Indonesian Journal of Electrical Engineering and Computer Science
Vol. 41, No. 1, January 2026, pp. 394~408
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v41.i1.pp394-408

Cryptojacking detection using model-agnostic explainability

Elodie Ngoie Mutombo[1], Mike Wa Nkongolo[2], Mahmut Tokmak[3]
[1]Department of Computer Science, University of Pretoria, Pretoria, South Africa
[2]Department of Informatics, Faculty of Engineering, Built Environment and Information Technology, University of Pretoria, Pretoria, South Africa
[3]Department of Management Information Systems, Bucak Zeliha Tolunay School of Applied Technology and Management, Burdur Mehmet Akif Ersoy University, Burdur, Turkey

Article Info

Article history:

Received Jun 16, 2024
Revised Aug 24, 2025
Accepted Dec 13, 2025

Keywords:

Cryptojacking
Cybersecurity
Explainable AI
Forensic analysis
Large language models
Process memory data
Transformers
UGRansome dataset

ABSTRACT

Cryptojacking is the illicit use of computing resources for cryptocurrency mining. It has emerged as a serious cybersecurity threat that degrades critical system performance and increases operational costs. This paper proposes an advanced machine learning (ML) framework that integrates transformer-based language models with post hoc explainable artificial intelligence (XAI) to detect cryptojacking using complementary network traffic and process memory data. Numerical and categorical features are discretized and tokenized to enable semantic modelling and contextual learning. Experimental results show that transformer models effectively capture cryptojacking-related behavioral patterns, with decoding-enhanced BERT with disentangled attention (DeBERTa) achieving high detection performance and recall exceeding 80%. bidirectional encoder representations from transformers (BERT) attains comparable recall with lower computational overhead, making it well suited for real-time environments, while robustly optimized BERT approach (RoBERTa) and DeBERTa are more appropriate for offline or batch-based analysis. Model performance is evaluated using standard classification metrics, and XAI techniques provide interpretable insights into feature relevance, supporting transparent and reliable detection. In general, the proposed framework delivers an effective and deployment-ready solution for cryptojacking detection.

Corresponding Author:

Mike Wa Nkongolo
Department of Informatics, Faculty of Engineering, Built Environment and Information Technology
University of Pretoria
Pretoria, South Africa
Email: mike.wankongolo@up.ac.za

1. INTRODUCTION

Cryptojacking attacks have emerged as a major cybersecurity threat that exploits computing resources without consent to mine cryptocurrencies for profit [1]. Attackers use methods such as malicious websites, compromised servers, and file-less techniques, with in-browser mining becoming increasingly common [1]. These attacks degrade critical system performance, increase energy consumption, and cause financial losses. The rise of cryptojacking-as-a-service (CaaS) has lowered barriers for cybercriminals, amplifying the threat [1]. The inherent decentralization and anonymity of cryptocurrencies further impede efforts to track and analyze cryptojacking activity [1]. Proactive mitigation using large language models (LLMs) and explainable artificial intelligence (XAI) is crucial, since traditional signature-based machine learning (ML) approaches struggle to detect zero-day attacks [1]. While LLMs can classify malicious activity, their interpretability is limited, motivating the use of XAI to improve transparency and reliability.

Despite extensive research on cryptojacking, early detection remains underexplored [1], [2]. Promising approaches involve monitoring application programming interface (APIs), registries, file activity, and network metadata, yet most tools are evaluated in simulations, facing challenges in data collection, ground truth accuracy, and interpretability [3]-[5]. This study uses network traffic and process memory data to detect cryptojacking and improve interpretability.

The research evaluates three LLMs for early detection and applies XAI techniques like local interpretable model-agnostic explanations (LIME) and shapely additive explanations (SHAP), to explain feature contributions. The experiment aims to identify the most effective LLM that enhances model explainability and determine the optimal dataset for cryptojacking detection. The study hypothesizes that i) LLMs accurately detect cryptojacking, ii) LIME improves interpretability, and iii) the datasets provide comprehensive information for effective classification. Studies highlight shortcomings in current cryptojacking defenses [6], demonstrating the need for more effective and early detection. This is critical as evolving cryptojacking techniques exploit computing devices such as GPUs and CPUs for cryptocurrency mining, thereby harming system performance and security. Effective early prevention is essential to mitigate the impact of cryptojacking to protect critical systems from data loss and financial damage by identifying attacks before devices are exploited [7]. LLMs can support this process by analyzing extensive network traffic datasets to detect patterns of malicious activity to enable timely detection and response [7].

For example, Adigun *et al.* [8] present a method to detect cryptojacking activities related to Bitcoin (BTC) traffic using six ML algorithms. The random forest (RF) model achieved the best performance even though LLMs and XAI were not applied. A hybrid ML method combining internet protocol (IP) blacklisting and payload inspection is implemented by Danesh *et al.* [9] to detect cryptojacking at the network edge. The method achieved high accuracy (97.02%), but lacked LLMs and XAI. This limits its ability to provide interpretable explanations. Cao *et al.* [10] introduced MagInspector, an unsupervised approach that leverages GPU magnetic signatures and adversarial autoencoders to boost the detection accuracy of mutable cryptojacking by 25.5% on NVIDIA GPUs and 17.8% on AMD GPUs.

Advanced models like MagInspector have been widely used to detect various network attacks via traffic analysis [11], showing potential for cryptojacking detection. As noted above, LLMs combined with XAI techniques have not yet been applied in this domain. The proposed method demonstrates that integrating LLMs with XAI can enhance cryptojacking pattern recognition. Nevertheless, challenges such as model opacity, high-dimensional representations, and context dependencies highlight the need for explainability [12]. XAI methods like SHAP and LIME can provide insights into crucial features to enhance the interpretability of LLM results [13]. Building on this, the study introduces a hybrid and model-agnostic framework to improve LLM interpretability for robust cryptojacking analysis.

The limitations of existing cryptojacking detection literature arise from several factors. Firstly, there is a lack of publicly available datasets. Many approaches also lack interpretability in identifying cryptojacking, malicious addresses, and cryptocurrency-mining activities, often focusing on broader malware detection [8]-[10]. For example, while ML has been used to classify zero-day exploits, the specific characteristics and interpretability of cryptojacking remain largely unaddressed [11], [12]. Some recurrent neural network (RNN)-based methods, including long short-term memory (LSTM), gated recurrent unit (GRU), and simple RNN, rely heavily on legacy datasets like UNSW-NB15 and NSL-KDD [14]. This limits their applicability to novel malware detection. Traditional ML techniques may also produce false positives. Therefore, future research should explore hybrid and agnostic LLM to accurately detect, predict, mitigate, and interpret cryptojacking threats across diverse cryptocurrency networks [15]. Current detection methods show promise, but vary with attack type, system setup, and conditions [13]-[15].

The manuscript is organized as follows: section 2 describes the methodology with experimental datasets. In section 3 details the proposed approach, including the chosen LLMs and evaluation metrics. Section 4 presents and analyses the results while section 5 concludes the study.

