Chapter 7

Resnet50-Based Deep Convolutional Neural Network For Zero-Day Attack Prediction And Detection

7.1 Introduction

The proposed hybrid framework is a combination of complementary strategies to address different aspects of prediction of a zero-day attack. ANN-based Autoencoder, Phase 2 (feature processing) is an input feature reduction method that attempts to reduce the high-dimensional input features and remove the redundancy before sequential learning. Phase 2 and Phase 3 apply the Modified Bi-LSTM to acquire forward and backward temporal relationships among the attack patterns so that are capable of predicting the varying patterns of zero-days precisely. Phase 2 includes Hybrid Game Theory module, to simulate the strategic correlation between attackers and defenders, and help to make risk-conscious predictions in the condition of uncertainty. These elements together with the traditional inertial classifier can overcome the limitations of the traditional inertial classifiers, as an ensemble of them learns small-scale representations, temporal strategies, and strategic actions, resulting in a higher prediction rate and allowing them to resist any new previously unknown zero-day attacks.

The framework that has been proposed in this chapter based on the idea of the Modified Bi-LSTM with Game Theory and Autoencoder-based feature compression suggested predicting the Zero-day attacks. The proposed model was found to have achieved the maximum accuracy of up to 95.4 percent that results in an improved prediction performance compared to the existing classifiers of the ML when compared to the previous models with the capacity to jointly model the temporal attacker behavior and the strategic interaction between an attacker and a defender rather than relying on the contrived patterns of the features.

Despite the fact that has been improved, the stage primarily deals with sequential and strategic properties whereby it can hardly do much to affect deeper structural expressions of an attack behavior. Therefore, the second step extends the framework with deep feature learning via ResNet50 to add value to a greater spatial-behavioral reference and enhanced strength and extrapolation to more complicated zero-day attack styles.

The framework offers a defense against zero-day attacks and is multi-faceted and temporal prediction as an input in Chapter 5 and spatial-adversarial prediction as an input in Chapter 6. Temporal modeling involves the sequential behavior and node-based strategies and spatial-adversarial modeling enables the generalization of the patterns and the threat detection which results in the augmentation of accuracy of detecting the threats, the decrease in false positive, and the augmentation of preparedness.

It is known that the proposed DC-nZDASN is the one to predict the attack occurrence with the notion of a zero-day, however, it can also detect attempted attacks in real-time and minimize the number of the cases of the false positive and false negative. The latency of real time detection was discovered to consist of the time interval between the onset of an attack and its identification to the system, and this was reported in the results section as the exploits being successfully and timely intercepted by the framework that had not been detected previously.

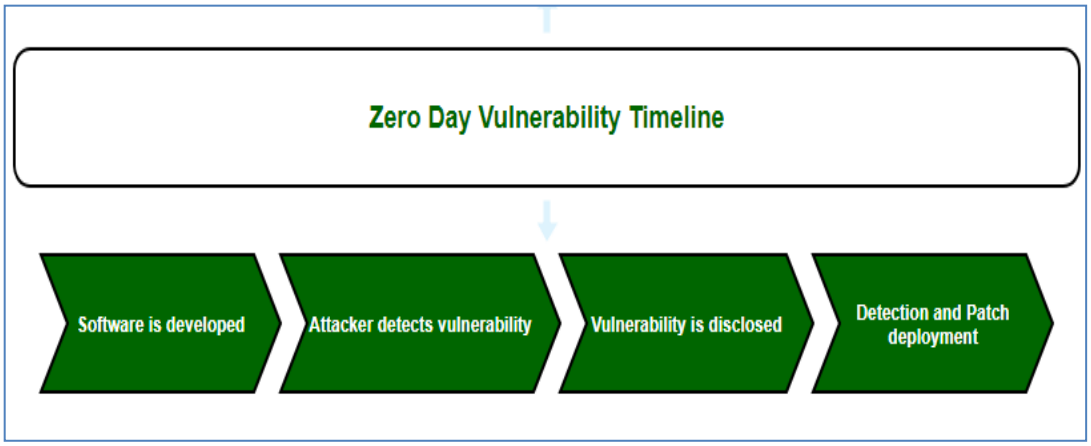

Figure 7.1 Zero-Day Vulnerability Timeline

Figure 7.1 Zero-day attack (ZDA) prediction Zero-day vulnerability chronology is also an important issue of IDS. It exploits the unknown vulnerabilities of the network. The lifecycle of the zero-day attack is presented in Figure 6.2. The chart shows the date on which the programme has been identified by the software manufacturers until the date on which the update has been provided or repaired. Software vulnerabilities are a start of a zero-day attack. It builds objectives of attack and theft. A software vendor of vulnerabilities anticipates bugs and addresses them once it has been attacked. The fix is not issued till the vulnerability exists.

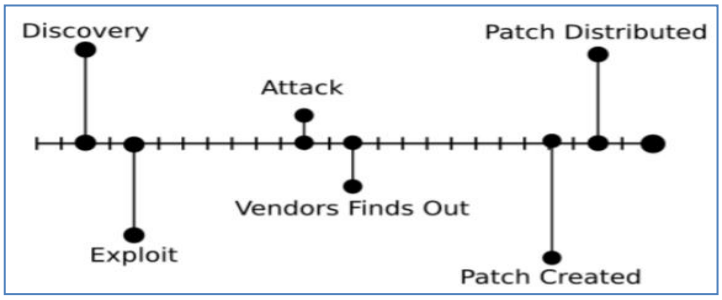

Figure 7.2 Lifecycle of ZDA

Chapter 7 provides a research on zero-day attack prediction and detection by beginning with an introduction in Section 7.1 in order to provide a context regarding the significance of this field in cybersecurity. Section 7.2 is a detailed explanation of the proposed methodology, sub-sections are Zero-Day Attack Prediction and Detection, Pseudocode for Proposed Zero-Day Attack Prediction and Detection and Algorithms, in which the approach, algorithm framework and individual algorithms are described. Section 7.3 provides the experimental setup and results, including a simulated experiment, performance measures to be used in evaluation and a detailed analysis of the results and discussion of its implications. Lastly, the article ends in Section 7.4 with a summary of the main findings and contributions of the zero-day attack prediction and detection framework in strengthening cybersecurity systems against emerging threats.

7.2 Proposed Methodology

Computer security requires the identification of malicious nodes. Among the different malwares, zero-day malware is a challenging malware because it cannot be eliminated by antivirus systems. The current malware detection method used prevails on malware stored capabilities and thus hinders the identification, detection and predicting the zero-day attack where state of the art malware is built to avoid detection. This research would expect a new methodology known as deep-convolutional n -zero-day network in which it generates synthetic malware and trains to distinguish between fake and original malware. The data that is generated in the random distribution is similar rather than identical to original data: it include of altered features as compared to actual data. The model on offer develops different malware characteristics using actual information and path dataset, standard scalar utilizing Decision Tree regressor is used to pre-process the datasets, the finest features are identified by randomly utilizing the Random Forest (RF) using Logistic Regression (LR). The testing and training are done using a convolutional neural network Long Short Term Memory (LSTM) with Residual Network (ResNet50).

