# Enhanced Intrusion Detection Using Stacked FT-Transformer Architecture

**S. Phani Praveen [*1], Thulasi Bikku[2], P. Muthukumar[3], K. Sandeep[4], Jampani Chandra Sekhar[5], V. Krishna Pratap[6]**

[1]Department of CSE, PVP Siddhartha Institute of Technology, Kanuru, Vijayawada, A.P, India.
[2]Department of Computer Science & Engineering, Amrita School of Computing Amaravati, Amrita Vishwa Vidyapeetham, AP, India
[3]Professor,Department of Electrical and Electronics Engineering, Saveetha School of Engineering, Saveetha Institute of Medical and Technical Sciences, Tiruvallur, Chennai, Tamilnadu, India-602105
[4]Department of Information Technology, Dhanekula Institute of Engineering & Technology, Vijayawada 521139,A.P, India
[5]Professor, Department of CSE, NRI Institute of Technology, Visadala, Guntur, Andhra Pradesh, India
[6]Assistant Professor, Department of CSE, NRI Institute of Technology, Visadala, Guntur, Andhra Pradesh, India

Emails: spppraveen@pvpsiddhartha.ac.in;thulasi.bikku@gmail.com; muthukumarvlsi@gmail.com; kottesandeep@gmail.com; jcsekhar9@gmail.com; pratapv9@gmail.com
* Corresponding Author: spppraveen@pvpsiddhartha.ac.in

**Abstract**

The function of network intrusion detection systems (NIDS) in protecting networks from cyberattacks is crucial. Many of the more conventional techniques rely on signature-based approaches, which have a hard time distinguishing between various types of assaults. Using stacked FT-Transformer architecture, this research suggests a new way to identify intrusions in networks. When it comes to dealing with complicated tabular data, FT-Transformers—a variant of the Transformer model—have shown outstanding performance. Because of the inherent tabular nature of network traffic data, FT-Transformers are an attractive option for intrusion detection jobs. In this area, our study looks at how FT-Transformers outperform more conventional machine learning (ML) methods. Our working hypothesis is that, in comparison to single-layered ML models, FT-Transformers will achieve better detection accuracy due to their intrinsic capacity to grasp long-range correlations in network traffic data. We also test the FT-Transformer model on several network traffic datasets that include various protocols and attack kinds to see how well it performs and how generalizable it is. The purpose of this research is to shed light on how well and how versatile FT-Transformers perform for detecting intrusions in networks. We aim to prove that FT-Transformers can secure networks from ever-changing cyber threats by comparing their performance to that of classic ML models and by testing their generalizability.

**Keywords :** Intrusion detection Ft Transformer; Stacking; cybersecurity; machine learning.

## 1. Introduction:

To safeguard vital infrastructure and private data, stringent security measures are required due to the increasing dependence on interconnected computer networks. In order to prevent unwanted access and harmful actions, networks rely on network intrusion detection[1,2] systems (NIDS). In order to detect intrusion attempts, these systems constantly scan network data, analyse its properties, and flag patterns that don't follow the standard.
Intrusion detection strategies[3,4] have always depended on methods that use signatures. These techniques compare patterns of network traffic with databases that contain known attack signatures. Although signature-based techniques are useful for detecting known assaults, they do have their limits. Networks are left open to zero-day vulnerabilities because they are unable to identify new assaults or variations of old ones.
There have been some encouraging developments in the field of intrusion detection using ML and DL models in the last several years. These models are able to detect both known and undiscovered assaults by learning



Doi : https://doi.org/10.54216/JCIM.130202
Received: July 24, 2023 Revised: November 17, 2023 Accepted: February 09, 2024

intricate patterns from labelled data on network traffic[5-8]. The current state of intrusion detection ML and DL models is not without its problems, though:

- Feature Engineering: An essential part of building ML and DL models is feature engineering, which entails picking and preparing pertinent features from data on network traffic[9-12]. The relevancy and quality of the engineered characteristics determine how well these models work. In addition to being domain-specific and sometimes time-consuming, manual feature engineering runs the risk of missing important data linkages.
- Limited Learning Capacity: Because they only have one layer of learning, traditional ML models may miss detections or false alarms because they can't grasp the complex correlations between different characteristics of network data.
- Evolving Attack Landscape: Because cybercriminals are always coming up with new ways to avoid detection, the nature of network attacks is also changing. Security personnel may have more work on their hands if traditional ML models need to be retrained with fresh data in order to accommodate these changes.

Using stacked FT-Transformer architecture, this research suggests a new way to identify intrusions in networks. Notably effective in dealing with complicated tabular data are FT-Transformers, which are a variant of the Transformer model. Because of the inherent tabular nature of network traffic data, FT-Transformers are an attractive option for intrusion detection jobs.

We present a method that takes advantage of FT-Transformer capabilities to discover complex correlations among different types of network traffic. The model may be able to better detect harmful network behaviour by stacking numerous Transformer layers, which capture long-range dependencies within the data.

Using stacked FT-Transformer architecture, this research suggests a new way to identify intrusions in networks. In order to overcome these drawbacks, FT-Transformers do the following:

- Automated Feature Learning: FT-Transformers can learn informative representations from raw network traffic data automatically, avoiding the need for manual feature engineering. This is in contrast to standard ML.
- Enhanced Learning Capacity: The model can better detect known and undiscovered attacks by stacking several Transformer layers, which capture complex relationships and long-range dependencies within the data.
- Adaptability: Because of their built-in learning capabilities, FT-Transformers can adapt to new attack vectors by constantly learning from data collected from network traffic.

The Objectives of this paper include

- A novel approach for detecting intrusions in networks using FT-Transformers is presented in this paper.
- Examine FT-Transformers in Perspective of their benefits over more conventional ML methods.
- In order to assess the FT-Transformer model's generalizability, its efficacy should be evaluated across a variety of network traffic datasets.

This paper is structured as follows.. In Section 2, FT-Transformers are discussed in detail, including the model architecture suitable for detecting intrusions in networks. Both the experimental results and the methods used for evaluation are detailed in Section 3. Section 4 concludes the research and examines its future directions.

## 2. Methodology:

The proposed methodology is shown in figure 1.