2. PROPOSED METHOD

Experimental steps delineating the research methodology for implementing the proposed model consists of preprocessing experimental datasets, refining features, choosing appropriate LLMs, and carrying out experimental tests to validate the selected approach (Figure 1). These steps have been stratified into i) data collection, ii) preprocessing, iii) feature extraction, iv) LLM classification, and v) XAI (Figure 1).

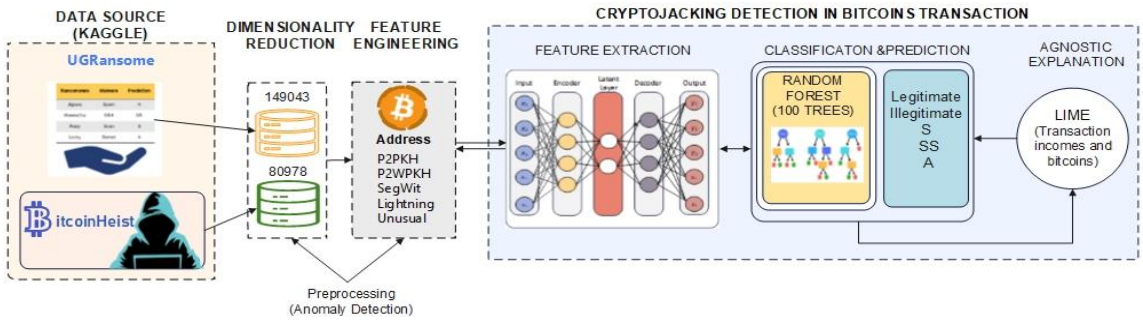


Figure 1. The proposed hybrid ML model

2.1. Experimental datasets

2.1.1. UGRansome dataset

UGRansome is the first experimental dataset. It is a key resource for identifying zero-day exploits [16]. Created in 2021, the dataset includes previously undocumented zero-day attacks [16]. And contains 207,533 samples each with 14 attributes (Table 1). Its large size supports effective ML training and testing. Recently, Zhang *et al.* [16] used the UGRansome by employing F-measures to enhance model interpretability. Compared to other datasets, UGRansome offers a larger feature set and more robust model evaluation [2], [5]. The dataset is publicly available in CSV format (10.0 MB), covering 17 ransomware families (https://www.kaggle.com/datasets/nkongolo/ugransome-dataset/data).

Table 1. Attributes in the UGRansome dataset

Column	Feature	Type	Description
1	Timestamp	Quantitative	Numeric time taken by a transaction to occur (e.g., 45 seconds).
2	Protocol	Qualitative	Categorical network protocol used for the communication.
3	Flag	Qualitative	Categorical flags associated with the network communication.
4	Family	Qualitative	Ransomware family (e.g., CryptoLocker, Locky, NoobCrypt).
5	Cluster	Quantitative	Numerical groups categorizing malware.
6	Seed address	Qualitative	The initial or source address involved in the transaction.
7	Expended address	Qualitative	The recipient or destination address involved in the transaction.
8	Bitcoin	Quantitative	The bitcoin amount involved in transactions (e.g., 3.0 BTC).
9	USD	Quantitative	The equivalent value of bitcoin in US dollars.
10	Network flow	Quantitative	The size of data transferred in the transaction (e.g., 454 bytes).
11	IP	Qualitative	The IP address.
12	Threats	Qualitative	Malware associated with malicious activities (e.g., phishing).
13	Port	Quantitative	The network port (e.g., 5061).
14	Prediction	Qualitative	Target variable (signature (S), anomaly (A), synthetic signature (SS)).

2.1.2. Process memory dataset

The PM is the second experimental dataset (Table 2) collected via dynamic malware analysis using the MalFe platform. The ransomware and benign executables are compiled in a sandbox environment [17] with API call traces recorded as JSON reports. These reports are used to extract categorical and temporal features, including call types, timestamps, frequencies, intervals, and sequences [17]. After preprocessing, the dataset contains approximately 280–285 unique API call features, supporting malware visualization and classification based on runtime behaviour (https://www.kaggle.com/code/thashannaick/ransomware-detection-using-llm-and-xai-techniques).

Table 2. Attributes in the PM dataset

Column	Feature	Type	Description
1	r	Numerical	Count of read-related API calls executed by the process.
2	rw	Numerical	Count of combined read and write API calls.
3	rx	Numerical	Count of read and execute API calls.
4	rwc	Numerical	Count of read, write, and create API calls.
5	rwx	Numerical	Count of read, write, and execute API calls.
6	rwxc	Numerical	Count of read, write, execute, and create API calls.
7	Category	Categorical	Category of the API call (e.g., file, registry, process, network).
8	Family	Categorical	Ransomware family associated with the sample.
9	Label	Categorical	Class label indicating benign or malware.

2.1.3. Suitability of experimental datasets for cryptojacking detection

The UGRansome and PM datasets are suitable for cryptojacking detection because they capture complementary malicious behaviors [18]. UGRansome provides cryptocurrency transaction and network features that reflect illicit mining activity (Table 1), while the PM dataset captures runtime process behavior through API call patterns that reveal stealthy cryptomining operations (Table 2). Together, these datasets can potentially enable LLMs to learn complex behavioral dependencies across network and host levels. Their structured features further support XAI methods, allowing transparent interpretation of feature contributions in cryptojacking detection [7].

2.2. Feature preprocessing and encoding for LLM-based cryptojacking analysis

Numerical and categorical features from the UGRansome and PM datasets are preprocessed and transformed to enable effective fine-tuning of transformer-based LLMs [18]. Missing numerical values are imputed using median statistics, while categorical attributes are completed using mode values to preserve data integrity and generalization. Label inconsistencies across datasets are standardized prior to training. Continuous numerical features are discretized using quantile-based binning and encoded into token-like representations [7], [18].

These tokens are concatenated into sequential text inputs to allow structured cryptojacking data to be processed as linguistic sequences by LLMs. This preprocessing and encoding pipeline ensures compatibility with transformer architectures while retaining behavioural semantics essential for cryptojacking detection, and provides a reliable foundation for downstream explainable analysis. Both datasets were free of missing values and duplicates after removing negative timestamps. Categorical features were encoded using Python's label encoder.

3. METHOD

An autoencoder is a type of artificial neural network (ANN) used for unsupervised deep learning (DL) [19]. It consists of two main components: an encoder that compresses the input data into a lower-dimensional representation, and a decoder that reconstructs the original input from this representation.

Let X be the input data, $f_{encoder}(X)$ be the encoding function, and $f_{decoder}(X)$ be the decoding function. The goal of training an autoencoder is to minimise the reconstruction error measured using a loss function such as mean squared error (MSE) [19]. In the autoencoding process, θ represents the parameters of both the encoder and decoder, and n is the number of training examples [19]. The encoder maps the input data X to a lower-dimensional representation $Z = f_{encoder}(X)$. Through this process, the autoencoder learns a compact representation of the input data, capturing its essential features in a lower-dimensional space.

