Cybersecurity in Deep Learning Techniques: Detecting Network Attacks

Shatha Fawaz Ghazal[1]
Department of Administrative Sciences, Finance and Computer, Al- Balqa Applied University (BAU)
Zarqa, Jordan

Salameh A. Mjlae[2]
Prince Abdullah bin Ghazi Faculty of Information Technology, Al- Balqa Applied University (BAU)
AL- Salt, Jordan

***Abstract*—Deep learning techniques have been found to be useful in a variety of fields. Cybersecurity is one such area. In cybersecurity, both Machine Learning and Deep Learning classification algorithms can be used to monitor and prevent network attacks. Additionally, it can be utilized to identify system irregularities that may signal an ongoing attack. Cybersecurity experts can utilize machine learning and deep learning to help make systems safer. Eleven classification techniques, including eight machine learning algorithms (Decision Tree, Random Forest, and Gradient Boosting) and one statistical technique, were employed to examine the popular HTTP DATASET CSIC 2010. (K-Means). Along with XGBoost, AdaBoost, Multilayer Perceptrons, and Voting, three deep learning algorithms are Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM), and LSTM plus CNN. To evaluate the performance of such models, precision, accuracy, f1-score, and recall are often used metrics. The results showed that when comparing the three deep learning algorithms by the aforementioned metrics, the LSTM with CNN produced the best performance outcomes in this paper. These findings will show that our use of this algorithm allows us to detect multiple attacks and defend against any external or internal threat to the network.**

Keywords*—*HTTP DATASET CSIC 2010; deep learning; cybersecurity attacks; detection attacks; network attacks

I. Introduction

Cybersecurity (Cyber_Security) [1] protects computers, servers, and networks from intrusion, theft, and other forms of harm to their data, hardware, and software, as well as from interruptions in service or misuse of the resources they provide. Our reliance on computers, the Internet, and wireless network standards like Bluetooth and Wi-Fi, as well as the proliferation of "smart" devices like smartphones, televisions, and the myriad other devices that make up the Internet of things, have all contributed to the Internet of Things growing significance (IOT) [1]. Due to the complication of its political application and technological implementation, cybersecurity is one of the greatest issues of the twenty-first century. The system's key objectives are dependable operation, integrity of data, and safeguarding of sensitive information. Due to the complication of its political application and technological implementation, cybersecurity is one of the greatest issues of the twenty-first century. The system's key objectives are dependable operation, intact data, and the security of sensitive information [2].

A cyber-attack is any hostile attempt to hack an IT system or its users in order to gain unauthorized access to the system's data or information [3]. Cyber attackers are often crooks looking to profit financially from the attack in the great majority of instances. In other cases, the intention is to stop operations by preventing users from accessing IT systems or destroying physical equipment [3]. State agents or cybercriminals working for them are frequently involved in state-sponsored cybercrime. Cyberattacks can be wide in scope, impacting a number of businesses across several regions and nations, or they can be directed at specific companies or persons [4]. Targeted attacks frequently expand beyond their intended targets, posing a threat to all firms. A state-sponsored strike against Ukrainian banks and utilities most likely caused the global NotPetya outbreak in June 2017. The clean-up achieved the desired effect on Ukraine, but it also had a worldwide impact, costing almost $10 billion in IT system recovery and lost productivity, according to publications documenting the cleanup [2, 3, and 4].

Deep learning can be classified as a sort of machine learning. Research into algorithms is crucial to the growth and development of this discipline. Deep learning employs artificial neural networks created to simulate human cognitive processes, whereas machine learning focuses on generalization and empirical data. Until recently, the available computational power limited the complexity of neural networks [5]. Systems in the cybersecurity sector may use deep learning to recognize and learn from patterns in order to foil future attacks and adapt to evolving adversarial tactics. It can improve the speed with which cybersecurity teams respond to actual attacks and mitigate risks [5]. Because it reduces the amount of time spent on routine tasks, it allows businesses to better allocate their resources. In conclusion, deep learning can make cybersecurity easier, more preventative, cheaper, and more effective [4, 5]. However, this will only be possible if the data that drives machine learning provides an accurate depiction of the setting. As the adage goes, "if you put garbage in, you'll get garbage out." [5].

The following are some key takeaways from this paper that outline its contribution:

1) This thesis makes use of a publicly available dataset pertinent to the topic at hand, allowing a wide range of classification algorithms to be put to use.

2) This category includes deep learning and machine learning. Building a robust model to recognize different network assaults utilizing a variety of machine learning and deep learning methods.

Examining the ability of each method to distinguish between the numerous network assaults that have been documented in the dataset. Several classification techniques, including machine learning and deep learning algorithms, are applied to a well-known network intrusion dataset in this study in order to determine whether or not each sample is typical and to identify attacks that are not typical. The remainder of this study is organized as follows: The second component of this paper provides a bibliography of relevant archival materials for this investigation. The methods employed are thoroughly detailed in Section III. Some examples are provided to illustrate the results in Section IV. Section V presents our conclusion and suggestions for future research.

II. Related Work

Several authors applied machine learning algorithms in the cybersecurity field to detect different types of attacks based on real-time datasets or existing datasets from several resources. Kim et al. [6] using a Convolutional Neural Network (CNN) with long short-term memory (LSTM) and a Deep Neural Network (DNN), the sample was identified as normal or abnormal (DNN). They used the 61,065 instances and 16 features from the HTTP DATASET CSIC 2010 that were classified as either normal (36,000 samples) or anomalous (25,065 samples). The results demonstrated that the CNN equipped with LSTM was 91.54 percent accurate. Vartouni et al. [7] used a novel algorithm known as a Stacked Auto-Encoder to determine whether the sample was typical or out of the ordinary. They used the 61,065 instances and 16 features from the HTTP DATASET CSIC 2010 that were classified as either normal (36,000 samples) or anomalous (25,065 samples). According to the findings, the Stacked Auto-Encoder was able to achieve an accuracy of 88.32 percent.

Betarte et al. [8] The sample was classified as normal or abnormal using three distinct machine learning techniques: Random Forest, K-Nearest Neighbors (K-NN) with a k value of three, and Support Vector Machine (SVM). They utilized the HTTP DATASET CSIC 2010 dataset for their analysis, which covers normal (36,000 samples) and abnormal (61,065 cases) categories (25,065 samples). Compared to the other methods, Random Forest achieved 91.54 percent more accuracy. Tuan et al. [9] examined their five machine learning techniques using the common dataset UNSW-NB15, which includes numerous types of network assaults.

