Online-Journals.org

iJOE International Journal of Online and Biomedical Engineering

iJOE | eISSN: 2626-8493 | Vol. 20 No. 15 (2024) | OPEN ACCESS

https://doi.org/10.3991/ijoe.v20i15.50873

PAPER

Advanced Machine Learning Techniques for Enhancing Network Intrusion Detection and Classification Using DarkNet CIC2020

Archana Kalidindi[1], B. Raja Koti[2], CH. Srilakshmi[3], Krishna Mohan Buddaraju[4](✉), Akash Reddy Kandi[1], Gopi Sai Srinivas Makutam[1]

[1]Department of CSE-AIML & IoT, VNR VJIET, Hyderabad, Telangana, India

[2]Department of CSE, GITAM School of technology, GITAM University, Visakhapatnam, Andhra Pradesh, India

[3]Department of CSE-IoT, CBIT, Hyderabad, Telangana, India

[4]Department of Mechanical Engineering, GRIET, Hyderabad, Telangana, India

krishnamohan@griet.ac.in

ABSTRACT

The increase of illegal endeavors and malware traffic inside the Darknet presents a detracting challenge to cybersecurity. This study deals with the problems by applying leading machine learning (ML) techniques to reduce the use of the Darknet misuse while still preserving the aura of mystery, anonymity, and privacy. Leveraging the Darknet CIC2020 dataset, the research performs binary and multiclass classification, which are based on modern algorithms, with autoencoder being one of them. Convolutional neural networks (CNN), long short-term memory (LSTM), and XGBoost were used to discriminate the complicated systems. Results indicate XGBoost performed better in cases of both binary and multi-class classifications showing tremendous reliability, accuracy, recall, and F1-score. Furthermore, the study extends its scope by introducing ensemble techniques such as voting classifier and stacking classifier, aiming to enhance predictive accuracy by joining diverse base estimators. Combining autoencoder and XGBoost, alongside investigating the CNN+LSTM architecture, enhances the model's effectiveness. These hybrid approaches are implemented so that they can affect the components of various algorithms, creating more finely grained acting metrics on a per-order basis. This approach is to analyze Darknet traffic patterns that enable us to understand the importance of Internet security measures.

KEYWORDS

Darknet traffic, XGBoost, long short-term memory (LSTM), ensemble learning, autoencoders, voting classifier, stacking classifier

1 INTRODUCTION

The unexploited space that fails to establish contact with other computers in the globe is determined as Darknet. Due to its anonymity, as well as the existence of virtual markets and cryptocurrencies, it is called dark. However, Kumar et al. in their work,

Kalidindi, A., Raja Koti, B., Srilakshmi, CH., Buddaraju, K.M., Kandi, A.R., Srinivas Makutam, G.S. (2024). Advanced Machine Learning Techniques for Enhancing Network Intrusion Detection and Classification Using DarkNet CIC2020. *International Journal of Online and Biomedical Engineering (iJOE)*, 20(15), pp. 141–154. https://doi.org/10.3991/ijoe.v20i15.50873

Article submitted 2024-07-02. Revision uploaded 2024-09-09. Final acceptance 2024-09-09.

say that there is no such thing as "real users" in the dark web; any activity detected is generally considered unwanted, and they could be trial messages, probes for flaws in a system, or an actual cyberattack [1]. The work done by Lashkari et al. states that Darknet is usually famous as the epicenter of illegitimate web activities. Analysis of Darknet traffic is crucial to observe real-time activity happening over the Darknet. Darknet identification and the spotting of crimes on the Internet are made possible by understanding the Internet traffic that targets unused IPs [2]. Various perils such as distributed denial of services (DDoS) attacks, botnets, spoofing, probing, and scanning are found on Darknet traffic, as mentioned in the work done by Sarwar et al. After finishing extracting network traffic attributes, they found out how advanced threats were able to identify Darknet traffic patterns [3]. In the work proposed by Narisetty et al. a group of corrupted computer systems that perform an attack all at the same time on a victim, which could be a website, corporate server, or other network channel, is known as a DDoS attack. In the event of an attack, a corporation's accessibility and functionality might be heavily impaired. The excessive communications from these infected machines overwhelm the target systems, to the point where they either slow down or shut down completely, denying legitimate users or systems of some services [4].

This study has the goal to explore the Darknet using emerging intelligent machine approaches to overcome the problems in the Darknet traffic. Particularly, the research applies autoencoders, an artificial intelligence-based model that gained much popularity in the past, which are well-known in data analysis for their skill to uncover deeper patterns and structures that are hidden within the datasets. Allhusen et al. come up with the best practices of Darknet trafficking to increase the efficiency and accuracy of this highly challenging area [5]. Through the usage of autoencoders' inherent capacity to self-encrypt and decipher incoming data, investigators may find alternatives to typical network acts. These new inclusive rules are in the process of development and will eventually be laid out for more exact and beneficial discriminative criteria of dangerous behavior. The work covers different types of models and emphasizes which model fits better. The focus shifts to exploring different possibilities for machine intelligence that are going to be mentioned in the later part.

The Darknet CIC2020 is a dataset for network intrusion detection that includes both normal traffic and attacks. They encompass a broad class of attacks, including DDoS, botnets, and many more. As a result of high dimensionality, autoencoders can be used to decrease the number of input features while preserving crucial information present in the Darknet CIC2020 dataset. This is advantageous in terms of filtering noise and indicating the important characteristics of intrusion detection. Autoencoders can also perform well and have the ability to learn and identify anomalies, which is crucial to NIDS. They can teach the system how normal traffic should be, and anything that gets out of this can be considered an intrusion.

