Threat Detection Using Dilated Convolutional Neural Networks

[1]Gunturu.Maheswarareddy,[2]Alaparthi.Yaswanth,[3]Janga.Vedeshreddy,
[4]Andagula.Trinadh, [5]Mr. Y.V. Narayana, Assistant Professor.

[1,2,3,4,5]Dept. of Information Technology, VVIT, Andhra Pradesh, India

---

Abstract

With cyber threats like Distributed Denial of Service (DDoS) attacks becoming more sophisticated, protecting online services has never been more critical. Traditional threat detection systems often Depend on preset rules, which limits their effectiveness.at spotting new and evolving cyber threats, including Previously unknown attacks. To mitigate this, this paper presents a threat detection system powered by dilated convolutional neural networks (DCNNs), which are highly effective at analyzing network traffic patterns by capturing long-range dependencies. This allows for more accurate and efficient threat detection. The model is trained using supervised learning on a dataset containing both normal and unwanted traffic, with preprocessing and data normalization techniques applied to handle issues like data imbalance and noise. Performance is Assessed based on essential classification Indicators such as accuracy, precision, recall, F1-score, and ROC-AUC are used for evaluation, ensuring reliability in identifying Online threats. Testing results indicate tha DCNN-based system significantly outperforms traditional CNN and RNN models, achieving 99.03% accuracy, 98.88% precision, 99.03% recall, 98.95% F1-score, and 99.27% ROC-AUC. These results highlight the effectiveness and scalability of the proposed model in real-time threat detection, offering organizations a powerful tool to strengthen cybersecurity and proactively defend against emerging cyber attacks..

Key Words:

Cybersecurity, Threat Detection, DDoS Mitigation, Network Security, Deep Learning, Dilated Convolutional Neural Networks, Anomaly Detection, Real-Time Security.

1. INTRODUCTION

With the rise in cyber threats, especially Distributed Denial of Service (DDoS) attacks, ensuring network security has become a major concern. Conventional Threat Detection Systems (TDS) largely depend on static rules, which are inadequate for identifying new and advanced attacks. To improve upon these limitations, this study introduces a novel TDS utilizing Dilated Convolutional Neural Networks (DCNNs) to effectively recognize both known and emerging threats. DCNNs are particularly effective in processing network traffic patterns by capturing long-range dependencies, thereby improving accuracy in threat detection. vulnerable.

2. LITERATURE REVIEW

In today's digital world, cybersecurity is more crucial than ever, with organizations constantly battling increasingly sophisticated cyber threats, from the internet, especially Distributed Denial of Service (DDoS) attacks Traditional Threat Detection Systems (TDS) They mainly depend on established rules and recognizable attack patterns, which makes them less effective against emerging and evolving threats, such as unknown attacks

To handle these challenges, machine learning (ML) and deep learning (DL) have been utilizedhave emerged as promising approaches in network security. Among these, **Dilated Convolutional Neural Networks (DCNNs)** stand out due to their ability to detect cyber threats by capturing long-range dependencies in network traffic. This section reviews existing research on ML/DL-based threat detection techniques and highlights the advantages of DCNNs over conventional models.

2.1 Traditional Threat Detection Approaches

Threat Detection Systems (TDS) Are typically classified into two main categories:

Pattern-Based TDS (PBTDS): These systems, like **Snort** and **Suricata**, rely on predefined attack signatures. Although they work well against known threats, they are unable to identify unknown attacks(Garcia et al., 2019).

- **Anomaly-Based TDS (ABTDS):** These systems use statistical analysis or machine learning to identify unusual traffic patterns (Santos et al., 2020). While more adaptive than SBTDS, they often struggle with high false-positive rates and evolving attack tactics.

Given these challenges, there's a growing need for more intelligent and adaptive solutions.

2.2 Machine Learning-Based TDS

Researchers have explored various ML techniques to enhance threat detection, including:

- **Support Vector Machines (SVM):** Useful for classifying attacks, but resource-intensive when handling large datasets.(Wang et al., 2021).

- **Random Forest (RF):** Efficient for processing large datasets but struggles when dealing with highly imbalanced network traffic (Liu et al., 2022).

