JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

AN IMPROVED NETWORK TRAFFIC CLASSIFICATION BASED ON DUAL FEATURE LEARNING WITH A HYBRID DEEP LEARNING MODEL

[1*]Dr.P.Arulprakash, [2]A.Balamurugan, [3]S.Avinashlingam, [4]S.Dhanushshankar, [5]J.Mohammed Jameal

[1]Professor & Head, Department of Computer Science and Engineering, Park College of Engineering and Technology, Coimbatore.*

[2,3,4,5]Department of Computer Science and Engineering, Rathinam Technical Campus, Coimbatore.

ABSTRACT

Traffic classification methods encounter a major issue of identifying the most suitable feature combination, which has a significant influence on the classification process. It is vital to maintain a balance between the classifier's ability to generalize and its exposure to risk. This paper proposed model focuses on traffic classification using a combination of DL and ML techniques. The input data is taken from a preprocessed dataset that has undergone data cleaning and data standardization. Feature extraction and selection are then performed using machine learning and deep learning techniques. The machine learning technique involves separate feature extraction and selection operations, while the deep learning technique utilizes a single DBN for both operations. In the ML technique, the feature extraction stage includes features like source port, destination port, Time to live, protocol, and packet size, and the feature selection is performed using a CESOA. The selected features are concatenated, and traffic classification is performed using a Bi-LSTM model. This model aims to improve the accuracy of traffic classification by optimizing feature selection and leveraging the strengths of both ML and DL techniques. The MATLAB software is used for implementation and an accuracy of 98.37% is achieved for the higher-performing proposed model than the current models.

Keywords: *Machine Learning, Deep Learning, Deep Belief Network, Chaotic enriched Seagull Optimization Algorithm, and Bidirectional Long Short-Term Memory.*

1. INTRODUCTION

In WSN, the term "traffic classification" refers to the process of recognizing and categorizing various forms of traffic that are transmitted over the network. This is a crucial role in WSNs as it enables effective resource utilization, performance optimization, and enhanced service quality. The WSN is a group of nodes with the ability to sense, process, and communicate on their own [1], [2]. To sense and keep track of the interesting happenings, these tiny sensor nodes ad hoc interconnect with one another. The data packets are sent through each node via a short-range transmitter through intermediary nodes and toward a sensor node, also known as a base station. Traffic in WSNs is categorized based on a number of features, including packet size, data rate, packet type, packet content, and application layer protocol. Regarding bandwidth, latency, dependability, and energy usage, different traffic kinds have distinct needs. As a result, network changes are possible using traffic classification to accommodate the specific needs of different traffic kinds and applications [3], [4], [5], [6].

Traffic classification in WSNs faces a number of difficulties, including the necessity for real-time processing, the need for constrained resources [7], [8], [9] (such as processing speed, memory, and battery life), and changeable network topology. To solve these issues, researchers have created a variety of algorithms and methods, including rule-based systems, pattern recognition systems, and machine learning-based strategies. The ML classifiers have demonstrated their suitability in this situation since they are able to classify encrypted communication without explicitly relying on port information [10], [11], [12], [13]. Nevertheless, their typical form relies on the laborious, inefficient, and unsuitable automation process of producing handcrafted (domain-expert driven) features, such as packet sequence statistics, which cannot keep up with the rate of evolution of network traffic. The ability to train classifiers directly from input data by automatically extracting structured (and sophisticated) feature representations makes DL the first step towards achieving excellent performance in dynamic and difficult (encrypted) TC situations.

Deep learning has led to the emergence of classification techniques based on CNN and RNN networks. Currently, there are numerous articles on the categorization of network traffic, most of which use both conventional and ML classification techniques. There are numerous publications concerning various deep

learning algorithms, as well as some studies on enhancing the performance of deep learning algorithms.The main contribution of the paper is as follows,

- To effectively classify the network traffic, this work proposed both machine learning and deep learning model for feature selection. After selecting the features from both ML and DL techniques, concatenation is performed to improve the classification process.
- To improve the efficiency of the feature selection process, the movement of the seagull is improved using the Chaotic Optimization algorithm called Chaotic enriched Seagull Optimization Algorithm (CESOA), while utilizing the chaotic values at the f_c factor, the optimal features are selected.

The organization of the paper is as follows, the recent existing papers related to network traffic classification are given in section 2, the detailed proposed methodology for the traffic classification is discussed in section 3, the results obtained for the proposed model are discussed in section 4 and the conclusion is presented in section 5.

2. LITERATURE REVIEW

The recent existing papers relevant to traffic classification in a network are given in this section.

In 2019, Lim,*et. al.,*[16]have suggested that without the involvement of the network operator, traffic must be categorized using deep learning. A software-defined network-based deep learning model for traffic classification was introduced in that paper. With the internal network traffic pre-processing, the paperprovides flow-based payload datasets to train two deep learning models. To perform network traffic categorization, two models are used such as the multi-layer LSTM model, and the combination of CNN and single-layer LSTM models. Additionally, to perform a model tweaking approach for determining the two deep learning models' ideal hyper-parameters.

In 2020, Lotfollahi,*et. al.,* [17] have introduced adeep Packet, a system for autonomous feature extraction from network data utilizing deep learning algorithms for traffic classification. To the best of our knowledge,

Program recognition and traffic characterization task was both handled by deep Packet, which is the first traffic classification system to use SAE and one-dimensional CNN. Furthermore, believed that Deep Packet represents the beginning of a broad trend toward the use of deep learning algorithms in traffic classification tasks due to the state-of-the-art outcomes it has obtained.

In 2020, Bu,*et. al.,* [18] have introduced the classification of encrypted network traffic in that research was accomplished using a neural network model with deep and parallel NIN architecture. In contrast to CNN, to enhance local modelling, NIN applies a micro-network next to the convolution layer. Furthermore, NIN makes use of a pooled global average before final classification rather than the conventional fully connected layers, which drastically cuts down on the total amount of design variables. In that approach, Fixed-length packet vectors are mapped to traffic or program tags using deep NIN models with several MLP layers.

In 2021, Ren,*et. al.,* [19] have provide a brand-new classification scheme for encrypted traffic. That model doesn't need to extract and choose features, unlike conventional traffic classification techniques. The binary tree type is used in this model for the purpose of ensuring that all classifier in the tree-like architecture completes the little categorization, and it provides the precise splitting principles for whichever variety of traffic categories. The RNN techniquewas used through the classifier to learn the time-related characteristics of the traffic. That utilizes the cosine to evaluate the similarity across classes and identify which categories had the same node as them.

In 2021, Sadeghzadeh,*et. al.,* [20] have suggested a DL-based network traffic classifiers' resistance to ANT. ANT made it difficult for DL-based network traffic classifiers to use UAP generating methodologies. That suggests three additional UAP-injection approaches that would produce ANT through network traffic. A UAP is included in a bogus packet's payload via the AdvPay attack to test how well flow content classifiers hold up under pressure. The purpose of the AdvBurst attack was to assess a flow's burst by injecting a predetermined the quantity of mock packets with built-in statistical features using a UAP.

In 2019, Cherif,*et. al.,* [21] have evaluated the performance of the XGBoost algorithm using a dataset made up of genuine traces that were recorded using a significant French ISP and involved more than 34k residential users. Also, that contrasted various ML algorithms while accounting for various performance factors. The XGBoost algorithm ends up being the most precise one. The work was not focused on the categorization of web

traffic. In order to assist the implementation of such a strategy in the field, it was also concentrated on establishing an effective retraining solution. Also, not suitable for SDN and cloud-based categorization strategies.