This dataset consists of nine separate attack types (DoS, Reconnaissance, Backdoor, Fuzzers, Analysis, Exploits, Worms, Shellcode, and Generic) and common attacks (44 characteristics). The machine learning algorithms are SVM, ANN, Naive Bayes (NB), and Unsupervised Learning (USML). Using this dataset, they demonstrated the USML's high degree of precision (94.78 percent). Anwer et al. [10] used four machine learning techniques to detect malicious network traffic based on a well-known dataset. They used a popular cybersecurity dataset named NSL-KDD that has 148,517 samples divided into training (125,973) and testing (22,544) datasets. There are a total of five classes in this dataset, split evenly between normal attacks and non-normal attacks, with subclasses within each subclass. For example, DoS attacks (Smurf, Back, Land, Processtable, Neptune, Pod, Apache2, Udpstorm, Worm, Teardrop) and R2L attacks (Xsnoop, Guess Password, Named, Ftp write, Phf, Multihop, Imap, Warezmaster, Warezclient, Snmpgetattack, Spy, Xlock, Snmpguess, Httptunnel, Sendmail) are all in existence. Machine learning algorithms include things like the Support Vector Machine (SVM), Gradient Boosted Decision Trees (GBDT), and Random Forest (RF). These algorithms were evaluated on four fronts: accuracy, specificity, training time, and prediction time. The 85.34 percent accuracy achieved by RF was the highest of any of the tested algorithms.

Su et al. [11] presented the BAT model, a deep learning technique for spotting hostile network infiltration. Utilization of the NSL-KDD dataset, which is commonly utilized in the investigation of network intrusion. The same number of data points were used for training and testing, totaling 125,973 points (22,544). Each of the five classes in this dataset is further subdivided into two additional groups: normal attacks and non-normal attacks. To name just a few examples, there are denial-of-service (DoS) attacks (Smurf, Back, Land, Processtable, Neptune, Pod, Apache2, Udpstorm, Worm, Teardrop), reconnaissance-to-leak (R2L) attacks (Xsnoop, Guess Password, Named, Ftp write, Phf, Multihop, Imap, Warezmaster, Warezclient, Snmpgetattack, Spy, Xlock The intrusion detection accuracy of this model was 84.25%.

Xu et al. [12] proposed a new model based on a five-layer autoencoder (AE) that is more adept at detecting network anomalies. Because of its prominence in the realm of network assault, the NSL KDD dataset was deployed. The same number of data points were used for training and testing, totaling 125,973 points (22,544). Each of this dataset's five classifications is then separated into two additional groups: normal attacks and non-normal attacks. There are denial-of-service (DoS) assaults (Smurf, Back, Land, Processtable, Neptune, Worm, Pod, Apache2, UDPstorm, and others). Teardrop), reconnaissance-to-leak (R2L) attacks (Xsnoop, Guess Password, Named, Ftp write, Phf, Multihop, Imap, Warezmaster, Warezclient, Snmpgetattack, Spy, Xlock), and reconnaissance-to-attack (R2A) attacks (Xsnoop, Guess Password, Named, Ftp write, Phf, Multihop, I. The study demonstrated that the model's accuracy in identifying intrusions is 90.61 percent.

Kavitha et al. [13] presented a new technique for network intrusion detection based on One Class Support Vector Machine (OCSVM). They utilized the popular NSL-KDD dataset, which consists of a total of 148,517 samples split evenly between a training set of 125,973 samples and a testing set of 22,544 samples. This dataset has a total of five classes, evenly divided between normal and non-normal attacks, with subclasses inside each subclass. New technique for network intrusion detection based on One Class Support Vector Machine (OCSVM). They utilized the popular NSL-KDD dataset, which consists of a total of 148,517 samples split evenly between a training set of 125,973 samples and a testing set of 22,544 samples. This dataset has a total of five classes, evenly divided between normal and non-normal attacks, with subclasses inside each subclass. To name just a few examples, there are denial-of-service (DoS) attacks (Smurf, Back, Land, Processtable, Neptune, Pod, Apache2, Udpstorm, Worm, Teardrop), reconnaissance-to-leak (R2L) attacks (Xsnoop, Guess Password, Named, Ftp write, Phf, Multihop, Imap, Warezmaster, Warezclient, Snmpgetattack, Spy, Xlock. The model was found to be 81.29 percent accurate when used for intrusion detection.

Ferriyan et al. [14] developed several machine learning models for detecting cyberattacks, and used a new dataset called ALLFLOWMETER HIKARI2021. Background, Benign, Bruteforce, Bruteforce-XML, Probing, and XMRIGCC CryptoMiner are the six types of attacks represented among the 555,278 instances and 86 features extracted by Zeek [https://zeek.org/] in this dataset. The normal attacks include Bruteforce, Bruteforce-XML, Probing, and XMRIGCC CryptoMiner, while the malicious attacks include Background and Benign. KNN, SVM, RF, and MultiLayer Perceptron are the ML models to use (MLP). They demonstrated that these models can achieve detection accuracy of roughly 0.99 percent as shown in Table I.

TABLE I. PREVIOUS RELATED WORK FOR CYBERSECURITY ATTACKS DETECTION

Ref	Year	Attack	Cybersecurity Dataset	Algorithm	Result
[6]	2020	Normal Anomalous	CSIC 2010 dataset	CNN and LSTM DNN	CNN and LSTM accuracy = 91.54%
[7]	2018	Normal Anomalous	CSIC 2010 dataset	Stacked Auto-Encoder	Accuracy = 88.32
[8]	2018	Normal Anomalous	CSIC 2010 dataset	Random Forest KNN-3 SVM	Random Forest accuracy = 72%
[9]	2019	• Analysis • Reconnaissance • DoS • Exploits • Fuzzers • Generic • Normal • Worms • Backdoor • Shellcode	UNSW-NB15	SVM, ANN, NB, DT, and USML	USML accuracy = 94.78%.
[10]	2021	• R2L • DoS • U2R • Probe • Normal	NSL-KDD	SVM, GBDT, and RF	RF Accuracy = 85.34%
[11]	2020	• R2L • DoS • U2R • Probe • Normal	NSL-KDD	BAT model	Accuracy = 84.25%
[12]	2021	• R2L • DoS • U2R • Probe • Normal	NSL-KDD	5-layer autoencoder (AE)-based model	Accuracy = 90.61%
[13]	2021	• R2L • DoS • U2R • Probe • Normal	NSL-KDD	One Class SVM	Accuracy = 81.29%
[14]	2021	• Background • Benign • Bruteforce • Bruteforce-XML • Probing • XMRIGCC CryptoMiner	ALLFLOWMETER HIKARI2021	KNN, SVM, RF, and MLP	Accuracy = 0.99

III. Proposed Methodology

Fig. 1 depicts the proposed strategy utilized in this study to identify threats in a wide variety of cybersecurity datasets. The subsequent sections demonstrate how the proposed method operates in practice.