- **K-Nearest Neighbors (KNN):** Performs well on smaller datasets but lacks the scalability required for real-time detection (Rizvi et al., 2023).

While ML techniques improve detection capabilities, they still require extensive feature engineering and often fail to generalize well to newly emerging threats.

2.3 Deep Learning for TDS

Deep learning offers a more advanced approach by automatically learning patterns in network traffic without manual feature extraction. Several architectures have been explored:

- **Convolutional Neural Networks (CNNs):** CNNs are able to detect local traffic patterns but struggle with capturing sequential dependencies, Restricting their capability to identify prolonged attack patterns(Fu et al., 2019; Zhang et al., 2021).

- **Recurrent Neural Networks (RNNs):** For sequential data analysis, RNNs improve upon CNNs but Encounter vanishing gradient problems, which hinder the retention of long-term dependencies(Hassan et al., 2021; Qiu et al., 2020). Additionally, they require high computational power, making real-time implementation challenging.

- **Long Short-Term Memory Networks (LSTMs):** LSTMs overcome vanishing gradient challengesproblems and can model complex attack sequences (Yue et al., 2021). However, they require extensive training time and consume significant computational resources.

- **Transformers for TDS:** Attention-based models improve long-range dependency learning (Zhang et al., 2023). However, their high training costs make them impractical for real-time threat detection.

Key Advantages of DCNNs in TDS:

1. **Captures both local and global dependencies** in network traffic without needing recurrent layers, making it more efficient (Gupta et al., 2022).
2. **More computationally efficient** compared to LSTMs and Transformers.
3. **Highly scalable**, allowing for real-time network traffic analysis.
4. **Proven effectiveness in recent research:**
5. **Li et al. (2022):** Demonstrated that DCNN outperforms CNN and RNN models in detecting DDoS, Probe, and R2L attacks, achieving an impressive accuracy of **97.5%**.
6. By leveraging DCNNs, cybersecurity systems can improve their ability to detect both known and emerging threats, offering a more adaptive and reliable solution for modern network security challenges.

2.5 Comparative Analysis of TDS

Models/Parameters	**DCNN**(Proposed)	**CNN**	**RNN**
Accuracy(%)	99.03	98.34	94.13
Precision(%)	98.88	98.34	92.96
F1 Score(%)	98.95	98.29	93.23
Recall(%)	99.03	98.34	94.13
ROC-AUC(%)	99.27	96.52	84.65

Fig- 1:Comparative Analysis of TDS

Conclusion:

The literature review traces the advancements in Threat Detection Systems (TDS), showcasing how DCNNs outperform both traditional and other deep learning-based models. With their ability to recognize intricate attack patterns, DCNNs prove to be a highly effective solution for cybersecurity. Building on previous research, this study introduces a robust, real-time TDS powered by DCNNs, trained on the CIC-IDS2017 dataset to enhance threat detection capabilities.

3. PROPOSED SYSTEM

The proposed system enhances conventional Threat Detection Systems (TDS) by incorporating **Dilated Convolutional Neural Networks (DCNNs)**, which are highly effective at recognizing subtle and complex patterns in network traffic. By utilizing deep learning, the system continuously evolves, improving Its capability to distinguish between legitimate and malicious activity. With its capability to analyze long-range dependencies, the DCNN-based model ensures real-time detection of emerging cyber threats, strengthening overall network security. This advanced approach provides a more **efficient, adaptable, and scalable** solution to counter sophisticated cyberattacks.