1. Data Pre-processing: Data preparation and collection is the focus of this phase of developing the machine learning model. Cleaning the data, dealing with missing values, and translating it into a model-friendly format are all possible steps in this process. Gathering data on network traffic and transforming it into a numerical representation that the FT-Transformer model can comprehend might be part of this stage in the context of network intrusion detection.
2. Data Analysis: At this stage, we will investigate and comprehend the facts. We can find trends, patterns, and correlations in the data by using exploratory data analysis (EDA) approaches. This might be useful when deciding which machine learning model and features to be used.
3. Model Training:. Training machine learning and deep learning models on some subset of the pre-processed data is the objective of this stage. In order to discover the hidden connections between the characteristics and the dependent variable, the parameters of the model are fitted using the training data. To implement network intrusion detection, one would train the model to recognise suspicious patterns in data collected from network traffic.
4. Performance Evaluation: Next, the model is tested on a different dataset to see how well it performed after training. An objective evaluation of the model's generalizability can be achieved by using this test set, which is not utilised during training. Measuring the model's accuracy in detecting intrusions on unseen network traffic data is an important part of performing network intrusion detection.



Doi : https://doi.org/10.54216/JCIM.130202
Received: July 24, 2023 Revised: November 17, 2023 Accepted: February 09, 2024

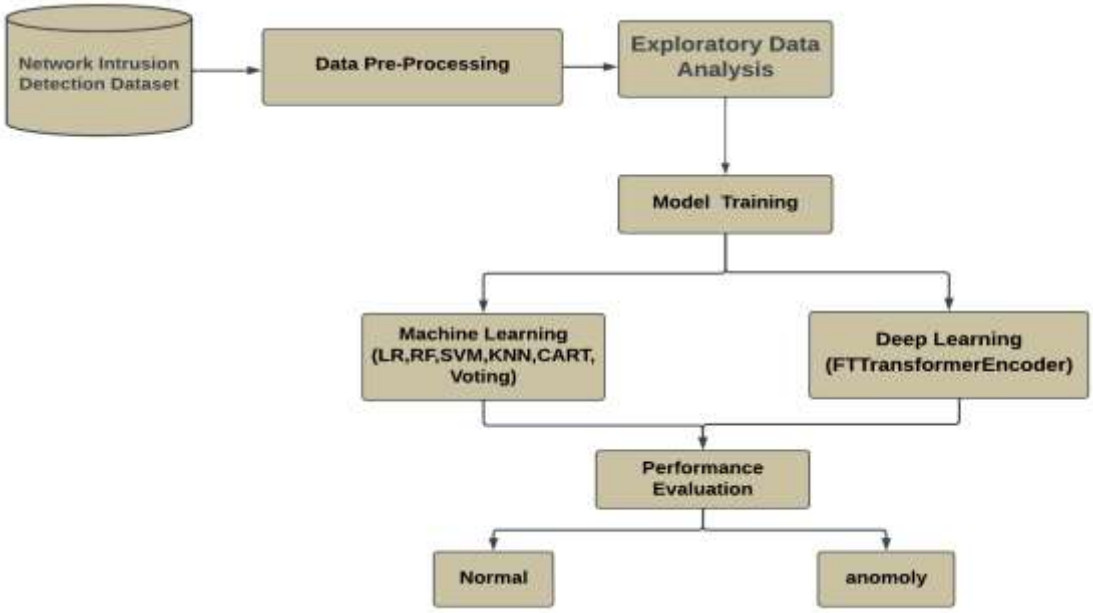



Figure 1: Methodological overview

*2.1 Data Insights*

The information was extracted from a dataset called "network-intrusion-detection," which was accessed through the following link: https://www.kaggle.com/datasets/sampadab17/network-intrusion-detection. Figure 2 displays the dataset's features, whereas Figure 3 depicts the classifications.

| # | Column | Non-Null Count | Dtype |
|---|---|---|---|
| 0 | duration | 25192 non-null | int64 |
| 1 | protocol_type | 25192 non-null | object |
| 2 | service | 25192 non-null | object |
| 3 | flag | 25192 non-null | object |
| 4 | src_bytes | 25192 non-null | int64 |
| 5 | dst_bytes | 25192 non-null | int64 |
| 6 | land | 25192 non-null | int64 |
| 7 | wrong_fragment | 25192 non-null | int64 |
| 8 | urgent | 25192 non-null | int64 |
| 9 | hot | 25192 non-null | int64 |
| 10 | num_failed_logins | 25192 non-null | int64 |
| 11 | logged_in | 25192 non-null | int64 |
| 12 | num_compromised | 25192 non-null | int64 |
| 13 | root_shell | 25192 non-null | int64 |
| 14 | su_attempted | 25192 non-null | int64 |
| 15 | num_root | 25192 non-null | int64 |
| 16 | num_file_creations | 25192 non-null | int64 |
| 17 | num_shells | 25192 non-null | int64 |
| 18 | num_access_files | 25192 non-null | int64 |
| 19 | num_outbound_cmds | 25192 non-null | int64 |
| 20 | is_host_login | 25192 non-null | int64 |
| 21 | is_guest_login | 25192 non-null | int64 |
| 22 | count | 25192 non-null | int64 |
| 23 | srv_count | 25192 non-null | int64 |
| 24 | serror_rate | 25192 non-null | float64 |
| 25 | srv_serror_rate | 25192 non-null | float64 |
| 26 | rerror_rate | 25192 non-null | float64 |
| 27 | srv_rerror_rate | 25192 non-null | float64 |
| 28 | same_srv_rate | 25192 non-null | float64 |
| 29 | diff_srv_rate | 25192 non-null | float64 |
| 30 | srv_diff_host_rate | 25192 non-null | float64 |
| 31 | dst_host_count | 25192 non-null | int64 |
| 32 | dst_host_srv_count | 25192 non-null | int64 |
| 33 | dst_host_same_srv_rate | 25192 non-null | float64 |
| 34 | dst_host_diff_srv_rate | 25192 non-null | float64 |
| 35 | dst_host_same_src_port_rate | 25192 non-null | float64 |
| 36 | dst_host_srv_diff_host_rate | 25192 non-null | float64 |
| 37 | dst_host_serror_rate | 25192 non-null | float64 |
| 38 | dst_host_srv_serror_rate | 25192 non-null | float64 |
| 39 | dst_host_rerror_rate | 25192 non-null | float64 |
| 40 | dst_host_srv_rerror_rate | 25192 non-null | float64 |
| 41 | class | 25192 non-null | int64 |