While autoencoders are effective at learning compact and task-agnostic representations of data, modern natural language processing (NLP) often requires modeling complex sequential dependencies and contextual relationships in large corpora [18], [20], [21]. This limitation motivates the use of transformer-based architectures (Figure 2), such as LLMs, which extend the concept of learned representations to sequences of arbitrary length.

Transformers replace the fixed encoding–decoding paradigm of autoencoders with attention mechanisms that dynamically capture relationships between elements in a sequence to enable state-of-the-art performance in tasks such as text generation, summarization, and classification [7], [18], [20], [21]. In this study, cryptojacking detection using the UGRansome and PM datasets is performed by using an autoencoder for feature extraction (Figure 1), transformer-based LLMs for classification (Figure 2), and explainability is provided via SHAP and LIME for feature attribution (Figure 1).

The study uses pre-trained transformer-based language models (PLM), such as bidirectional encoder representations from transformers (BERT), robustly optimized BERT approach (RoBERTa), and decoding-enhanced BERT with disentangled attention (DeBERTa), to extract rich, and context-aware feature representations for improved classification. The BERT is a pre-trained language model that utilizes bidirectional attention to capture the contextual meaning of tokens in a sequence [21].

By understanding both preceding and succeeding tokens simultaneously, BERT generates rich embeddings that represent complex relationships in textual or sequential data [20]. In the context of cryptojacking detection using network traffic of the UGRansome dataset and process memory features, BERT can encode system logs, process names, or memory traces into embeddings that capture behavioral patterns indicative of malicious activity (Figure 2). The RoBERTa is an improved variant of BERT that employs dynamic masking, larger batch sizes, and more training data to achieve better generalization and representation quality [21]. For cryptojacking detection, RoBERTa can provide more robust feature representations from the UGRansome and PM datasets to improve the model's ability in distinguishing between normal and malicious process behaviors (Figure 2).

The DeBERTa enhances BERT by using disentangled attention mechanisms and an improved decoding structure to capture both content and positional information more effectively [21]. In cryptojacking detection, DeBERTa can extract fine-grained, and context-aware features from memory snapshots or malware execution patterns, which can improve classification accuracy when combined with downstream LLM classifiers (Figure 2). Together, these PLMs can be fine-tuned on cryptojacking datasets (Figure 2) by juxtaposing their ability to learn high-dimensional, and semantically rich representations, which are then fed into classifiers to accurately detect and differentiate malicious activities.

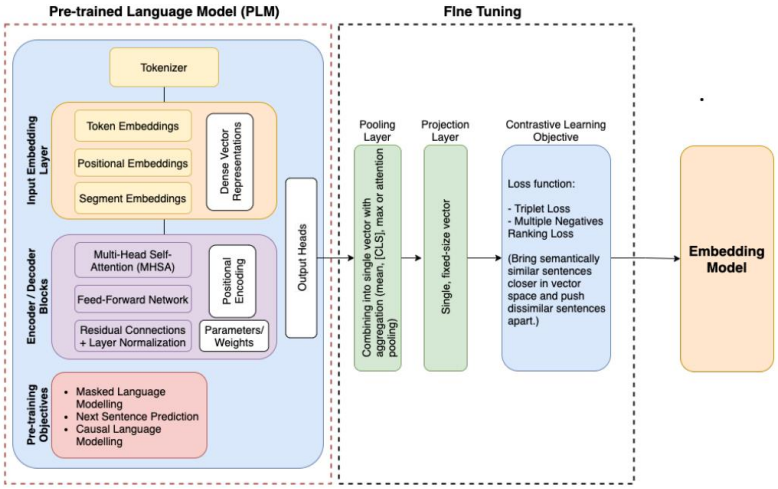


Figure 2. Conceptual flow: PLM, fine-tuning, and embedding model

3.1. Anomaly detection techniques

The isolation forest (IF) algorithm is used to detect anomalies by isolating observations through random splits (Figure 1). For a given dataset *X* with *n* observations and *d* features, the IF is described (1) as:

$$Isolation(x) = \sum_{i=1}^{n} \frac{1}{h(x_i)} \quad (1)$$

where $h(x_i)$ is the path length from the root node to the terminal node for observation x_i. The anomaly score $S(x)$ for x is also defined as (2):

$$S(x) = 2^{-\frac{E(h(x))}{c(n)}} \quad (2)$$

where $E(h(x))$ is the average path length of x and $c(n)$ is the average path length of unsuccessful searches (3).

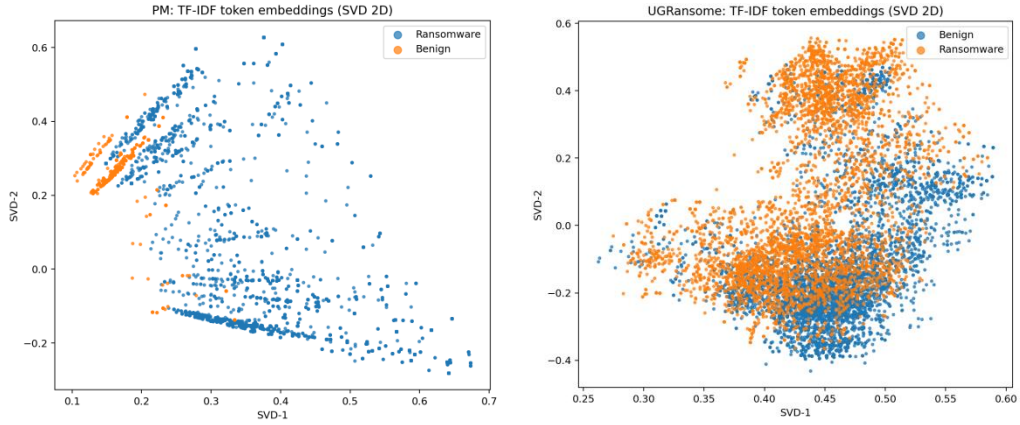
$$c(n) = 2H(n-1) - \frac{2(n-1)}{n} \quad (3)$$

With $H(i)$ being the i-th harmonic number. If $S(x)$ is close to 1, then x is considered an outlier. Otherwise, it is considered normal. The autoencoder detects anomalies based on the reconstruction error $\mathcal{L}$ computed as the MSE between x and its reconstruction $\hat{x}$ (4).

$$\mathcal{L}_{(x,\hat{x})} = \frac{1}{d}\sum_{i=1}^{d}(x_i - \hat{x}_i)^2 \quad (4)$$

A threshold ϵ is set for $\hat{x} > \epsilon$, when x is flagged as an anomaly. Preprocessing these anomalies is crucial for enhancing the performance and reliability of the proposed model. Anomalies, such as outliers and missing values could skew results and lead to inaccurate predictions. The study addressed these issues by preprocessing anomaly data, which enhances the accuracy and improves the robustness of the proposed model, making it less sensitive to noise and variations in the data. Furthermore, clean and well pre-processed data reduced training time, and the computational resources required, making the entire modelling process more efficient. Figure 3 shows the process by which anomaly data is filtered and transformed into embeddings.