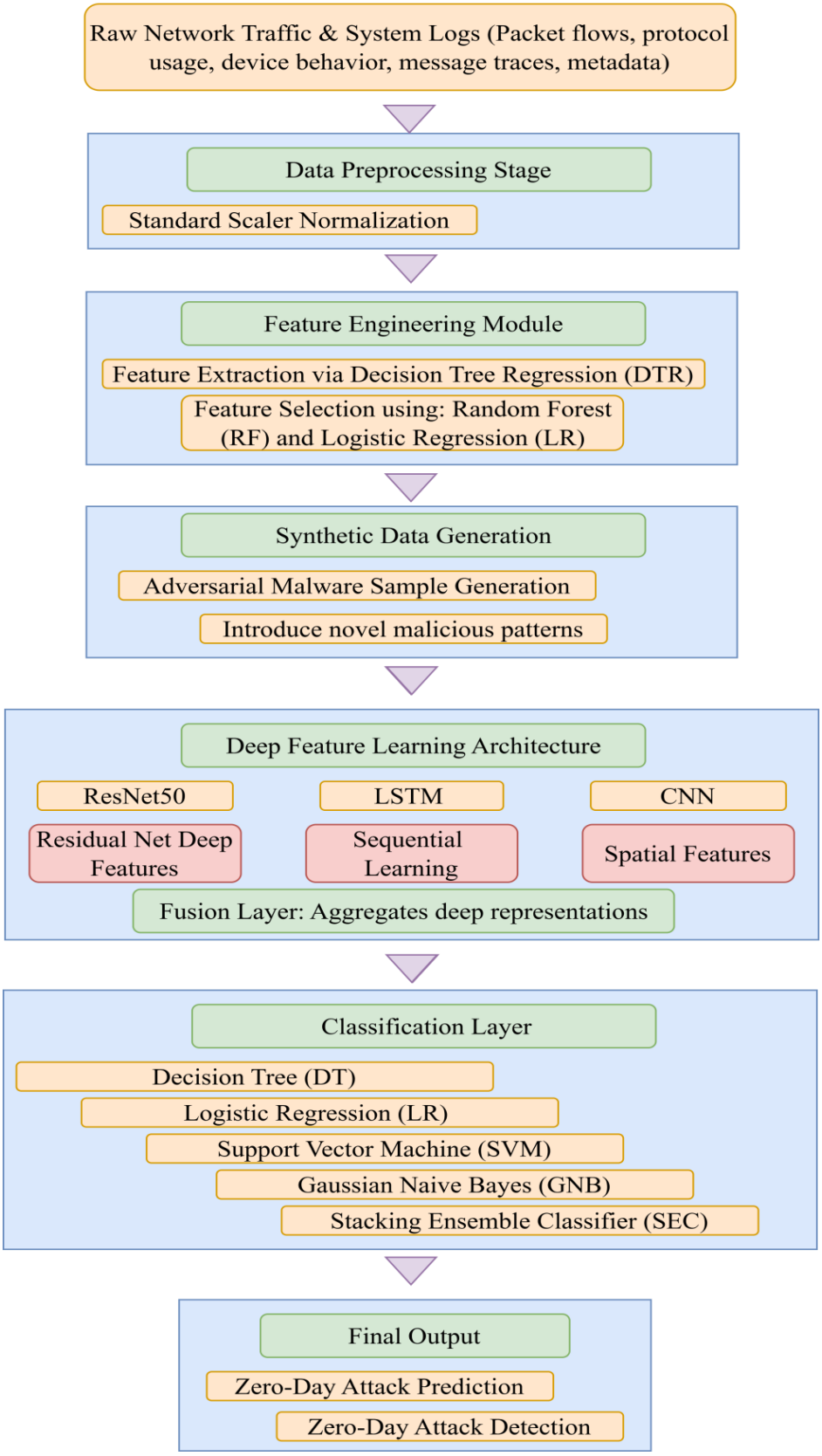

Figure 7.3 Framework for DC-nZDASN Model

To identify which model is more appropriate in predicting zero day attacks, the classification has been performed on various machine learning algorithms that include Decision Tree (DT), Support Vector Machine (SVM), Logistic regression (LR), Gaussian naïve bayes (GNB) and stacking Ensemble classification. The classification accuracy proposed is expected to be higher than the current techniques, and the accuracy is more likely to enhance learning stability. It stands well against zero-day attacks as opposed to others.

The DC-nZDASN model, as proposed, is a hybrid between deep convolutional learning of features, temporal and spatial models to predict and detect zero-day attacks as demonstrated in figure 7.3. It is a process that is instigated by simple scaling and feature selection of raw network traffic and systems logs (Decision Tree Regression, Random Forest, and Logistic Regression). The synthetically generated adversarial malware samples are developed to reproduce the undetectable attack patterns. The deep features in terms of spatial patterns, CNN in terms of hierarchical features and Bi-LSTM in terms of time dependencies are extracted using ResNet50. Finally, there is a Stacking Ensemble Classifier which is a mixture of two or more ML classifiers (DT, SVM, LR, GNB) to provide strong prediction of zero-day attacks.

7.2.1 Robustness against Adversarial Evasion

The DC-nZDASN uses an Adversarial Safety Network (ASN) to deal with evasive attacks, such as input perturbation and obfuscation. With the assistance of synthetic adversarial samples, the hybrid ResNet50LSTM model will be trained, and, thus, the model will not only be trained to achieve high results, in the conditions of the real world, but also, minimize false negatives, in the process of detecting a zero-day attack.

7.2.2 Zero-Day Attack Prediction and Detection

The proposed DC-nZDASN model solves the problem of predicting and identifying zero-day attacks, as well as the capabilities of evasive behavior advanced malware.

Prediction: The model predicts possible zero-day attacks using sequential attack behavior. Bi-LSTM layers are fed with preprocessed and filtered features and learn temporal effects and produce node-specific risk likelihoods of every network entity. The synthetic adversarial malware samples are introduced to provide additional data on previously unseen attack conditions and enhance generalization of the model.

Detection: Detection is used to identify the existing attacks on the fly by deriving spatial and hierarchical patterns. ResNet50 can capture deep feature hierarchies, CNN layers can extract local spatial patterns and the combination of these with Bi-LSTM can help the model to identify bonafide attacks as opposed to artificial adversarial examples. The last categorization is conducted by Stacking Ensemble of ML classifiers (DT, SVM, LR, GNB) in order to guarantee strong detection.

The proposed DC-nZDASN model deals with the problem of prediction and detection of zero-day attacks, and capabilities of evasive behavior designed malware.

Prediction: The model provides predictions of the possible zero-day attacks which rely on the successive behavior of attack. The Bi-LSTM layers are learned on processed and filtered features and learned temporal effects and produce likelihood of risk of each network entity. The artificial adversarial malware samples are provided to provide additional information about the circumstances of attack that have never been observed previously and enhance the process of model generalization.

Detection: Detection is carried out in order to detect the existing attacks on the fly through making spatial and hierarchical pattern. ResNet50 can get deep feature hierarchies, CNN layers may get local spatial patterns and adding them to Bio-LSTM may help the model to identify bonafide attacks in comparison to artificial adversarial examples. A Stacking Ensemble of ML classifiers (DT, SVM, LR, GNB) is used to classify the final classification in order to assure high detection.

Pseudocode for Proposed Zero-Day Attack Prediction and Detection
Initialize ResNet50, CNN, and Bi-LSTM model components. **Preprocess dataset**: standard scaling, Decision Tree Regression for feature mapping, and Random Forest + Logistic Regression for feature selection. **Generate synthetic zero-day samples** and combine with original data. **Train** CNN-BiLSTM-ResNet50 model with adversarial samples to enhance robustness. **Classify** using Stacking Ensemble Classifier to produce predictions for each node. **Output**: Node-level attack probabilities, confidence scores, and real-time detection alerts.

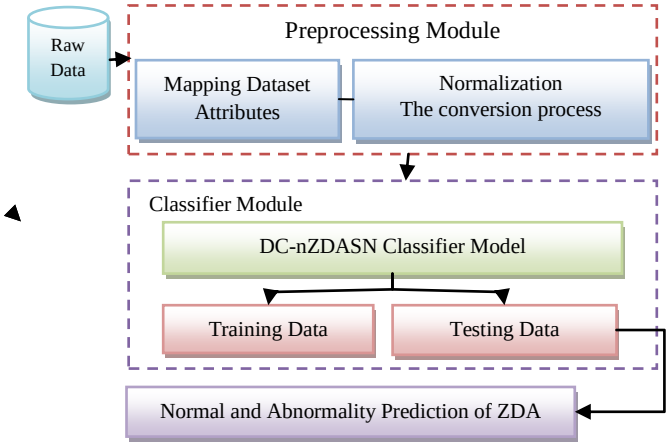

Figure 7.4 Flow Diagram of DC-nZDASN Model

Figure 7.4 indicates the DC-nZDASN model of forecasting and identifying zero-day attacks. Preprocessing of raw network traffic and system logs and the selection of features is done with Random Forest and Logistical Regression. Malware samples that are adversarial are produced synthetically as imitations of hidden attacks. Deep features are extracted by using ResNet50, CNN and Bi-LSTM then the results are combined with Stacking Ensemble Classifier to produce node-level attack probabilities, confidence scores, and real-time detection alerts.

Deep-Convolutional n-Zero-Day Network malware detection is an improved malware detector especially against zero-day attacks that integrate a variety of features and preprocessing strategies to enhance better performance of the model. Behavioral indicators and signature-like traits are important characteristics, and Standard Scalar preprocessing and Decision Tree Regression preprocessing are used to provide the most effective data preparation. Random Forest and Logistic Regression feature selection narrows down the input by underscoring the most pertinent features. The model uses deep pattern recognition, which is based on ResNet50; temporal dependencies, which are based on LSTM; and spatial hierarchies, which are based on CNN; and classification algorithms which include Decision Tree, SVM, Logistic Regression, Gaussian Naive Bayes, and Stacking Ensemble which enhance the accuracy. The model boosts its capacity to detect new threats by creating new features through the creation of false malware cases. This overall methodology yields better classification accuracy of 95.09 indicating that the technique has a greater learning stability and capability against zero day attacks as compared to traditional techniques.

7.2.3 Justification of Methods used in Prediction and Detection Process

Prediction and Detection Approaches: It is Bi-LSTM with temporal dependencies, ResNet 50, CNN with spatial and hierarchical feature extraction and Stacking Ensemble Classifier, the weight of which is able to guarantee the effective prediction of the potential zero-day attacks and the correct detection of the threat in real-time. Artificial malware antagonist helps to improve the generalization to the unknown attack design.

Proposed Model Justification DC-nZDASN is a deep convolutional model which is trained through an adversarial learning culture to acquire rich latent feature of network behavior capable of being used to predict known and unknown zero-days attacks in advance, reduce false positives, and maximize response time in dynamic network applications.

7.2.4 Data Pre-Processing

The most significant process is data pre-processing. A greater number of data analytics include missing values and other contaminants, which might impede the results of the data. The data mining increases the quality and efficiency of the outcome that is generated in the mining. The ML Techniques should be applied to the data effectively in order to achieve the right findings and make adequate predictions. Preprocessing of the Zero-Day data must be done in two stages.