Lashkari et al. and Almomani in their work state that convolutional neural networks (CNNs) among the suggested models are used to separate sequentially due to their ability to extract the contextual words in conjunction with the nearby words, splitting spatial and temporal characteristics. Long short-term memory (LSTM) networks are also being developed to process sequential data and identify infinite dependencies. The proposed CNN+LSTM structure could be cognitive of the major hints by combining the benefits of CNNs and LSTMs. We can improve the model performance significantly through training the model on the given dataset [2, 6].

We know that CNNs can be used to extract spatial features (such as patterns in the network traffic data), and LSTMs can capture temporal dependencies, such as changes in traffic behavior over time. This combination is powerful for detecting sophisticated attacks. By using CNNs to first process the data and then applying LSTMs, you can build a model that understands both the content and context of network traffic.

Additionally, Almomani in their study employed XGBoost, an effective and proven method of machine learning (ML) used to study the hidden solutions in Darknet traffic data. Moreover, in contrast to relying on a single model choice, ensemble methods, such as a voting classifier and other algorithm combinations, aggregate the strengths of multiple models, leading to improved accuracy and robustness in intrusion detection. The stacking classifier performs the task of stacking various individual classifiers to boost their accuracy. Ensemble forms are liberating and amplifying the outputs of multiple classification models, and that's how their effect gets better by keeping a balance between traffic accuracy and free data usage. The most reliable algorithms are ensemble, hybrid ML models with high precision rates. By fusing different ML methods and soft computing approaches, hybrid ML models can be created. The development of the ensemble methods involves the use of many grouping techniques, such as boosting or bagging techniques for multiple ML classifiers [6]. Findings from studies indicate that XGBoost is designed to capture intricate patterns in data.

This study aims to influence the ongoing work in strengthening cybersecurity by using different approaches that integrate the features of various ML methods. It also aims to enhance cybersecurity infrastructure and protect against illicit activities within the Darknet environment.

2 RELATED WORK

Lashkari et al. present a novel approach called Deep Image Processing, which feeds a two-dimensional CNN with the gray image created by selecting the most significant features to detect and describe Darknet activity. An 86% accuracy is achieved by merging two encrypted traffic datasets to produce a Darknet dataset for the evaluation of the suggested method [2]. Sarwar et al. proposed a deep learning classifier based on CNNs and sophisticated, optimized ML algorithms that are used for Darknet traffic detection and categorization. The research was conducted on a state-of-the-art dataset, which includes eight categories of network packets. XGB was the most accurate classifier, with an 85% average F-score [3].

Kumar et al. [1] built a framework by utilizing Darknet traffic to identify threats. ML algorithms created a pattern that can identify both known and undiscovered threats by extracting the appropriate network traffic features of the threats from the Darknet data. Asad et al. [7] proposed a unique back-propagation-based deep neural network detection system to identify various application layer DDoS attacks effectively. Their suggested neural network architecture achieved an accuracy of 98%. Narisetty et al. recommended that activation and loss functions may be useful to defenders. By using the CICIDS 2017 dataset, the effect of these functions is evaluated with an SVM-RBF classifier. Their research tried to estimate a non-supervised stacked autoencoder for feature reduction using support vector machines (SVM) [4].

Almomani [6] proposed a stacking ensemble learning classifier for the analysis and classification of Darknet traffic. His novel approach makes use of predictions generated by three learning techniques to address Darknet attack issues. In the training and testing phases, the classifiers achieved accuracy of more than 99% and 97%, respectively. To detect evidence of illicit activity on the Darknet, Fernandez et al. [8] investigated the automatic classification of Tor Darknet images using semantic attention key point. Filtering, a technique that combines saliency maps with a bag of visual words (BoVW) to filter non-significant features at the pixel level that do not belong to the object of interest. Soro et al. [9] proposed a community detection-based approach to make it easier to analyze massive volumes of Darknet data. The proposed system automatically detects and isolates vertical, focused, and horizontal scans.

Choshen et al. [10] concluded that texts on the Darknet, whether legal or illicit, can be distinguished not only by the words they include but also by the superficial syntactic structure in the distribution of POS tags. Al-Haija et al. [11] proposed an efficient automated Darknet traffic detection system. The system achieved an accuracy of 99.5% with bagging decision trees. Jin et al. [12] proposed a DarkMor framework for detecting traffic on the dark web. The feature fusion model and traffic perception model make up DarkMor's key components. This model outperforms cross-modal algorithms by achieving an accuracy of 97.78%.

The between-class learning algorithm named CDBC was initially proposed by Song et al. [13] based on the Chebyshev distance. They also presented a Darknet detection framework based on CDBC. Their study uses two datasets and eleven different classifier types tested both in CDBC and non-CDBC contexts. The experimental findings demonstrate an accuracy of 99.99%. Saleem et al. [14] provide an extensive analysis of the existing methods for classifying encrypted network traffic within the Darknet and anonymous traffic. Their research also provides an in-depth analysis of Darknet traffic approaches that employ ML techniques to track and identify traffic attacks within the Darknet. To enhance the detection of Darknet traffic, Rust-Nguyen and M. Stamp in their study [15] evaluate the classification of traffic using SVM, random forest (RF), CNN, and auxiliary-classifier generative adversarial networks (AC-GAN). Using the CIC-darknet2020 dataset, they found that the RF model performs better than the cutting-edge ML methods. Within this context, the study done by Allhusen et al. [5] aims to evaluate the best prediction models through the analysis of dark web traffic. The results demonstrated the significance of feature selection. The accuracy of the model can occasionally be improved by selecting the appropriate features. Narisetty et al. proposed a hybrid intrusion detection system that utilizes a constraints optimized stacked autoencoder (COSAE) for dimensionality reduction to improve intrusion detection efficiency [16].