Figure 2: Features in Network Intrusion Dataset



Doi : https://doi.org/10.54216/JCIM.130202
Received: July 24, 2023 Revised: November 17, 2023 Accepted: February 09, 2024

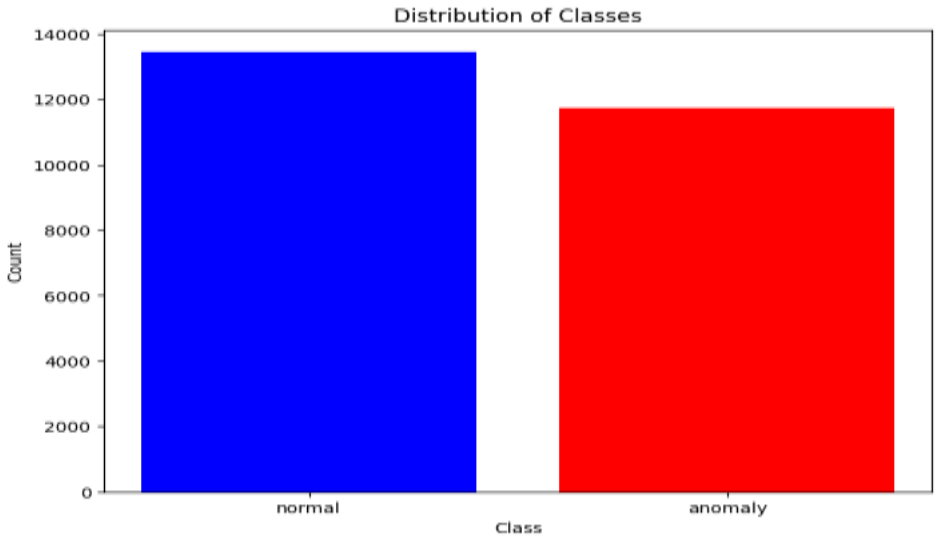



Figure 3: Classes in Network Intrusion Dataset

*2.2. FT-Transformer Approach*

A stacked FT-Transformer is a variation of the standard FT-Transformer[13,14,15] model that utilizes multiple Transformer layers stacked on top of each other and is shown in figure 4 and 5.

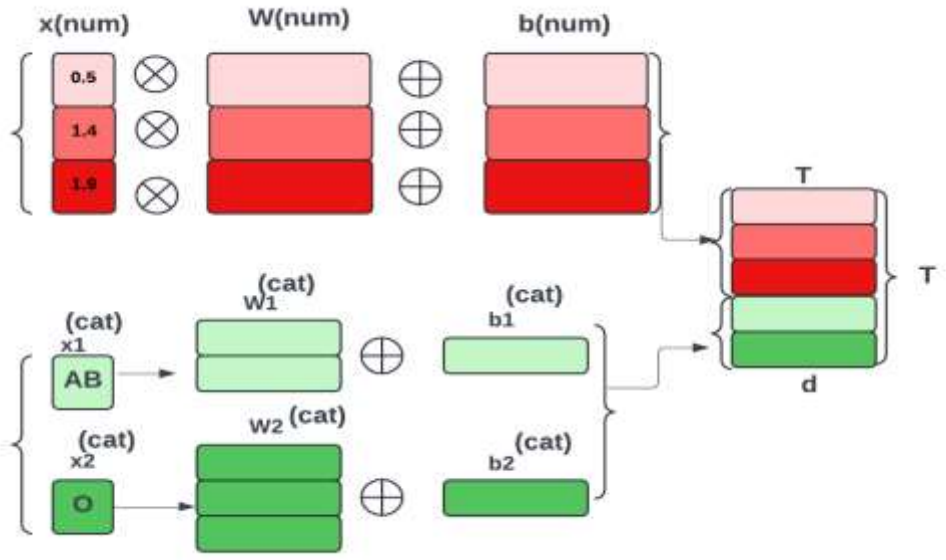



Figure 4:Feature Tokenizer (FT-Transformer Architecture with 3 numerical and 2 categorical features)



Doi : https://doi.org/10.54216/JCIM.130202
Received: July 24, 2023 Revised: November 17, 2023 Accepted: February 09, 2024

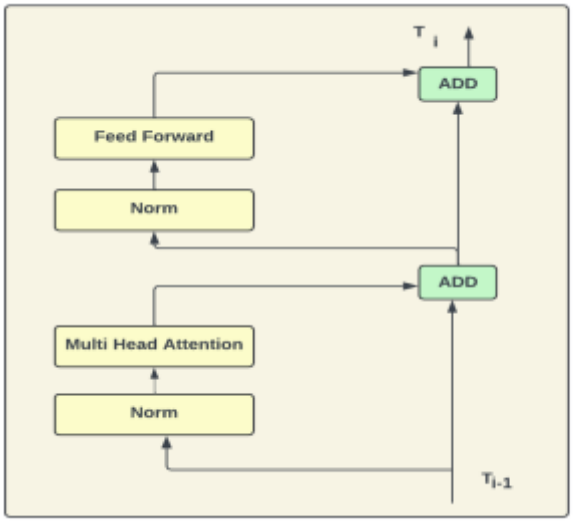



Figure 5: Stacked Transformer Layer (One Layer)

- FT-Transformer (Feature Tokenizer + Transformer): It is a model developed for deep learning that is specifically tailored to analyze tabular data. To make it function with structured data, it modifies the Transformer architecture, which is often used for NLP. At its heart, the concept is to tokenize numerical and categorical characteristics such that they can be processed by Transformer layers.
- Stacked Transformer Layers: The tokenized features are processed by a single stack of Transformer layers in a conventional FT-Transformer. Multiple stacked Transformer layers are utilized in a stacked FT-Transformer. By focusing on the interrelationships of features, each successive layer enhances the comprehension of the data.
- The purpose of stacking these layers is to:

Increase Model Capacity: The model's ability to understand intricate correlations between data elements increases as the number of layers increases. This has the ability to enhance performance on tasks such as classification or prediction.