7.2.4.1 Label Encoding Method

Label Encoding is the conversion of labels into a special number form to enable it to be readable by the computer. The machine learning algorithms can then be more skillful in choosing how such labels should be applied. It is a significant step of pre-processing of structured knowledge in supervised learning.

7.2.4.2 Standard Scalar Method

The Z-score normalization (or Standard Scalar method) is a method of data preprocessing, which is used to transform features of numerical data. It involves the scaling of data points to have zero means and unit variance. To put the values into a common scale, the mean value is calculated and multiplied by the standard deviation of the feature; therefore, making it easier to compare and analyze. Standard Scalar apprehends machine learning and statistics to ensure that the characteristics of the various scales do not impact

the learning algorithm in an unreasonable manner. It helps in achieving high quality model performance and strength as it standardizes the features to a distribution.

To standardize the set of features by dividing and normalizing to unit variance. The mean score will be obtained as

$$Ds = z = \frac{x-\mu}{\sigma} \text{ ---------- (7.1)}$$

Most machine learning estimators are based on a standardized data set, otherwise, it can be very bad when the features are not close to the normal distribution.

After data cleaning, the data is then normalized in such a way that it may be used in training and testing of the models. In a data-partitioning, the test set is maintained apart with the training set, where the algorithm is optimized. The underlying logic and algorithms of the features and the values in the training data are used in building the training model. Normalization aims at homogeneity in every aspect.

7.2.5 Feature Engineering Module

The following section describes the feature extraction using decision tree regression and feature importance process through random forest and logistic regression.

7.2.5.1 Decision Tree

Decision tree is a simple classification and regression technique. The decision tree model is structured in the form of a tree, and it is capable of outlining how instances of classification are made according to features. It may be regarded more as an arrangement of the if-then rules, which also may be regarded as conditional probability distributions described in feature space and class space.

Attribute A information gain:

$$Gain(A) = Info(D) - InfoA(D) \text{ ---------- (7.2)}$$

Pre-processing information entropy

$$Info(D) = Entropy(D) = -\sum_j p(j|D)logp(j/d) \text{--------(7.3)}$$

Distribution information entropy

$$InfoA(D) = \sum_{i=1}^{y} \frac{n_i}{n} Info(Di) \text{ --------- (7.4)}$$

```
Function standard_scalar(data):
    mean = calculate_mean(data)
std_deviation =
calculate_standard_deviation(data)
standardized_data = (data - mean) /
std_deviation
    return standardized_data
Function calculate_mean(data):
    mean = sum(data) / length(data)
    return mean
Function calculate_standard_deviation(data):
    mean = calculate_mean(data)
squared_diff = sum((data - mean)^2)
    variance = squared_diff / length(data)
std_deviation = square_root(variance)
    return std_deviation
# Usage
standardized_features =
standard_scalar(features)
```

7.2.5.2 Feature Importance

The second step will deploy the technique of feature importance in the network. The importance of features technique creates the association between the attribute and the target set. The input dataset is subjected to RF method known as the dataset. LR method will predict a characteristic that is the most frequent in the dataset.

The features are then selected after pre-processing the data using a hybrid processing algorithm that integrates RF and LR algorithms.

7.2.5.2.1 Random Forest

Random Forest is used in estimating importance of features and baseline classification. It assists in establishing influential characteristics without a structural adjustment.

7.2.5.2.2 Logistic Regression

Regression is a statistical approach for analyzing and quantifying the connection between dependent variables and one or more independent variables. The logistic regression algorithm works by estimating the parameters of a logistic function, also known as the sigmoid function.

7.2.6 Training with LSTM and ResNet50

This stage aims at training a deep hybrid model that combines information of both temporal sequence learning and spatial feature extraction so as to support the detection of zero-day attacks. Using LSTM and ResNet50, the architecture can both generate the changing attack behavior over time and the complex structural patterns of network traffic data.

7.2.6.1 Bi-LSTM and LSTM

Sequential dependencies are obtained through the use of LSTM and Bi-LSTM to extract sequential dependencies in network traffic data. The standard versions are compared to those of the modified architecture of Bi-LSTM**.**

7.2.6.2 CNN with Bi-LSTM

CNN

As has been discussed above, deep learning and the use of convolutional neural networks in image processing are continuing to grow. CNN is capable of assuming numerous different forms. Even so, as a rule, neural networks that do not have fully connected layers but use convolutional layers instead of them or together with them. The last job will determine the number, type and way of assembly that is required.

M kernels of dimension (K K) create a general convolutional layer. For N-band input x(l), the lth generic convolutional layer produces an M-band output z.

$$z^{(l)} = W^{(l)} * x^{(l)} + b^{(l)} \text{ ----- (7.5)}$$

whose m-th component is a combination of 2D convolutions:

$$z^{(l)}.(m,.,.) = \sum_{n=1}^{N} w^{(l)}(m,n,.,.) * y^{l}(n,.,.) + b^{(l)}(m).\text{------ (7.6)}$$

combination provides CNN with its overall function

$$f(x;\ \emptyset) = fL(fL-1(\ldots f1(x;\emptyset 1),\ldots,\emptyset L-1);\emptyset L)\text{------ (7.7)}$$

where ∅ (∅1,…, ∅L) represents all parameters that must be learned. A cost function, an optimization procedure, and numerous training pairs of input and output reference samples are needed to train the parameters. When gauging how well-anticipated results match up to reference results, the L(∅;∅) cost function is used. An optimization technique is performed to decrease L, and the parameters are adjusted accordingly.

CNN with Bi-LSTM

CNN with Bi-LSTM is an architecture of the artificially intelligent system based on the combination of the Convolutional Neural Networks (CNNs) with the Bidirectional Long Short-Term Memory (Bi-LSTM) networks. It is usually applied in tasks that follow sequences, e.g., natural language processing and speech recognition. The architecture consists of a CNN layer that identifies local patterns and features of the input sequence with the help of convolutional filters and a Bi-LSTM layer that identifies both the past and the future context by processing the sequence forward and backwards. The combination also enables the model to learn complex dependencies as well as learn long-term dependencies of the data. The CNN layer does local feature extraction whereas the Bi-LSTM layer does the sequential dependencies. The Bi-LSTM layer output can be followed by the tasks of classification or sequence generation. Such a hybrid architecture has proven very successful in a number of different applications, which require sequential data analysis, and has become a widespread system of choice when both spatial and temporal insight is needed.

The use of CNN, which could be a ground-breaking computer vision technology, in E.C. prediction and load forecasting by researchers is encouraging due to the encouraging outcomes that researchers have received so far. As a result, the research introduces a CNN- and Bi-LSTM-based hybrid network in energy consumption prediction. The second layer of analysis is Bi-LSTM that follows CNN and identifies and interprets the output sequence of CNN. The assistance of a CNN network makes it easier to extract time-specific features of E.C. variables. It is based on this fact that the Bi-LSTM network is more reliable in predicting electrical conductivity (E.C.). A CNN has convolutional and pooling layers that are not visible.

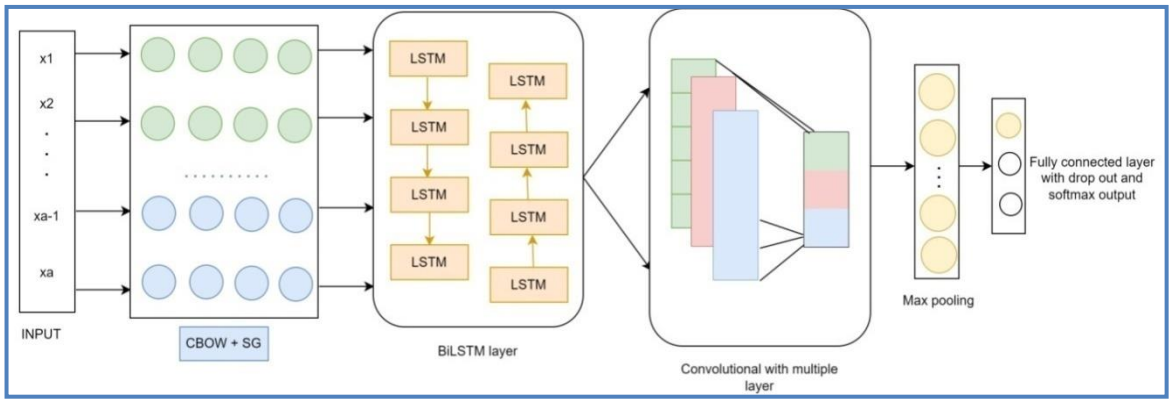

Figure 7.5 CNN with Bi-LSTM

The figure below (Figure 7.5) is the combination of Skip-Gram (SG) and CBOW to effectively learn word embedding and learn semantic relationship through the input corpus and then fed to a Bi-LSTM layer where sequential and contextual relationships are obtained in both forward and backward directions. The outcome is then sent through a multi-layer convolutional neural network (CNN) in a bid to acquire local spatial relations and hierarchical attributes. The feature maps are then flattened after max pooling and subjected to dropout layer to regulate them and the final classification using softmax layer making them suitable in identifying the advanced patterns of zero-day attacks using text.

7.2.7 Applying with ResNet

It is presumed that ResNet is a high-level spatial extractor, provided by pre-trained deep convolutional network. It is an element of profound learning and then optimization and hybridization.