3 PROPOSED MODEL

The main focus of this work was to collect features from a CIC Darknet2020 dataset to recognize prominent features that would provide the most efficient results. To achieve this, we used the recursive feature elimination with cross-validation (RFECV) method in addition to the XGBoost. Through RFECV, we iteratively recognized and picked the most appropriate features according to their contribution to the model's performance, eventually diminishing the dimensionality of the dataset to a more controllable size. After feature selection, we try to analyze various ML models that determine the ultimate acceptable approach for classifying the data. Our assessment included autoencoder-CNN-LSTM, CNN-LSTM, XGBoost, voting classifier, and stacking classifier.

3.1 Data-set collection

In this study, we used the CIC Darknet 2020 dataset, incorporating approximately 140,000 rows and 85 columns, which is an excellent resource for cybersecurity studies and analysis. It uses a two-layered approach to achieve results for both Darknet and benign traffic. The origination of traffic occurs at the first layer. This layer generates both benign and Darknet traffic.

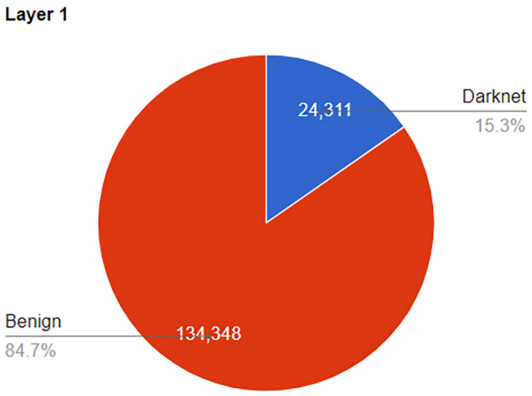

Fig. 1. The number of samples of both Darknet and benign traffic

Figure 1 indicates the division of the samples between Darknet and benign data. Benign data is 85% of the total dataset. Benign web traffic normally includes regular Internet actions, while Darknet traffic holds more suspicious or malicious activities. The second layer of the dataset is intact with different communication protocols, each represented as a specific category reflected in Figure 2. It exhibits eight categories, of which audio streaming is the largest portion, representing 55%. Darknet traffic is further categorized into various types, each representing different communication protocols.

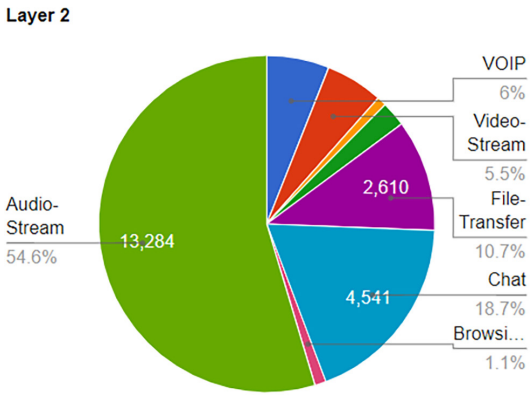

Fig. 2. Number of encrypted flows in Darknet traffic

Table 1. Details of Darknet network traffic

Traffic Category	Description	Applications Used
Audio-Stream	Transmission of audio data for real-time communication.	Vimeo and YouTube
Browsing	Web browsing activities involving HTTP/HTTPS requests and responses.	Firefox and Chrome
Chat	Instant messaging communication between users.	ICQ, AIM, Skype, Facebook and Hangouts
Email	Exchange of electronic mail messages using SMTP, IMAP, or POP3 protocols.	SMTPS, POP3S and IMAPS
P2P	Direct communication between network nodes for file-sharing and decentralized services.	uTorrent and Transmission (BitTorrent)
Transfer	Data transfers between endpoints using FTP, SFTP, or SCP protocols.	Skype, FTP over SSH (SFTP) and FTP over SSL (FTPS) using FileZilla and an external service
Video-Stream	Transmission of video data for real-time communication or streaming services.	Vimeo and YouTube

Table 1 presents a collection of various types of internet traffic along with the application of each category. Each of these classifications simulates specific network behaviors and communication protocols.

3.2 Normalization

Normalization is implemented using the min-max scaling method, which adjusts the attributes in the range of 0 and 1 by leveraging the minimum and maximum values of each feature.

3.3 Feature selection

Recursive feature elimination with cross-validation is used for feature selection. It continuously removes the features from the dataset and selects the optimum features. The initial step in RFECV is to train a ML model on the entire dataset and rank the significance of each feature. Then, the least significant features are detached, and the model can be applied to the reduced dataset. Until the optimal subset of features is obtained, the process is repeated recursively.