3.1 Proposed System Architeture:

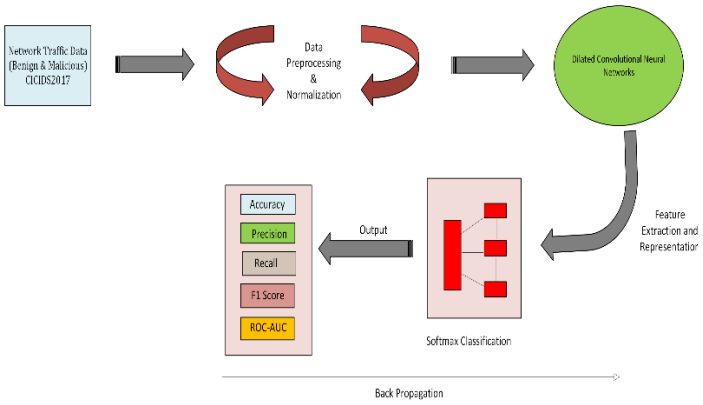

Fig-2:System Architeture

3.2 Proposed System Algorithm

Step 1: Load and Prepare the Dataset

1. Start by loading the dataset from the given file location.
2. Since the target labels are categorical (like class names), convert them into numbers using LabelEncoder. This helps the model understand them better.
3. Separate the dataset into two parts: **features (X)**, which the model will use to learn patterns, and **labels (y)**, which are the correct answers.
4. Standardize the feature values using StandardScaler so that all the The input data is within a comparable range
5. , making learning more efficient.
6. Convert the numerical labels into a categorical format to work well with classification models.
7. Use 80% of the data for training and 20% for testing. The training data helps the model learn, and the testing data checks how well it performs.

8. Step 2: Define the Dilated CNN Model

1. Set up a sequential neural network, which processes data step by step.
2. Add the first one-dimensional convolutional layer with 64 filters and a kernel size of 3, a dilation rate of 1, and ReLU activation to help detect patterns.
3. Add another **1D convolutional layer**, Comparable to the first but with a **dilation rate of 2**, allowing the model to recognize patterns at a larger scale.
4. Add a third **1D convolutional layer** with a **dilation rate of 4**, helping the model capture even more complex relationships in the data.
5. Use **Global Average Pooling** to reduce Reduces the number of features while preserving essential information
6. Add A fully connected dense layer containing 128 neurons with ReLU activation to further process the extracted features.
7. Include a **Dropout layer (30% dropout rate)** to Avoid overfitting and enhance the model's performance.
8. Finally, add an output A dense layer with softmax activation to enable the model to output probability distributions for each class in a multi-class classification task.

Step 3: Train and analyse the Model

1. Train the **Dilated Convolutional Neural Network (DCNN)** using The training dataset (X_train, y_train), where the model identifies patterns from the data
2. Set the training guidelines: run the model for **20 epochs** (cycles of learning) and use a batch size of 32, indicating that the design processes 32 samples at once before making updates to its learning.
3. During training, Validate the model using the test data (X_test, y_test) to monitor its performance on unseen data.
4. Once training is complete, Assess the model's accuracy on the test dataset to determine its capability of spread new data.

5. Step 4: Display Results and Save the Model

1. Print the final test accuracy in a percentage format.
2. Save the trained model as "dcnn_threat_detection_model.h5" for future use.

4 .OPTIMIZATION TECHNIQUE

Softmax classification is widely used for multi-class classification problems, including threat detection systems. It converts the raw output values (logits) from a neural network into probability scores, making it easier to classify an instance into one of several categories.

4.1 What is Softmax Classification?

Softmax is an activation function used in the result layer of a multi-class neural network. It ensures that the outputs are interpretable as probabilities, meaning

The total of all class probabilities is 1, with the highest probability indicating the predicted class.

In a Threat Detection System, Softmax helps in categorizing network traffic into different classes, such as:

1. Benign Traffic
2. DDoS Attack
3. Malware Attack

Softmax Formula

For a given input vector z containing raw scores (logits) for N classes, the Softmax function is defined as:

$$P(y_i) = \frac{e^{zi}}{\sum_{i=1}^{i=z} e^{zi}} \quad \text{.....(1)}$$

Where:

- $p(y_i)$ = Probability of class iii.
- zi= Logit (raw score) for class iii.
- e^{zi} = Exponential function applied to the logit.
- $\sum_{i=1}^{i=z} e^{zi}$ = Sum of exponentials across all NNN classes (normalization factor).

This ensures that all output probabilities sum to 1.