7.2.7.1 The Bi-LSTM with ResNet

In this instance, these methods provide a method of doing precisely that which is the prediction of domestic energy consumption on historical training models.

1The pre-processing of the data involved the use of the mean value to represent the missing values on the column in the research to indicate that the missing values are many in the data. To handle data, which is not related to the standard scale, research pre-process the entire data points with min-max scalar, that are in the range of [0, 1], and the formula serving in this case is given.

$$m_n = \left[\frac{M - M_{min}}{M_{max} - M_{min}}\right] \text{-------- (7.8)}$$

M (max), M (min) and M n are normalized values of the dataset and M is a value at a given time step. Preparation of data will also allow saving time and money on calculations since no important information will be missed.

2. The sliding window model will entail the division of research information into features and labels and the resulting selection of features. The feature in the research was the current value and the past values in the column as the label. The proposed technique has proposed a window size of 60, whereby the non-final values of 60 calculate the feature set of the 61 st value (the label).

3. The architecture of the proposed approach will adopt four network architectural designs namely LSTM, Bi-Bi-LSTM, CNN-LSTM, and CNN-LSTM. Depending on the design choices made during the construction, a model can be characterized by a high or a low network efficiency.

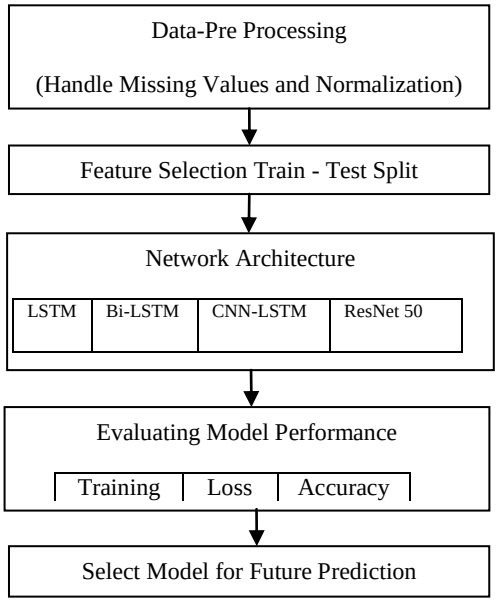

Figure 7.6 Training and Testing Model

The test has a number of procedures that are carried out to demonstrate the effect. This is a parameter where the network performance is influenced with regard to data. All these including dense, CNN and LSTM are all integrated into a single model, CNN-Bi-LSTM. All the layers of CNN begin with time-distributed convolution and finish with a max pooling layer with Relu activation function. After the CNN layer, the output is passed through a time-distributed flattening, the LSTM bilayer and dense layers. CNN-LSTM model contains 75 Bi-LSTM units, a single dense unit, and 64 convolution filters in the layer. max pooling A single kernel is in each layer.

Figure 6.6 outlines the way the proposed method of establishing the use of energy. To summarize, the following are the major points that one should remember:

1) The first stage is when the information is cleansed and organized.

2) The features and labels are retrieved and analyzed and then it is split into training and testing data sets.

3) The network topology of every model is confirmed through the assistance of various performance measures.

4) The best model and architecture can be used to predict the future energy consumption.

7.2.8 Classifications using ML

This step is the final decision step of the suggested DC-nZDASN framework where the extracted and learned features are mapped to labels, i.e., benign, known attack, or zero-day attack. This framework tests the traditional and the state of the art classifiers in order to make them robust, relatively valid, and extrapolate against unknown attack behavior.

The following types of classification can be used and analyzed by the proposed work:

- Decision Tree (DT)- a rule based baseline classifier which is used in the formation of plain decision boundaries.
- Support Vector machine (SVM) - applied to evaluate the capacity of separation in high dimensional feature space margin.
- Gaussian Naive Bayes (GNB) Gaussian Naive Bayes (GNB) is a probability based baseline, and it is making an assumption that the features are conditionally independent.
- Logistic Regression (LR) - applied when making a linear decision and the interpretation is required.
- Stacking Ensemble Classifier - it uses a combination of multiple base learners to increase the stability of classification and overall prediction.
- Modern Hybrid Game Theory and ANN-Autoencoder Modified Bi-LSTM (Chapter 5) - which is used in the resolution of the problem of temporal prediction and strategic threat.
- DC-nZDASN (Proposed) - a deep convolutional based classifier which applies the ResNet50 based transfer learning and adversarial augmented samples in detecting the final zero day attacks.

The framework combines the temporal prediction of the Bi-LSTM model with the GT and the AE model and spatial/behavioral feature learning with DC-nZDASN, which results into a higher detection accuracy and a lower rate of false positives and a greater resistance to the transformation of adversarial strategies in comparison with the standalone classifiers.

7.2.8.1 Decision Tree Algorithm

The decision tree can also be utilized as a starting point classifier and a starting point in data treatment in the prediction phase. It does provide classification based on rules, and it provides a comparative yardstick on which to measure the performance of zero-day attack prediction.

7.2.8.2 Categorization with Linear SVM

SVM is a conventional classification technique, which is used to separate normal and suspicious samples using optimum hyperplanes. Its usage is only compared with the proposed models in terms of performance.

7.2.8.3 Naive Bayes classifier

GNB is an independent-feature based classifier that is based on probabilistic basis. It provides identical results under the comparison of the effectiveness of advanced deep learning models.

7.2.8.4 Logistic Regression

One of the common linear classifiers that are used to approximate probability of attacks is Logistic Regression. It is a zero-day attack prediction baseline, which is applied in predicating attacks comparatively.

7.2.8.5 Ensemble Learning

In the context of the proposed DC-nZDASN, ensemble learning is built on the basis of stacking to improve the classification performance of the zero-day attack prediction and detection. ResNet50, CNN, and Bi-LSTM modules are deep features, which are trained on base classifiers, including Decision Tree (DT), Support Vector Machine (SVM), Logistic Regression (LR), and Gaussian Naive Bayes (GNB). The output of these classifiers is combined with a meta-learner to produce the final prediction which produce more accurate and hardy node-level attack probabilities. The methodology will ensure that the ensemble exploits divergent strong points of the model aspects, minimizing false classification and improving the overall unseen zero-day attacks.

7.2.8.6 Stacking Ensemble

The proposed DCnZDASN framework would rely on a stacking ensemble to enhance the accuracy of prediction in zero-day attacks. The deep features that are obtained

after the ResNet 50, CNN and Bi-LSTM modules are the inputs to various base classifiers, including Decision Tree (dt), Support Vector machine (SVM), Logistic Regression (LR) and Gaussian Naive Bayes (GNB). The output of these base classifiers is combined with the help of a meta-learner that produces the end-node level attack probability and confidence score. It exploits the complementing capabilities of individual classifiers to improve generalization to new patterns of attacks and reduce false classification, without using of linear or statistical time-series assumptions.

7.2.9 Final Prediction and Detection using DC-nZDASN

DC-nZDASN is a deep learning-based and ensemble classification hybrid network that predicts and detects attacks in zero-day attacks. Predictive application works with synthetic samples of malware which are instances of what the future might represent in terms of attack and enhances the learning power of the model to distinguish features against signatures. This is followed by detection, which is conducted by using a combination of ResNet50, LSTM, and CNN, which individually identify hierarchical, temporal, and spatial patterns in the input. Decision Tree Regression, Random Forest and Logistic regression have extraction and selection of the decision tree optimized and final classification is done by a Stacking Ensemble Classifier of various ML algorithms. The combination of this solution enables the model to sensibly categorize known and unknown threats and form an effective advancement over current technology on signature-based technology through active discovery and prediction of zero-day vulnerabilities in real-time.

7.3 Experimental Setup and Results

Prediction threshold of ZDA prediction is done, where rate, bandwidth computation traffic, and mean square error are also examined. The number of active nodes across the network should be summed; when ZD possibilities of nodes are considered to avoid the formation of additional communication. In this case, MSE and threshold prediction are augmented with the additional likelihood of mitigation of ZDA. Table 6.4 illustrates the parameters of the CNN model having a multi-layer convolution layer, a convolution layer, a pooling layer, and fully connected layers, respectively. This table will contain size, padding, activation function step size, and classification results.