Cross-validation is employed all along each redundancy of feature elimination to assess the model's performance on various subsets of the data, guaranteeing that the feature selection process is robust and not relying on a single train-test split. RFECV is used to accompany a 10-fold cross-validation strategy utilizing an XGBoost classifier as the estimator. XGBoost can assess feature importance by tracking how frequently features are used in decision-making across the ensemble's trees. This evaluation helps identify which features significantly enhance the model's predictive performance. RFECV leverages this information to determine which features can be retained or removed without substantially impacting the model's accuracy. Specific features are chosen or removed based on their importance and relevance to the model's performance. Features with high-importance scores or a strong link to the target variable are kept because they help the model make accurate predictions. Features that are highly correlated with each other or add unnecessary noise are removed to simplify the model and prevent overfitting.

The number of features chosen by RFECV is decided based on accuracy and other performance metrics. Eventually, the chosen feature subset is utilized for analysis and modeling.

4 METHODOLOGY

An autoencoder with a CNN-LSTM framework is utilized for anomaly detection. The choice of utilizing a CNN-LSTM algorithm for the autoencoder is motivated because of the characteristics of the input data. This architecture is appropriate for processing the sequential data that contains both geographical and temporal ranges, which are common in applications such as time-series analysis. The encoder's CNN layers can efficiently capture spatial dependencies in the input data, while the decoder's LSTM layers can model temporal dependencies over time. Experimentation

with various architectures and hyperparameters is necessary to decide the optimum model for a given task.

The encoder part of the autoencoder is a collection of various thick layers accompanying various activation functions along with 'tanh' and 'relu.' These layers reduce the dimensionality of the input data to a great extent by compressing it into a latent representation accompanying fewer dimensions. The encoder layers evenly extract features from the input data and capture its latent patterns. Batch normalization is used after every dense layer to standardize next-layer inputs for the stable training process. The encoder design in the layer accompanying seven units represents the compressed latent space. This layer serves as the bottleneck of the autoencoder, urging it to gain a compressed representation of input data.

In essence, the decoder of the autoencoder tries to recreate the original input data from the compressed latent variable that is produced by the encoder. The work of the decoder layers mirrors that of the encoder layers in a descending manner. In every thick, the decoder is built to reassemble the original message using the compressed form. It searches for the most essential attributes only when decoding the input while neglecting the redundant information. In this way, it is forced to come up with the features of the data with the highest possibility for recognition. The important applications of autoencoders include unsupervised learning tasks such as anomaly detection and dimensionality reduction. They aim to reconstruct input data from compressed latent space by learning to represent the most significant features of the data.

Convolutional neural networks can capture spatial patterns from sequential data, and LSTMs are good at modeling temporal relationships. The CNN-LSTM model can get the benefits of both procedures by joining these two structures. CNNs are used in completing the tasks that include images or sequential data because of the ability to extract features and understand the underlying patterns of input data. LSTMs, on the other hand, are responsible for labeling temporal patterns and general dependencies, making them appropriate for subsequent data processing tasks where the input's order matters. When the input data has both spatial and temporal relationships, the CNN-LSTM architecture performs well.

The CNN-LSTM model begins with a series of convolutional layers (Conv1D) that extract relevant features from the input data by identifying spatial patterns. Max-pooling layers (MaxPooling1D) then reduce the dimensionality of these feature maps while preserving essential information. Following this, LSTM layers capture temporal dependencies, making them well-suited for sequential data analysis due to their ability to retain information over long sequences. This allows the model to learn complex patterns and relationships over time. Finally, the fully connected layers (Dense), with dropout regularization, perform classification based on the extracted data.

Figure 3 illustrates the training and test accuracies for CNN+LSTM models. Generally, the model will be more accurate as it gets exposed to larger amounts of data for more epochs. The test accuracy is generally lower than the training accuracy, and it could be either rising or falling based on time elapse. This is because the test accuracy reflects how well the model generalizes to new, unseen data. A model that overfits, perform well on training data but poorly on unseen data. In this case, the CNN+LSTM model appears to generalize well, as its test accuracy continues to improve over time.