This technique removes extreme values beyond a defined threshold, preventing distortion of the mean or variance (Figure 3(a)). As shown in Figure 3(b), the resulting datasets exhibit reduced random variability, highlighting key features that contribute to improved model efficiency, particularly for the UGRansome dataset.



(a) (b)

Figure 3. Embedding generation from (a) PM and (b) UGRansome data

3.2. Local interpretable model-agnostic explanations

LIME aims to explain individual predictions by perturbing the input data and observing changes in the output [22]. The formula for computing the value (or weight) of a feature involves the following steps:

– Generate perturbed samples (5)

$$Z = \{z_{1,} z_{2,} \dots, z_m\} \tag{5}$$

Where z_i are the perturbed samples generated around x.

– Model predictions (6)

$$\hat{f}(z) = \{\hat{f}(z_1), \hat{f}(z_2), \dots, \hat{f}(z_m)\} \tag{6}$$

Where $\hat{f}$ is the black-box model, in this case, PLMs.

– Weighting samples (7)

$$\pi_x(z) = \frac{exp(-\frac{(D(x,z))^2}{\sigma^2})}{\sum_{z' \in Z} exp(-\frac{D(x,z')}{\sigma^2})} \tag{7}$$

Where *D (x, z)* is a distance function (e.g., Euclidean distance) between *x* and *z*, and σ is a kernel width parameter.

– Linear model fitting (8)

$$g(z) = \beta_0 + \beta_{1z_1} + \beta_{2z_2} + \cdots + \beta_{nz_n} \tag{8}$$

Here, the coefficients β represent the importance of each feature, and the linear model g is used to interpret the weighted samples [20]-[22]. Hence, β derived from g fitted to these samples explain the importance of features in influencing the model's prediction. This study utilizes LIME as a post-hoc method to generate explanations after the autoencoder and LLMs have been trained (Figure 4). Additionally, SHAP is employed to quantify the contribution of each feature to the model's predictions [23], providing a complementary perspective on model interpretability. The aim is to enhance the explanation of cryptojacking detection in ML-based XAI systems. In the experiments, ransomware-derived features are used to detect cryptojacking because both malware types exhibit abnormal system behaviors, such as excessive CPU and memory usage, enabling anomaly-based detection approaches to be effectively applied (Figure 4).

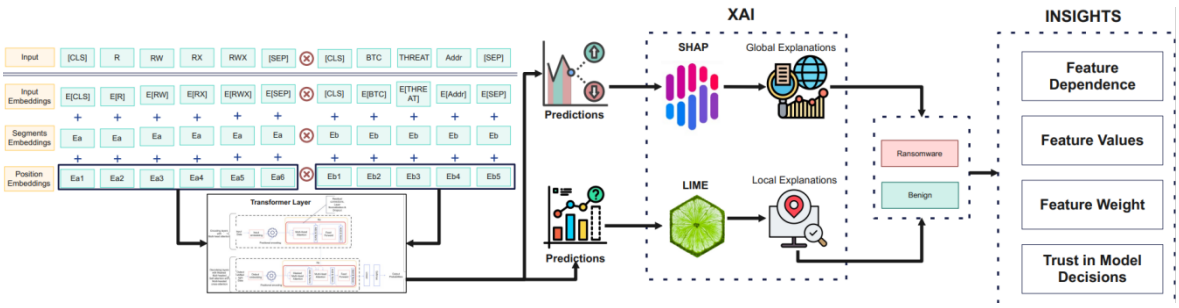


Figure 4. Experimental XAI setup

3.3. Parameters fine-tuning

Table 3 outlines parameters of the proposed hybrid model. For the autoencoder, the parameters include the Adam optimizer, MSE loss function, rectified linear unit (ReLu) as the activation function for hidden layers, and Sigmoid activation function for the output layer. These parameters are used for training the autoencoder model [20], [21]. In turn, Gini and entropy measured the quality of a RF split, with the number of trees in the forest set to 100. The LIME model is set to RF with a Logit link function to generate local explanations for LLM's predictions (Table 3). The link function specifies transformations applied to the output of the interpretable model [21]. The study uses Python libraries such as numpy, Keras, TensorFlow, pandas, matplotlib, seaborn, scikit-learn, and lime to implement the hybrid model (Table 3). Transformer-based models, BERT (max_seq_len=128, batch=32), RoBERTa (learning_rate=2e-5, epochs=3), and DeBERTa (attention_heads=12, epochs=3) were used for contextual feature extraction.

Table 3. Parameters of the proposed hybrid model

Algorithm	Parameters	Description
Autoencoder	Hidden layers=2, epochs=50, Adam optimizer, ReLU, MSE, and Sigmoid	Feature extraction. Neural network (NN) for anomaly detection.
RoBERTa	Learning_rate=2e-5, epochs=3	Optimized BERT variant with robust pre-training.
DeBERTa	Attention_heads=12, epochs=3	Transformer model with disentangled attention.
BERT	Max_seq_len=12, batch=32	Transformer-based model for contextual classification.
LIME	RF and Logit: num_features=14	Post-hoc explainer that approximates local model behavior.
SHAP	Model_type=tree	Post-hoc explainer using Shapley values to assign feature importance.

3.4. Data split

The study partitioned the datasets into 80% training and 20% testing sets using scikit-learn's train-test split while preserving class distribution, and employed four-fold cross-validation to evaluate model performance and reduce bias.

3.5. Evaluation metrics

The accuracy of a binary classification model is assessed by dividing the count of correctly classified samples (true positive (TP) and true negative (TN)) by the total number of samples including false positive (FP), and false negative (FN) [22], [23]. This metric serves as an indicator of the proportion of instances that are accurately classified within the hybrid model (9).

$$Accuracy = \frac{TP+TN}{N} = \frac{TP+TN}{TP+TN+FP+FN} \quad (9)$$

Precision represents the proportion of samples accurately classified as positive, thus measuring the proportion of TP predictions (10).

$$Precision = \frac{TP}{TP+FP} \quad (10)$$

The recall is defined as the ratio of correctly classified positive samples to the total number of actual positive samples. This metric serves to quantify the proportion of TP accurately identified by the model (11).

