2024 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS) | 979-8-3503-9172-5/24/$31.00 ©2024 IEEE | DOI: 10.1109/ANTS63515.2024.10898195

KNOXRT: Knowledge-Networked Optimization for eXploratory Real-time Training

Sanket Mishra, Jeeveth K, Mithun Shivakoti, Aravindan V, Rajkanwar Singh
School of Computer Science & Engineering,
VIT-AP University, Amaravati,India
sanketmishra@live.com, jeevethkoka@gmail.com ,mithun.shivakoti11@gmail.com,
aravindanvemula2@gmail.com, rajkanwars15@outlook.com

***Abstract*—Effective threat detection and classification are essential in real-time data stream processing. In this paper, KNOXRT, an architecture that combines traditional tree-based models, such as Random Forest, XGBoost, and CART, with deep learning models, including Convolutional Neural Networks (CNNs), Long Short-Term Memory (LSTM), and Gated Recurrent Units (GRUs), is presented. Real-time data is ingested using Apache Kafka, and feature extraction is performed in parallel through CNN, CNN+LSTM, and CNN+GRU models. The resulting features are merged and passed through multiple classifiers, achieving high classification accuracy. The best-performing model, CART, achieved perfect training accuracy (1.00) and strong production testing accuracy (0.9927), precision (0.9927), recall (0.9927), and F1-score (0.9927). Class imbalances are addressed using GSmote for balanced training data, and real-time predictions are supported by containerized microservices. Continuous performance monitoring and online retraining ensure the system adapts to dynamic environments without interruption.**

***Index Terms*—Real-time data processing, Threat detection, Classification, Deep Learning**

I. Introduction

In today's digital landscape, unprecedented challenges have been created in the fields of cybersecurity and threat detection due to the rapid growth of data generated from diverse sources. Traditional methods of data processing are often found to struggle in keeping pace with the increasing volume and velocity of information, leading to critical gaps in the identification and classification of potential threats in real time [11]. As malicious actors become more sophisticated, a pressing need for advanced systems that can dynamically adapt to emerging patterns and anomalies in data streams is recognized.

KNOXRT is introduced in this paper as an innovative architecture designed to enhance threat detection and classification capabilities through a hybrid approach that combines traditional machine learning models with cutting-edge deep learning techniques. By integrating established tree-based models such as Random Forest and XGBoost with recurrent neural networks (RNNs) like Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRU), along with Convolutional Neural Networks (CNNs), a comprehensive framework capable of delivering high classification accuracy in complex environments is offered by KNOXRT.

Apache Kafka is leveraged by the architecture to facilitate scalable and efficient real-time data ingestion, ensuring that the demands of continuous streaming from multiple sources can be met by the system. Additionally, the preprocessing stages—including feature extraction utilizing parallelized LSTM and deep neural network models, as well as dataset balancing through GSmote. This enhances not the data preparation process and significantly boosts the predictive performance of KNOXRT.

II. Literature Review

Recent advancements in deep learning have significantly enhanced the capabilities of models utilized for feature extraction across various domains, particularly in time series and sequential data analysis. The integration of Convolutional Neural Networks (CNNs) with Long Short-Term Memory (LSTM) networks and Gated Recurrent Units (GRUs) has emerged as a promising approach, effectively leveraging the strengths of both architectures.

The detection of network intrusions has become more challenging due to the rise of sophisticated attacks. In this context, [10] propose a robust Intrusion Detection System (IDS) using a hierarchical Convolutional Neural Network model called "TreeNets". The TreeNet approach builds a tree-like structure to classify network intrusions in a binary fashion, providing a unique hierarchical learning mechanism. To optimize feature selection, the Binary Grey Wolf Optimization (BGWO) algorithm is incorporated, which significantly improves the detection accuracy of intrusions. The authors benchmark their model on the NSLKDD dataset, achieving an accuracy of 0.8216 on KDDTest+ and 0.6637 on KDDTest-21.