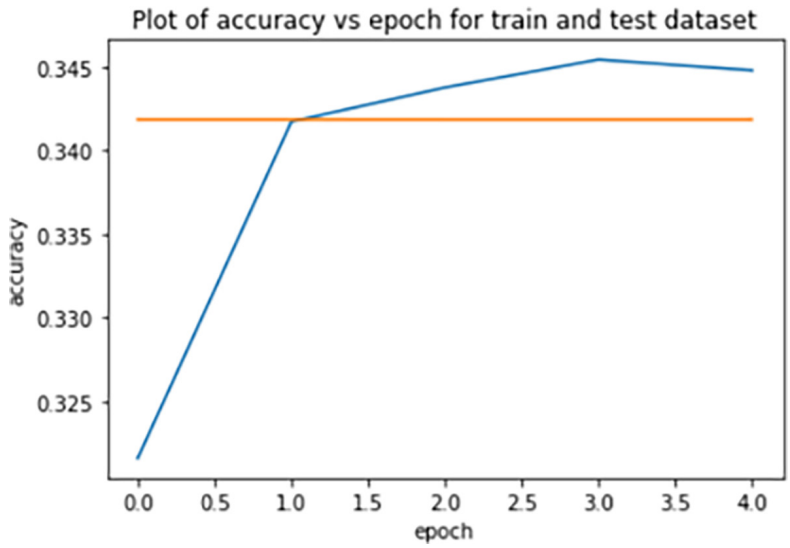

Fig. 3. CNN+LSTM plot of accuracy vs. epoch

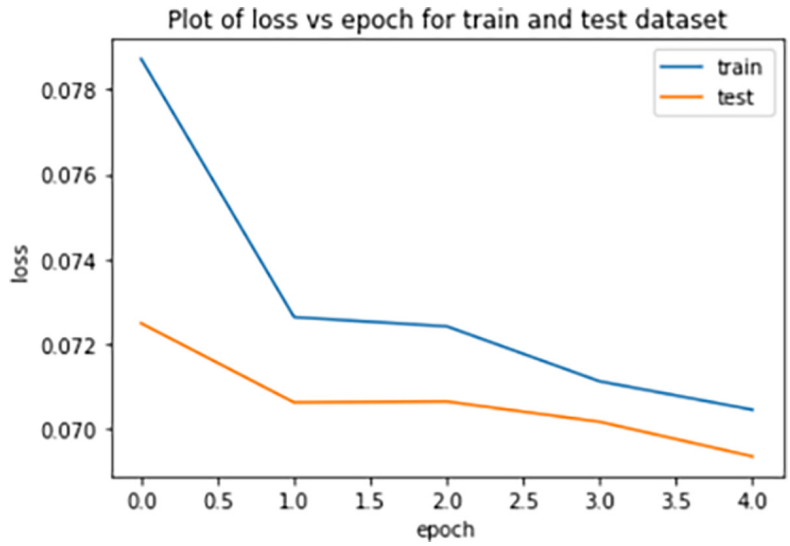

Fig. 4. CNN+LSTM plot of loss vs. epoch

Figure 4 illustrates how the loss varies with the epoch number for both training and test data sets. In this scenario, the training loss should decrease whenever the model recognizes the patterns of the training data. The test loss is used for keeping track of how well the model generalizes to the data. Here, the training loss seems to be dropping gradually as the CNN+LSTM model learns different patterns in the training data. The value of test loss is also decreasing; this may be an indication that the model is performing well on the unseen data.

One of the ensembles learning techniques for classification tasks is the voting classifier. It forecasts which class will obtain the most votes (the mode) or which class will have the highest average probability (soft voting) among all the classifiers by combining the predictions of several different classifiers. Here, the RF classifier and the decision tree classifier are the two predictors to be fed to the voting classifier in order to create its predictions. The technique of bagging is the base for the RF classifier, where the forest is built by many decision trees.

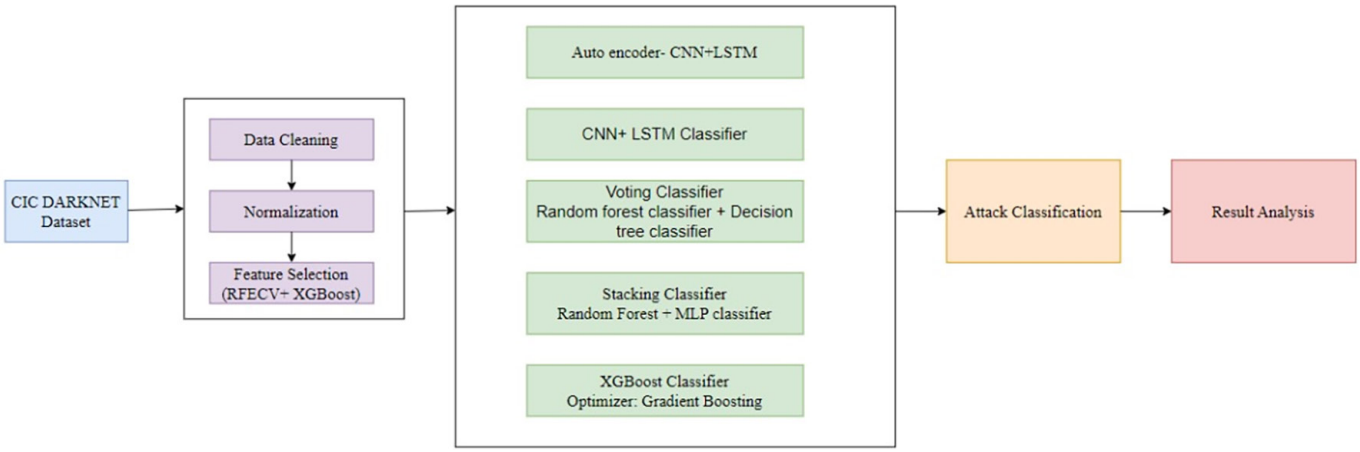

Fig. 5. Workflow diagram of proposed methodology

Figure 5 shows the proposed approach to classify different types of attacks. The diagram illustrates various stages in the process of network traffic classification. This involves several stages: collecting network data, cleaning, normalization, feature selection with RFECV and XGBoost, assessing five ML models to see which model is best, and then classifying it as Darknet or normal categories.

The RF classifier is a well-known algorithm for classification because of its resilience capacity to handle high-dimensional data with ample characteristics. Decision trees, which divide the feature space into smaller sections, and effective classification algorithms. They are renowned for their ability to understand complex relationships within the data and for being interpretable.

The voting classifier makes its final prediction using either the weighted average of probabilities (soft voting) or the most common class label (hard voting), incorporating the outputs of the two classifiers. In this case, "soft" voting is used, meaning that the final prediction is based on the probability scores from each base classifier. These two models were chosen for the voting classifier because of their complementary strengths: the decision tree classifier excels at capturing complex decision boundaries, while the RF classifier provides stability and robustness through its ensemble method. By combining these models, the voting classifier aims to reduce overfitting and enhance generalization performance.