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

F1 score represents the harmonic mean of precision and recall (12).

$$F1 = \frac{2}{\frac{1}{Precision}+\frac{1}{Recall}} = \frac{2\times Precision\times Recall}{Precision+Recall} \quad (12)$$

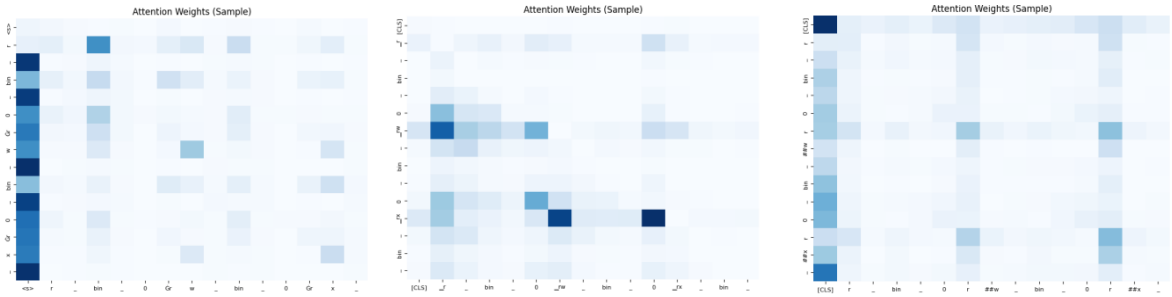
The receiver operating characteristic – area under the curve (ROC-AUC) is a performance metric for classification models [23]. The ROC curve plots the TP rate (sensitivity) against the FP rate (specificity) at different threshold settings. The AUC measures the area under this curve, representing the model's ability to distinguish between classes. A value of 1 indicates perfect classification, while 0.5 corresponds to random guessing [23]. Table 4 provides a confusion matrix used to evaluate the classification performance in identifying cryptojacking transactions [22]. In the context of cryptojacking detection, TP refers to transactions correctly classified as cryptojacking. FP denotes transactions incorrectly identified as cryptojacking. Conversely, FN represents misclassified cryptojacking transactions. TN encompasses transactions accurately classified as non-cryptojacking.

Table 4. Confusion matrix

Actual/predicted	Positive (cryptojacking)	Negative (non-cryptojacking)
Positive (cryptojacking)	TP	FP
Negative (non-cryptojacking)	FN	TN

4. RESULTS AND DISCUSSION

The hybrid model underwent evaluation using a four-fold cross-validation approach, and the reported performance is an average across all folds. As highlighted in Figure 5, the attention visualizations reveal that transformer models effectively learn a contextual representation of discretized features. DeBERTa exhibits a highly dynamic and non-uniform attention distribution (Figure 5(a)), allocating focus across multiple bin tokens and operation-specific indicators (e.g., rx, rw). This demonstrates its ability to interpret the engineered token sequence as a structured behavioral language. In contrast, RoBERTa (Figure 5(b)) and BERT (Figure 5(c)) show more concentrated attention, primarily focusing on initial tokens, suggesting strong performance but reduced sensitivity to feature variation (Figure 5).



(a) (b) (c)

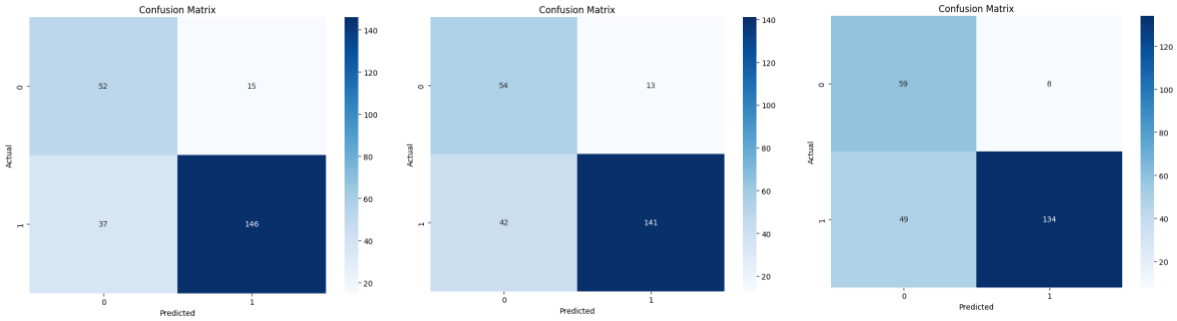
Figure 5. Attention weight for; (a) RoBERTa, (b) DeBERTa, and (c) BERT on tokenized features

These results confirm the effectiveness of the feature transformation and tokenization process, as well as the transformers' ability to selectively prioritize salient behavioral features. The visualizations enhance model transparency by showing that critical related features are explicitly identified and weighted during classification, reinforcing the suitability of transformer architectures for explainable malware detection [24].

4.1. Classification results

The transformer models demonstrated strong detection capabilities with varying trade-offs between recall and precision. DeBERTa achieved an overall accuracy of 77.2%, correctly identifying most malware instances, with a high recall of over 90%, indicating effective detection of malicious activity (Figure 6(b)). Its elevated FP rate (FPR) suggests an overly cautious behavior, potentially increasing false alarms. Similarly, BERT exhibited high malware recall (≈90%), successfully identifying the majority of malicious samples, but at the cost of a moderate precision due to FPs, reflecting a tendency to misclassify unusual benign activity as

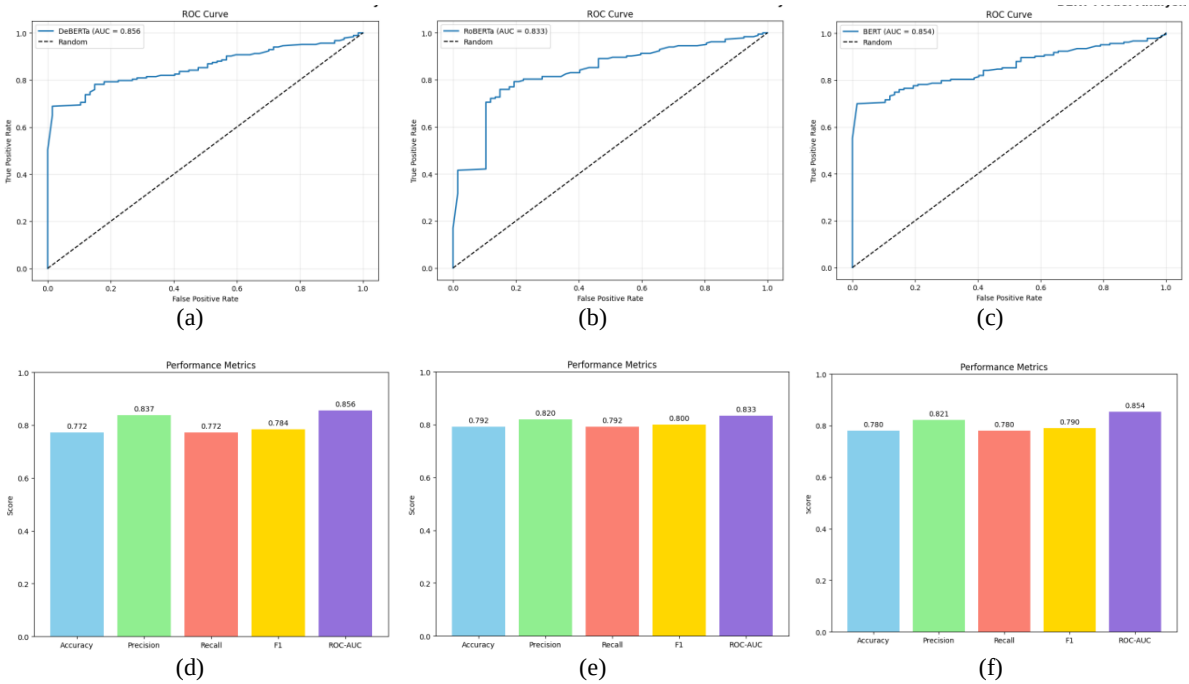
malicious (Figure 6(c)). RoBERTa showed comparatively improved performance, achieving more TPs with fewer FPs than BERT (Figure 6(a)), indicating a better balance between detection accuracy and false alarm reduction.