In 2020, Elnawawy, *et. al.,* [22] have introduced an ML-Based FPGA for Network traffic classification. Random forest feature selection and stepwise regression were used in that article to assess the applicability of packet-level and flow-level features. Also, the best proportion of packets to extract flow-level information from was calculated. Using the publicly accessible datasets from the UNB and the UNIBS, several experiments were carried out utilizingk-nearest neighbors, naive Bayes, SVM, artificial neural networks, and random forests.

2.1. Problem statement

Traffic classification is the process of categorizing network traffic based on its characteristics and identifying the type of application or service that generated it. The exponential increase in network traffic and the increasing diversity of applications and services have made traffic classification a critical task in security, network management, and performance optimization. The problem of traffic classification involves designing algorithms and techniques thataccurately identify different types of traffic based on a variety of features such as packet size, flow duration, payload content, and protocol behavior. The challenge is to handle the high volume and variability of network traffic while maintaining high accuracy, low latency, and scalability.

The goal of traffic classification is to provide network administrators and security analysts with a better understanding of the network traffic and to enable them to enforce policies, detect anomalies, and mitigate threats. The applications of traffic classification range from QoS provisioning to traffic engineering, from intrusion detection to content filtering, and from network forensics to network optimization.

3. PROPOSED METHODOLOGY

The process of classifying the different types of network traffic that are being sent across a network is known as network traffic classification. This is done through analyzing various attributes of network packets, such as the source and destination IP address, port numbers, protocol type, and packet size. There are many reasons why network traffic classification is important. It helps the network administrators to monitor and manage network traffic, detect security threats, optimize network performance, and prioritize network traffic for specific

applications or users. There are different approaches to network traffic classification, including rule-based, statistical, and ML-based methods. The proposed traffic classification model is shown in figure 1.

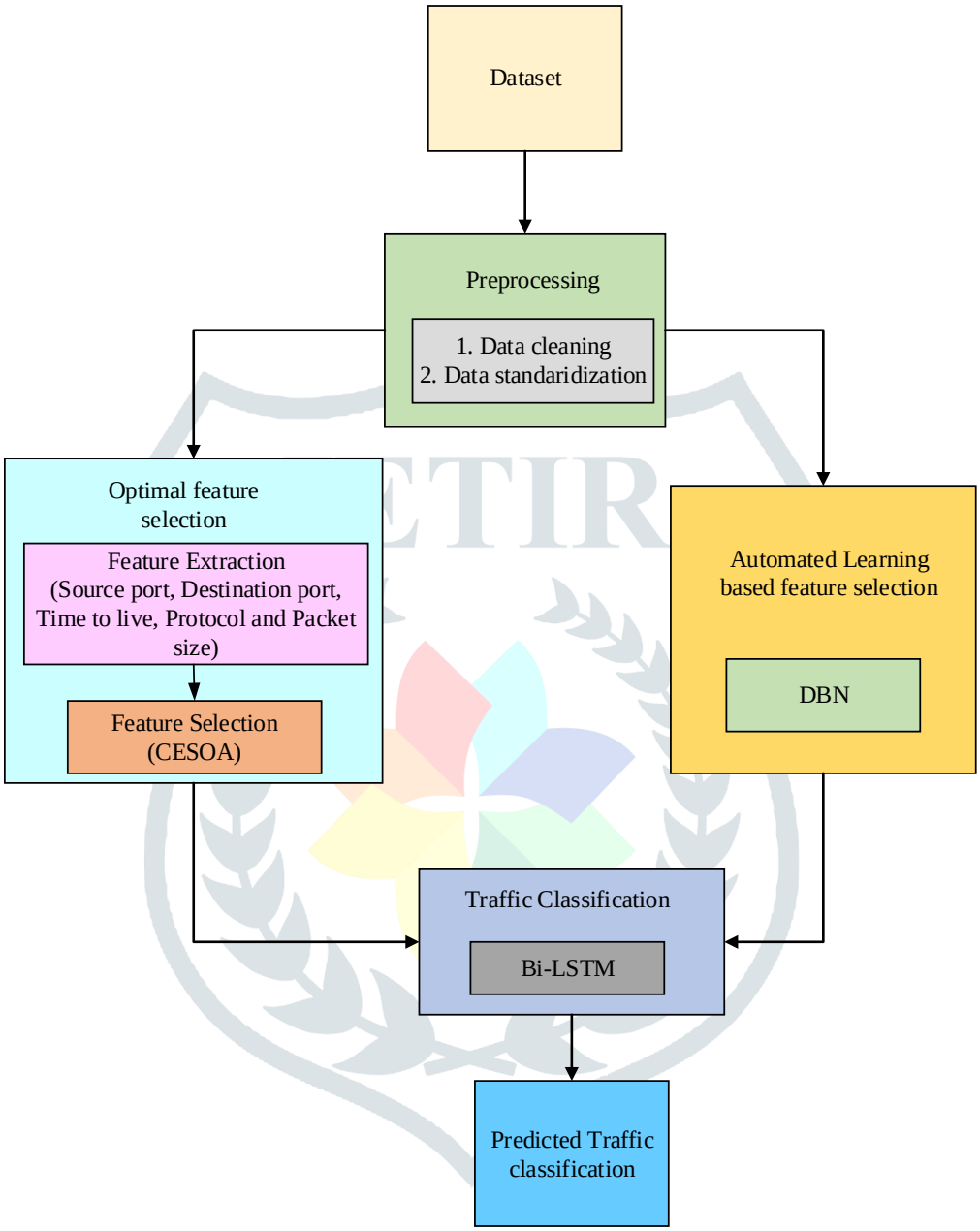

Figure 1: Block diagram of the network traffic classification model

In this proposed model, the input data is taken from the dataset which is preprocessed using data cleaning and data standardization. Then the feature selection is performed using both machine learning and deep learning techniques. The optimal feature selection is performed using the ML technique and the Automated learning is used for selecting the features in the DL technique. In the ML technique, the feature extraction and feature selection are separately performed using different techniques and in the DL technique, a single DBN is used for

both the feature extraction and selection operation. After selecting the features, both features are concatenated and the traffic classification is performed using the Bi-LSTM model.

3.1. Pre-processing

Preprocessing is required since the dataset needed to be cleaned up because it had noise and outliers. Estimating missing values and removing noise, such as anomalies, normalization, and checking for unbalanced data are all included in the preprocessing stage.

3.1.1. Data cleaning

Data cleaning is an important step in the data preprocessing stage for any machine learning project, including traffic classification. Here are some techniques that is used for data cleaning in traffic classification:

- Handling missing values: Traffic data may have missing values due to various reasons, such as sensor malfunction or data transmission issues. It is important to identify and handle missing values appropriately, either by imputing them with a reasonable value or by removing the corresponding samples.
- Removing redundant features: Traffic data may contain redundant features that do not contribute to the classification task or may even introduce noise. These features are identified and removed using feature selection techniques.
- Normalization and scaling: Traffic data may have features with different scales or ranges. Normalizing and scaling is help to bring all features to the same scale, making them comparable and improving the performance of the classification model.