The stacking classifier is an effective ensemble model used to boost predictive performance by combining the predictions of several base classifiers. Although its benefits include adaptability, model variety, and metaknowledge, the best choice depends on the dataset's features. RF is a powerful ensemble learning method based on decision trees. It is known for its ability to manage complex data interactions. It is one of the fundamental models, as it does well in categorization tasks and is less likely to overfit. A multi-layer perceptron classifier is a type of artificial neural network which can discover difficult connections and understand complex patterns.

5 RESULTS AND DISCUSSIONS

The efficiency of each model is assessed using metrics such as accuracy, precision, F1-score, and recall.

Table 2. Comparison table for binary classification

S. No.	ML Model	Accuracy	Precision	Recall	F1-Score
1	Autoencoder-CNN-LSTM	0.830	0.805	0.830	0.811
2	CNN-LSTM	0.827	1.000	0.827	0.905
3	XGBoost	0.984	0.984	0.984	0.984
4	Voting Classifier	0.998	0.998	0.998	0.998
5	Stacking Classifier	0.993	0.993	0.993	0.993

Table 2 displays the accuracy, precision, and F1-score metrics for various ML models in binary classification tasks. Similarly, Table 3 presents the corresponding metrics for multi-class classification tasks.

Table 3. Comparison table for multi classification

S. No.	ML Model	Accuracy	Precision	Recall	F1-Score
1	Autoencoder-CNN-LSTM	0.117	0.014	0.117	0.025
2	CNN-LSTM	0.338	1.000	0.338	0.505
3	XGBoost	0.864	0.880	0.864	0.869
4	Voting Classifier	0.972	0.972	0.972	0.972
5	Stacking Classifier	0.911	0.919	0.911	0.912

Accuracy: This factor is the ratio of the correct predictions made by the model, and all the predictions it made.

$$Accuracy = (TP + TN)/(TP + TN + FP + FN)$$

where TP: true positives, TN: true negatives, FP: false positives, FN: false negatives.

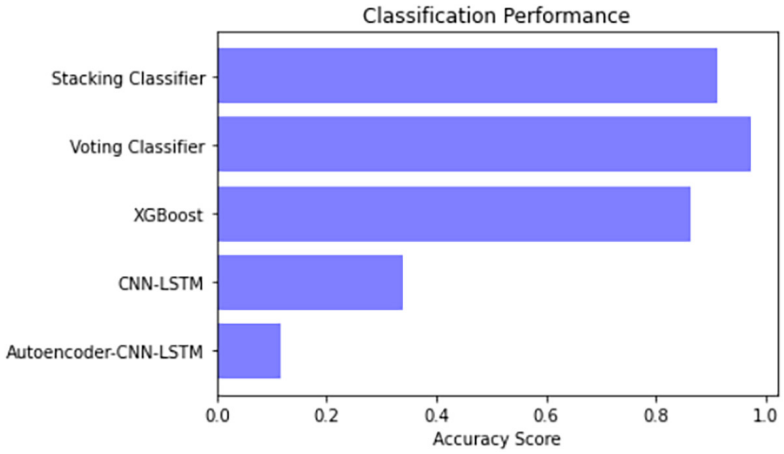

Fig. 6. Accuracy comparison

Figure 6 portrays the classification performance of five ML model attempts on a single task. Here, the voting classifier model is the most accurate, with a considerable margin ahead of the stacking classifier model or XGBoost model. Among the five

models are the CNN-LSTM model and the autoencoder-CNN-LSTM model, which has the rank of the lowest accuracy.

Precision: It is the fraction of positive instances that occur as predicted. This metric is generally used when the false positive rate is high. The comparison of precision scores is shown in Figure 7.

$$Precision = TP/(TP + FP)$$

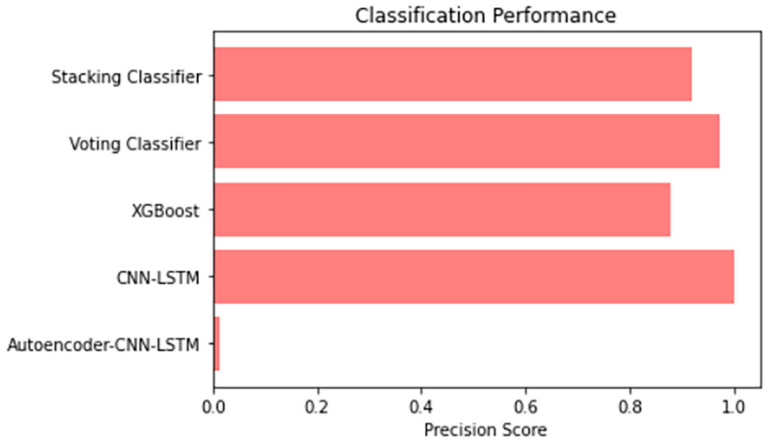

Fig. 7. Precision comparison

Recall: It measures the capability of a model to correctly predict all examples of positive class. A high recall value means the model is very efficient at discovering positive samples. On the other hand, a low recall value has some of the true samples missed. The bar graph in Figure 8 gives us a clear understanding of recall scores for different models.