(a) (b) (c)

Figure 6. LLM's confusion matrix for (a) RoBERTa, (b) DeBERTa, and (c) BERT

These results highlight that while all models are effective at detecting malware, managing false positives (FPs) remains a key challenge for practical deployment (Figure 7 and Table 5). The RoBERTa model achieved strong discriminative performance with an AUC of 0.833 (Figure 7(a)), as further reflected in its performance metrics (Figure 7(d)). The DeBERTa model obtained the highest AUC of 0.856 (Figure 7(b)), benefiting from its disentangled attention mechanism and high recall, as shown in Figure 7(e). BERT followed closely with an AUC of 0.854 (Figure 7(c)), demonstrating strong and consistent classification capability (Figure 7(f)). BERT was the most computationally efficient model, requiring only 6 minutes to train, making it well suited for real-time environments (Figure 8). RoBERTa required approximately 82 minutes, offering improved performance at a higher computational cost suitable for GPU-enabled systems (Figure 8). DeBERTa had the highest training cost at 138 minutes, indicating its suitability for offline or infrequently retrained detection systems rather than real-time deployment (Figure 8). These results indicate that BERT is the most practical choice for real-time cryptojacking detection, as it can be trained and updated quickly under limited computational resources, while RoBERTa and DeBERTa are better suited for offline or batch-based detection where higher accuracy justifies longer training times (Figure 8).



(a) (b) (c)

(d) (e) (f)

Figure 7. LLMs overall results for (a, d) RoBERTa, (b, e) DeBERTa, and (c, f) BERT

Table 5. Performance comparison

Model	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	ROC-AUC (%)
DeBERTa	77	84	77	78	86
RoBERTa	79	82	79	80	83
BERT	78	82	78	79	85

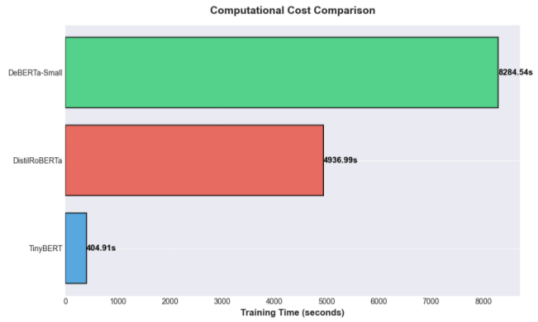
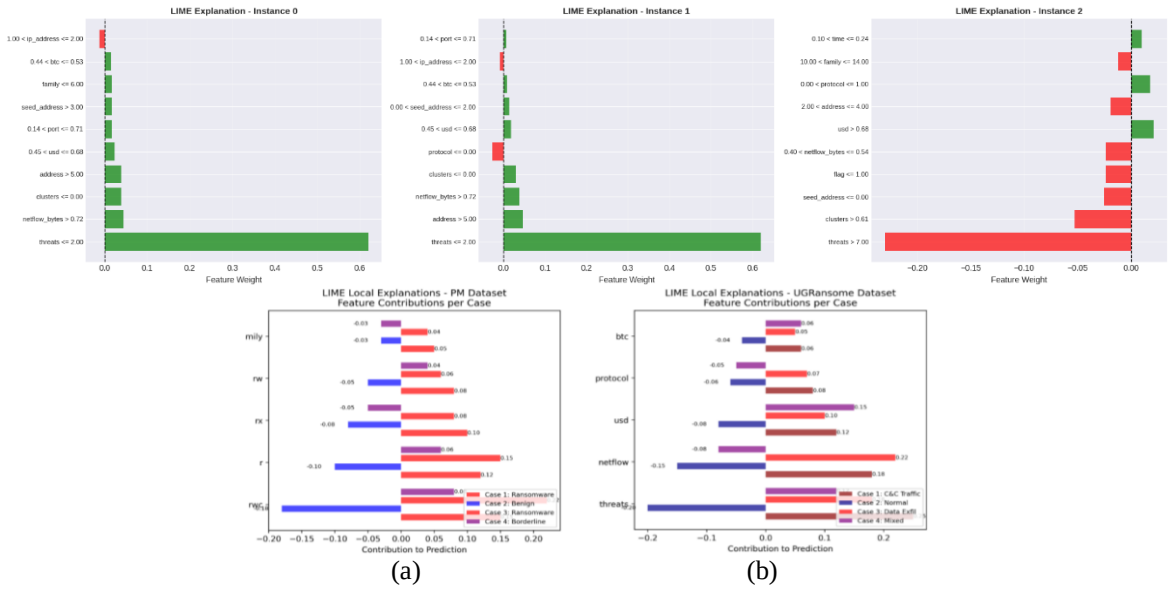


Figure 8. Computational cost

4.2. LIME explanation results

The LIME analysis on the UGRansome and PM datasets provides interpretable evidence of how the proposed model identifies cryptojacking intrusions. For both datasets, LIME highlights that predictions are strongly influenced by features associated with ransomware family labels, indicating that certain ransomware behaviors overlap with cryptojacking activity (Figure 9).

In particular, process memory access patterns, such as read–write (rw) and read–execute (rx) operations, emerge as dominant contributors (Figure 9(a)), reflecting the intensive and persistent memory usage required for illicit cryptocurrency mining [25]. Protocol indicators, including BTC, USD, and specific communication protocols, are also assigned high importance (Figure 9(b)), suggesting attempts to monetize compromised resources and maintain mining pool connectivity. Furthermore, network traffic (NetFlow) features are consistently emphasized in capturing abnormal outbound connections and sustained traffic volumes typical of cryptojacking campaigns (Figure 9(b)). Finally, addresses and threat indicators contribute to the model's decisions by linking observed behaviors to known malicious traffic. In general, the LIME results confirm that the model relies on semantically meaningful network traffic and PM features to distinguish cryptojacking from benign activity, thereby increasing trust and transparency in the detection process.