4.2 Softmax Classification Algorithm :

1. Input Layer:
    - The model receives an input feature vector X.
    - It is processed through the neural network layers.
2. Compute Logits (Raw Scores):
    - The final fully connected (dense) layer produces a logit vector Z where each element represents the raw score for a class.
    - Logits are given by:

$$Z = W \cdot X + bZ \quad \text{.....(2)}$$

Where:

W= Weights

X = Input features

b = Bias terms

3. Apply the Softmax Function:

$$P(y_i) = \frac{e^{zi}}{\sum_{i=1}^{i=z} e^{zi}} \quad \text{.....(3)}$$

Where:

$p(y_i)$ = Probability of class iii.

zi= Logit (raw score) for class iii.

e^{zi} = Exponential function applied to the logit.

$\sum_{i=1}^{i=z} e^{zi}$ = Sum of exponentials across all NNN classes

4. Predicted Class Selection:
    - The class with the highest probability is selected as the final prediction

$$y_{pred} = \arg max\ P(y_i) \quad \text{.....(4)}$$

5. Compute Loss (Cross-Entropy):
    - The cross-entropy loss function measures how closely the predicted probabilities align with the actual labels.:

$$L = -\sum_{i=1}^{i=n} y_i\, log\big(P(y_i)\big) \quad \text{.....(5)}$$

6. Backpropagation & Optimization:
    - The gradient descent algorithm updates the model weights based on the loss function to minimize errors.
    - Common optimizers: Adam, SGD, RMSprop.
7. Model Training & Evaluation:
    - The training process utilizes mini-batch gradient descent.
    - Performance is measured using accuracy, precision, recall, F1-score, and ROC-AUC.

4.3 Pictorial Representation of Softmax

Below is a visual representation of how the Softmax function transforms raw network outputs into class probabilities.

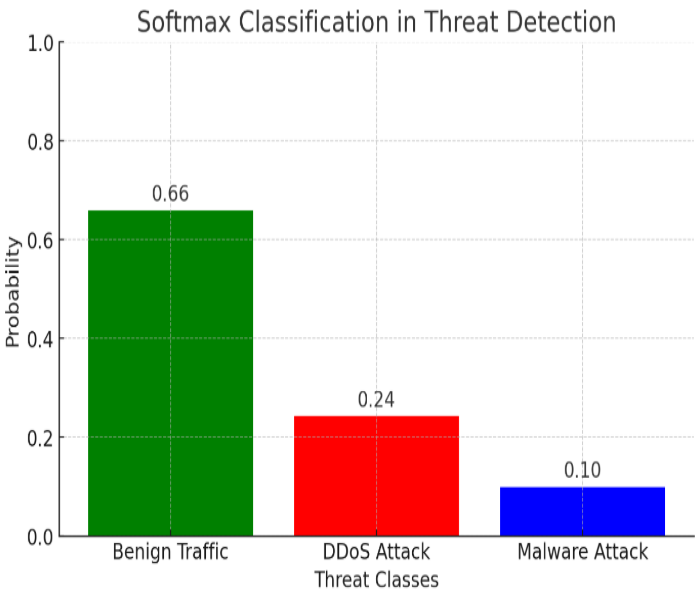

Chart-1: Softmax Classification

5. PERFORMANCE ANALYSIS

This part provides the experimental assessment of the DCNN-based Threat Detection System. The proposed model's performance is evaluated in comparison to CNN and RNN models using common metrics like accuracy, precision, recall, F1-score, and ROC-AUC

Performance Metrics

The model's performance is evaluated using the following metrics.:

5.1 Accuracy :

Accuracy is a mostly used metric for analysing classification models, including Threat Detection Systems (TDS). It indicates the frequency of correct predictions made by the model

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad \text{.....(6)}$$

Where:

- TP (True Positives): Correctly predicted threats.
- TN (True Negatives): Correctly predicted benign traffic.
- FP (False Positives): Incorrectly classified benign traffic as a threat (false alarm).
- FN (False Negatives): Incorrectly classified threats as benign (missed attacks).