3.1.2. Data standardization

Data standardization, also known as normalization, is a technique used to transform the values of a dataset to have a standard scale. This is often done to compare variables that are measured in different units or have vastly different ranges. The formula for standardization is given in Eq. (1),

$$z = (x - \mu) / \sigma \quad (1)$$

When the original value is x, the standardized value is z, the standard deviation is σ, and its dataset's mean is μ. To standardize a dataset, simply apply this formula to each value in the dataset. The result is a new dataset with a mean of 0 and a standard deviation of 1.

3.2. Feature Extraction

After calculating the missing values, it is necessary to find the critical traits that strongly and favorably correlate with those critical to traffic classification. Building a robust traffic classification is prevented through extracting the vector features, which removes irrelevant and worthless features for prediction.

3.2.1. Source port

In network traffic classification, the source port refers to the port number on the sending device or application that the data is being sent from. A port number is a 16-bit integer value that is used to identify specific applications or services running on a device.

3.2.2. Destination port

In network traffic classification, the destination port refers to the port number on the receiving device or application to which the data is being sent to. Like the source port, a port number is a 16-bit integer value that is used to identify specific applications or services running on a device.

3.2.3. TTL

In network traffic classification, A packet's IP header contains a field called TTL that contains information about how many routers it is pass through in total before being deleted. The TTL field helps prevent packets from looping indefinitely in a network, which causes network congestion and reduce network performance.

3.2.4. Protocol

In network traffic classification, the protocol refers to the rules and procedures that govern the communication between devices over a network. In the context of network traffic, the protocol is usually identified using a protocol number in the IP header of a packet, which specifies which transport protocol is being used to transmit

the data, such as ICMP, UDP, TCP, etc. Each protocol has its own set of rules and procedures for transmitting data and responding to errors.

3.2.5. Packet size

Packet size is measured in terms of the total size of the packet, including the headers and payload, or just the payload size. The payload size is usually the more relevant feature for traffic classification, as it reflects the actual data being transmitted. When using packet size as a feature for traffic classification, it's important to consider the distribution of packet sizes for each traffic class. Different traffic classes may have distinct patterns in their packet size distributions, which is used to differentiate between them.

3.3. Feature Selection using the Seagull optimization algorithm

Seagulls are regarded as marine birds. Seagulls come in a variety of species, and they all have different physical traits including length and mass. According to studies, seagulls are highly intelligent creatures that allegedly use their feet to make rain-like noises to entice earthworms to the surface. Seagulls drink both freshwater and saltwater, unlike the majority of creatures. The most crucial behavioural characteristics of seagulls are their assault and migration habits. When searching for the most plentiful food sources, seagulls make use of their seasonal travels, also known as their migration behaviour or pattern.

Seagulls typically reside in colonies. They hunt for their prey and attack it using their knowledge. The primary traits of seagulls are their migratory habits and tendency for aggression. Seagulls migrate from one region to another throughout certain seasons in search of the tastiest and most plentiful food sources that is give them enough energy. In the following manner this behaviour is described:

- They migrate in groups and move about. In order to prevent collisions, seagulls take various starting locations.
- Seagulls in a flock is moved in the direction of the most viable candidates for survival, or one whose fitness value is lower than the rest.
- The other seagulls are adjusted their starting places based on the fittest seagull.

When birds travel across the water to get from one spot to another, seagulls frequently attack them. When attacking, they might move in a spiral-like natural shape. When migrating birds travel over sea water inorder for moving from one location to another, seagulls used commonly attack them. During the time of attacking, the seagulls used to naturally transform their spiral form. Figure 2 represents a conceptual model of these characteristics.

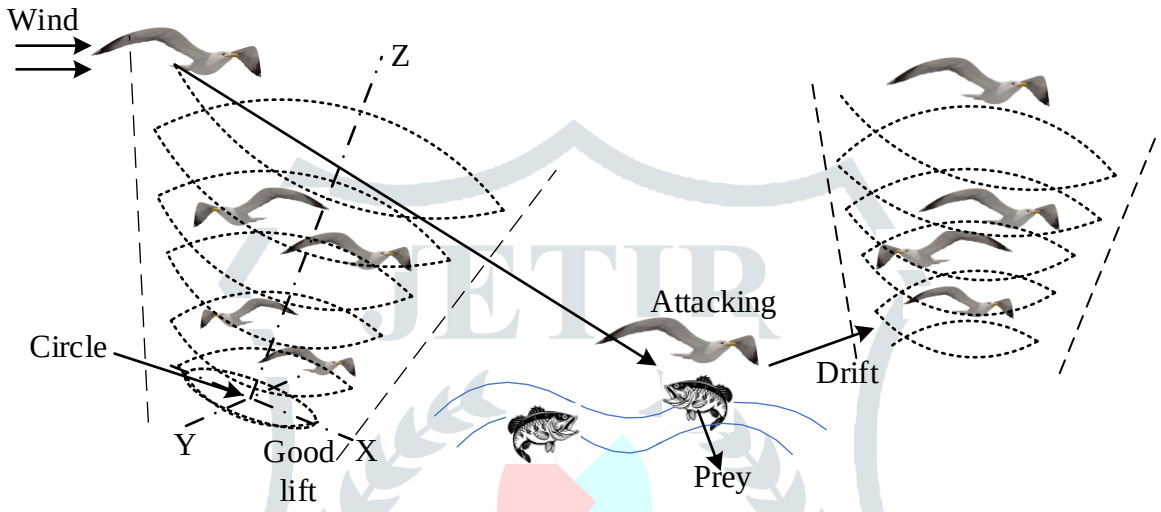

Figure 2: Behaviors of seagulls during migration and during attacks

These traits would be demonstrated in a way that would enable them to stay focused on the improvised goal. This leads to the development of a novel kind of optimization method. This study focuses mostly on the seagulls' natural characteristics.

3.2.2.2 Mathematical model

It has been explored how seagulls migrate and how their prey's attacking strategies are mathematically structured.

i. Migration (Exploration)

The algorithm mimics the movement of a flock of seagulls as they fly from one spot to another. At this point, a seagull would be able to fulfill all three of their requirements.

• **Avoiding the collisions:** During the process of identifying the new search agent's position, a new extra variable *A* was introduced in Eq. (2) and used to stop collisions from happening with nearby seagulls, shown in figure 3.

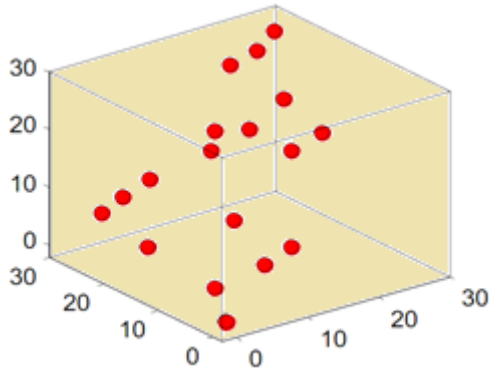

Figure 3: Avoiding collisions between search agents

$$\overrightarrow{C_s} = A \times \overrightarrow{P_s}(u) \quad (2)$$

Where the present iteration is denoted as$u$, A denotes the movement characteristics of the search agent on the available search space, $\overrightarrow{C_s}$means the location of the search agent where it won't be able to collide with other search agents, and the search agent's current position is represented as $\overrightarrow{P_s}$. The value of A is given in Eq. (3),

$$A = f_c - \left(u \times \left(f_c / Max_{iteration}\right)\right) \quad (3)$$

Where: $u = 0,1,2, \ldots, Max_{iteration}$

Where $f_c$was offered as a means of regulating the frequency of the employing variable, f_c was then linearly minimized towards 0. The value of $f_c$was set to 2 in the current job.