A. Dataset Description

In this study, we utilized the widely-used and exhaustive HTTP DATASET CSIC 2010 dataset, which comprises a variety of cyberattack kinds. The types of cyberattacks, the total number of Instances in the respective datasets, and the total number of characteristics are detailed in Table II.

Our department's online store's traffic is logged in the HTTP dataset CSIC 2010 for analysis. Registering with this web app requires the input of personal data, and once registered, users can make purchases from a cart interface. Due to the fact that it is a Spanish-language website, the collected information may contain some Latin characters [15]. In the dataset, 36,000 requests are legitimate, while another 25,000 were created fraudulently by bots. Threats like cross-site scripting, CRLF injection, server-side inclusion, parameter manipulation, and SQL injection are all part of the package. There have been successful web detection experiments using this dataset [15]. Fig. 2 depicts the frequency of attacks on the HTTP dataset CSIC 2010, while Table III details the characteristics of this dataset.

B. Preparing Dataset

In order to apply different deep learning algorithms to each dataset, we encode their non-numerical properties into numerical features using a standard method [16]. Label Encoder is a technique for converting data that isn't numerical into a form that computers can understand by assigning each value a number starting at zero [16]. All the features in this dataset are categories, so we need to transform them into numbers. We used the Hold out method to divide the data set. As a result, in our experiments, we use only 0.20 percent of the total dataset for evaluation purposes, while the remaining 0.80 percent serves as training data.

C. Classification Algorithms

In this paper, we detail the detection procedure and the classification algorithms that enabled it. Some machine learning and deep learning formula examples are provided.

TABLE II. Information of Cybersecurity Datasets

Dataset Name	No. of Instances	No. of Features	Cybersecurity Attacks	
HTTP DATASET CSIC 2010	61,065	16	Normal	36,000
			Malicious	25,065

TABLE III. Features of the HTTP Dataset CSIC 2010.

No.	Feature
1	Method
2	User-Agent
3	Pragma
4	Cache-Control
5	Accept
6	Accept-encoding
7	Accept-charset
8	language
9	host
10	cookie
11	content-type
12	connection
13	length
14	content
15	URL
16	label

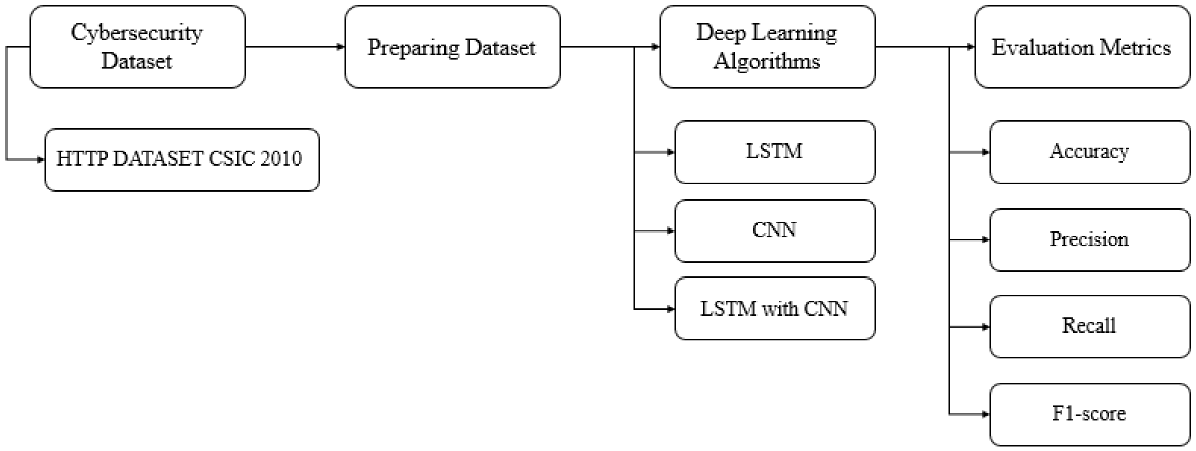


Fig. 1. Illustrate Flow-chart of the Proposed Methodology.

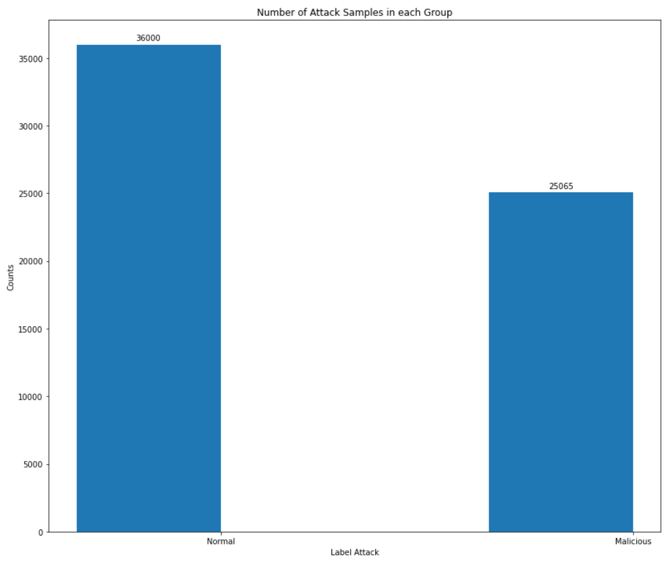


Fig. 2. Attacks Frequency in HTTP Dataset CSIC 2010.

1) Machine learning algorithms: After preparing the dataset, it is fitted to seven machine learning algorithms, each with a test size of 0.1, in order to detect the aforementioned cybersecurity attacks. This was accomplished by utilizing a hold-out approach to divide the dataset into training and testing datasets, with 0.9% of the total datasets acting as training datasets and 0.1% serving as testing datasets. Multiple machine learning models are constructed utilizing the training dataset, and their efficacy is assessed utilizing the testing dataset.

