil et al., (2021) A Big Data Analytical Approach for Prediction of Cancer Using Modified K-Nearest Neighbour Algorithm, Journal of Medical Imaging and Health Informatics, Vol. 11, No. 8, pp. 2120-2125 (6).

[63] Keerthana Nandakumar et al. (2021) Sudhakar Sengan, Securing data in transit using data-in-transit defender architecture for cloud communication, Soft Computing.

[64] Huidan Huang et al. (2021) Emotional intelligence for board capital on technological innovation performance of high-tech enterprises, Aggression and Violent Behavior, 101633.

[65] Sudhakar Sengan et al. (2021) Detection of false data cyber-attacks for the assessment of security in smart grid using deep learning, Computers & Electrical Engineering, Vol. 93, 107211.

[66] Sudhakar Sengan et al., (2021). Cost-effective and efficient 3D human model creation and re-identification application for human digital twins, Multimedia Tools and Applications, 2021. DOI:10.1007/s11042-021-10842-y.

[67] Omnia Saidani Neffati et al., (2021). Migrating from traditional grid to smart grid in smart cities promoted in developing country, Sustainable Energy Technologies and Assessments, Vol. 45.

[68] Prabhakaran Narayanan et al. (2021) Novel Collision Detection and Avoidance System for Mid-vehicle Using Offset-Based Curvilinear Motion. Wireless Personal Communication.

[69] Balajee Alphonse et al., (2021). Modeling and multi-class classification of vibroarthographic signals via time domain curvilinear divergence random forest, J Ambient Intell Human Comput.

[70] Omnia Saidani Neffati et al., (2021). An educational tool for enhanced mobile e-Learning for technical higher education using mobile devices for augmented reality, Microprocessors and Microsystems, Vol. 83, 104030.

[71] Prabhakaran Narayanan et al., (2021). Analysis and design of fuzzy-based manoeuvring model for mid-vehicle collision avoidance system. J Ambient Intell Human Comput.

[72] Sudhakar Sengan et al., (2021) Network Embedding Architecture using Laplace Regularization-Non-Negative Matrix Factorization for Virtualization, Microprocessors and Microsystems.

[73] K. Sathish Kumar et al. (2020). Area Based Efficient and Flexible Demand Side Management To Reduce Power And Energy Using Evolutionary Algorithms, Malaysian Journal of Computer Science, 27335.

[74] L. Arokia Jesu Prabhu et al., (2020) Medical Information Retrieval Systems for e-Health Care Records using Fuzzy Based Machine Learning Model, Microprocessors and Microsystems, DOI:10.1016/j.micpro.2020.103344.

[75] N. Satheesh et a., (2020). Flow-based Anomaly Intrusion Detection using Machine Learning Model with Software Defined Networking for OpenFlow Network, Microprocessors, and Microsystems.

[76] Sengan Sudhakar et al., (2020). A fuzzy-based high-resolution multi-view deep CNN for breast cancer diagnosis through SVM classifier on visual analysis, Journal of Intelligent & Fuzzy Systems, pp. 1-14.

[77] Sengan Sudhakar et al. (2020). Images super-resolution by optimal deep AlexNet architecture for medical application: A novel DOCALN, Journal of Intelligent & Fuzzy Systems, pp. 1-14.

[78] V. Vijaya Kumar et al. (2020). Design of peer-to-peer protocol with sensible and secure IoT communication for future internet architecture, Microprocessors and Microsystems, Vol. 78.

[79] Sudhakar Sengan et al., (2020). Enhancing cyber-physical systems with hybrid smart city cyber security architecture for secure public data-smart network, Future Generation Computer Systems.

[80] Ganesh Kumar, K and Sudhakar Sengan (2020) Improved Network Traffic by Attacking Denial of Service to Protect resources using Z-Test

Based 4-Tier Geomark Traceback (Z4TGT), Wireless Personal Communications.

[81] S. Sudhakar et al. (2020). Unmanned Aerial Vehicle (UAV) based Forest Fire Detection and monitoring for reducing false alarms in forest fires, Computer Communications 149, pp. 1–16.

[82] Sudhakar Sengan and Chenthur Pandian S, (2016), Hybrid Cluster-based Geographical Routing Protocol to Mitigate Malicious Nodes in Mobile Ad Hoc Network, International Journal of Ad Hoc and Ubiquitous Computing, Vol. 21, No. 4, pp. 224-236.