In the traditional Seagull Optimization algorithm, the f_c-factor is linearly decreased from 2 to 0. This helps to update the A factor to balance primarily. To improve the performance of Seagull Optimization, a chaotic strategy has been utilized in place of the f_c-factor. Therefore, adopted four different chaotic values to optimize the performance. Moreover, using the chaotic strategy ensures the optimization does not fall into the problem of local optimization. Mathematical formulas have been considered to improve the A factors instead of f_c is given in Eq. (4) to Eq. (7).

$$f_c(1) = 1.95 - 2t^{\frac{1}{4}}/T^{1/3} \quad (4)$$

$$f_c(2) = 1.95 - 2t^{\frac{1}{3}}/T^{1/4} \quad (5)$$

$$f_c(3) = \left(-\frac{3t^3}{T^3}\right) + 1.5 \quad (6)$$

$$f_c(4) = \left(-\frac{2t^3}{T^3}\right) + 1.5 \quad (7)$$

The aforementioned chaotic values are alternatively utilized in the place of f_c; therefore, a highly efficient optimal solution is obtained.

• Moving in the direction of the best neighbor

Later, in order to minimize collisions between the neighbouring herds, the herd's search agents travel closer to the other herds given in Eq. (8) in order to identify which herds are the most cooperative, shown in figure4.

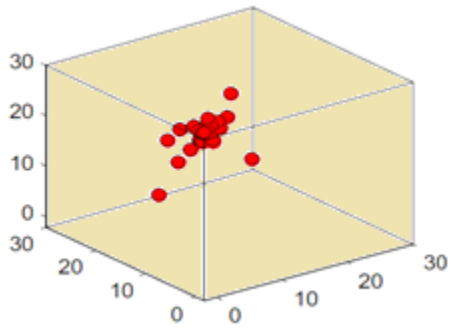

Figure 4: Agents of search moving in the direction of the best neighbour

$$\overrightarrow{M_s} = B \times \left(\overrightarrow{P_{bs}}(u) - \overrightarrow{P_s}(u)\right) \quad (8)$$

Where the search agent $\overrightarrow{P_s}$ is positioned towards the search agent $\overrightarrow{P_{bs}}$, the fittest seagull, in the areas where $\overrightarrow{M_s}$describes the search agents. The B behaviour was randomly generated, which is what has proven to be a successful balance between exploitation and exploration. The value of B is calculated using Eq. (9),

$$B = 2 \times A^2 \times rd \quad (9)$$

Where rd was a random number that would fall between [0, 1] approximately.

• Stay in close to the top search agent

The search agent is also be able to update its location in reference to the provided ideal search agent, as seen in figure 5.

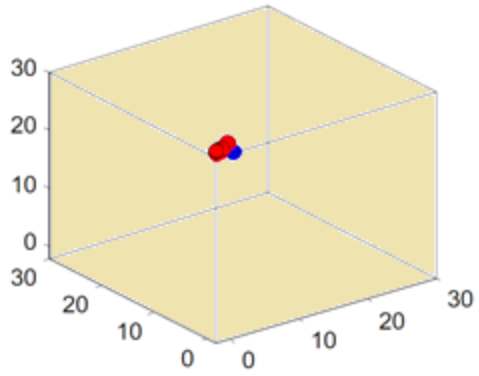

Figure 5: Movement in the direction of the optimal search agent

The updated location is given in Eq. (10),

$$\overrightarrow{D_s} = \left|\overrightarrow{C_s} + \overrightarrow{M_s}\right| \quad (10)$$

It is recognized as the best seagull, whose fitness value was minimal, where $\overrightarrow{D_s}$ was marked as the separation of the search agents and the distance to the most suitable search agent.

ii. Attacking (Exploitation)

The purpose of the exploitation isto capitalize on both the experience and the prior success of the search. The seagulls had the ability to often change their attacking angle and speed during the period of migration. They assist themselves in maintaining their height by using both their body weight and their wings. The seagull bird attacks its prey in a spiral motion while it is in the air, shown in figure 6.

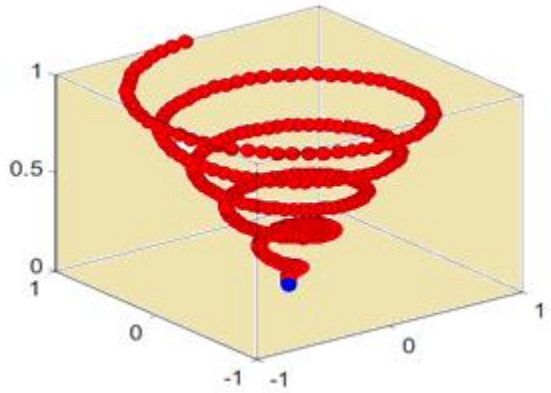

Figure 6: Seagulls' natural attacking methods

The u, v, and w planes were used to show the following properties in Eq. (11), Eq. (12), and Eq. (14) respectively.

$$u' = r \times \cos(k) \quad (11)$$

$$v' = r \times \sin(k) \quad (12)$$

$$w' = r \times k \quad (13)$$

$$r = q \times e^{ks} \quad (14)$$

Where r denoted the diameter of each spiral turn and k denoted a random value falling within the range of $[0 \leq k \leq 2\pi]$. Where e was used as the base for the natural logarithm,q *and* s were the constants used to define the spiral shape. Eq. (15) is used to determine the search agents' most recent location.

$$\overrightarrow{P_s}(u) = \left(\overrightarrow{D_s} \times u' \times v' \times w'\right) + \overrightarrow{P_{bs}}(u) \quad (15)$$

Where $\overrightarrow{P_s}(u)$ changes the position and then saves the best response from various search agents.

The population is produced randomly at the start of the seagull optimization algorithm (SOA) that is being presented. The search agents were able to inform the positions according to the most effective search agent during iterations. The A was linearly minimized from $f_c$to 0. The seamless transition between exploitation and exploration was due to the variable B. The seagull optimization algorithm has earned a reputation as the global best optimizer due to its superior exploitation and exploration capabilities.

3.4. Automated learning using DBN

Deep learning performs deep feature extraction after discovering the fundamental expression of the data by mapping expression for layer-by-layer nonlinearity. For fault diagnosis, good feature extraction is crucial. The DBN is a common DL approach that is capable of successfully extracting feature information from the original information.

3.4.1. RBM

The RBM uses high-dimensional space to address the feature division problem. Each layer's individual units are connected in both directions, but there is no link within a layer itself. A perfect match exists between the input vector and the RBM's visible vector. The characteristic that is taken out of the visible vector is the hidden vector. The following Eq. (16) is a definition of the RBM unit configuration's energy:

$$E(v,h) = -\sum_{i=1}^{I} v_i x_i - \sum_{j=1}^{J} y_j h_j - \sum_{i=1}^{I}\sum_{j=1}^{J} h_i v_i W_{ij} \quad (16)$$

Where W_{ij} is the connection weight between the visible layer unit i and the hidden layer unit j, and x_i and y_j are the offsets of the v_i and the h_j, respectively. The following is how Eq. (17) represents the visible layer's conditional distribution function.

$$P(h_j = 1|v) = \sigma\left(\sum_i w_{ij} v_i + y_j\right) \quad (17)$$

The following Eq. (18) describes the way the hidden layer's conditional distribution is expressed:

$$P(h_j = 1|h) = \sigma\left(\sum_i w_{ij} h_j + x_i\right) \quad (18)$$

Where the activation function σ is typically a sigmoid function. The combined probability distribution function between the hidden layer and the visible layer is described in the following equation (19).

$$P(v,h) = \frac{1}{S} e^{-E(v,h)} \quad (19)$$

Where S is derived through summing all feasible hidden and visible vectors, as shown in Eq. (20),

$$S = \sum_v \sum_h e^{-E(v,h)} \quad (20)$$

The buried layer neuron $h$is determined using Eq. (17) the input v. Using Eq. (18) it is possible to recreate the visible layer once the hidden layer has been retrieved. It is possible to recreate the visible layer by setting each v_i in Eq. (18) to 1.