Table 7.1 LSTM with ResNet Parameters

Layers	Type	Size	Padding	Activation function	Step	Classification based on feature inputs
Layer 1	Multi-layer convolution	1*1, 3*3, 5*5, 7*7	012	ReLU	1	11 * 11 * 64
Layer 2	Convolution layer	3*3	-	ReLU	1	9 * 9 *64
Layer 3	Multi-layer convolution	1*1, 3*3, 5*5, 7*7	012	ReLU	1	9 * 9 *128
Layer 4	Convolution layer	3*3	-	ReLU	1	7 * 7 * 128
Layer 5	Multi-layer convolution	1*1, 3*3, 5*5, 7*7	012	ReLU	1	7 * 7 *256
Layer 6	Pooling layer	3*3	-	ReLU	2	3 * 3 * 256
Layer 7	Fully connected layer	-	-	Sigmoid	-	256
Layer 8	Fully connected layer	-	-	Sigmoid	-	64
Layer 9	Fully connected layer	-	-	Sigmoid	-	5

Table 7.2 Attack Types and Corresponding Training/Testing Set

S. No	Attacks	Training set	Testing set
1	U2R	Ps, buffer overflow, rootkit, load module	Perl, ps, buffer overflow, xterm, sqlattack
2	R2L	Warezmaster, phf, multi-hop, imap, guess password, ftpwrite, spy, warezclient	Ftpwrite, httptunnel, imap, named, phf, multi-hop, send mail, snmpgetattack, wxlock, snmpguess, warezmaster, xsnoop
3	Probe	Satan, portsweep, nma, ipsweep	Msacn, saint, satan, nmap, portsweep, ipsweep
4	DoS	Neptune, smurf, back, land, pod, teardrop	Udpstrom, smurf, worm process table, teardrop, pod, Neptune, back, land, apache2, mailbomb

In Table 7.2 Four major types of attacks are provided (U2R, R2L, Probe and DoS) and training and testing strategies of the datasets were selected to balance out the performance as well as to be in a position to replicate the results of the proposed DC-nZDASN model.

7.3.1 Simulated Experiment

The data set that was used in this step is based on the data sets that are described in Chapter 4, i.e. Dataset D1 (PATH Dataset) and Dataset D2 (Celosia Zero-Day Attack Dataset). First, 25 attributes containing numerical and binary data are acquired to capture the behavior of networks and characteristics of attacks that may be applied in zero-day attacks.

The feature importance analysis with the assistance of the Boruta and Chi-square test represents the preprocessing in the proposed work. Once this is analyzed, 15 strongest features are selected and it used continuously in chapter 6 (prediction) and chapter 7 (detection using DC-nZDASN). The selected features are applicable in the process of classifying zero-day attack patterns and reducing the redundancy and cost of computation. Polished feature set is then subjected to DC-nZDASN model so as to identify precisely the zero-day attacks.

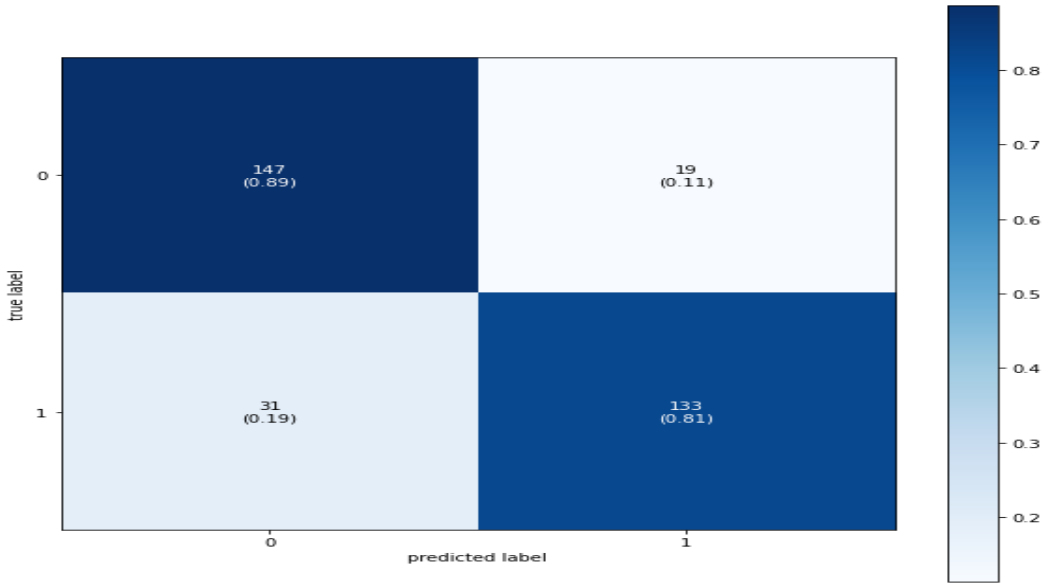

Figure 7.7 Confusion Matrix

The values of the confusion matrix are TP, FP, TN, and FN, as indicated in Figure 7.7. The model estimated TP and TN 28 and 17, respectively and FP and FN 0.

7.3.2 Performance Metrics

The framework of numerous classifications was used as a method of analyzing the research result. Data on the multiclassification were divided into two classification problems and the one-versus-the-rest method was used.

TPd: prediction is category d and the reality is category d.

TNd: the forecast is other classes of category D; the fact is other classes of category d.

FPd: category D has been predicted; other category d classes have been the reality.

FNd: the predictive other classes of category

D; the reality category d.

All categories were taken as positive samples to find out the overall accuracy, precision, and recall values. The precision can be represented as follows using equation, (7.9).

$$\text{Accuracy} = \frac{\text{Numberofsamplescorrectlyclassified}}{\text{Numeberofsamplesforallcategories}} \text{------ (7.9)}$$

This accuracy of a certain category can be viewed as a forecast of a sample accuracy since it can be seen in Equation (7.10):

$$\text{Precision}_i = \frac{TP_d}{TP_d + FP_d} \text{-------- (7.10)}$$

The recall of a given category may be considered the quantity of that category which the sample of category d was predicted to cover, in the sample set as indicated in Equation (7.11),

$$\text{Recall}_i = \frac{TP_d}{TP_d + FN_d} \text{--------- (7.11)}$$

The Equation is used to give F measure.

$$F - \text{Measure} = 2.\frac{\text{Precison.recall}}{\text{Nprecison+recall}} \text{---------- (7.12)}$$

Table 1 presents the results of training and testing of the selected dataset through the proposed model in the current research based on the usage of Modified Bi-LSTM Deep Neural Networks.

Table 7.3 Training and Testing Values with 10 Epochs

Epoch	Training Loss	Validation Loss	Training Accuracy	Testing Accuracy
1	0.0039	0.0626	0.9987	0.9846
2	0.0035	0.0762	0.9989	0.9835
3	0.0030	0.0699	0.9991	0.9853
4	0.0039	0.0643	0.9986	0.9862
5	0.0017	0.0684	0.9995	0.9866
6	0.0030	0.0747	0.9991	0.9858
7	0.0026	0.0887	0.9991	0.9844
8	0.0023	0.0776	0.9992	0.9855
9	0.0023	0.0749	0.9992	0.9868
10	0.0013	0.0767	0.9996	0.9876

The table 7.3 that follows constitutes training and validation results of a model measured in terms of epochs. Every epoch is defined by one full cycle of the training data. Each training, validation, training accuracy and testing accuracy are calculated and documented in relation to each epoch. The following is an interpretation of the data:

Epoch 1: The training loss of the model was 0.0039 and validation loss was 0.0626. The training accuracy was 0.9987 which means that the model achieved the training data with high accuracy of 99.87. The accuracy of the testing was 0.9846 which is an indication that the model had good performance in unseen testing data with a model accuracy of 98.46%.

Epoch 2: The loss of the model training dropped to 0.0035 and the validation loss dropped marginally to 0.0762. The training accuracy was also high at 0.9989 and the testing accuracy was a bit lower and was 0.9835.

Epoch 3: The model continued to record more improvements in the training and validation losses with values of 0.0030 and 0.0699 respectively. The training accuracy was high of 0.9991 and testing accuracy rose to 0.9853.

Epoch 4: The training loss was higher to 0.0039, whereas the validation loss was lower to 0.0643. The training accuracy became a little smaller which was 0.9986, and the testing accuracy was 0.9862.

Epoch 5: The model obtained much lower training loss of 0.0017 and validation loss of 0.0684. The training accuracy was now 0.9995 and the testing accuracy was now 0.9866.

Epoch 6: There was a minor increase in the training losses and the validation losses to 0.0030 and 0.0747 respectively. The accuracy of training was also good at 0.9991 and the testing accuracy went down a bit to 0.9858.

Epoch 7: The validation loss became 0.0887, which can be seen as overfitting. Nevertheless, the accuracy of training was high (0.9991) and the testing accuracy reduced to 0.9844.

Epoch 8: The validation loss of the model was 0.0776. Training accuracy was 0.9992 and 0.9855 was the test accuracy.

Epoch 9: This model also attained a validation loss of 0.0749. The training accuracy was also high at 0.9992 and the testing accuracy was also improved to 0.9868.

Epoch 10: The model obtained much lower training loss of 0.0013 and validation loss of 0.0767. The training accuracy was 0.9996 and the testing accuracy had risen to 0.9876.

All in all, the model has shown good results in terms of reducing losses as well as the process of improving the level of accuracy during training. Nevertheless, the validation loss and the testing accuracy appear to be slightly different between different epochs. Analysis and tuning may be required further to deal with overfitting or changing performance.