- Random Forest Algorithm (RF)

Random forests, an ensemble supervised learning method, trains many decision trees to perform regression and classification. The most commonly selected class by trees is the one that is expected to be the solution to classification problems. When it comes to regression jobs, we give you the typical prediction from all of our trees [17]. Overfitting the training set is a common problem for decision trees; random decision forests are capable of mitigating this effect. Therefore, we utilize the Random Forest classifier to categorize each cyber-attack dataset [18, 19]. Due to the discontinuous nature of our experiment's label, we employed a random forest of the classification variety with the following parameters: n_estimators (number of decision trees) = 100, max_features = sqrt, max_depth = None, random_state = 42.

- Decision Tree Algorithm (DT)

Decision Tree is a supervised machine learning system that, similar to people, makes decisions based on predefined criteria. [20]. The fields of data mining, statistics, and machine learning all use decision tree learning (or induction of decision trees) as a method of predictive modeling [21]. The use of a decision tree, it goes from observing a sample (representing the trunk) to drawing conclusions about the sample's target value (representing the leaves, which are attack types) [22].

Classification trees are a type of tree model where the target variable is discrete, and where the "leaves" represent the different types of attacks and the "branches" represent characteristics of the dataset that aid in predicting class labels [23, 24]. One type of decision tree, called a regression tree, has a continuous objective variable (typically real numbers) [23]. One of the most well-known machine learning algorithms, decision trees are prized for their clarity and ease of use. Using a Decision Tree with a classification type was necessary due to the discrete nature of the label in our experiment. These are the parameters we used for DT: criterion = gini and random_state = 42.

- Multilayer Perceptron Algorithm (MLP)

The feedforward artificial neural network is based on the multilayer perceptron (MLP). Many-layer Perceptron Networks (MLPs) are a type of feedforward artificial neural network (ANN) that can also be thought of as "deep neural networks" (with threshold activation). Decision Tree is a supervised machine learning system that, similar to people, makes decisions based on predefined criteria [20].

Data mining, statistics, and machine learning all use decision tree learning as a technique for predictive modeling (or induction of decision trees) [21]. The use of a decision tree Input, hidden, and output layers are the three types of node layers an MLP needs to operate. Except for the input nodes, all other nodes are neurons with nonlinear activation functions. [25, 26]. As part of its training process, MLP uses backpropagation, a supervised learning method. An MLP differs from a linear perceptron because to its several layers and non-linear activation. It can distinguish between data that can be cleanly categorised and data that cannot [27]. Since our experiment's label is discrete, we used an MLP of the classification variety; the corresponding MLP parameters were as follows: activation = relu and random_state = 42.

- eXtreme Gradient Boosting Algorithm (XGBoost)

Shortened to "XGBoost," Extreme Gradient Boosting is a popular method for both classification and regression. Gradient boosting has been parallelized and optimized [28]. Through the use of parallelization, the training time for the boosting process can be cut in half. As an alternative to training a single optimal model on the entire dataset, we train hundreds of models on different subsets of the dataset [28]. Afterwards, have a vote to determine which model did the best (as in traditional approaches). As opposed to more traditional gradient boosting methods, XGBoost often yields better results [29]. The Python implementation provides access to numerous hidden characteristics that can be changed to increase accuracy and precision [30].

This algorithm's overarching goal is to strengthen weak learners (decision trees) so that they can generate the final prediction label (the weighted average of each weak classifier's predictions) [30]. Among the many notable features of the XGBoost [28, 29, 30] are: The model is 1) parallelized, meaning it can run in parallel on multiple CPU cores. 2) Regularization, XGBoost offers a variety of regularisation penalties to prevent overfitting. A properly trained model can successfully generalize thanks to regularizations with penalties that improve training. XGBoost can recognize and learn from non-linear patterns in data. 3) it's possible to use cross-validation right now because it's built in. 4) XGBoost's ability to run on distributed servers and clusters like Hadoop and Spark makes it possible to manage massive amounts of data. C++, Java, Python, and Julia are just some of the supported programming languages. Because the label in our experiment is discrete, we utilised this approach with classification type. The XGBoost parameters we used are as follows:

colsample_bylevel = 1, learning_rate = 0.1, gamma = 0, n_estimators = 100, and random_state = 42.

- AdaBoost Algorithm

Adaptive boosting, or the statistical classification meta-algorithm AdaBoost, is an example of such a program. It complements a variety of learning strategies to boost efficiency and effectiveness. In a boosted classifier, the final output is a weighted sum of the outputs of other learning algorithms, or "weak learners" [31]. AdaBoost is adaptive because it adjusts subsequent weak learners to prioritize instances that earlier classifiers incorrectly labeled. It has the potential to be more resistant to the overfitting problem than other learning algorithms. Even if individual learners perform badly, the model as a whole will eventually converge on a highly effective learner if and only if it outperforms random guessing [32]. Combining powerful base learners, such as deep decision trees, with AdaBoost has been shown to work well to create a more accurate model [33]. It is common practice to use AdaBoost to combine weak base learners (such as decision stumps). While some learners may perform poorly, the model as a whole will eventually converge to a highly effective learner if and only if it can outperform random guessing [32]. Combining powerful base learners, such as deep decision trees, with AdaBoost has been shown to work well to create a more accurate model [33]. It is common practice to use AdaBoost to combine weak base learners (such as decision stumps).

Most learning algorithms are better suited to certain problem classes than others, and the optimal performance of an algorithm on a dataset depends on a wide range of tuning parameters and settings. Since it employs decision trees as weak learners, AdaBoost has been deemed by many to be the best out-of-the-box classifier [31, 32, 33]. AdaBoost, when combined with decision tree learning, employs information obtained at each stage about the relative "hardness" of each training sample to instruct subsequent trees to prioritize the most difficult-to-classify samples [33]. Because the label was discrete, we employed AdaBoost with classification-type training data in our experiment. Our final GB settings are as follows: algorithm = SAMME.R, learning_rate = 1.0, n_estimators = 50 and random_state = 42.