3.4.2. Automated Learning based feature extraction using DBN

The DBN training approach works by extracting deep representation details from the initial features. Figure 7 illustrates a DBN structure.

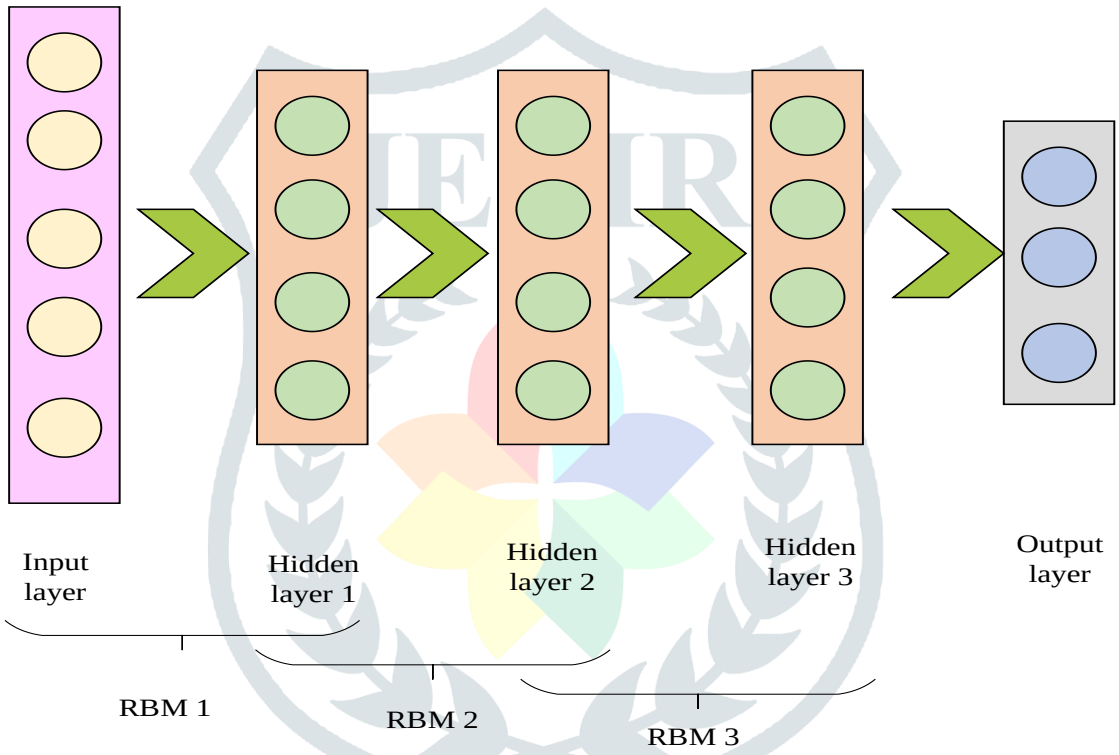

Figure 7: Layered architecture of the DBN model

The DBN is built through stacking several RBMs, and training is done from low to high layers, which is shown in Eq. (21) to Eq. (23).

$$\Delta w_{ij} = \varepsilon(\langle v_i h_j \rangle_{P(h|v)} - \langle v_i h_j \rangle_{rec}) \quad (21) \qquad \Delta y_j = \varepsilon(\langle h_j \rangle_{P(h|v)} - \langle h_j \rangle_{rec}) \quad (22)$$

$$\Delta x_i = \varepsilon(\langle v_i \rangle_{P(h|v)} - \langle v_i \rangle_{rec}) \quad (23)$$

Where learning rateis represented using ε and*rec* is a representation of the expectation of the partial derivative under the reconstructed model distribution. The DBN network is trained in reverse using the back-propagation technique, and each connection's weight is adjusted during the fine-tuning phase to enhance feature extraction and lower diagnostic error. Learning with guidance is a component of the entire fine-tuning phase. Following careful adjustment, the final RBM's feature extraction matches the DBN's feature extraction.

3.5. Bi-LSTM

The capacity of the bidirectional LSTM network to communicate weights has allowed it to become well-established in the field of sequence analysis methods. It iscomprehended long-term dependencies and effectively deal with the vanishing gradient problem. The bi-directional LSTM, as opposed to the traditional unidirectional LSTM, is capable of recording dynamic data from both earlier and later parts of the feature sequence. A backward LSTM layer and a forward layer are both present in the bi-directional LSTM network, as depicted in Figure 8.

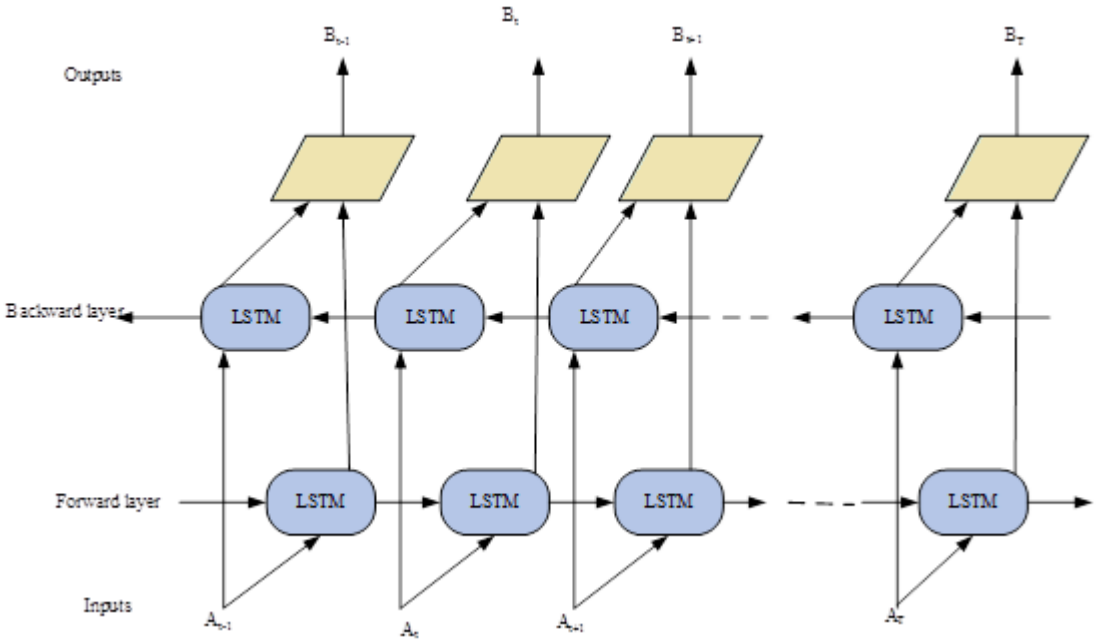

Figure 8: Architecture of the Bi-LSTM

The forward layer's hidden output sequence, h_t, is constructed using the selected features from time index 1 to t. While using an features from time index t+1 to the end, the backward layer's hidden output, denoted usingh_t, is calculated. By using the conventional LSTM theory, both hidden output states are updated. To achieve information storage, LSTM needs three specially constructed gates. In the original architecture, the current input