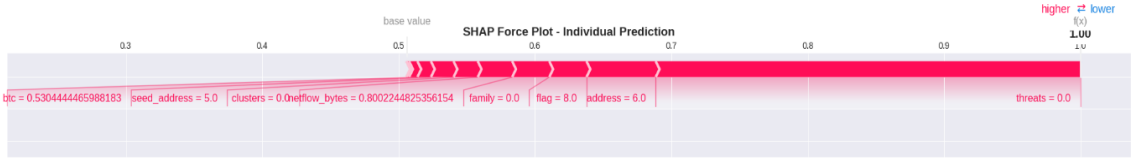


(a) (b)

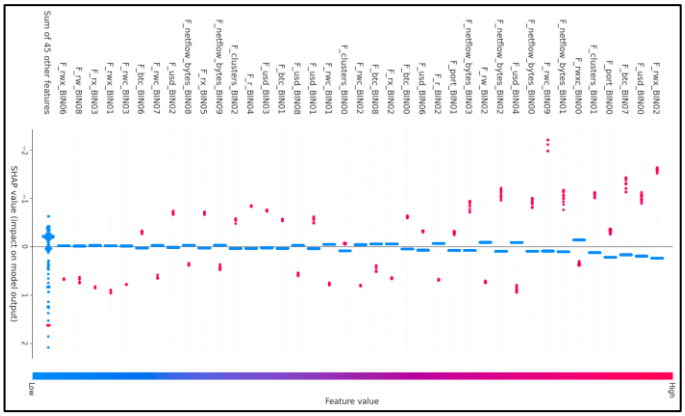
Figure 9. LIME results for (a) PM and (b) UGRansome data

4.3. SHAP explanation results

Figure 10 presents the SHAP explanation results for detecting cryptojacking. On the UGRansome dataset, the SHAP force plot shows that memory operations read–write–create (RWC), read–write–execute (RWX), BTC, cluster, and NetFlow features are the most influential factors in predicting individual cryptojacking (Figure 10(a)). Meanwhile, the SHAP summary plot on the PM dataset reveals a similar pattern of influential features at the global level (Figure 10(b)). Compared with LIME, which stresses RW, RX, BTC, protocol, USD, network traffic, addresses, and threats, SHAP not only confirms the importance of memory access patterns, cryptocurrency indicators, and network traffic, but also identifies higher-level features. In particular, RWC and clusters capture process creation behavior and aggregated activity patterns that LIME did not highlight (Figure 9), indicating that cryptojacking malware not only manipulates memory but also coordinates processes and network behaviors systematically. While LIME provides local and instance-specific explanations, SHAP offers a global perspective, revealing consistent behavioral signatures across the datasets. Results show that SHAP uncovers structural and behavioral patterns that complement LIME's local insights to enhance interpretability and trust in the detection of cryptojacking activity.



(a)



(b)

Figure 10. SHAP results for (a) UGRansome and (b) PM datasets

4.4. Comparative analysis with existing studies

Table 6 compares the findings with existing literature to highlight the contribution of this study [26]–[29]. It presents current detection approaches against the proposed hybrid host-based model, which surpasses state-of-the-art techniques in accuracy and precision, achieving more than 80% (Figure 11). In contrast, the attention-based LSTM model by Ma *et al.* [26] reported lower performance, while the proposed host-based method provides a more accurate solution for early cryptojacking detection. The model also outperforms Li *et al.* [27] approach, which achieved 89% precision using autoencoders (Table 6). The proposed host-based LIME matches or exceeds these metrics, demonstrating superior overall performance (Figure 11). Moreover, while Olayah *et al.* [28] and Abbasi *et al.* [29] reached 97% accuracy using Grey Wolf optimization and statistical code analysis; the proposed model improves interpretability [27]–[29]. These results support the hypothesis that host-based model-agnostic methods achieve high performance, similar to traditional ML models like RF and K-NN (Figure 11). Potential limitations include insufficient data quality, inadequate feature selection, computational cost, imbalanced datasets, or parameter tuning issues. Moreover, LLMs require complex processing time than RF or K-NN due to their deep architecture, high-dimensional embeddings, and sequential token processing, which increase inference complexity in real-world scenarios. Furthermore, model-agnostic host-based methods may be too complex or unsuitable for certain cryptojacking behaviors. As cryptojacking tactics evolve, performance may vary, highlighting the value of integrating advanced ML techniques for effective detection.

Table 6. Comparison of methods in literature

Reference	ML model	Method	Metric	Results	Limitations
[26]	DL	In-browser	MAPE	65%	Underfitting
[27]	Autoencoders	In-browser	Precision	89%	Traffic periodicity
[28]	Grey wolf	In-browser	Confidentiality	97%	XAI
[29]	Blacklisting	Host-based	Accuracy	97%	Multiple parameters
[30]	Ensemble learning	Host-based	F1-score	97%	SHAP
[31]	Sequential analysis	In-browser	Thresholds	-	Experimentation
[32]	Ensemble learning	Host-based	Accuracy	98%	Attacks categories
This study	Hybrid model	Host-based	ROC-AUC	93%	Datasets

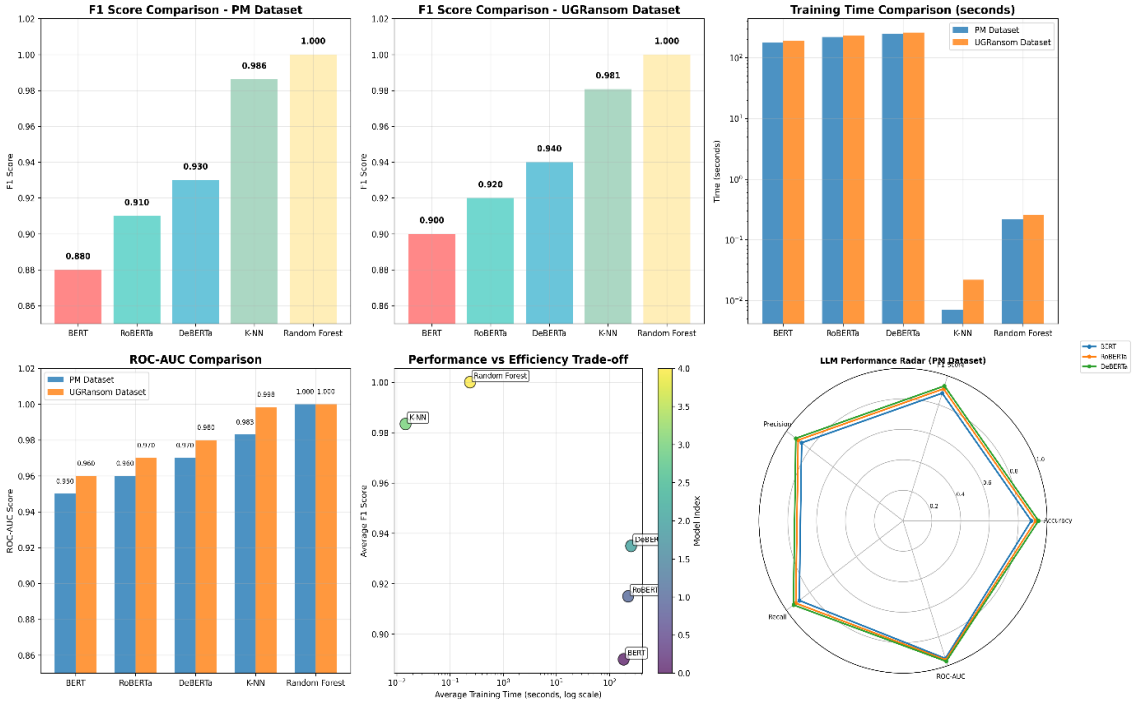


Figure 11. Comparative results

4.5. Discussion

This work explores the use of LLMs in combination with XAI techniques for detecting cryptojacking activities. While previous studies have applied various ML approaches to anomaly detection, very few have examined the explicit integration of LLMs with XAI methods to identify cryptojacking patterns [30]-[32]. The study findings indicate that LLMs can effectively capture normal transactional behavior and flag deviations associated with cryptojacking. Transformer-based architectures, with their capacity to manage complex datasets, were able to accurately distinguish cryptojacking states, while LIME and SHAP offered valuable insights into the key features driving these predictions. In particular, BTC incomes played a significant role in illicit transactions, and seed addresses were useful for predicting cryptojacking attacks [11], [33]. Despite demonstrating strong predictive capabilities, the approach faced certain constraints, such as the potential for overfitting and the limited availability of cryptojacking-specific datasets. Although the semi-supervised framework improves generalization by leveraging both labeled data, the small PM dataset size may restrict broader applicability. Additionally, while LIME and SHAP provide interpretability for individual feature contributions, they are limited in capturing interactions between features.