Capture Long-Range Dependencies: By stacking layers, the model is able to capture long-range dependencies between features, which would be hidden in a single layer. When dealing with complicated datasets, this can be really useful to detect intrusions[16,17,18,19].

When working with complicated tabular data, stacked FT-Transformers are a powerful tool, but before we use them, we need evaluate our dataset and available computing resources to determine the optimal trade-off between model complexity and performance.

*2.3 Model Building using ML*

*2.3.1 Support Vector Machine*

For regression and classification jobs, supervised machine learning algorithms like Support Vector Machine (SVM) come in handy. In a space with n dimensions (where n is the number of characteristics), it locates a hyperplane that clearly divides data points into discrete classes. SVM's primary objective is to maximise the distance between the hyperplane and the closest data points, which are called support vectors. Because it may use many kernel functions for complicated decision boundaries, it is both flexible and effective in high-dimensional spaces. When dealing with datasets of small to medium size, SVM shines. Equation 1 represents the decision function of svm.

$$D(z) = sign\left(\sum_{j=1}^{n} \propto_j y_j K(z_j, z) + b_j\right) \quad (1)$$

Where D(z) is Decision function
z-input feature vector
$\propto_j$ -Lagrange multipliers
$y_j$ -class labels
$K(z_j, z)$-Kernel function
$b_j$- Bias



Doi : https://doi.org/10.54216/JCIM.130202
Received: July 24, 2023 Revised: November 17, 2023 Accepted: February 09, 2024

*2.3.2 Logistic Regression*

Binomial classification tasks are handled by supervised learning algorithms like logistic regression. The decision function in Logistic Regression is based on the logistic function (sigmoid function), which maps the input features to a value between 0 and 1, representing the probability of the input belonging to the positive class. The decision function of Logistic Regression is as follows:

$$p(y = \frac{1}{x}) = \frac{1}{1 + e^{-z}} \quad (2)$$

Where p(y=1/x) -Probability of input x.
z-input features and their weights.

*2.3.3 KNN Classifier*

In both classification and regression, K-Nearest Neighbors (KNN) is a straightforward but powerful supervised learning algorithm. KNN initially stores all available data points during its training phase, along with their corresponding class labels (for classification) or target values (for regression). KNN calculates the distances between new data points presented for prediction and all other points in the dataset, using metrics like Euclidean distance.

$$d(t_{k1}, t_{k2}) = \sqrt{\sum_{k=1}^{k=n} (t_{k1} - t_{k2})^2} \quad (3)$$

*2.3.4 CART Model*

Based on the thresholds of the most discriminative features, the CART (Classification and Regression Trees) model recursively partitions the feature space based on features. For classification tasks or regression tasks, the algorithm optimizes criterion like Gini impurity during the training phase. The process is repeated until a specific stopping criteria is met, like a maximum sample depth or a minimum sample size per leaf. At the leaf nodes of a decision tree, labels are assigned (for classification) or predicted values are predicted (for regression). The algorithm classifies or predicts new data points by traversing the decision tree from the root node to the leaf node. A CART model based on decision trees, however, is prone to overfitting, especially with deep trees, and requires techniques like pruning and hyperparameter tuning to optimize performance. However, they are useful tools for various machine learning tasks due to their interpretability and ability to capture complex relationships.*3.3.5 2.3.5 Random Forest*

It is a method of ensemble learning that uses multiple decision trees to construct multiple decision trees when training is performed. In order to create more diversity in the trees, subsets of features and data samples are randomly selected. It is possible to combine the predictions of individual trees (for classification) or to average them (for regression). It's robust against overfitting and performs well on a variety of datasets. Random Forest is widely used due to its simplicity, scalability, and high accuracy.

*2.3.6 Voting Classifier*

Voting Classifiers combine multiple base classifiers in an ensemble learning method to make predictions. Depending on the dataset or subsets of the dataset, each base classifier can be trained on a different type of data. Base classifiers make their predictions independently and then aggregate them to make a final prediction. Voting can be either hard or soft; hard voting involves selecting the class that has the highest average probability; soft voting involves selecting the class that is the majority. By leveraging the strengths of the different models and reducing overfitting risks, the Voting Classifier often outperforms any single classifier in the ensemble. In practice, it is used often for classification tasks involving a variety of models.

**3. Empirical findings and performance evaluation:**

*3.1. Configuration settings*

An intrusion detection system (IDS) is shaped by its hyperparameters, which are crucial to determining its performance and behaviour. These parameters encompass various facets of the training process, including algorithm selection, hyperparameter tuning, feature engineering, sampling techniques, regularization, optimization algorithms, and evaluation metrics. To determine the best algorithm or model architecture to use, it is important to take into account the characteristics of the dataset. It involves using techniques such as grid search or random search to optimize hyperparameters, like learning rate or regularization parameters. The engineering hyperparameters and feature selection play a crucial role in learning relevant features from the data. In order to control the dataset's composition, sampling techniques such as oversampling and undersampling are employed. Overfitting is prevented and generalization ability is improved with regularization hyperparameters. Furthermore, training speed and convergence quality are determined by optimization algorithms' hyperparameters, while model performance is assessed and validated by evaluation metrics' hyperparameters. In



Doi : https://doi.org/10.54216/JCIM.130202
Received: July 24, 2023 Revised: November 17, 2023 Accepted: February 09, 2024