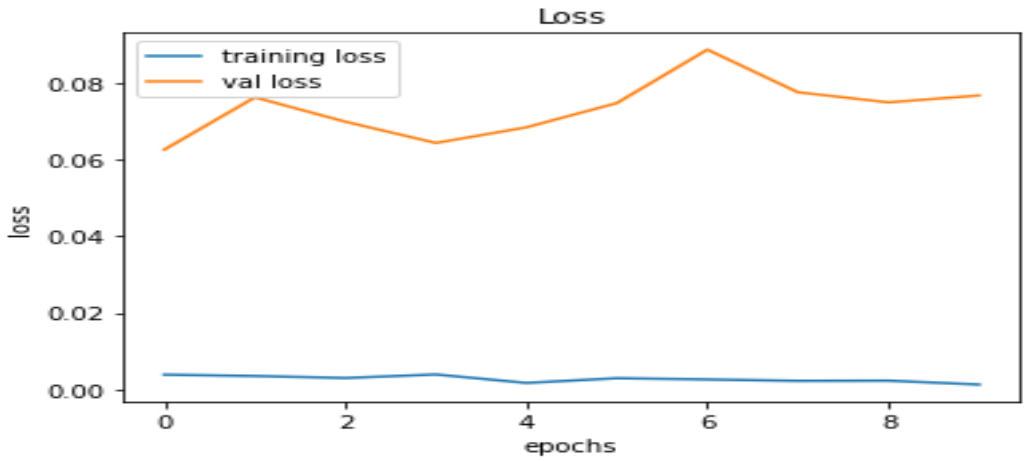

Figure 7.8 Training and Testing Loss

The loss values are used to train the proposed model as illustrated in Figure 7.8. In X-axis represents the number of Epochs and in Y-axis represents the loss.

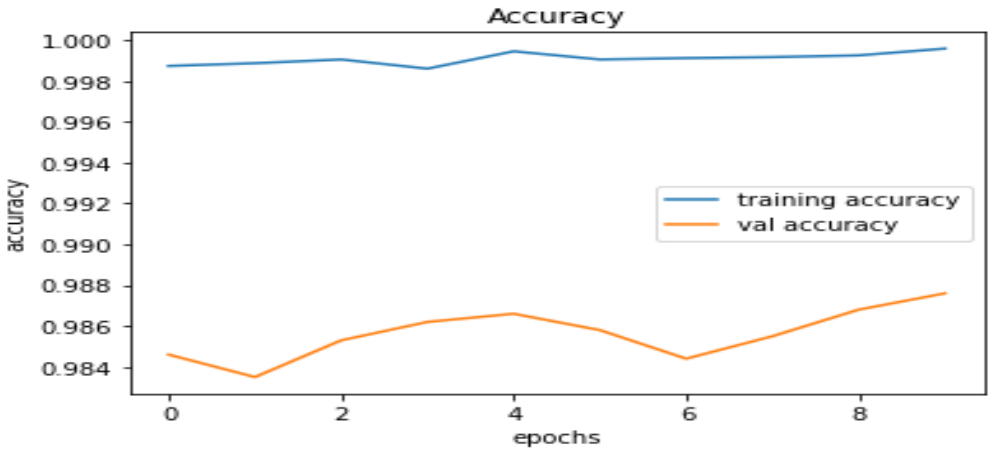

Figure 7.9 Training and Testing Accuracy.

The CNN-ResNet has been trained using 10 Epochs of Training, and the accuracy in the testing has been indicated in Fig 7.9. The X-axis will indicate the Epoch number, and Y-axis will indicate the accuracy. According to the CSV file, the accuracy and the f-measure values will be obtained with the help of the ML algorithms.

7.3.3 Analysis of Results and Discussions

Table 7.4 Comparison Table of Classifiers

Dataset	Algorithm	Accuracy (%)	Precision	Recall	F-measure
Dataset 1	DT	88	87	90	88
	SVM	83	81	88	84
	GNB	90	91	89	90
	LR	85	83	89	85
	Stacking Ensemble Classifier	95.9	89.5	88.4	89
Dataset 2	DT	91.75	91	92	92
	SVM	71	71	70	71
	GNB	83	87	78	82
	LR	71	72	70	71
	Stacking Ensemble Classifier	95.9	92.3	91.1	91.7

In Dataset 1 and Dataset 2 Stacking Ensemble Classifier performs better than all of the individual standalone baseline models (DT, SVM, GNB, LR) always. It achieves a top accuracy of 95.9 on Dataset 1, surpassing even the maximum score of GNB of 90, and F-measure of 89, and with a high balanced performance. It also performs the best in Dataset 2 with 95.9 percent accuracy and 91.7 F-measure, with SVM and LR coming way behind (around 71 percent). To be more exact, both datasets also have better precision and recall, which makes Stacking model good in recognizing and classifying complicated attack patterns and labeling them appropriately. This makes it beneficial in sophisticated cybersecurity threats detection processes such as zero-day vulnerabilities.

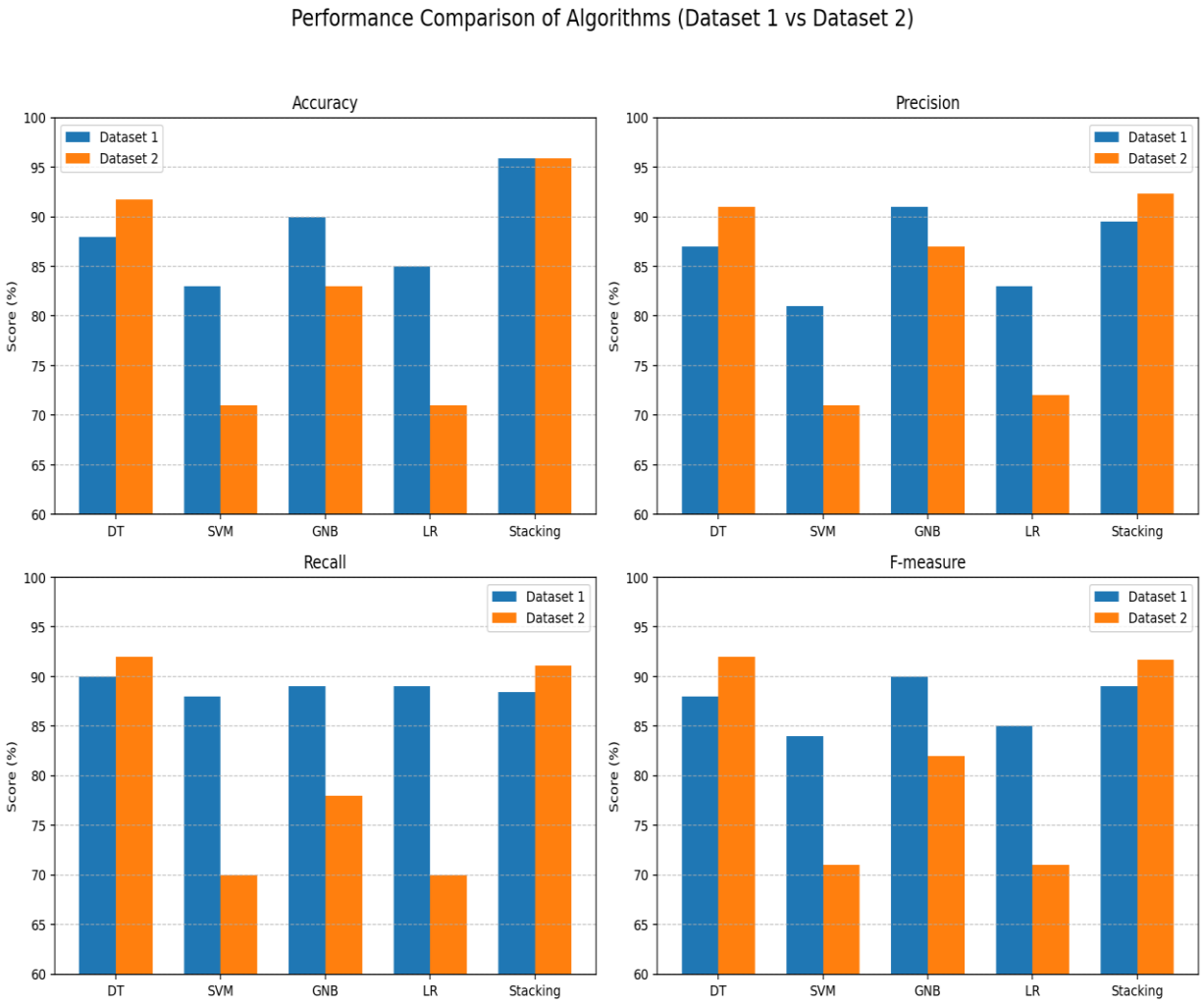

Figure 7.10 Classifiers Comparison Chart for Datasets 1 and 2

Figure 7.10provides the comparison of classification of the different algorithms. The x-axis in this chart represents the algorithms and the y-axis represents the metric score.