CNNs are renowned for their ability to extract spatial features from data, making them particularly

effective in image processing and scenarios where local patterns are critical. For instance, Ibra Him [4] explored the integration of AI and ML in real-time threat intelligence frameworks, utilizing CNNs to analyze network traffic patterns and demonstrating their proficiency in identifying intricate features indicative of anomalous behavior. However, while CNNs excel at spatial feature extraction, they are limited in their capacity to capture temporal dependencies inherent in sequential data. To address this limitation, researchers have increasingly turned to LSTM and GRU networks, both designed to manage long-range dependencies in sequences. A notable study by Surianarayanan et al. [5] proposed an ensemble model combining CNN and LSTM for real-time anomaly detection. Their findings indicated that the integration of CNNs for initial feature extraction, followed by LSTMs to capture temporal dynamics, resulted in superior performance compared to standalone models. This model effectively utilized the spatial hierarchies identified by the CNN while leveraging the sequential memory capabilities of LSTM to enhance classification accuracy. [9] integrates machine learning models like Long Short-Term Memory (LSTM) to enhance the predictive capabilities of CEP engines thus allowing for the proactive detection of events before they happen.

Similarly, GRUs, which are a simplified version of LSTMs, have been incorporated into hybrid architectures. Research by Mudgal and Bhalla [6] employed a CNN-GRU model for anomaly detection within a Honeypot Intelligence-enabled intrusion detection system, achieving precision rates of 0.9966 while reducing latency by 30%. This demonstrates the effectiveness of combining spatial and temporal feature extraction mechanisms in improving detection accuracy.

The combination of CNNs with LSTMs and GRUs extends beyond anomaly detection and classification tasks. In the realm of natural language processing (NLP), Deepthi et al. [7] introduced a Flexible Real-Time Traffic Stream Processing System (FRTSPS) that utilizes a CNN-LSTM model for sentiment analysis. The CNN component was employed to extract local n-gram features from text, while the LSTM captured contextual information over longer sequences, yielding state-of-the-art performance on benchmark datasets.

Moreover, the integration of machine learning techniques with CNNs has shown promise in various cybersecurity applications. For example, Kajiura and Nakamura [8] analyzed the performance of a distributed processing framework for machine learning-based network intrusion detection systems (NIDS), deploying multiple classifiers and identifying significant differences in throughput and latency. Their research emphasizes the need for selecting suitable machine learning algorithms to optimize performance.

Overall, the integration of CNNs with LSTMs and GRUs represents a significant advancement in feature extraction methodologies. By combining the strengths of spatial and temporal processing, these hybrid models are well-suited to tackle complex tasks across various domains, including cybersecurity, video analysis, and natural language processing. As the field continues to evolve, further research into optimizing these hybrid architectures will likely yield even more robust solutions for feature extraction challenges.

III. Methodology

A. Data Ingestion and Pipeline Structure

The data ingestion layer is managed by a Flask API, which collects real-time sensor data via HTTP POST requests. This raw data is subsequently produced to Kafka topics, with each API endpoint corresponding to a distinct Kafka topic. These topics serve as the communication backbone between various processing jobs within the framework.

- **Flask API:** The collection of sensor data and the sending of it to the Kafka topic for further processing are facilitated by this API. The delivery status of the messages is logged, ensuring that data is processed correctly within the pipeline.
- **Apache Kafka:** This messaging infrastructure acts as a highly scalable and fault-tolerant message broker for real-time data ingestion and distribution throughout the system.

B. Resampling and Class Balancing

To ensure robustness against class imbalances, GSmote from the smote-variants library is implemented in KNOXRT.This adaptive resampling ensures balanced classes before model training, enhancing classifier performance on imbalanced datasets.