and the previous hidden layer output are both necessary for updating the cell output state. Also, they had a peephole contact and were using the previous cell state as a parameter. Inp_t is the input stage,c_{t-1} is the prior cell output stage, andh_{t-1} is the prior hidden stage at time t. Eq. (24) to Eq. (25) is used to calculate the outputs of three gates. The input gate Inp_t determines whether Because of the Inp_t, the status was modified, the forget gate Fg_t determines whether the c_{t-1} was retained, and the output gate Out_tdetermines whether the h_{t-1} was transmitted to the subsequent cell,A_t is the potential candidate to update the memory cell at each timestamp t. Eq. (27) to Eq. (29)is used to calculate the current hidden state $h_t$and the output of the current LSTM cell c_t.

The processes inside the LSTM cell may be expressed as follows given the current input vector A_t, the most recent hidden state h_{t-1}, and the most recent memory cell state c_{t-1}.

$$Inp_t = \sigma(I_{A.Inp}A_t + R_{h.inp}h_{t-1} + b_v(inp) \quad (24)$$

$$Fg_t = \sigma(I_{A.Fg}A_t + R_{h.Fg}h_{t-1} + b_v(Fg) \quad (25)$$

$$Out_t = \sigma(I_{A.Out}A_t + R_{h.Out}h_{t-1} + b_v(Out) \quad (26)$$

$$m_t = \sigma(I_{At}A_t + R_{hc}h_{t-1} + b_v(m) \quad (27)$$

$$c_t = Fg_t * c_{t-1} + m_t.Inp_t \quad (28)$$

$$h_t = Out_t.tanh(c_t) \quad (29)$$

Where Inp_t, Fg_t, Out_t, and m_t stand for the input, forget, output, and input modulation gates, respectively, at time t. Input weights, recursive weights, and bias vectors are denoted using the letters I_A, R, and b_v.Eq. (30) contains the hidden layer's output,

$$h_t = \sigma(R_h[\overrightarrow{h_t}, \overleftarrow{h_t}] + b_h) \quad (30)$$

Where b_h is the weight bias of the hidden layer. The Bi-LSTM isutilizing data from earlier and later sequences, which aids in proper classification of network traffic.

4. RESULT AND DISCUSSION

In this section, the results obtained for the proposed model are compared with the existing models like SVM, MLP, LSTM, and CNN. The implementation is performed using MATLAB software.

❖ **Dataset Description (Labeled Network Traffic flows dataset with 141 Applications)**

In this dataset, there are 50 features. Each time an IP flow is formed using a network device, its data is recorded, including ports, the source and destination IP addresses, interarrival periods, packet sizes, flow durations, and the class of layer 7 protocol (application) the flow employed. Numerical types make up the majority of the characteristics, while nominal kinds do exist as well. The various PCAP files were processed using the computation of flow statistics, a program made to carry out the aggregation of packets into flows, Flow Labeler, and the labelling of the flows with their respective applications through the nDPI library, to obtain the dataset displayed here.

4.1. Performance metrics

The performance metrics like Accuracy, Time elapsed, MSE, RMSE, and MAE.

i) Accuracy

Accuracy is a common indicator for assessing the effectiveness of network traffic classification models. It represents the percentage of correctly classified instances out of the total number of instances. Using the confusion matrix, Eq. (31)is calculate the accuracy as follows,

$$Accuracy = \frac{(TP + TN)}{(TP + TN + FP + FN)} \quad (31)$$

ii) Elapsed Time

Elapsed time for network traffic refers to how long it takes for data to be transferred between devices via a network. The elapsed time for network traffic is affected through various factors, including the distance between

the devices, the speed of the network, the amount of data being transmitted, and any network congestion or errors. It is shown in Eq. (32,

$$Elapsed\ time\ =\ End\ time\ -\ Start\ time \tag{32}$$

iii) MSE

With regard to network traffic, MSE is used to measure the accuracy of predictions for the time it takes for network traffic to travel from one point to another. This is useful in assessing the performance of network protocols, routing algorithms, and other network components. The formula for MSE is given in Eq. (33),

$$MSE\ =\ 1/n\ *\ \Sigma\ (y_i\ -\ \hat{y}_i)^2 \tag{33}$$

Where y_i is the actual value for the i^{th} data point, n is the number of data points, and the predicted value for the i^{th}data point is $\hat{y}_i$.

iv) RMSE

The reliability of models that forecast how long it takes network traffic to transit between two points is assessed in the context of network traffic using RMSE. This is helpful in a variety of applications, including quality of service (QoS) management, traffic engineering, and network optimization.

$$RMSE\ =\ \sqrt{(1/N\ *\ sum((actual\ -\ predicted)^2)} \tag{34}$$

v) MAE

In order to calculate MAE for time for network traffic, first require a collection of anticipated values and actual values for the traffic at various time intervals. In order to get the MAE, compute the absolute difference between each anticipated value and its matching actual value. The formula for MAE is:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - x_i|$$

Where n is the total number of data points, y_i is the actual value of the dependent variable for the i^{th} data point, x_i is the predicted value of the dependent variable for the i^{th}data point, | | denotes the absolute value operator and Σ denotes the sum operator over i=1 to n.

4.2. Comparison of the performance metrics

The suggested method's performance measures are compared with those of already-used methods like CNN, LSTM, SVM, and MLP. The comparison is given in table 1.

Table 1: Comparison of the performance metrics

Techniques	Accuracy(%)	Time Elapsed (S)	MSE (%)	RMSE (%)	MAE (%)
Proposed	98.37	2.31	0.02	0.03	0.02
CNN	92.32	4.46	0.17	0.21	0.34
LSTM	87	7.31	0.46	0.69	0.57
SVM	85.28	12.62	1.32	2.86	3.14
MLP	82.91	16.32	1.91	4.21	3.56

Based on the table 1, the Proposed technique outperforms the other techniques in terms of accuracy (98.37%), the time elapsed (2.31 seconds), MSE (0.02%), RMSE (0.03%), and MAE (0.02%). The CNN technique has the second-best performance with an accuracy of 92.32% but takes longer time elapsed (4.46 seconds) and has higher error metrics (MSE, RMSE, and MAE) compared to the Proposed technique. The LSTM, SVM, and MLP techniques have lower accuracy and longer time elapsed with higher error metrics compared to the Proposed and CNN techniques.

i) Accuracy

This metric counts how many cases out of all the examples were successfully categorised. The technique will perform better the more accurate it is. Figure 9 shows the accuracy comparison between the existing techniques and the proposed model.