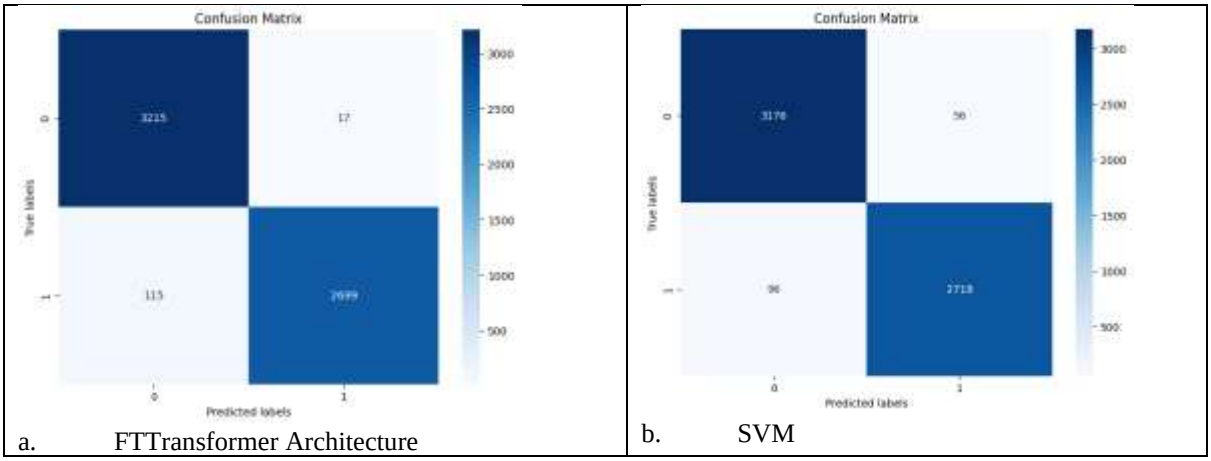
identifying intrusion attempts and classifying them accurately, IDS models can be trained with careful configuration of these hyperparameters. Table 1 shows the hyperparameters used in configuring and training the FTTransformer model for intrusion detection.

Table 1: Hyperparameters of Stacked Transformer Model

| **Hyperparameter** | **Value** |
|---|---|
| Numerical Embedding Type | Linear |
| Embedding Dimension | 32 |
| Depth | 3 |
| Number of Attention Heads | 6 |
| Attention Dropout Rate | 0.3 |
| Feedforward Dropout Rate | 0.3 |
| Output Dimension | 1 |
| Output Activation Function | Sigmoid |
| Epochs | 2 |
| Learning Rate | 0.001 |
| Weight Decay | 0.0001 |
| Optimizer | AdamW |
| Loss Function | BinaryCrossentropy |
| Evaluation Metric | Binary Accuracy |
| Early Stopping Patience | 10 |

*3.2 Evaluation of different classification methods*

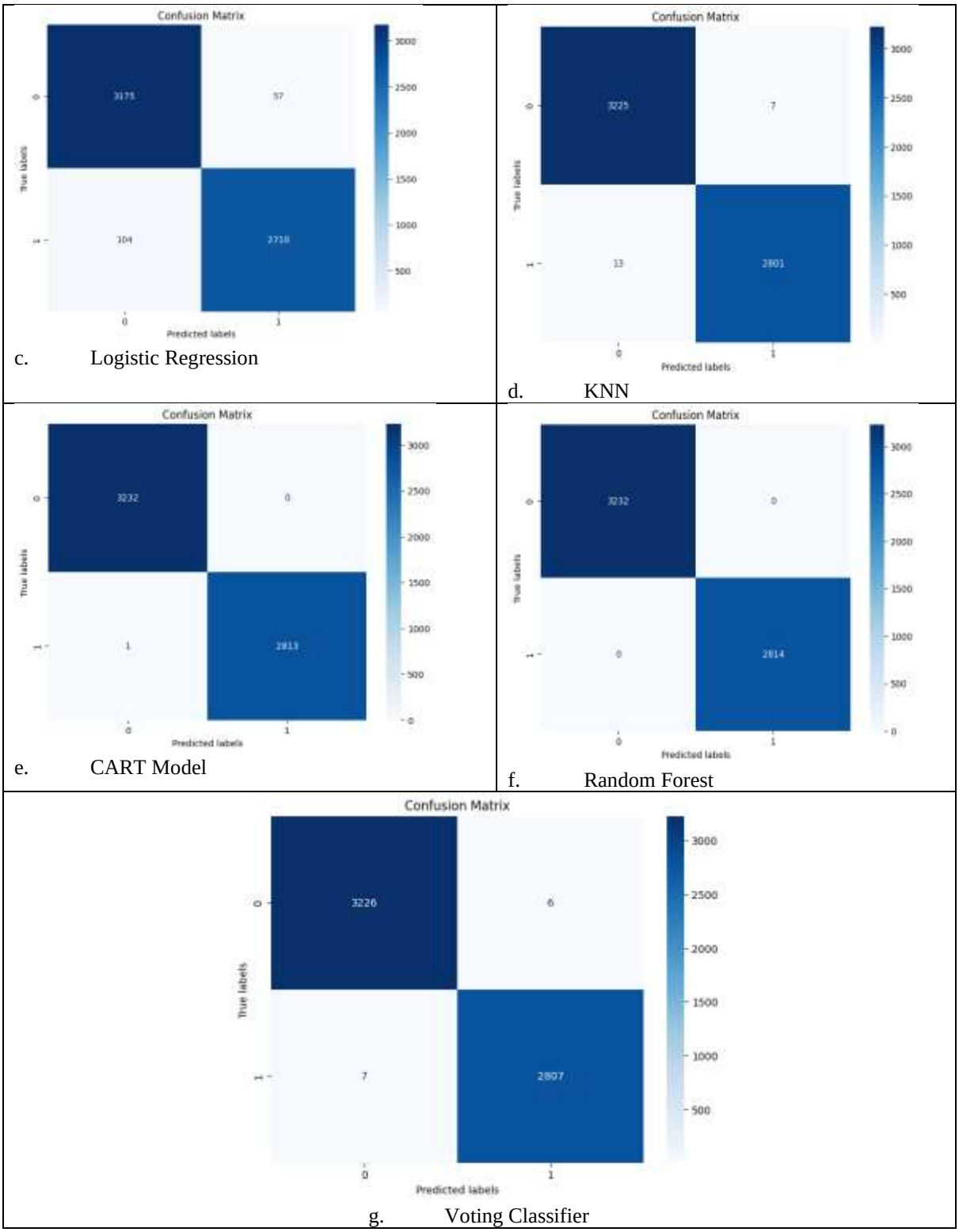
To assess the effectiveness of model configurations and models, various assessment metrics were used. An accuracy, precision, recall, as well as a F1 score were measured. Using this approach, the various intrusion detection techniques could be evaluated thoroughly, enabling comparisons between them. In addition, we assessed the impact of features on classification outcomes by examining their relevance. In our study, traditional machine learning models are evaluated holistically against FT-transformer architectures to illuminate their suitability to intrusion detection. This matrix (Figure 6) exposes the model's predictions against actual outcomes, indicating how the machine learning model performs when categorizing different anomalies.




a. FTTransformer Architecture b. SVM



Doi : https://doi.org/10.54216/JCIM.130202
Received: July 24, 2023 Revised: November 17, 2023 Accepted: February 09, 2024




c. Logistic Regression

d. KNN

e. CART Model

f. Random Forest

g. Voting Classifier

Figure 6: Confusion matrices with regard to Various architectures.