C. Feature Extraction and Optimization

Feature extraction is performed using Convolutional Neural Networks (CNNs), which are combined with recurrent networks such as LSTM and GRU to enhance the ability to capture temporal patterns in the data. The combinations are listed as under

- CNN
- CNN+LSTM
- CNN+GRU

These CNN-based models are trained in parallel leveraging PyFlink, and resulting features are sent to Kafka topics. Another PyFlink job subscribes to these topics and sends a consolidated set of features by combining all results.

D. Model Training and Classifiers

The output features from the CNN models are passed through a series of optimized classifiers, which include CatBoost, ExtraTrees, CART, RandomForest, and XGBoost. This multi-model approach ensures that a diverse set of classifiers is evaluated, allowing the system to select the most accurate model for deployment.

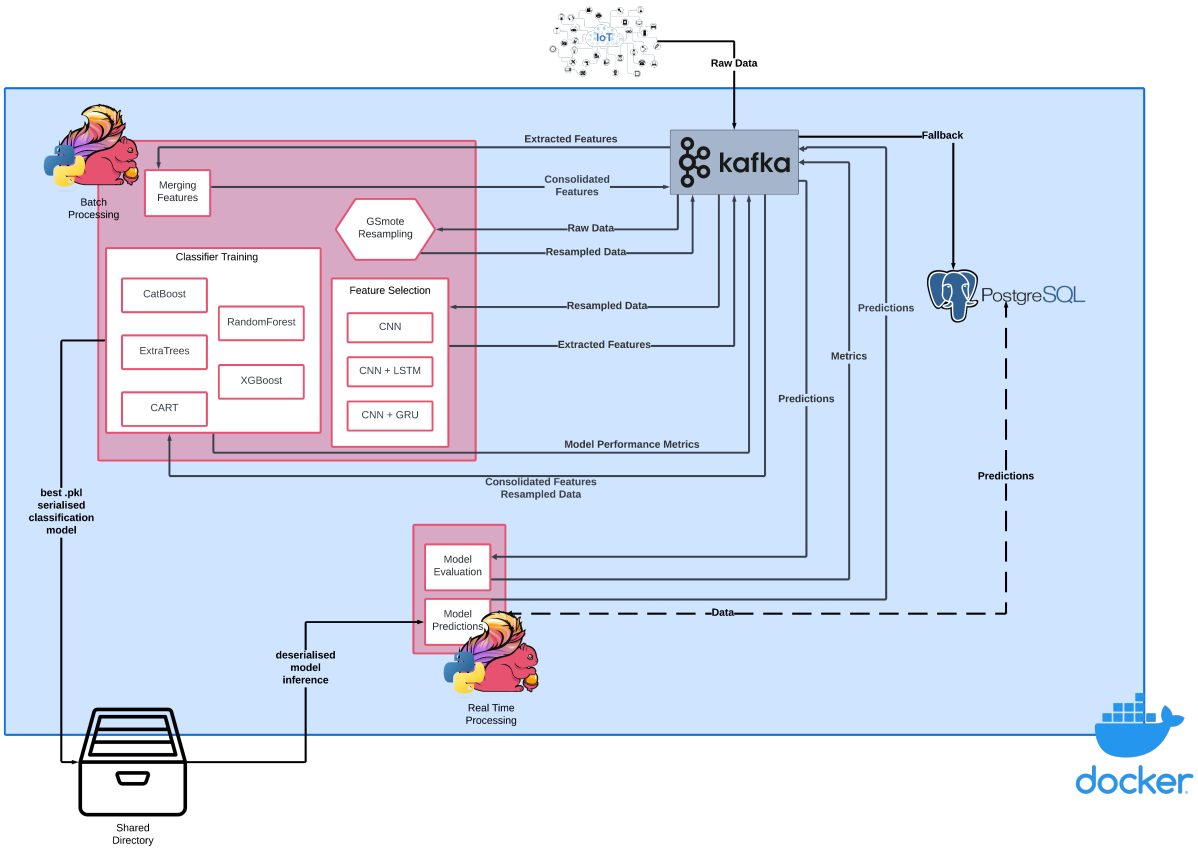

Fig. 1. KNOXRT Framework

- **Classifier Diversity:** The use of multiple classifiers ensures that different decision-making algorithms are leveraged, each offering unique strengths for different types of data distributions. Tree-based models such as Random Forest, ExtraTrees, and XGBoost excel at handling structured data with varying levels of complexity, while CART and CatBoost offer advantages in terms of faster training times and efficient memory usage.
- **Hyperparameter Optimization:** Each classifier is optimized using hyperparameter tuning techniques to ensure that the best possible configuration is employed. This ensures that models are neither underfitted nor overfitted to the training data.
- **Parallelized Training:** The classifiers are trained in parallel to reduce computational time. The performance of each model is evaluated based on accuracy, precision, recall, and F1-score, and the best-performing model is selected for real-time predictions.

E. Model Deployment and Real-time Prediction

The KNOXRT architecture provides the framework for real-time model deployment and prediction. Once the best-performing model is trained, it is serialized as a pickle file and deployed in the pipeline for real-time predictions.

- **Model Serialization:** Trained models are serialized as .pkl files and stored in a shared volume between the Flink Task-Manager and Job-Manager, ensuring efficient retrieval for real-time predictions.
- **Prediction Pipeline:** The deployed model consumes real-time data from the Kafka topics and makes predictions, which are then posted to a new Kafka topic, providing higher accuracy and faster prediction times.

F. Performance Monitoring and Model Retraining