- Gradient Boosting Algorithm (GB)

Classification and regression are just two examples of the many applications of the machine learning technique known as "gradient boosting." It provides a predictive model in the form of a series of decision trees, which are in general unreliable [34]. Using a technique called gradient boosting, we can combine the strengths of multiple less-effective learners (decision trees) into a single, more robust one. In this setting, individual decision trees are inefficient learners [35]. Each successive tree in the line is linked to the one before it and functions to correct the defect of the one before it in the chain. Boosting algorithms require extensive training time but provide high accuracy thanks to this causal connection. Statistic learning favors models with a slower learning rate [34, 35].

The following are our final GB settings: As the model improves, each new member of the group of slow learners can be accommodated within the residuals of the previous stage. The final model synthesizes insights from all three to create a robust learner. The residuals are calculated by using a loss function. For instance, in classification, mean square error (MSE) can be used, and in regression, logarithmic loss (log loss) is employed. No alterations occur when a new tree is added to the model.

The residuals of the existing model fit well with the additional decision tree [34, 35, and 36]. We used a GB with categorization type because our experiment's label is discrete. Here are our preferences for configuring GB: subsample= 1.0, learning_rate = 0.1, criterion= friedman_mse, n_estimators = 100, and random_state = 42.

- Voting Algorithm

As stated by [37], an ensemble machine learning model known as a "voting classifier" forecasts a class based on the likelihood that the output (class) corresponds to the target class. The only thing it does is add the results of each classifier that was fed into the voting classifier to predict which output class will receive the greatest number of votes [38]. Instead of building and analysing numerous specialised models, we propose a single model that trains on many models and predicts output based on the total number of votes for each output class. There are two voting processes. [39, 40] that are compatible with Voting Classifier. The projected output class with the most votes, and thus the highest likelihood of being predicted by each classifier, is the one chosen by "hard voting," as described in (1). The second method is known as "soft voting," and it involves basing the forecast on the output class rather than a majority vote.

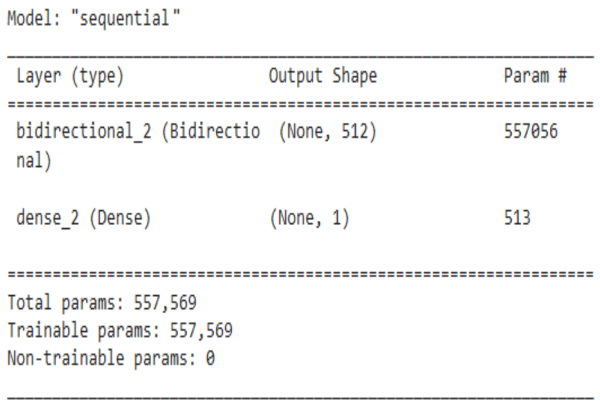
Given the dichotomous nature of the labels in our experiment, we opted for voting with classification, setting the parameters as follows: estimators = DT, RF, and XGB, and voting type = hard.

2) Deep learning algorithms: Once the dataset is ready, it is possible to use three deep learning algorithms to identify cybersecurity attacks using the dataset. There are three methods: LSTM, CNN, and LSTM combined with CNN.

- Long Short-Term Memory (LSTM) Algorithm

Long Short-Term Memory (LSTM) neural networks are used in the fields of deep learning and artificial intelligence. Because the LSTM can develop itself through feedback connections, it has a distinct advantage over traditional feedforward neural networks. Long Short-Term Memory (LSTM) neural networks are used in the fields of deep learning and artificial intelligence. Because the LSTM can develop itself through feedback connections, it has a distinct advantage over traditional feedforward neural networks. This recurrent neural network can analyse complete data sequences in addition to individual data points (such as photos, speech or video). LSTM has demonstrated success in the fields of healthcare, video game creation, healthcare analytics, and networked, unsegmented handwriting recognition. The longest short-term memory (LSTM) model of neural networks has been the subject of the most research over the past 100 years [41]. The remaining components of a typical LSTM unit are made up of cells, input gates, output gates, and forget gates. The values stored inside the cell can be retained there indefinitely, and the flow of data entering and leaving the cell is controlled by the cell's three gates. [41, 42]. LSTM networks are excellent for jobs involving the categorization, processing, and prediction of time series data because delays of variable lengths may exist between significant occurrences in a time series. Since the training of standard RNNs can result in the vanishing gradient problem, long short-term memory (LSTMs) was created as a solution. Because it is less sensitive to gap length, LSTM outperforms RNNs, hidden Markov models, and other sequence learning strategies. [42].

Fig. 3 depicts the two layers of the LSTM architecture that we employed: 1) a Bi-LSTM layer made up of an LSTM layer, 64 units, and relu as the activation function; and 2) a dense layer made up of a single unit and sigmoid as the activation function.



```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 bidirectional_2 (Bidirectio  (None, 512)              557056
 nal)

 dense_2 (Dense)             (None, 1)                 513

=================================================================
Total params: 557,569
Trainable params: 557,569
Non-trainable params: 0
_________________________________________________________________
```

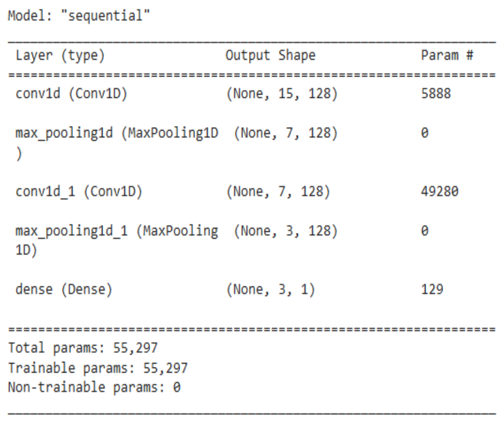
Fig. 3. LSTM Architecture.

- Convolutional Neural Network (CNN) Algorithm

Visual imaging evaluation typically makes use of an ANN class called a convolutional neural network (CNN, or ConvNet). Feature maps, which are translation-equivariant responses generated by convolution kernels or filters based on their shared-weight design and slide along input features, CNNs are also known as artificial neural networks that are shift- or space-invariant (SIANN). Contrary to common opinion, the majority of convolutional neural networks down sample the input, they do not translate invariantly. They are used in natural language processing, picture and video recognition, brain-computer interfaces, classification, segmentation, recommender systems, medical image analysis, and financial time series [43].