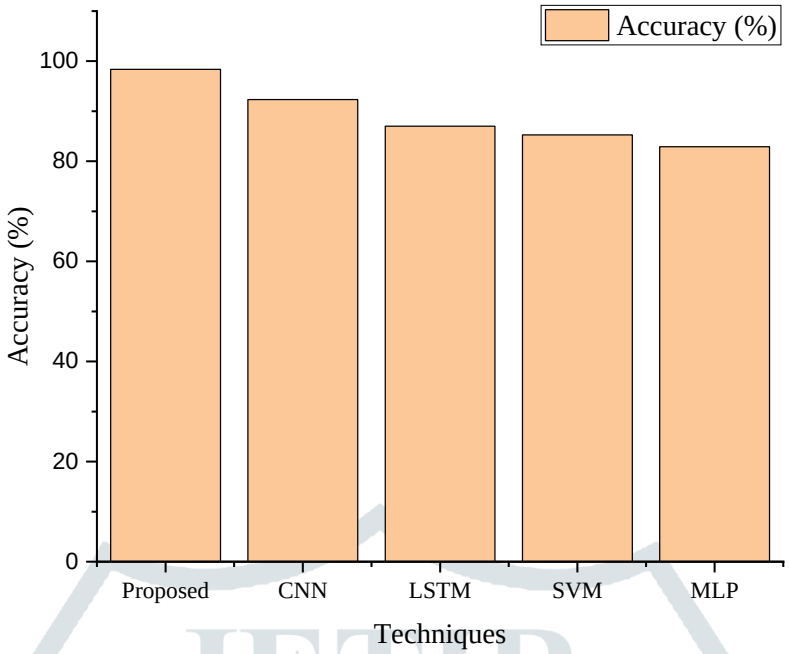

Figure 9: Comparison of Accuracy between the proposed and the existing techniques

The proposed technique has the highest accuracy compared to the other existing techniques like CNN, LSTM, SVM, and MLP.Which represents that the proposed model is an efficient one compared to other existing methods.

ii) Time Elapsed

This metric represents the time taken by each technique to complete its training and testing phases. The lower the time elapsed, the faster the technique. The time elapsed comparison between the suggested model and the currently used strategies is shown in Figure 10.

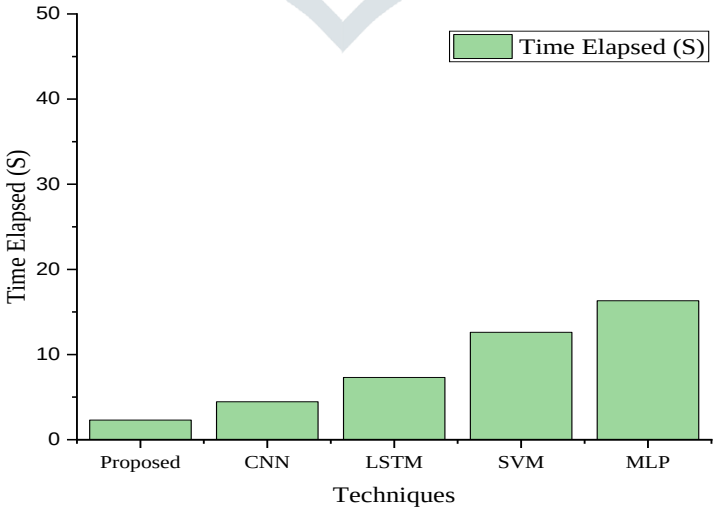

Figure 10: Comparison of time elapsed between the proposed and the existing techniques

When compared to other methods like CNN, LSTM, SVM, and MLP, the suggested method has the lowest time elapsed. It demonstrates that the suggested strategy outperforms other currently employed ones.

iii) MSE

The average squared difference between the expected and actual values is measured by this metric. A lower MSE suggests that the model and the data are more closely matched. Figure 11 compares the suggested model's MSE to that of the already-used methodologies.

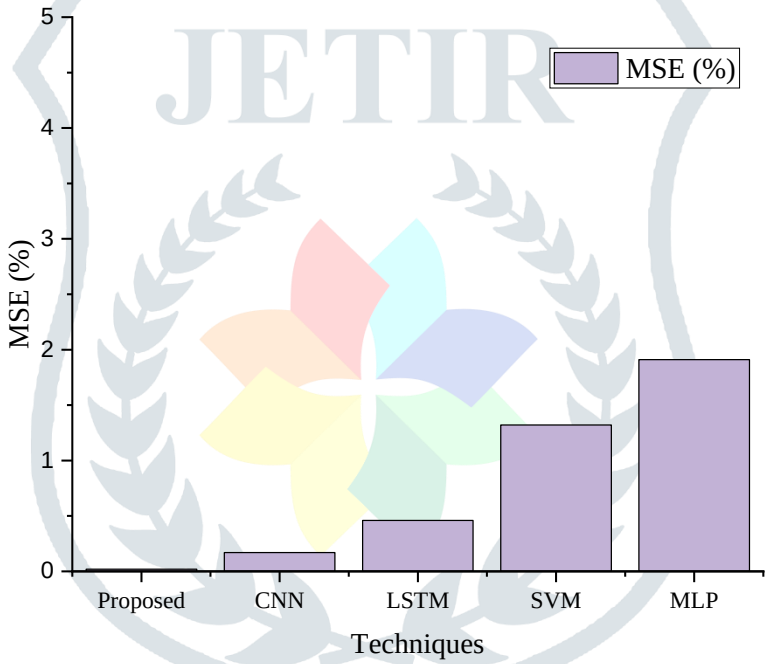

Figure 11: Comparison of MSE between the proposed and the existing techniques

In comparison to other techniques like CNN, LSTM, SVM, and MLP currently in use, the proposed technique has a low MSE value. It shows that the suggested model is effective in comparison to other existing strategies.

iv) RMSE

By measuring the error in the same units as the target variable, this metric, which is the square root of MSE, is used. When the RMSE is lower, the model more closely matches the data. The comparison of the suggested model's RMSE to the currently used methodologies is shown in Figure 12.

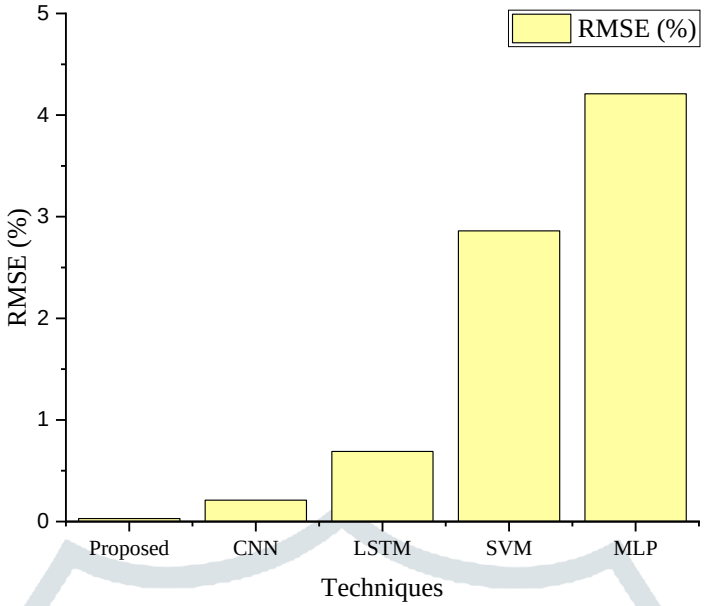

Figure 12: Comparison of RMSE between the proposed and the existing techniques

Compared to other methods now in use, such as CNN, LSTM, SVM, and MLP, the proposed method has a low RMSE value. Indicating that, in comparison to other existing techniques, the proposed model is an efficient one.

v) MAE

This statistic calculates the mean absolute difference between the expected and observed values. An improved fit of the model to the data is shown using a decreased MAE. Figure 13 compares the MAE of the suggested model to the currently used methodologies.