In the KNOXRT framework, model performance is continuously monitored using Apache Flink jobs that compute performance metrics in fixed windows and publish the results to Kafka.

- **Real-Time Monitoring:** A PyFlink job is responsible for monitoring model performance (accuracy, precision, recall, F1-score) in real-time. Performance metrics are computed at regular intervals and posted to Kafka topics for live dashboard monitoring.
- **Online Model Retraining:** Based on these performance metrics, models can be retrained and redeployed without interrupting the production

pipeline, allowing the system to adapt to changing data patterns and maintain high accuracy.

G. Production Environment and Infrastructure

- **Docker:** All components of the system are containerized using Docker, ensuring that the framework can be easily deployed and scaled across different environments.
- **PostgreSQL:** All predictions are stored in a PostgreSQL database, which serves as a fallback in case of Kafka failures.

IV. Results and Discussion

The ensemble CART algorithm is the best-performing model. Training metrics are listed in Table I, Validation in Table II, and Testing in Table III.

TABLE I
TRAINING MODEL PERFORMANCES

SNo	Model	Accuracy	Precision	Recall	F1 Score
1	CatBoost	0.99623	0.99624	0.99623	0.99623
2	ExtraTrees	0.99917	0.99917	0.99917	0.99917
3	CART	1	1	1	1
4	RandomForest	0.99979	0.99979	0.99979	0.99979
5	XGBoost	0.99914	0.99914	0.99914	0.99914

Table I shows the performance metrics of various models during the training phase. CART achieves perfect performance with 1.00 across all metrics, demonstrating that it fully learns the patterns from the training data. ExtraTrees and RandomForest models also perform exceptionally well, with accuracies of 0.99917 and 0.99979, respectively, suggesting strong predictive capability without overfitting. XGBoost and CatBoost have slightly lower but still excellent accuracy and precision, at around 0.999 and 0.996, respectively, indicating their robustness.

TABLE II
VALIDATION MODEL PERFORMANCES

SNo	Model	Accuracy	Precision	Recall	F1 Score
1	CatBoost	0.99637	0.99637	0.99637	0.99637
2	ExtraTrees	0.99917	0.99917	0.99917	0.99917
3	CART	1	1	1	1
4	RandomForest	0.99979	0.99979	0.99979	0.99979
5	XGBoost	0.99919	0.99919	0.99919	0.99919

In the validation phase (Table II), CART continues to deliver perfect performance, indicating its ability to generalize well. Random Forest and ExtraTrees maintain their near-perfect performance from training, showing minimal overfitting. XGBoost and CatBoost maintain their positions slightly behind the top performers, with accuracy and other metrics just under 1.00, demonstrating robust generalization to unseen validation data.