The structure of the visual cortex of animals served as inspiration for the development of convolutional networks. In particular, a cortical cell will only respond to stimuli that fall within the cell's receptive field [44]. All of the visible world is covered by the partially overlapping receptive fields of many neurons. Comparatively, CNNs require much less pre-processing than other image classification methods. The implication is that, unlike conventional methods, the network figures out how to optimize the filters (or kernels) on its own. First and foremost, feature extraction doesn't need any context or human input [43, 44].

The five-layered LSTM architecture utilized in our experiment is depicted in Fig. 4. Two Conv1Ds with the following parameters are provided: 128-element filters, three-element kernels, the same amount of padding, and the relu activation function. There are a total of three layers, including two max pooling layers and one dense layer containing one unit and a sigmoid activation function.

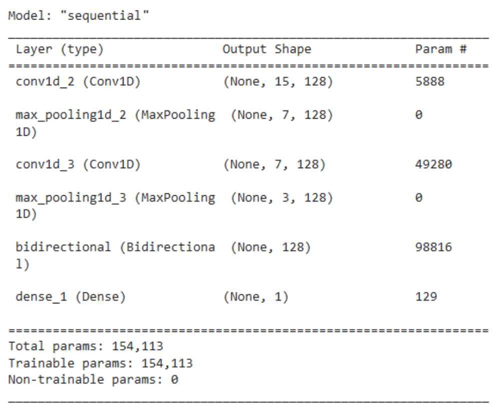


```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv1d (Conv1D)             (None, 15, 128)           5888

 max_pooling1d (MaxPooling1D  (None, 7, 128)           0
 )

 conv1d_1 (Conv1D)           (None, 7, 128)            49280

 max_pooling1d_1 (MaxPooling  (None, 3, 128)           0
 1D)

 dense (Dense)               (None, 3, 1)              129

=================================================================
Total params: 55,297
Trainable params: 55,297
Non-trainable params: 0
_________________________________________________________________
```

Fig. 4. CNN Architecture.

- LSTM with CNN Algorithm

We integrated the layers of LSTM and CNN to improve each algorithm's performance results. Fig. 5 depicts the LSTM with CNN architecture and includes: Two Conv1D are included: filters are 128, kernel-size = 3, padding = same, and relu as an activation function. One Bi-LSTM layer with an LSTM layer, 64 units, and relu as an activation function, as well as two max pooling layers and one dense layer with one unit and sigmoid as an activation function.



```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv1d_2 (Conv1D)           (None, 15, 128)           5888

 max_pooling1d_2 (MaxPooling  (None, 7, 128)           0
 1D)

 conv1d_3 (Conv1D)           (None, 7, 128)            49280

 max_pooling1d_3 (MaxPooling  (None, 3, 128)           0
 1D)

 bidirectional (Bidirectiona  (None, 128)              98816
 l)

 dense_1 (Dense)             (None, 1)                 129

=================================================================
Total params: 154,113
Trainable params: 154,113
Non-trainable params: 0
_________________________________________________________________
```

Fig. 5. LSTM with CNN Architecture.

IV. Results and Discussion

Based on four evaluation metrics, this section summarizes the experimental findings for each cybersecurity dataset for each deep learning method.

A. *Evaluation Metrics*

There are several assessment measures to examine the machine learning algorithms that were utilized, includes f1-score, recall, accuracy, and precision [45]. These metrics' formulas are as follows where:

- TP: The outcome is a true positive when the algorithm accurately predicts the positive class.
- TN: When the algorithm correctly predicts the negative class, the result is said to be "True Negative."
- FP: False positive results occur when the algorithm forecasts the positive class incorrectly.
- FN: A False Negative result occurs when the algorithm guesses the negative class incorrectly.

1) Accuracy: The ratio of correctly predicted samples to total samples, which is just a ratio of correctly predicted samples to total samples, is the most evident performance metric.

$$Accuracy = \frac{\text{TP+TN}}{\text{TP + TN + FP + FN}} \quad (1)$$

2) Precision: is the proportion of positively anticipated tweets that actually occurred to all positively predicted samples.

$$Precision = \frac{TP}{TP+FP} \quad (2)$$

3) Recall: is the ratio of positive samples that were accurately predicted to those that were generally forecast to be positive samples.

$$\text{Recall} = \frac{TP}{TP+FN} \quad (3)$$

4) F1-score: consider the weighted average of Precision and Recall.

$$\text{F1-score} = 2 * \frac{Precision * Recall}{Precision+ Recall} \quad (4)$$

B. *Machine Learning Results*

Table IV and Fig. 6 provide effectiveness measurements for the used machine learning techniques (precision, accuracy, f1-score, and recall). Voting and DT classifiers perform superiorly when it comes to the detection attack technique.

C. *Deep Learning Results*

The aforementioned three deep learning approaches were implemented with the following parameters: accuracy as the evaluation metric, binary cross entropy as the loss function, Adam as the optimizer function, 15 epochs as the number of training sessions, and 128 as the batch size. Table V and Fig. 7 exhibit the experimental outcomes for various methods. Each algorithm's performance is measured by four different criteria: accuracy, f1-score, recall, and precision:

LSTM: 0.93, 0.96, 0.87, and 0.91.

CNN: 0.92, 0.93, 0.88, and 0.91.

LSTM with CNN: 0.995, 1.0, 0.99, and 0.995.

The LSTM with CNN gave the best performance results, which means that this algorithm can detect these attacks efficiently and effectively compared with LSTM and CNN.

TABLE IV. Binary Classification Performance Results

Models	**Accuracy**	**Precision**	**Recall**	**F1-score**
DT	0.894056	0.878958	0.864066	0.871448
RF	0.879974	0.857993	0.852246	0.85511
GB	0.829376	0.733313	0.92632	0.818593
XGB	0.888489	0.852908	0.884161	0.868253
AdaBoost	0.774685	0.731106	0.724192	0.727633
MLP	0.735713	0.747059	0.550433	0.633848
Voting	0.893074	0.869463	0.873916	0.871684