$$Recall = TP/(TP + FN)$$

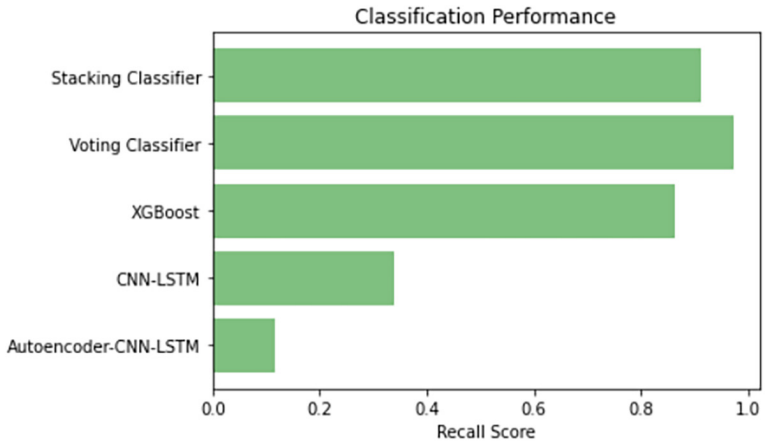

Fig. 8. Recall comparison

F1 Score: It is a metric for solving binary classification problems, which is given by the harmonic mean of precision and recall. It considers both precision and recall to present a comprehensive evaluation of the model.

$$F1\ Score = 2*(Precision * Recall)/(Precision + Recall)$$

The autoencoder-CNN-LSTM model may be good at telling normal traffic from abnormal, but it struggles to identify different types of attacks. Because the dataset is a bit imbalanced, the model might focus more on the most common type of data (such as attack vs. no attack) instead of learning to identify the less common attack types. This could be why it doesn't perform well in tasks where it needs to classify multiple types of attacks.

6 ALGORITHM

1. Load the dataset and handle missing values, if any.
2. Drop unnecessary columns such as 'Flow ID', 'Src IP', 'Dst IP', 'Timestamp', 'Flow Bytes/s', 'Flow Packets/s'.
3. Encode categorical labels using label encoding.
4. Split the dataset into features (X) and labels (y).
5. Use XGBoost with RFECV to select relevant features:
   - Initialize XGBClassifier and RFECV.
   - Fit RFECV on the dataset.
   - Get the selected features.
6. Split the dataset into training and testing sets.
7. Train various models:
   - Train autoencoder-CNN-LSTM:
   - Build an autoencoder with CNN and LSTM layers.
   - Train the autoencoder on the negative class instances.
   - Use the encoded representation as input for a classifier.
   - Train the CNN-LSTM model with Conv1D, MaxPooling1D, LSTM, and dense layers.
   - Train the XGBoost classifier.
   - Train voting classifier combining RF and Decision Tree classifiers.
   - Train stacking classifier combining RF and MLP classifiers with a final LGBM classifier.
8. Evaluate the models:
   - Evaluate models using accuracy, precision, recall, and F1-score.
9. Select the model with the best performance metrics.

The proposed algorithm aims to improve cybersecurity threat detection using ML. First, it cleans and prepares the dataset by removing duplicates. Then, it uses XGBoost with RFECV to select the best features for the model. Several models, including autoencoder-CNN-LSTM, CNN-LSTM, XGBoost, voting classifier, and stacking classifier, are trained on the processed data. The models are evaluated based on accuracy, recall, and F1-score to see how well they perform. Finally, the best-performing model is chosen and saved for use in real-world cybersecurity threat detection.

7 CONCLUSION

The proposed system architecture in this study addresses a wide range of issues related to Darknet activities by utilizing binary and multi-class classification techniques. By incorporating advanced algorithms such as autoencoder, CNN+LSTM, and XGBoost, the system effectively analyzes complex network traffic. The implementation of voting and stacking classifiers enhances accuracy, while the combination

of autoencoder and XGBoost, along with the CNN+LSTM experiment, strengthens the model's robustness. Future research should focus on refining the Darknet traffic classifier, exploring new deep learning networks, developing more realistic datasets, and innovating methods to improve accuracy. However, challenges remain, such as class imbalance in the dataset, which leads to the model favoring majority classes and the CNN-LSTM model's difficulty in distinguishing similar classes. These limitations suggest the need for more diversified data and improved generalization to enhance model accuracy.

8 REFERENCES

[1] S. Kumar, H. Vranken, J. van Dijk, and T. Hamalainen, "Deep in the dark: A novel threat detection system using darknet traffic," in *2019 IEEE International Conference on Big Data (Big Data),* Los Angeles, CA, USA, 2019, pp. 4273–4279. https://doi.org/10.1109/BigData47090.2019.9006374

[2] A. Habibi Lashkari, G. Kaur, and A. Rahali, "DIDarknet: A contemporary approach to detect and characterize the darknet traffic using deep image learning," in *Proceedings of the 2020 10th International Conference on Communication and Network Security,* 2020, pp. 1–13. https://doi.org/10.1145/3442520.3442521

[3] M. B. Sarwar, M. K. Hanif, R. Talib, M. Younas, and M. U. Sarwar, "DarkDetect: Darknet traffic detection and categorization using modified convolution-long short-term memory," *IEEE Access*, vol. 9, pp. 113705–113713, 2021. https://doi.org/10.1109/ACCESS.2021.3105000