During production testing (Table III, XGBoost achieves the highest metrics (accuracy, precision, recall, F1 score at 0.99723), demonstrating its strong capability to perform well in a real-world environment.

TABLE III
PRODUCTION TESTING MODEL PERFORMANCES

SNo	Model	Accuracy	Precision	Recall	F1 Score
1	CatBoost	0.99388	0.99390	0.99388	0.99389
2	ExtraTrees	0.99556	0.99557	0.99556	0.99556
3	CART	0.99267	0.99267	0.99267	0.99267
4	RandomForest	0.99712	0.99712	0.99712	0.99712
5	XGBoost	0.99723	0.99723	0.99723	0.99723

Random Forest follows closely with similar performance. ExtraTrees and CART show slightly lower metrics, indicating solid but not top-tier performance. CatBoost also performs well, though it falls behind other models in this set with an accuracy of 0.99388.

Figures 2, 3 and 4 compare performances of various models with features from each of the feature selection techniques for training, validation and testing respectively. CNN-GRU with ExtraTrees and CatBoost consistently deliver the best results across training, validation, and test sets, maintaining strong generalization with minimal performance drop-off. CNN-LSTM models generally show a more noticeable drop in test performance, especially with CART and XGBoost, indicating that LSTM might not be as effective for this dataset compared to GRU. CNN models (without LSTM/GRU) perform well across most cases, though CART shows more significant drops during testing, suggesting that tree-based models like ExtraTrees and Random Forest are better suited for CNN-extracted features.

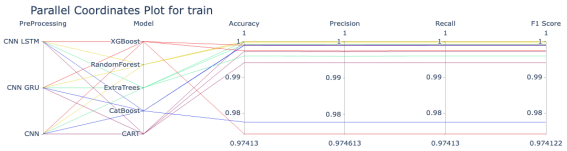

Fig. 2. Model Metrics on Train

In Fig.2, CNN-GRU with Extra Trees and CART achieve perfect scores (accuracy: 1.00 across all metrics), indicating that these models capture patterns well. The training set analysis confirms that GRU-based models paired with advanced classifiers like CatBoost and Random Forest perform well across training, validation, and test sets, demonstrating minimal overfitting and strong generalization.

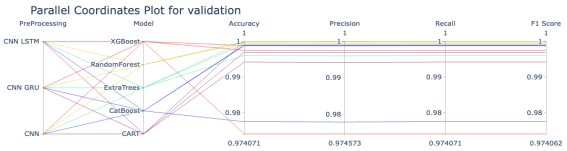

Fig. 3. Model Metrics on Valid

In Fig.3, models using just CNN for feature extraction, such as CNN with Extra Trees, show slightly

lower performance (accuracy: 0.996), reinforcing that the lack of temporal feature extraction can affect generalization. The validation results highlight that the models performing best on the test set also excel in validation, confirming that they are not overfitting and have robust generalization capabilities.

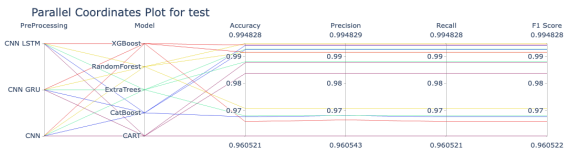

Fig. 4. Model Metrics on Test

In Fig.4, models like CNN-GRU with CART and Extra Trees exhibit slightly lower performance on the test set (accuracy: 0.9837 for CART, 0.9881 for Extra Trees), which implies that while CNN-GRU excels, certain tree-based models might struggle to generalize as well as boosting-based models like CatBoost and Random Forest. Models with temporal feature extraction (GRU/LSTM) paired with advanced tree-based classifiers like CatBoost and Random Forest perform better on unseen test data compared to models relying solely on CNN-based preprocessing.