Table 2: Performance of Various models with regard to different metrics

| Architecture | classes | Precision | Recall | F1-score | Accuracy |
|---|---|---|---|---|---|
| FT Transformer | normal | 0.98 | 0.99 | 0.97 | 0.999 |
| | anamoly | 0.99 | 0.95 | 0.97 | |
| SVM | normal | 0.97 | 0.98 | 0.98 | 0.97 |
| | anamoly | 0.98 | 0.97 | 0.97 | |



Doi : https://doi.org/10.54216/JCIM.130202
Received: July 24, 2023 Revised: November 17, 2023 Accepted: February 09, 2024

| | | | | | |
|---|---|---|---|---|---|
| Logistic Regression | normal | 0.97 | 0.98 | 0.98 | 0.97 |
| | anamoly | 0.98 | 0.96 | 0.97 | |
| KNN | normal | 0.97 | 0.98 | 0.97 | 0.991 |
| | anamoly | 0.98 | 0.97 | 0.98 | |
| CART Model | normal | 0.96 | 0.98 | 0.97 | 0.994 |
| | anamoly | 0.98 | 0.98 | 0.97 | |
| Random Forest | normal | 0.96 | 0.97 | 0.97 | 0.993 |
| | anamoly | 0.98 | 0.97 | 0.97 | |
| Voting Classifier | normal | 0.97 | 0.99 | 0.98 | 0.997 |
| | anamoly | 0.98 | 0.97 | 0.98 | |

For network intrusion detection, FT-Transformer appears to be a promising approach as shown in table 2. As a result, the system is highly accurate and offers both anomalous and normal traffic detection capabilities. Due to its highly accurate detection capabilities, the system can detect both anomalous and normal traffic. Here is a line graph showing the evaluation metrics of the various architectures from Figures 7-10.

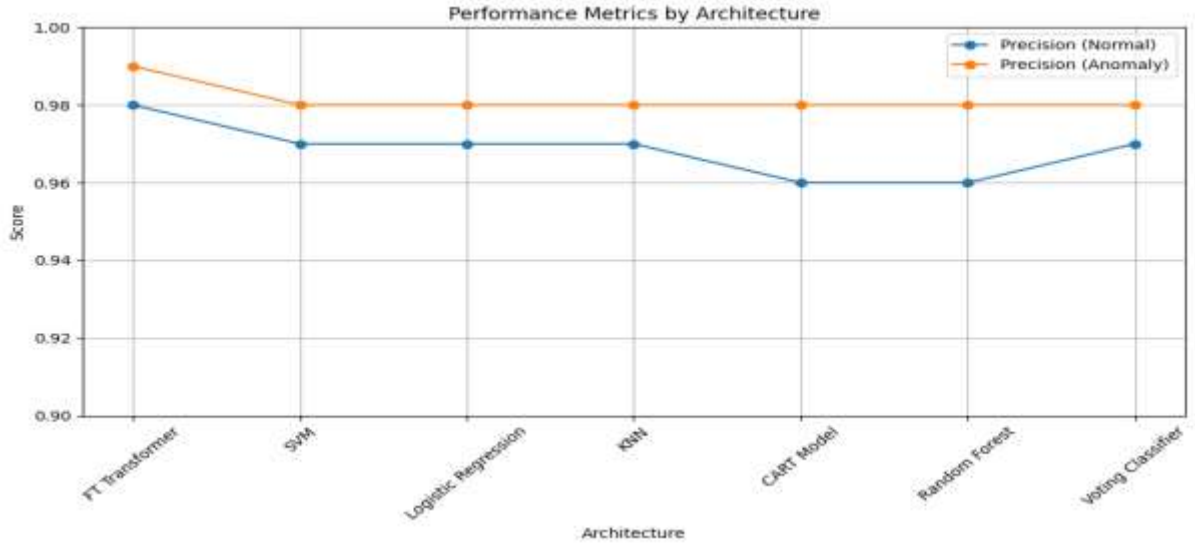



Figure 7: A line graph depicting the Precision of various architectures.

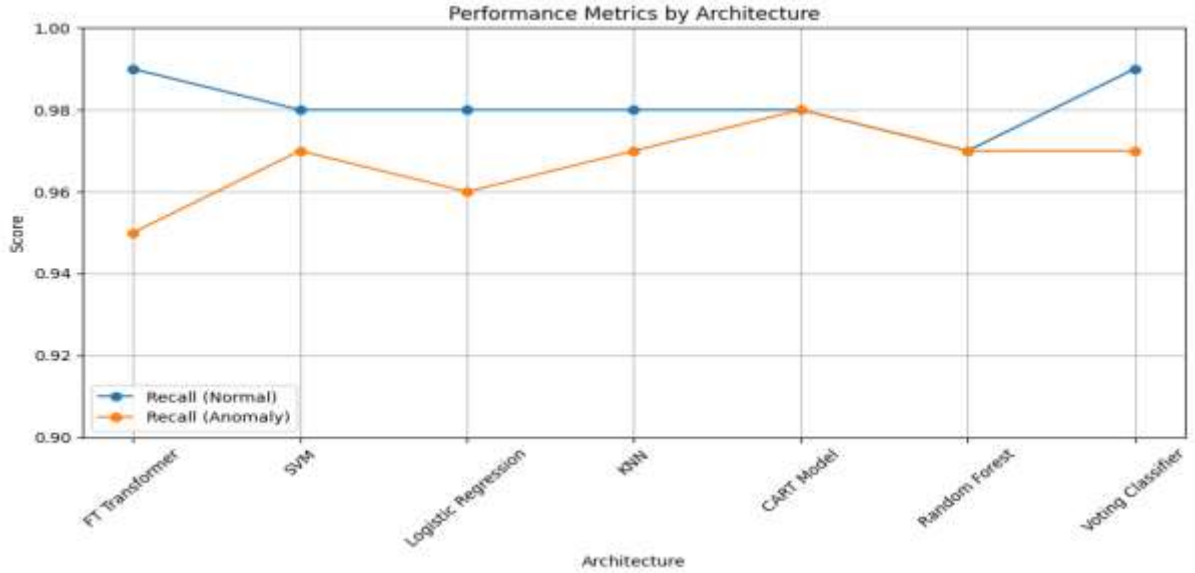



Figure 8: A line graph depicting the Recall of various architectures.