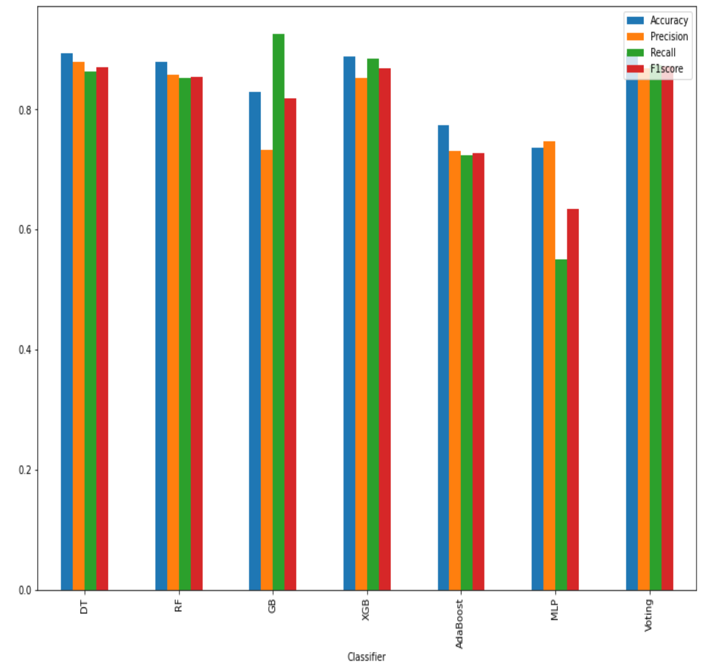


Fig. 6. Performance Results of Binary Classification.

TABLE V. Deep Learning Results

Algorithm	**Accuracy**	**Precision**	**Recall**	**F1-score**
LSTM	0.93	0.96	0.87	0.91
CNN	0.92	0.93	0.88	0.91
LSTM with CNN	**0.995**	**1.0**	**0.99**	**0.995**

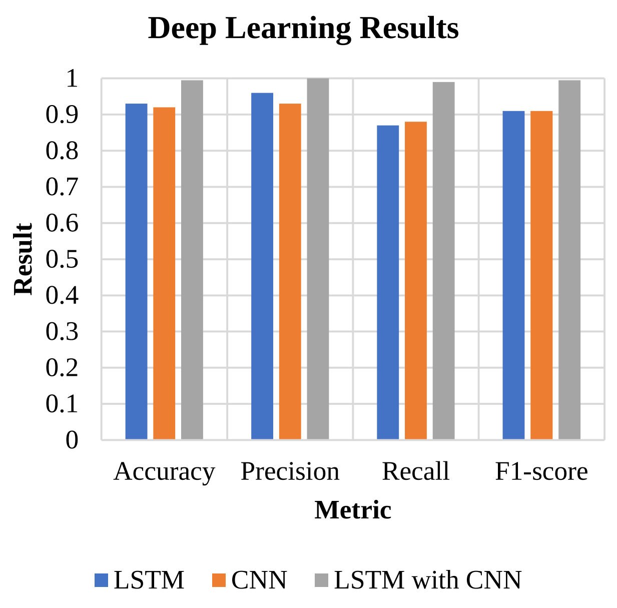


Fig. 7. Deep Learning Results.

V. Conclusion and Future Work

Eleven classification techniques, including three deep learning approaches [long short-term memory, convolutional neural networks, and long short-term memory + convolutional neural networks] [Decision Tree, Random Forest, Gradient Boosting, XGBoost, AdaBoost, Multilayer Perceptron, and Voting] were used in this study to identify suspicious assaults. Correctly predicted sample ratio is the most straightforward performance metric, as it is simply the predicted sample ratio. Using the Label Encoder method, we transformed the dataset from text to numbers. Algorithms are evaluated using the f1-score, along with precision, accuracy, and recall. In terms of detecting assaults, the LSTM with CNN outperformed the other models with scores of 0.995, 1.0, 0.99, and 0.995 for accuracy, precision, and recall, respectively. In this test, DT and voting classifiers outperformed the top machine learning methods. The results imply that a range of assault kinds can be detected using machine learning and deep learning. In a future study, we intend to assess the efficacy of these algorithms on a distinct dataset. In addition, we plan to implement ML, ML models, and DL approaches in our practice.

References

[1] Seemma, P. S., Nandhini, S., and Sowmiya, M. (2018), "Overview of cyber security", International Journal of Advanced Research in Computer and Communication Engineering, Vol. 7 No. 11, pp.125-128.

[2] Ervural, B. C., and Ervural, B. (2018), "Overview of cyber security in the industry 4.0 era", In Industry 4.0: managing the digital transformation, pp.267-284.

[3] Chowdhury, A. (2016), "Recent cyber security attacks and their mitigation approaches–an overview", In International conference on applications and techniques in information security, pp.54-65.

[4] El-Rewini, Z., Sadatsharan, K., Selvaraj, D. F., Plathottam, S. J., and Ranganathan, P. (2020), "Cybersecurity challenges in vehicular communications", Vehicular Communications, Vol. 23.

[5] Berman, D. S., Buczak, A. L., Chavis, J. S., and Corbett, C. L. (2019), "A survey of deep learning methods for cyber security", Information, Vol. 10 No.4.

[6] Kim, A., Park, M., and Lee, D. H. (2020), "AI-IDS: Application of deep learning to real-time Web intrusion detection", IEEE Access, Vol. 8, pp.70245-70261.

[7] Vartouni, A. M., Kashi, S. S., and Teshnehlab, M. (2018), "An anomaly detection method to detect web attacks using stacked auto-encoder". In 2018 6th Iranian Joint Congress on Fuzzy and Intelligent Systems (CFIS), pp. 131-134.

[8] Betarte, G., Pardo, Á., and Martínez, R. (2018), "Web application attacks detection using machine learning techniques", In 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA), pp.1065-1072.

[9] Tuan, T. A., Long, H. V., Kumar, R., Priyadarshini, I., and Son, N. T. K. (2019), "Performance evaluation of Botnet DDoS attack detection using machine learning", Evolutionary Intelligence, pp.1-12.

[10] Anwer, M., Farooq, M. U., Khan, S. M., and Waseemullah, W. (2021), "Attack Detection in IoT using Machine Learning", Engineering, Technology and Applied Science Research, Vol. 11 No. 3, pp.7273-7278.

[11] Su, T., Sun, H., Zhu, J., Wang, S., and Li, Y. (2020), "BAT: Deep learning methods on network intrusion detection using NSL-KDD dataset", IEEE Access, Vol. 8, pp.29575-29585.

[12] Xu, W., Jang-Jaccard, J., Singh, A., Wei, Y., and Sabrina, F. (2021), "Improving Performance of Autoencoder-Based Network Anomaly Detection on NSL-KDD Dataset", IEEE Access, Vol. 9, pp.140136-140146.