5.1.1

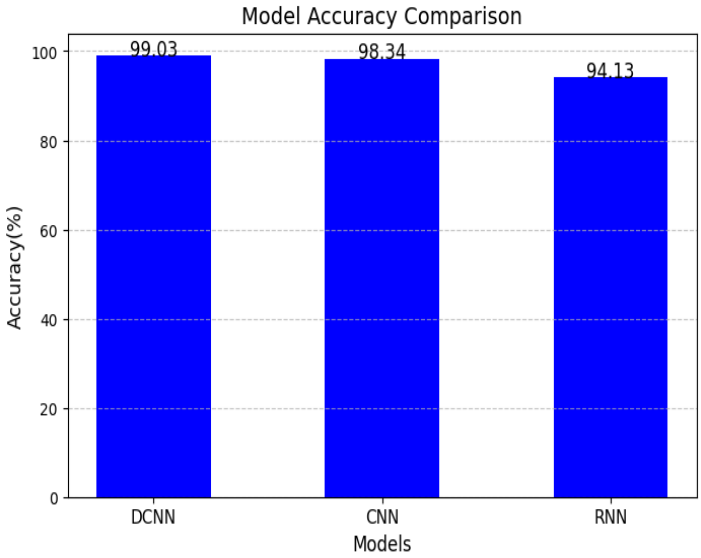

Chart -3: Model Accuracy

5.2 Precision

Precision, also referred to as Positive Predictive Value, determines the accuracy of predicted positive cases (threats).

It is crucial in Threat Detection Systems (TDS) to minimize false positives (FP) and avoid unnecessary security alerts.

Precision Formula

$$Precision = \frac{TP}{TP+FP} \quad \text{.....(7)}$$

Where:

TP (True Positives): Correctly predicted threats.

FP (False Positives): Incorrectly classified benign traffic as threats (false alarms).

5.2.1

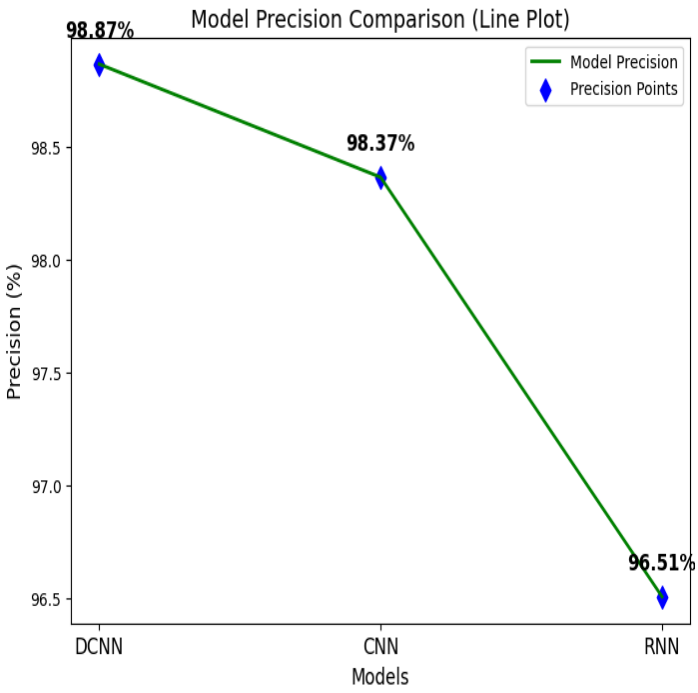

Chart-4: Model Precision

5.3 F1 Score:

The F1 score is a metric that assesses the balance between precision and recall in classification models. It is especially valuable for imbalanced datasets, where relying only on accuracy may not provide an accurate evaluation.,

calculated as:

$$F1 = \frac{Precision \times Recall}{Precision+Recall} \quad \text{.....(8)}$$

Where:

Precision (Positive Predictive Value)

Recall (Sensitivity)

TP = True Positives (correctly predicted threats)

FP = False Positives (incorrectly predicted threats)