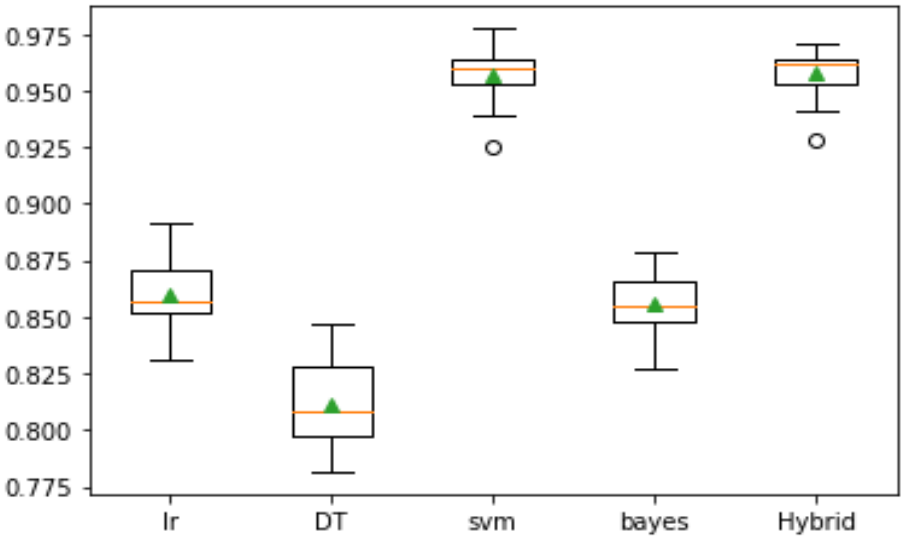

Figure 7.11 Stacking Ensemble Classification Comparison Chart

Figure 7.11 gives the comparison of performance of different classification models which include Logistic Regression, Decision Tree, SVM, Naive Bayes and Hybrid model. Among them, the Hybrid and SVM models are the most accurate and reliable as the fact of more advanced medians and fewer distributions testifies. Logistic Regression and Naive Bayes have medium-level levels of performance and the lowest accuracy and maximum variability is the Decision Tree model. No reasonably negative effect on the improved overall performance of SVM and Hybrid exists because of the existence of outliers. That means that the Hybrid model is the most successful and robust of the tried classifiers.

Table 7.5 Comparison Table for Prediction Methods

Dataset	Model	Accuracy (%)	Precision	Recall	F-Measure
Dataset 1	Hidden Markov Model	84.5	82.1	83.0	82.5
	BERT	89.2	86.8	87.5	87.1
	Modified Bi-LSTM and Hybrid GT with ANN-AE	95.4	91.3	91.5	90.4
	DC-nZDASN	96.9	92.3	92.1	91.7
Dataset 2	Hidden Markov Model	86.3	83.2	84.5	83.8
	BERT	88.5	85.9	86.4	86.1
	Modified Bi-LSTM and Hybrid GT with ANN-AE	95.0	90.3	90.8	89.1
	DC-nZDASN	96.0	91.7	91.9	90.4

Table 7.5 means that DC-nZDASN (Deep Contextual non-Zero-Day Attack Sequence Network) is more accurate in the two datasets and superior to deep learning and conventional models. The Hidden Markov Model (HMM) is the worst one as it is unable to comprehend the context in a broad sense. By Seyyar et al. (2022) BERT is also quite successful with its great language modeling, however, lower in position when compared to the Bi-LSTM hybrid. The Hybrid Game Theory and ANN-AE Modified Bi-LSTM is far better than HMM and BERT but it remains slightly below DC-nZDASN in F-Measure and the general accuracy. This confirms the suitability of integrating both the deep contextual and game-theoretic learning in predicting zero-day attacks.

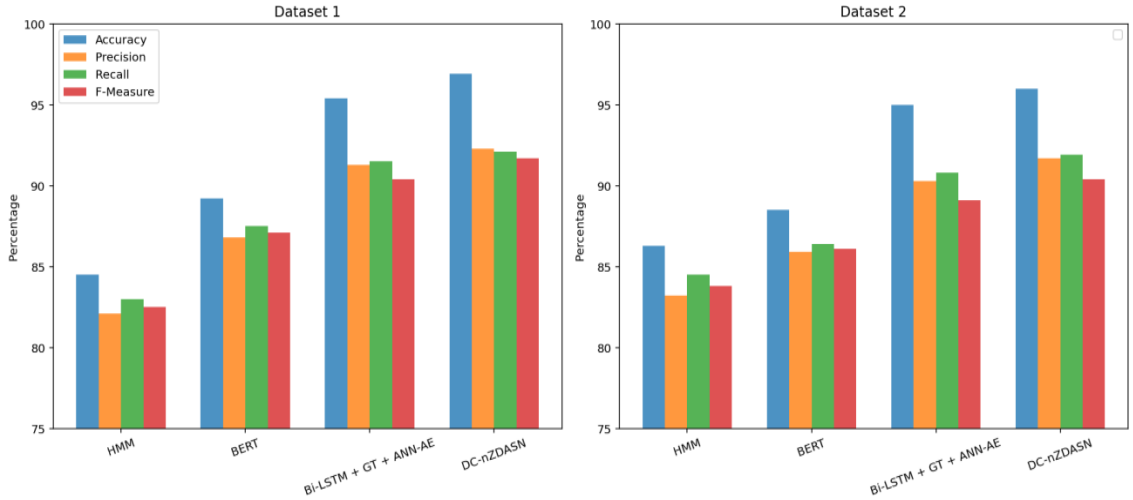

Figure 7.12 Prediction Comparison Chart

Figure 7.12 presents the comparison of prediction of the Proposed DC-nZDASN and the existing methods. The x-axis in this chart represents the algorithms and the y-axis represents the metric score.

Table 7.6 Comparison Table for Detection Methods

Dataset	Model	Accuracy (%)	Precision	Recall	F-Measure
Dataset 1	Bayesian GNN	87.2	84.5	85.3	84.9
	TransVAE Threat Detector	89.6	86.7	87.1	86.9
	HADE – One Class SVM	91.3	88.5	89.0	88.7
	DC-nZDASN	96.2	93.1	93.8	92.7
Dataset 2	Bayesian GNN	85.4	83.1	82.6	82.8
	TransVAE Threat Detector	88.8	86.2	85.9	86.0
	HADE – One Class SVM	90.1	87.8	87.4	87.6
	DC-nZDASN	96.9	93.3	93.4	92.9

Table 7.6 demonstrates better results of DC-nZDASN, compared to all the models, using both datasets in terms of accuracy, precision, recall and F-measure. Despite the fact that HADE is very good especially in identifying anomalies, it is lagging behind DC-nZDASN. TransVAE Threat Detector is a mediocre one that identifies sequence anomalies using Transformer and VAE and is best applied in cases that are not known. Bayesian GNN has the lowest scores which can probably be attributed to the fact that it is context-aware. DC-nZDASN as a whole is more flexible and detects zero-day attacks.

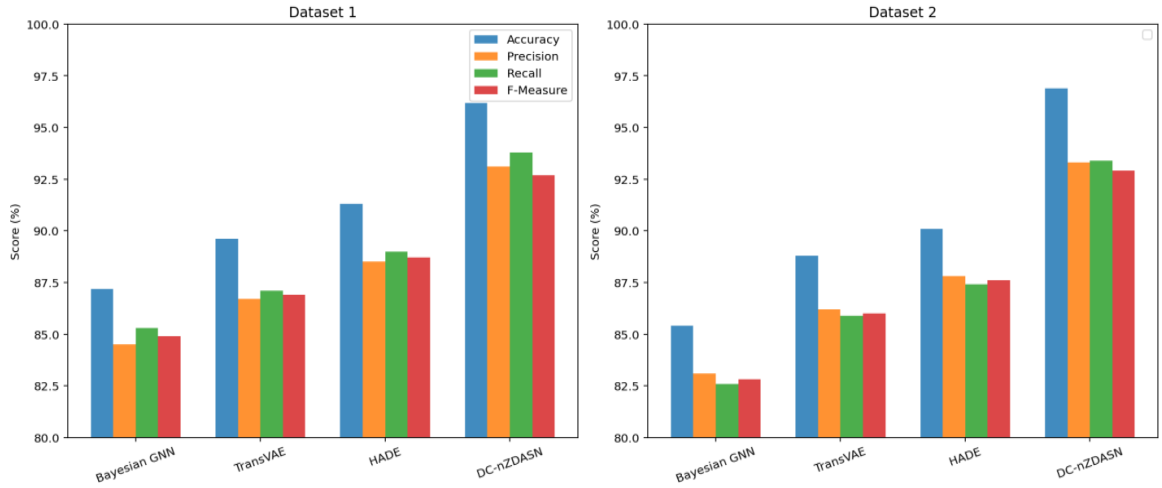

Figure 7.13 Detection Comparison Chart

Figure 7.13 presents the comparison of the Proposed DC-nZDASN detection with the current methods. The x-axis in this chart represents the algorithms and the y-axis represents the metric score.

Table 7.7 Performance Improvement Table for Prediction

	Compared Model	**Accuracy**	**Precision**	**Recall**	**F-Measure**
Dataset 1	Hidden Markov Model (HMM)	↑ 12.4%	↑ 10.2%	↑ 9.1%	↑ 9.2%
	BERT	↑ 7.7%	↑ 5.5%	↑ 4.6%	↑ 4.6%
	Modified Bi-LSTM + Hybrid GT + ANN-AE	↑ 1.5%	↑ 1.0%	↑ 0.6%	↑ 1.3%
Dataset 2	Hidden Markov Model (HMM)	↑ 9.7%	↑ 8.5%	↑ 7.4%	↑ 6.6%
	BERT	↑ 7.5%	↑ 5.8%	↑ 5.5%	↑ 4.3%
	Modified Bi-LSTM + Hybrid GT + ANN-AE	↑ 1.0%	↑ 1.4%	↑ 1.1%	↑ 1.3%

As indicated in the comparison table, the proposed DC-nZDASN is superior to the existing approaches by astronomic values of performance improvement. In Dataset 1, DC-nZDASN achieves the highest margin over HMM at +12.4% accuracy and other measuring indicators, i.e. +9%. BERT achieves small improvements, and the Modified Bi-LSTM + GT + ANN-AE achieves the same performance with insignificant improvements to the suggested solution. Similarly, DC-nZDASN in Dataset 2, again, beats with an upper limit of +9.7% better accuracy on HMM and consistent increases in the accuracy, recall, and F-measure, which validates its soundness in detection of the zero-day threats.