V. Conclusion and Future Scope

The KNOXRT framework has marked a significant leap in real-time data processing and threat detection by effectively combining traditional machine learning models with deep learning techniques. By integrating convolutional neural networks (CNNs) with tree-based models like Random Forest and XGBoost, KNOXRT has achieved impressive classification accuracy, with the CART model reaching an outstanding 0.9948. Its capability to manage large-scale, real-time data ingestion—supported by Apache Kafka and PyFlink—demonstrates its scalability and adaptability in fast-changing environments. Moreover, KNOXRT ensures optimal performance over time through continuous monitoring and the ability to retrain models online as data patterns shift. Advanced preprocessing methods, including parallelized feature extraction and class balancing using GSmote, have further enhanced the robustness and accuracy of its predictions. Overall, KNOXRT sets a new standard for real-time exploratory training and classification in cybersecurity. Looking ahead, future research will delve into integrating advanced anomaly detection systems to enhance the framework's resilience against unknown attack vectors. Additionally, improving scalability by incorporating more efficient distributed computing technologies will be a key focus.

References

[1] Adireddy, A., Kurnala, V., Patlolla, S. & Arora, G. An Ensemble Approach of CNN and GRU Models for Network and Server Intrusion. *2024 5th International Conference For Emerging Technology (INCET)*. pp. 1-5 (2024)

[2] Vikram, A. & Mohana Anomaly detection in Network Traffic Using Unsupervised Machine learning Approach. *2020 5th International Conference On Communication And Electronics Systems (ICCES)*. pp. 476-479 (2020)

[3] Vyas, S., Tyagi, R., Jain, C. & Sahu, S. Performance Evaluation of Apache Kafka – A Modern Platform for Real Time Data Streaming. *2022 2nd International Conference On Innovative Practices In Technology And Management (ICIPTM)*. **2** pp. 465-470 (2022)

[4] I. Him. "Leveraging Artificial Intelligence and Machine Learning for Real-Time Threat Intelligence: Enhancing Incident Response Capabilities,", 2024.

[5] C. Surianarayanan, S. Kunasekaran, and P. R. Chelliah, "A high-throughput architecture for anomaly detection in streaming data using machine learning algorithms," *International Journal of Information Technology*, vol. 16, no. 1, pp. 493-506, Nov. 2023, doi: 10.1007/s41870-023-01585-0

[6] A. Mudgal, S. Bhatia, "Big Data Enabled Intrusion Detection with Honeypot Intelligence System on Apache Flink (BDE-IDHIS)," in 2023 14th International Conference on Computing Communication and Networking Technologies (ICCCNT), 2023.

[7] Deepthi, B., et al. "An efficient architecture for processing real-time traffic data streams using apache flink," in Multimedia Tools and Applications, vol. 83, no. 13, pp. 37369–37385, 2023.

[8] Maho Kajiura, Junya Nakamura, "Practical Performance of a Distributed Processing Framework for Machine-Learning-based NIDS," 2024.

[9] S. Mishra, M. Jain, B. Siva Naga Sasank, and C. Hota, "An Ingestion Based Analytics Framework for Complex Event Processing Engine in Internet of Things," Lecture Notes in Computer Science. Springer International Publishing, pp. 266–281, 2018.

[10] Mishra, S., Dwivedula, R., Kshirsagar, V., & Hota, C. (2021). Robust Detection of Network Intrusion using Tree-based Convolutional Neural Networks. In Proceedings of the 3rd ACM India Joint International Conference on Data Science & Management of Data (8th ACM IKDD CODS & 26th COMAD) (pp. 233–237). Association for Computing Machinery.

[11] Qing-jun Yuan, Gaopeng Gou, Yanbei Zhu, Yuefei Zhu, G. Xiong, Yongjuan Wang (2024). MCRe: A Unified Framework for Handling Malicious Traffic With Noise Labels Based on Multidimensional Constraint Representation. IEEE Transactions on Information Forensics and Security, 19, 133-147.