5.3.1

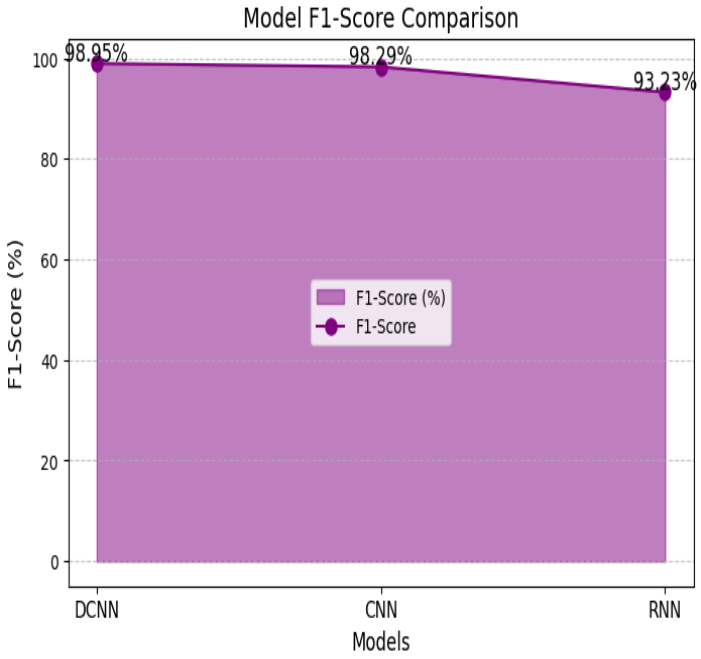

Chart- 5: Model F1 Score

5.4 ROC-AUC

The Receiver Operating Characteristic (ROC) curve and the Area Under the Curve (AUC) help measure how well a classification model distinguishes between different categories. They are especially useful for assessing performance in both binary and multi-class classification problems.

These rates are computed as:

$$TPR = \frac{TP}{TP+FN} \quad \text{.....(9)}$$

$$FPR = \frac{FP}{FP+FN} \quad \text{.....(10)}$$

Where:

TP = True Positives (correctly predicted positive cases)

FP = False Positives (incorrectly predicted positive cases)

TN = True Negatives (correctly predicted negative cases)

FN = False Negatives (incorrectly predicted negative case

The AUC (Area Under the ROC Curve)

It measures the ability of the model to distinguish between classes.

It is given by:

$$AUC = \int_0^1 TPR\, d(FPR) \quad \text{.....(11)}$$

This integral computes the total area under the ROC curve

5.4.1

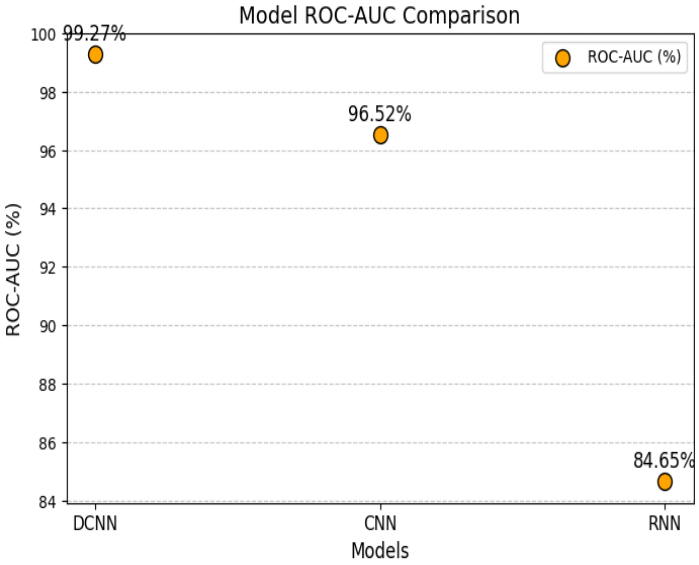

Chart -6: Model ROC-AUC

5.5 Recall:

Recall , or how well a system catches real threats, is key in cybersecurity. The higher the recall, the fewer threats slip through, keeping networks safer..

Recall Formula

$$Recall = \frac{TP}{TP+FN} \quad \text{.....(11)}$$

Where:

1. TP (True Positives): Correctly predicted positive cases (actual threats detected correctly).
2. FN (False Negatives): Missed positive cases (actual threats incorrectly classified as normal traffic).