Doi : https://doi.org/10.54216/JCIM.130202
Received: July 24, 2023 Revised: November 17, 2023 Accepted: February 09, 2024

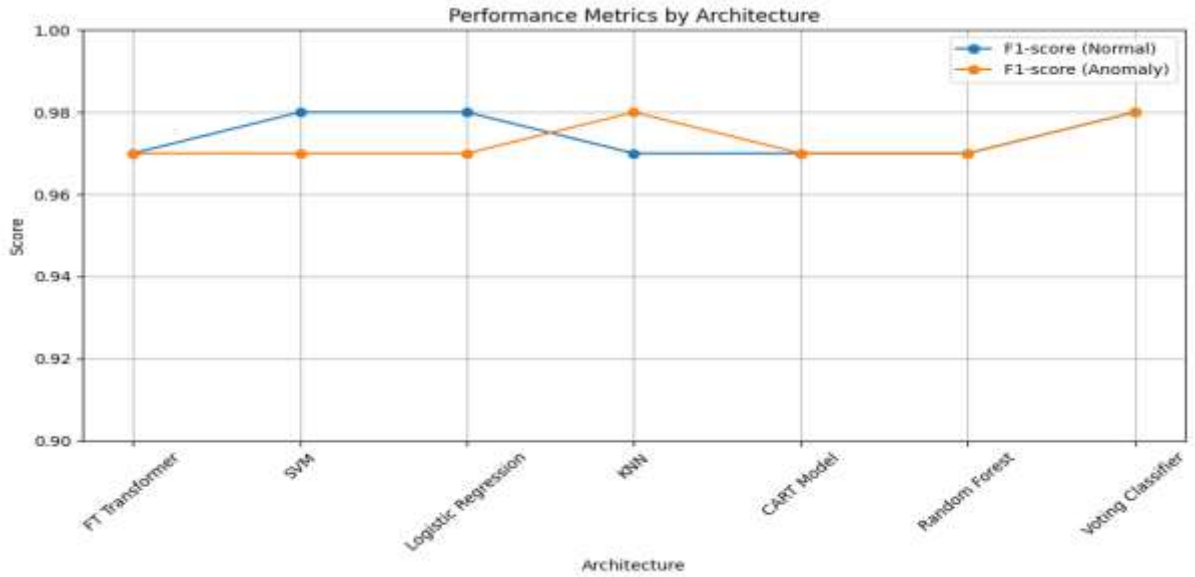



Figure 9: A line graph depicting the F1-score of various architectures.

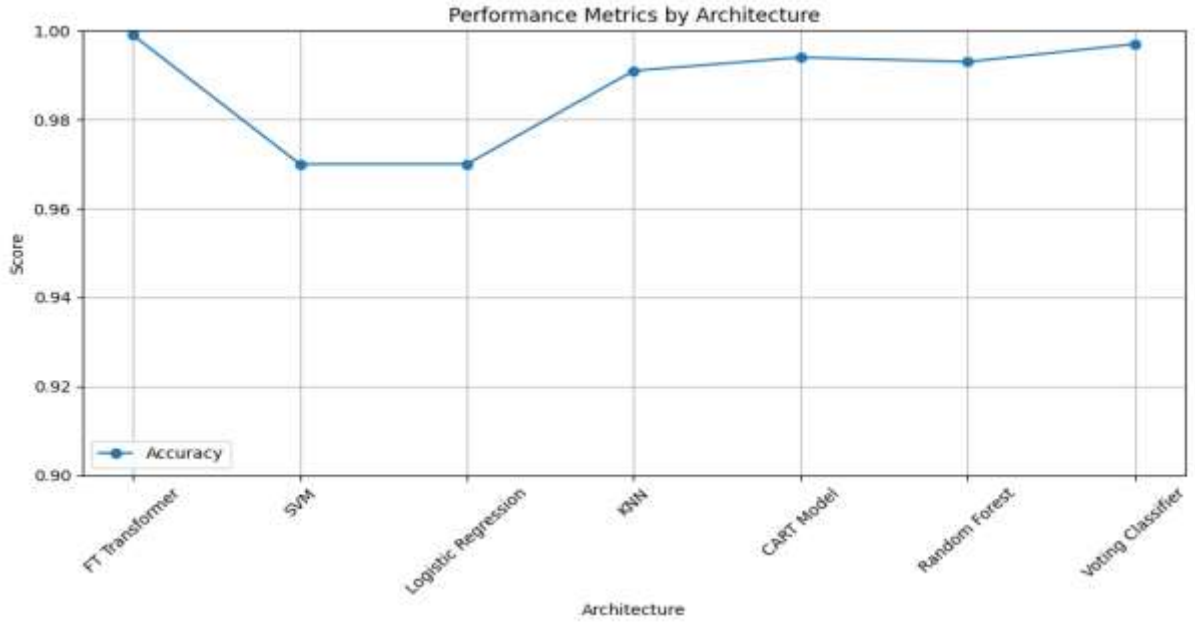



Figure 10 : A line graph depicting the Accuracy of various architectures.

The table 3 summarizes the performance of a Stacked FT-Transformer model on different network traffic datasets.