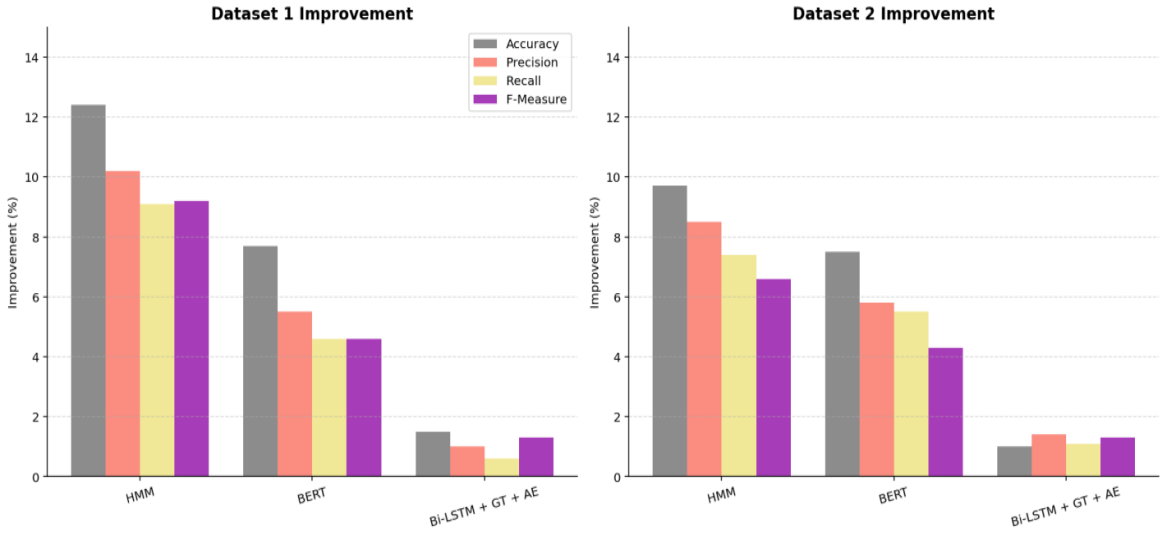

Figure 7.14 Performance Improvement Table for Prediction

Figure 7.14 shows the Performance Improvement table performance based on datasets 1 and 2 of prediction methods. The x-axis in this chart displays the algorithms and the y-axis displays the values of improvement.

Table 7.8 Performance Improvement Table for Detection

	Compared Model	Accuracy	Precision	Recall	F-Measure
Dataset 1	Bayesian GNN	↑ 9.0%	↑ 8.6%	↑ 8.5%	↑ 7.8%
	TransVAE Threat Detector	↑ 6.6%	↑ 6.4%	↑ 6.7%	↑ 5.8%
	HADE – One Class SVM	↑ 4.9%	↑ 4.6%	↑ 4.8%	↑ 4.0%
Dataset 2	Bayesian GNN	↑ 11.5%	↑ 10.2%	↑ 10.8%	↑ 10.1%
	TransVAE Threat Detector	↑ 8.1%	↑ 7.1%	↑ 7.5%	↑ 6.9%
	HADE – One Class SVM	↑ 6.8%	↑ 5.5%	↑ 6.0%	↑ 5.3%

Table 7.8 signifies that the suggested model of DC- nZDASN has enormous enhancement in each performance measure compared to Bayesian GNN, TransVAE, and HADE. Using Dataset 1, it reaches the maximum accuracy of a 9.0% improvement of regular improvisations in precision, recall, and F-measure. Using Dataset 2, the model is much better, and the accuracy is 11.5 percent higher than Bayesian GNN and the F-measure is also greater by over 10 percent. Although TransVAE and HADE are slightly improved, the patterned and high margins of DC-nZDASN indicate the effectiveness of DC-nZDASN in detecting zero-day attacks with diverse architectures.

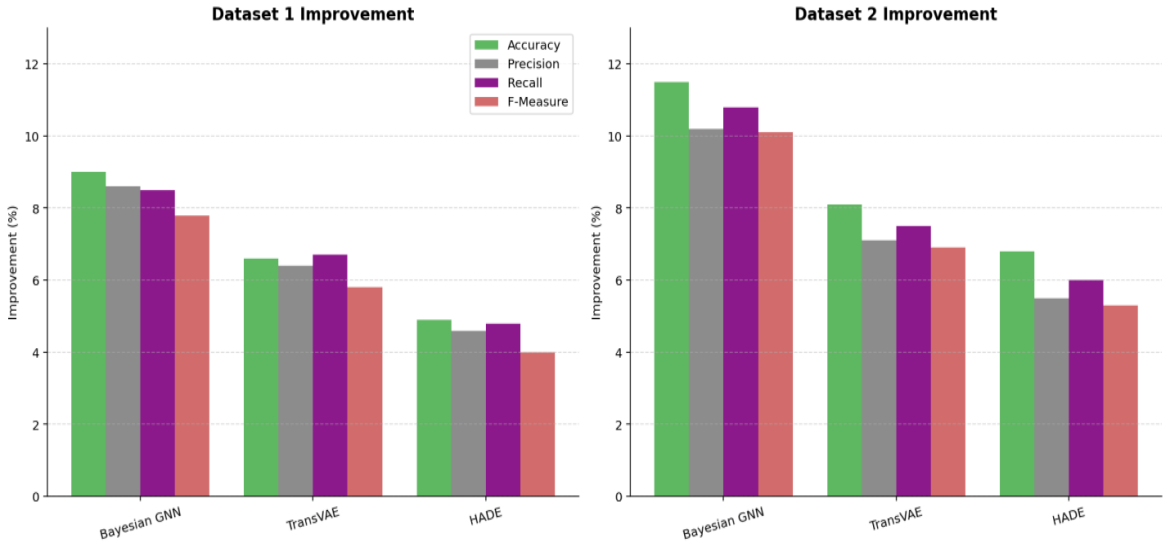

Figure 7.15 Performance Improvement Table for Detection

Figure 7.15 shows the Performance Improvement table of Detection methods based on dataset 1 and 2. The x-axis in this chart displays the algorithms and the y-axis displays the values of improvement.

7.4 Computational Expenses

Computational expense is one of the most crucial factors in evaluation of zero-day attack prediction and detection model since its complexity should be matched with corresponding improvements in its performance which can be quantified. In this case, the cost of computation is contrasted to the classification accuracy to identify whether the incremental processing cost of the state-of-the-art models can offer any significant distinction to the conventional ones.

The results of Dataset 1 and Dataset 2 show that the conventional classifiers such as Decision Tree, SVM, Gaussian Naive bayes and Logistic Regression have an accuracy ranging between 71 - 91.75 but fail to generalize the dynamism of the attacks thus detection capabilities are limited. The Stacking Ensemble Classifier as a trade-off between accuracy and cost maximizes the prediction accuracy at 95.9 to the dataset on both datasets, although it increases the cost of computation slightly.

The suggested deep learning models are associated with a moderate increase in the computational load as offered by the sequential and convolutional processing. This is however compensated by huge increases in performance. The hybrid game theory and ANN-AE based Modified Bi-LSTM ascertains the highest percent of accuracy of 95.4 and 95.0 (Dataset 1 and 2) and the DC-nZDASN detection model recognizes the highest percent of accuracy of 96.9 and 96.0 percent, respectively. Dc-nZDASN offers a 5-6 percent solution over current sophisticated detection models such as Bayesian GNN, TransVAE, and HADE-One Class SVM with less than 91.3 accuracy rates.

Overall, the results confirm that the proposed framework is trade-off followed by the consideration of the computational cost and the accuracy of detection using the method of feature reduction (15 selected features) and an effective model design and the additional processing cost can be justified by the real-life use of the zero-day attack prediction and detection.

7.5 Chapter Summary

The chapter presented the DC-nZDASN framework, which is the combination of deep convolutional learning (ResNet50) with several machine learning classifiers and adversarial generated malware samples that maximize the detection and classification of the zero-day attacks. The highest possible detection rate of the proposed model is 96.9 regardless of the fact that it achieves higher accuracy compared to more advanced baselines such as Bayesian GNN and HADE by almost 56 percent but the value of the model is that it is more robust due to adversarial generalization and less overfitting compared to optimization of the metrics alone. Since the model has a high rate of detection, it is limited to the fact that it requires quite high computation requirements and the model is not adversarial to scalable, low-latency zero-day protection in live networks.

Publications

- Akshaya S and Padmavathi G. ResNet50-based deep convolutional neural network for zero-day attack prediction and detection. International Journal of Advanced Technology and Engineering Exploration. 2025; 12(124):507-527. (Scopus)