[4] N. Narisetty, G. R. Kancherla, B. Bobba, and K. Swathi, "Hybrid intrusion detection method based on constraints optimized SAE and grid search based SVM-RBF on cloud," *International Journal of Computer Networks and Applications*, vol. 8, no. 6, pp. 776–787, 2021. https://doi.org/10.22247/ijcna/2021/210725

[5] A. Allhusen, I. Alsmadi, A. Wahbeh, M. Al-Ramahi, and A. Al-Omari, "Dark web analytics: A comparative study of feature selection and prediction algorithms," *SSRN Electronic Journal*, pp. 6, 2021. https://doi.org/10.2139/ssrn.3949786

[6] A. Almomani, "Darknet traffic analysis, and classification system based on modified stacking ensemble learning algorithms," *Information Systems and e-Business Management*, 2023. https://doi.org/10.1007/s10257-023-00626-2

[7] M. Asad, M. Asim, T. Javed, M. O. Beg, H. Mujtaba, and S. Abbas, "DeepDetect: Detection of distributed denial of service attacks using deep learning," *The Computer Journal*, vol. 63, no. 7, pp. 983–994, 2019. https://doi.org/10.1093/comjnl/bxz064

[8] E. F. Fernandez, R. A. V. Carofilis, F. J. Martino, and P. B. Medina, "Classifying suspicious content in Tor Darknet," *arXiv preprint arXiv:2005.10086*, 2020.

[9] F. Soro, M. Allegretta, M. Mellia, I. Drago, and L. M. Bertholdo, "Sensing the noise: Uncovering communities in darknet traffic," in *2020 Mediterranean Communication and Computer Networking Conference (MedComNet)*, 2020, pp. 1–8. https://doi.org/10.1109/MedComNet49392.2020.9191555

[10] L. Choshen, D. Eldad, D. Hershcovich, E. Sulem, and O. Abend, "The language of legal and illegal activity on the darknet," *arXiv preprint arXiv:1905.05543v2*, 2019. https://doi.org/10.48550/ARXIV.1905.05543

[11] Q. Abu Al-Haija, M. Krichen, and W. Abu Elhaija, "Machine-learning-based darknet traffic detection system for IoT applications," *Electronics*, vol. 11, no. 4, p. 556, 2022. https://doi.org/10.3390/electronics11040556

[12] Y. Jin *et al.*, "Darkmor: A framework for dark web traffic detection that integrates local feature and spatial characteristics," *SSRN Electronic Journal,* pp. 18, 2023. https://doi.org/10.2139/ssrn.4648074

[13] B. Song, Y. Chang, M. Liao, Y. Wang, J. Chen, and N. Wang, "CDBC: A novel data enhancement method based on improved between-class learning for darknet detection," *Mathematical Biosciences and Engineering*, vol. 20, no. 8, pp. 14959–14977, 2023. https://doi.org/10.3934/mbe.2023670

[14] J. Saleem, R. Islam, and M. Z. Islam, "Darknet traffic analysis: A systematic literature review," *IEEE Access*, vol. 12, pp. 42423–42452, 2024. https://doi.org/10.1109/ACCESS.2024.3373769

[15] N. Rust-Nguyen and M. Stamp, "Darknet traffic classification and adversarial attacks," *arXiv preprint arXiv:2206.06371*, 2022. https://doi.org/10.48550/ARXIV.2206.06371

[16] N. Narisetty, Ayesha, T. Abid, G. S. Rao, and M. S. Nidhya, "Supervised learning approach for detection of malicious nodes," in *2024 5th International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV)*, Tirunelveli, India, 2024, pp. 882–885. https://doi.org/10.1109/ICICV62344.2024.00145

9 AUTHORS

Archana Kalidindi is an Assistant Professor in CSE-AIML & IoT, VNR VJIET Hyderabad, India. She is currently pursuing Ph.D. in Computer Science in the area of IoT and Deep Learning. Her research interest includes Deep Learning, Data Science, and Machine Learning (E-mail: archana.buddaraju@gmail.com).

Dr. B. Raja Koti completed his Ph.D. in Information technology from GITAM University, Visakhapatnam, India. He is working as an Assistant Professor in the department of computer science and engineering, GITAM School of technology, GITAM University, Visakhapatnam. His research interest includes cyber security, computer networks security, cryptography, cloud computing and blockchain technology (E-mail: rbadugu@gitam.edu).

CH. Srilakshmi is working as an Assistant Professor in CSE-IOT Department, CBIT. She is currently pursuing Ph.D. in Computer Science. Her areas of interest include deep learning, machine learning and computer vision (E-mail: chsrilu.45@gmail.com).

Dr. Krishna Mohan Buddaraju completed his Ph.D. in Mechanical Engineering from National Institute of Technology, Andhra Pradesh. His research interests include finite element analysis, bio mechanics, robotics, IOT, and artificial intelligence and machine learning (E-mail: krishnamohan@griet.ac.in).

Akash Reddy Kandi has earned his bachelor's degree in Computer Science and Engineering, specializing in Artificial Intelligence and Machine Learning, from VNR VJIET. He is currently pursuing his Master's degree, with a focused interest in machine learning, deep learning, and data analysis (E-mail: akash.kandi24@gmail.com).

Gopi Sai Srinivas Makutam has earned his bachelor's degree in Computer Science and Engineering, specializing in Artificial Intelligence and Machine Learning, from VNR VJIET. His area of interest falls in the application of Deep Learning, Data Science, and Machine Learning in order to create more challenging solutions and embellish in the field (E-mail: gopisai1902@gmail.com).