Table 3: FT-Transformer performance on different datasets

| **Dataset Name** | **Stacked FT-Transformer Approach (accuracy)** |
|---|---|
| CICIDS2017 [ 20] | 0.97 |
| CICIDS2018[ 21] | 0.96 |
| CICDDoS2019 [22 ] | 0.94 |

*4.Conclusion*

This article explored the potential of stacked FT-Transformers to identify network breaches. We compared these models to more conventional machine learning algorithms to see how well they generalised across different datasets. Not only did FT-Transformers have great overall accuracy when classifying network traffic, but they were also able to capture data with long-range relationships. This might be because of their exceptional ability to identify anomalies in the data. Testing the model on several datasets confirmed its broad applicability, suggesting its potential application for intrusion detection in the real world. An attractive aspect of FT-Transformers is their capacity to automatically learn features, record complicated correlations in network data,



Doi : https://doi.org/10.54216/JCIM.130202
Received: July 24, 2023 Revised: November 17, 2023 Accepted: February 09, 2024

and react against new threats. Research into the explanation of models and the integration of real-time network monitoring can further reinforce FT-Transformer's practical applicability for network security.

**References**

[1] Asharf, J., Moustafa, N., Khurshid, H., Debie, E., Haider, W., & Wahab, A. (2020). A review of intrusion detection systems using machine and deep learning in internet of things: Challenges, solutions and future directions. Electronics, 9(7), 1177.

[2] Qazi, E. U. H., Faheem, M. H., & Zia, T. (2023). HDLNIDS: hybrid deep-learning-based network intrusion detection system. Applied Sciences, 13(8), 4921.

[3] Ahmad, Z., Shahid Khan, A., Wai Shiang, C., Abdullah, J., & Ahmad, F. (2021). Network intrusion detection system: A systematic study of machine learning and deep learning approaches. Transactions on Emerging Telecommunications Technologies, 32(1), e4150.

[4] He, K., Kim, D. D., & Asghar, M. R. (2023). Adversarial machine learning for network intrusion detection systems: A comprehensive survey. IEEE Communications Surveys & Tutorials, 25(1), 538-566.

[5] Abdulganiyu, O. H., Ait Tchakoucht, T., & Saheed, Y. K. (2023). A systematic literature review for network intrusion detection system (IDS). International Journal of Information Security, 22(5), 1125-1162.

[6] Sirisha, U., Chandana, B. S., & Harikiran, J. (2023). NAM-YOLOV7: An Improved YOLOv7 Based on Attention Model for Animal Death Detection. Traitement du Signal, 40(2).

[7] Khafaga, D. S., Karim, F. K., Abdelhamid, A. A., El-kenawy, E. S. M., Alkahtani, H. K., Khodadadi, N., ... & Ibrahim, A. (2023). Voting Classifier and Metaheuristic Optimization for Network Intrusion Detection. Computers, Materials & Continua, 74(2).

[8] Alkanhel, R., El-kenawy, E. S. M., Abdelhamid, A. A., Ibrahim, A., Alohali, M. A., Abotaleb, M., & Khafaga, D. S. (2023). Network Intrusion Detection Based on Feature Selection and Hybrid Metaheuristic Optimization. *Computers, Materials & Continua*, *74*(2).

[9] Sarhan, M., Layeghy, S., Moustafa, N., & Portmann, M. (2021). Netflow datasets for machine learning-based network intrusion detection systems. In *Big Data Technologies and Applications: 10th EAI International Conference, BDTA 2020, and 13th EAI International Conference on Wireless Internet, WiCON 2020, Virtual Event, December 11, 2020, Proceedings 10* (pp. 117-135). Springer International Publishing.

[10] Gamage, S., & Samarabandu, J. (2020). Deep learning methods in network intrusion detection: A survey and an objective comparison. *Journal of Network and Computer Applications*, *169*, 102767.

[11] Saheed, Y. K., Abiodun, A. I., Misra, S., Holone, M. K., & Colomo-Palacios, R. (2022). A machine learning-based intrusion detection for detecting internet of things network attacks. *Alexandria Engineering Journal*, *61*(12), 9395-9409.

[12] Kilincer, I. F., Ertam, F., & Sengur, A. (2021). Machine learning methods for cyber security intrusion detection: Datasets and comparative study. *Computer Networks*, *188*, 10784.

[13] Zhu, B., Shi, X., Erickson, N., Li, M., Karypis, G., & Shoaran, M. (2023). Xtab: Cross-table pretraining for tabular transformers. *arXiv preprint arXiv:2305.06090*.

[14] Sluis, E. (2023). Combining the FT-Transformer with the LSTM model to predict stock prices.

[15] Biyyapu, N., Veerapaneni, E. J., Surapaneni, P. P., Vellela, S. S., & Vatambeti, R. (2024). Designing a modified feature aggregation model with hybrid sampling techniques for network intrusion detection. Cluster Computing, 1-19.

[16] Aruna, R., Kushwah, V. S., Praveen, S. P., Pradhan, R., Chinchawade, A. J., Asaad, R. R., & Kumar, R. L. (2024). Coalescing novel QoS routing with fault tolerance for improving QoS parameters in wireless Ad-Hoc network using craft protocol. Wireless Networks, 30(2), 711-735.

[17] Phani Praveen, S., Ali, M. H., Jarwar, M. A., Prakash, C., Reddy, C. R. K., Malliga, L., & Chandru Vignesh, C. (2023). 6G assisted federated learning for continuous monitoring in wireless sensor network using game theory. Wireless Networks, 1-27.

[18] Swapna, D., & Praveen, S. P. (2020). An exploration of distributed access control mechanism using blockchain. In Smart Intelligent Computing and Applications: Proceedings of the Third International Conference on Smart Computing and Informatics, Volume 2 (pp. 13-20). Springer Singapore.

[19] Jyothi, V. E., Kumar, D. L. S., Thati, B., Tondepu, Y., Pratap, V. K., & Praveen, S. P. (2022, December). Secure data access management for cyber threats using artificial intelligence. In 2022 6th International Conference on Electronics, Communication and Aerospace Technology (pp. 693-697). IEEE.

[20] https://www.unb.ca/cic/datasets/ids-2017.html

[21] https://www.unb.ca/cic/datasets/ids-2018.html

[22] https://www.unb.ca/cic/datasets/ddos-2019.html



Doi : https://doi.org/10.54216/JCIM.130202
Received: July 24, 2023 Revised: November 17, 2023 Accepted: February 09, 2024