[13] Kavitha, S., and Uma Maheswari, N. (2021), "Network Anomaly Detection for NSL-KDD Dataset Using Deep Learning", Information Technology in Industry, Vol. 9 No. 2, pp.821-827.

[14] Ferriyan, A., Thamrin, A. H., Takeda, K., and Murai, J. (2021), "Generating Network Intrusion Detection Dataset Based on Real and Encrypted Synthetic Attack Traffic", Applied Sciences, Vol. 11 No. 17.

[15] Giménez, C. T., Villegas, A. P., and Marañón, G. Á. (2010), "HTTP data set CSIC 2010", Information Security Institute of CSIC (Spanish Research National Council).

[16] Hancock, J. T., and Khoshgoftaar, T. M. (2020), "Survey on categorical data for neural networks", Journal of Big Data, Vol. 7 No.1, pp.1-41.

[17] Pal, M. (2005). Random forest classifier for remote sensing classification. International journal of remote sensing, 26(1), 217-222.

[18] Farnaaz, N., & Jabbar, M. A. (2016). Random forest modeling for network intrusion detection system. Procedia Computer Science, 89, 213-217.

[19] Idhammad, M., Afdel, K., & Belouch, M. (2018). Detection system of HTTP DDoS attacks in a cloud environment based on information theoretic entropy and random forest. Security and Communication Networks, 2018.

[20] Kingsford, C., & Salzberg, S. L. (2008). What are decision trees?. Nature biotechnology, 26(9), 1011-1013.

[21] Quinlan, J. R. (1986). Induction of decision trees. Machine learning, 1(1), 81-106.

[22] De Ville, B. (2013). Decision trees. Wiley Interdisciplinary Reviews: Computational Statistics, 5(6), 448-455.

[23] Kotsiantis, S. B. (2013). Decision trees: a recent overview. Artificial Intelligence Review, 39(4), 261-283.

[24] Amor, N. B., Benferhat, S., & Elouedi, Z. (2004, March). Naive bayes vs decision trees in intrusion detection systems. In Proceedings of the 2004 ACM symposium on Applied computing (pp. 420-424).

[25] Noriega, L. (2005). Multilayer perceptron tutorial. School of Computing. Staffordshire University.

[26] Tang, J., Deng, C., & Huang, G. B. (2015). Extreme learning machine for multilayer perceptron. IEEE transactions on neural networks and learning systems, 27(4), 809-821.

[27] Ramchoun, H., Ghanou, Y., Ettaouil, M., & Janati Idrissi, M. A. (2016). Multilayer perceptron: Architecture optimization and training.

[28] Mitchell, R., & Frank, E. (2017). Accelerating the XGBoost algorithm using GPU computing. PeerJ Computer Science, 3, e127.

[29] Pan, B. (2018, February). Application of XGBoost algorithm in hourly PM2. 5 concentration prediction. In IOP conference series: earth and environmental science (Vol. 113, No. 1, p. 012127). IOP publishing.

[30] Dong, W., Huang, Y., Lehane, B., & Ma, G. (2020). XGBoost algorithm-based prediction of concrete electrical resistivity for structural health monitoring. Automation in Construction, 114, 103155.

[31] Hu, W., & Hu, W. (2005, September). Network-based intrusion detection using Adaboost algorithm. In The 2005 IEEE/WIC/ACM International Conference on Web Intelligence (WI'05) (pp. 712-717). IEEE.

[32] Jabri, S., Saidallah, M., El Alaoui, A. E. B., & El Fergougui, A. (2018, December). Moving vehicle detection using Haar-like, LBP and a machine learning Adaboost algorithm. In 2018 IEEE International Conference on Image Processing, Applications and Systems (IPAS) (pp. 121-124). IEEE.

[33] Yuan, L., & Zhang, F. (2009, July). Ear detection based on improved adaboost algorithm. In 2009 International Conference on Machine Learning and Cybernetics (Vol. 4, pp. 2414-2417). IEEE.

[34] Son, J., Jung, I., Park, K., & Han, B. (2015). Tracking-by-segmentation with online gradient boosting decision tree. In Proceedings of the IEEE international conference on computer vision (pp. 3056-3064).

[35] Peter, S., Diego, F., Hamprecht, F. A., & Nadler, B. (2017). Cost efficient gradient boosting. Advances in neural information processing systems, 30.

[36] Lusa, L. (2017). Gradient boosting for high-dimensional prediction of rare events. Computational Statistics & Data Analysis, 113, 19-37.

[37] Kumar, U. K., Nikhil, M. S., & Sumangali, K. (2017, August). Prediction of breast cancer using voting classifier technique. In 2017 IEEE international conference on smart technologies and management for computing, communication, controls, energy and materials (ICSTM) (pp. 108-114). IEEE.

[38] El-Kenawy, E. S. M., Ibrahim, A., Mirjalili, S., Eid, M. M., & Hussein, S. E. (2020). Novel feature selection and voting classifier algorithms for COVID-19 classification in CT images. IEEE Access, 8, 179317-179335.

[39] Khan, M. A., Khan Khattk, M. A., Latif, S., Shah, A. A., Ur Rehman, M., Boulila, W., ... & Ahmad, J. (2022). Voting classifier-based intrusion detection for iot networks. In Advances on Smart and Soft Computing (pp. 313-328). Springer, Singapore.

[40] Mahabub, A. (2020). A robust technique of fake news detection using Ensemble Voting Classifier and comparison with other classifiers. SN Applied Sciences, 2(4), 1-9.

[41] Monner, D., and Reggia, J. A. (2012), “A generalized LSTM-like training algorithm for second-order recurrent neural networks”, Neural Networks, Vol. 25, pp.70-83.

[42] Gers, F. A., Eck, D., and Schmidhuber, J. (2002), “Applying LSTM to time series predictable through time-window approaches”, In Neural Nets WIRN Vietri-01, pp.193-200.

[43] Sun, Y., Xue, B., Zhang, M., Yen, G. G., and Lv, J. (2020), “Automatically designing CNN architectures using the genetic algorithm for image classification”, IEEE transactions on cybernetics, Vol. 50 No. 9, pp.3840-3854.

[44] Dalianis, H. (2018), “Evaluation metrics and evaluation”, In Clinical text mining, pp.45-53.