5.5.1

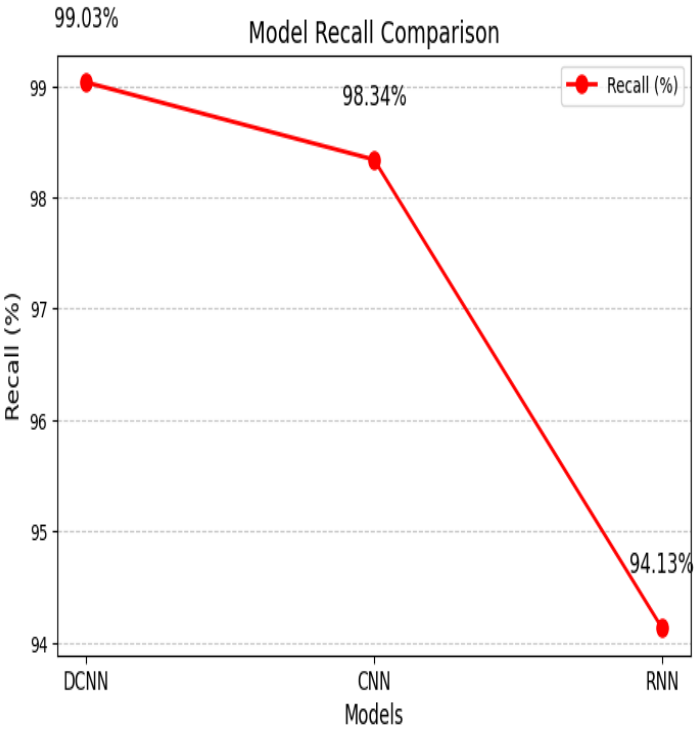

Chart -7: Model ROC-AUC

Analysis of DCNN Performance

High Accuracy**:** The DCNN model achieves **99.03%** accuracy, significantly outperforming CNN and RNN models.

Improved Recall: With a recall of **99.03%,** DCNN successfully identifies a higher number of actual threats.

Optimized Precision: The precision of **98.87%** ensures that false positives are minimized, making it a reliable classifier.

Superior F1-Score & ROC-AUC: The F1-score of **98.95%** and ROC-AUC of **99.27%** validate DCNN's ability to balance classification accuracy and threat detection efficiency.

Comparative Analysis of DCNN, CNN, and RNN

The performance evaluation of DCNN, CNN, and RNN reveals that DCNN significantly outperforms both traditional models across all key metrics. The following observations summarize the comparative analysis:

Accuracy Comparison

DCNN achieves the highest accuracy (99.03%), proving its superior ability to classify threats correctly.

CNN scores 98.34%, showing limitations in capturing complex patterns.

RNN reaches 94.13%, slightly improving on CNN but still lagging behind DCNN.

Precision & Recall Performance

1. DCNN excels in both precision (98.87%) and recall (99.03%), indicating a low false positive rate and high threat detection efficiency.
2. RNN's recall (94.13%) is lower, leading to missed threats.
3. CNN performs better than RNN but still falls short of DCNN

F1-Score & ROC-AUC Performance

1. DCNN achieves the highest F1-score (98.95%), balancing precision and recall optimally.
2. CNN and RNN have lower F1-scores, indicating weaker overall classification ability.
3. DCNN also secures the highest ROC-AUC score (99.27%), proving its ability to distinguish between normal and malicious traffic effectively.

6. Conclusion & Future Work

This research demonstrates that a DCNN-based Threat Detection System significantly improves the detection of cyber threats compared to CNN and RNN models. By utilizing dilated convolutions, DCNNs efficiently identify complex attack patterns, making them highly suitable for cybersecurity applications. Future work will explore real-time deployment and hybrid deep learning techniques for further enhancements.

Author contributions

The This research was a collaborative effort, with each author playing a crucial role in different aspects of the study:

- **Gunturu Maheswarareddy** led the research design and overall study framework, ensuring a structured and logical approach.
- **Andagula Trinadh** handled data collection and preprocessing, making sure the dataset was clean and ready for analysis.
- **Janga Vedeshreddy** focused on analyzing the results and interpreting the findings to draw meaningful insights.
- **Alaparthi Yaswanth** was responsible for drafting and refining the manuscript, ensuring clarity and coherence.
- **Mr. Y.V. Narayana** provided valuable guidance, supervised the research, and reviewed the final draft for accuracy and completeness.