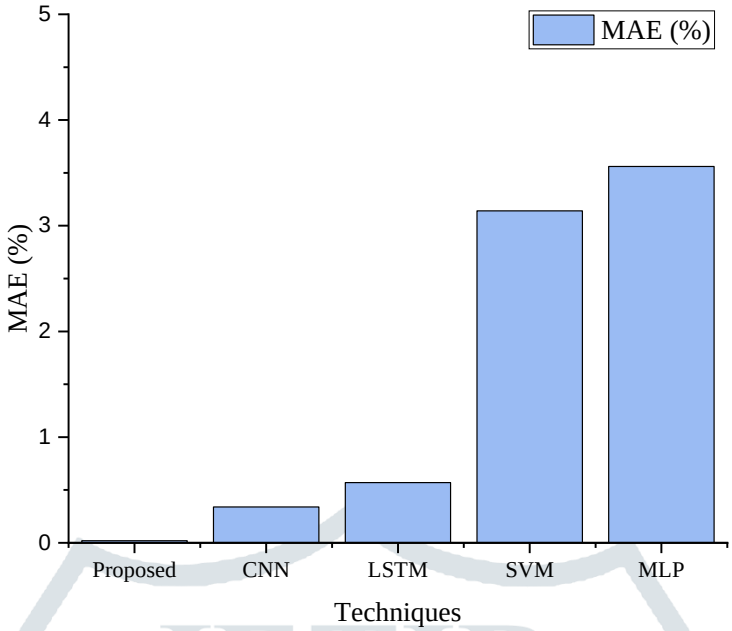

Figure 13: Comparison of MAE between the proposed and the existing techniques

The suggested method has the lowest MAE when measured against various approaches already in use, including CNN, LSTM, SVM, and MLP. That proving the effectiveness of the suggested model in comparison to other available techniques.

5. CONCLUSION

In conclusion, this paper proposed a novel model for traffic classification that combines ML and DL techniques. The proposed model uses a preprocessed dataset and performs feature extraction and selection using both machine learning and deep learning techniques. The outcomes show that the proposed model outperformed the current models with an accuracy of 98.37%. This model provides an effective solution for traffic classification, which is employed in a variety of applications like network security, traffic engineering, and resource allocation. Future research will focus on applying this model to large-scale datasets and exploring its performance under various traffic conditions.

REFERENCE

[1]. Ahmad, B., Jian, W., Ali, Z.A., Tanvir, S. and Khan, M.S.A., 2019. Hybrid anomaly detection by using clustering for wireless sensor network. Wireless Personal Communications, 106, pp.1841-1853.

[2]. Fotohi, R., Firoozi Bari, S. and Yusefi, M., 2020. Securing wireless sensor networks against denial-of-sleep attacks using RSA cryptography algorithm and interlock protocol. International Journal of Communication Systems, 33(4), p.e4234.

[3]. Zeng, Y., Gu, H., Wei, W. and Guo, Y., 2019. $ Deep-Full-Range $: a deep learning based network encrypted traffic classification and intrusion detection framework. IEEE Access, 7, pp.45182-45190.

[4]. Dong, S., 2021. Multi class SVM algorithm with active learning for network traffic classification. Expert Systems with Applications, 176, p.114885.

[5]. Montieri, A., Ciuonzo, D., Bovenzi, G., Persico, V. and Pescapé, A., 2019. A dive into the dark web: Hierarchical traffic classification of anonymity tools. IEEE Transactions on Network Science and Engineering, 7(3), pp.1043-1054.

[6]. Liu, X., You, J., Wu, Y., Li, T., Li, L., Zhang, Z. and Ge, J., 2020. Attention-based bidirectional GRU networks for efficient HTTPS traffic classification. Information Sciences, 541, pp.297-315.

[7]. Setiawan, R., Ganga, R.R., Velayutham, P., Thangavel, K., Sharma, D.K., Rajan, R., Krishnamoorthy, S. and Sengan, S., 2021. Encrypted network traffic classification and resource allocation with deep learning in software defined network. Wireless Personal Communications, pp.1-17.

[8]. Shahraki, A., Abbasi, M., Taherkordi, A. and Jurcut, A.D., 2022. A comparative study on online machine learning techniques for network traffic streams analysis. Computer Networks, 207, p.108836.

[9]. Bhushan, B. and Sahoo, G., 2020. Requirements, protocols, and security challenges in wireless sensor networks: An industrial perspective. Handbook of Computer Networks and Cyber Security: Principles and Paradigms, pp.683-713.

[10]. Aceto, G., Ciuonzo, D., Montieri, A. and Pescapè, A., 2019. MIMETIC: Mobile encrypted traffic classification using multimodal deep learning. Computer networks, 165, p.106944.

[11]. Aceto, G., Ciuonzo, D., Montieri, A. and Pescapé, A., 2020. Toward effective mobile encrypted traffic classification through deep learning. Neurocomputing, 409, pp.306-315.

[12]. Pacheco, F., Exposito, E. and Gineste, M., 2020. A framework to classify heterogeneous Internet traffic with Machine Learning and Deep Learning techniques for satellite communications. Computer Networks, 173, p.107213.

[13]. Ahmed, A.A. and Agunsoye, G., 2021. A real-time network traffic classifier for online applications using machine learning. Algorithms, 14(8), p.250.

[14]. Aceto, G., Ciuonzo, D., Montieri, A. and Pescapé, A., 2021. DISTILLER: Encrypted traffic classification via multimodal multitask deep learning. Journal of Network and Computer Applications, 183, p.102985.

[15]. Iliyasu, A.S. and Deng, H., 2019. Semi-supervised encrypted traffic classification with deep convolutional generative adversarial networks. IEEE Access, 8, pp.118-126.

[16]. Lim, H.K., Kim, J.B., Kim, K., Hong, Y.G. and Han, Y.H., 2019. Payload-based traffic classification using multi-layer lstm in software defined networks. Applied Sciences, 9(12), p.2550.

[17]. Lotfollahi, M., Jafari Siavoshani, M., Shirali Hossein Zade, R. and Saberian, M., 2020. Deep packet: A novel approach for encrypted traffic classification using deep learning. Soft Computing, 24(3), pp.1999-2012.

[18]. Bu, Z., Zhou, B., Cheng, P., Zhang, K. and Ling, Z.H., 2020. Encrypted network traffic classification using deep and parallel network-in-network models. IEEE Access, 8, pp.132950-132959.

[19]. Ren, X., Gu, H. and Wei, W., 2021. Tree-RNN: Tree structural recurrent neural network for network traffic classification. Expert Systems with Applications, 167, p.114363.

[20]. Sadeghzadeh, A.M., Shiravi, S. and Jalili, R., 2021. Adversarial network traffic: Towards evaluating the robustness of deep-learning-based network traffic classification. IEEE Transactions on Network and Service Management, 18(2), pp.1962-1976.

[21]. Cherif, I.L. and Kortebi, A., 2019, April. On using extreme gradient boosting (XGBoost) machine learning algorithm for home network traffic classification. In 2019 Wireless Days (WD) (pp. 1-6). IEEE.

[22]. Elnawawy, M., Sagahyroon, A. and Shanableh, T., 2020. Fpga-based network traffic classification using machine learning. IEEE Access, 8, pp.175637-175650.

[23]. Dataset is taken from https://www.kaggle.com/datasets/jsrojas/labeled-network-traffic-flows-114-applications dated on 15/02/2023.