REFERENCES:

[1] D. A. S. Resul and M. Z. Gündüz, "Analysis of cyber-attacks in infrastructures", IoT-based critical International Journal of Information Security Science, Vol. 8, No. 4, pp.122-133, 2020.
[2] T. T. Nguyen and V. J. Reddi, "Deep reinforcement learning for cyber security", IEEE Transactions on Neural Networks and Learning Systems. 2021.
[3] A. Samy, H. Yu, and H. Zhang, "Fog-based attack detection framework for Internet of things using deep learning", IEEE Access, Vol. 8, pp. 74571-74585, 2020.
[4] S. Chesney, K. Roy, and S. Khorsandroo, "Machine learning algorithms for preventing IoT cybersecurity attacks", In: Intelligent Systems and Applications: Proc. of the 2020 Intelligent Systems Conference (IntelliSys) Vol. 3, pp. 679-686, 2021.
[5] M. Kuzlu, C. Fair, and O. Guler, "Role of artificial intelligence in the Internet of Things (IoT) cybersecurity", Discover Internet of things, Vol. 1, pp. 1-14, 2021.
[6] N. Fu, N. Kamili, Y. Huang, and J. Shi, "A Novel Deep Intrusion Detection Model Based on a Convolutional Neural Network", Aust. J. Intell. Inf. Process. Syst., Vol. 15, No. 2, pp. 52-59, 2019.
[7] R. H. Hwang, M. C. Peng, C. W. Huang, P. C. Lin, and V. L. Nguyen, "An unsupervised deep learning model for early network traffic anomaly detection", IEEE Access, Vol. 8, pp. 30387-30399, 2020.
[8] W. Qiu, Q. Tang, Y. Wang, L. Zhan, Y. Liu, and W. Yao, "Multi-view convolutional neural network for data spoofing cyber-attack detection in distribution synchrophasor's", IEEE Transactions on Smart Grid, Vol. 11, No. 4, pp. 3457-3468, 2020.
[9] C. Yue, L. Wang, D. Wang, R. Duo, and H. Yan, "Detecting Temporal Attacks: An Intrusion Detection System for Train Communication Ethernet Based on Dynamic Temporal Convolutional Network", Security and Communication Networks, 2021, pp. 1-21, 2021.
[10] W. Liu, X. Xu, L. Wu, L. Qi, A. Jolfaei, W. Ding, and M. R. Khosravi, "Intrusion detection for maritime transportation systems with batch federated aggregation", IEEE Transactions on Intelligent Transportation Systems, 2022.
[11] M. Asam, S. H. Khan, A. Akbar, S. Bibi, T. Jamal, A. Khan, U. Ghafoor, and M. R. Bhutta, "IoT malware detection architecture using a novel channel boosted and squeezed CNN", Scientific Reports, Vol. 12, No. 1, pp. 1-12, 2022.

[12] Q. Liang, C. Liu, Y. Zhong, and X. Ren, “A Lightweight Flow-based DDoS Detection Approach using Dual Convolutional Kernels”, In: Proc. of 2022 China Automation Congress (CAC), pp. 2838-2843, 2022.

[13] S. H. Khan, T. J. Alahmadi, W. Ullah, J. Iqbal, A. Rahim, H. K. Alkahtani, W. Alghamdi, and A. O. Almagrabi, “A new deep boosted CNN and ensemble learning based IoT malware detection”, Computers & Security, Vol. 133, p. 103385, 2023.

[14]] Z. Liu, J. Yu, B. Yan, and G. Wang, “A Deep 1-D CNN and Bidirectional LSTM Ensemble Model with Arbitration Mechanism for DDoS Attack Detection”, IEEE Transactions on Emerging Topics in Computational Intelligence, Vol. 6, pp. 1396-1410, 2022.

[15] S. Rizvi, M. Scanlon, J. McGibney, and J. Sheppard, “Deep learning-based network intrusion detection system for resource-constrained environments”, In: Proc. of the 13th EAI International Conference on Digital Forensics and Cyber Crime, 2023.

[16] Y. V. Narayana and M. Sreedevi, "DDCATF: Deep Learning Approach for Detection of Cybercrime Activities Based on Temporal Features," 2023 International Conference on Self Sustainable Artificial Intelligence Systems (ICSSAS), Erode, India, 2023, pp. 462-469, doi: 10.1109/ICSSAS57918.2023.10331644.