Compared with other DL models that prioritize metrics such as mean absolute percentage error (MAPE) or confidentiality, the hybrid model-agnostic approach offers a more balanced solution by combining high accuracy with explainability. Traditional ML models often sacrifice transparency for performance, but the inclusion of LIME and SHAP enhances trust by clarifying individual predictions. This interpretability is particularly important in blockchain environments, where understanding the rationale behind cryptojacking detection can inform risk mitigation strategies [33]. The semi-supervised nature of the proposed model further allows the use of both labeled and unlabeled datasets, increasing adaptability to real-

world scenarios where fully annotated data is scarce. This combination of interpretability, predictive performance, and flexibility positions host-based semi-supervised model-agnostic techniques as a promising solution for cryptojacking detection in blockchain systems. Future work should prioritize the expansion of cryptojacking datasets and the development of engineered features to improve generalization. Investigating methods to capture feature interactions more effectively and integrating advanced DL techniques could further enhance detection capabilities. Additionally, designing scalable solutions capable of handling large volumes of transaction data will be essential as cryptojacking methods continue to evolve. Addressing these challenges will strengthen the reliability of cryptojacking recognition in complex critical system's environments.

5. CONCLUSION

This study aimed to investigate how LLMs, specifically BERT, RoBERTa, and DeBERTa, can be effectively combined to detect and understand cryptojacking attacks using the UGRansome and PM datasets. The research question is addressed through a series of evaluations and analyses. The findings demonstrate that LLMs achieve commendable performance across various metrics, showcasing high accuracy and precision. The model exhibited moderate recall, indicating reasonable predictions. Integrating LIME and SHAP provided more profound insights into feature values and model predictions by enhancing the interpretability of LLM's results. Specifically, BTC incomes contributed to cryptojacking attacks, with seed addresses playing a crucial role in enabling timely interventions. The study supports the hypothesis that LLMs combined with XAI techniques offers a robust and interpretable approach to cryptojacking recognition. The finding's insights can lead to timely interventions and mitigation of the damage caused by malicious cryptomining activities. Future research should explore the scalability of LLMs to ensure comprehensive and practical cryptojacking recognition. In summary, employing hybrid ML models that integrate LLMs with XAI for cryptojacking detection marks a significant advancement in addressing zero-day exploits recognition.

FUNDING INFORMATION

This study was conducted without external financial support.

AUTHOR CONTRIBUTIONS STATEMENT

This journal adopts the Contributor Roles Taxonomy (CRediT) to acknowledge individual author contributions, minimize authorship disputes, and promote collaborative research.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Elodie Ngoie Mutombo		✓	✓	✓	✓	✓	✓	✓	✓		✓			✓
Mike Wa Nkongolo	✓	✓	✓	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓
Mahmut Tokmak		✓	✓	✓	✓	✓	✓	✓		✓	✓	✓		

C : **C**onceptualization
M : **M**ethodology
So : **So**ftware
Va : **Va**lidation
Fo : **Fo**rmal analysis
I : **I**nvestigation
R : **R**esources
D : **D**ata Curation
O : Writing - **O**riginal Draft
E : Writing - Review & **E**diting
Vi : **Vi**sualization
Su : **Su**pervision
P : **P**roject administration
Fu : **Fu**nding acquisition

CONFLICT OF INTEREST STATEMENT

The authors declare no conflict of interest.

DATA AVAILABILITY

The data supporting the findings of this study are included within the article.

REFERENCES

[1] Y. M. Gajmal, P. More, K. D. Kale, and A. Jagtap, "A survey on a novel cryptojacking covert attack," in *2nd International Conference on Intelligent Data Communication Technologies and Internet of Things, IDCIoT 2024*, Jan. 2024, pp. 359–364, doi: 10.1109/IDCIoT59759.2024.10467823.

[2] A. A. Alhashmi, A. A. Darem, A. B. Alshammari, L. A. Darem, H. K. Sheatah, and R. Effghi, "Ransomware early detection techniques," *Engineering, Technology and Applied Science Research*, vol. 14, no. 3, pp. 14497–14503, Jun. 2024, doi: 10.48084/etasr.6915.

[3] R. Su, "Generative mathematical models for ransomware attack prediction using chi-square feature selection for enhanced accuracy," *Signal, Image and Video Processing*, vol. 19, no. 10, p. 789, Oct. 2025, doi: 10.1007/s11760-025-04396-x.
[4] D. Shankar, G. V. S. George, J. N. S. S. Janardhana Naidu, and P. S. Madhuri, "Deep analysis of risks and recent trends towards network intrusion detection system," *International Journal of Advanced Computer Science and Applications*, vol. 14, no. 1, pp. 262–276, 2023, doi: 10.14569/IJACSA.2023.0140129.
[5] A. Rege and R. Bleiman, "A free and community-driven critical infrastructure ransomware dataset," in *Springer Proceedings in Complexity*, 2023, pp. 25–37.
[6] L. M. Kadhum, A. Firdaus, S. I. Hisham, W. Mushtaq, and M. F. Ab Razak, "Features, analysis techniques, and detection methods of cryptojacking malware: a survey," *JOIV : International Journal on Informatics Visualization*, vol. 8, no. 2, p. 891, May 2024, doi: 10.62527/joiv.8.2.2725.
[7] S. Alhazbi, A. Hussain, G. Oligeri, and P. Papadimitratos, "LLMs have rhythm: fingerprinting large language models using inter-token times and network traffic analysis," *IEEE Open Journal of the Communications Society*, vol. 6, pp. 5050–5071, 2025, doi: 10.1109/OJCOMS.2025.3577016.
[8] T. O. Adigun, A. Oshinubi, and I. Eweoya, "Cryptojacking detection using ML techniques," *The IUP Journal of Computer Sciences*, vol. 19, no. 1, pp. 7–17, Jan. 2025, doi: 10.71329/iupjcs/2025.19.1.7-17.
[9] H. Danesh, M. B. Karimi, and B. Arasteh, "CMShark: a NetFlow and machine-learning based crypto-jacking intrusion-detection method," *Intelligent Decision Technologies*, vol. 18, no. 3, pp. 2255–2273, Sep. 2024, doi: 10.3233/IDT-240319.
[10] H. Cao *et al.*, "Unveiling the Superiority of Unsupervised Learning on GPU cryptojacking detection: practice on magnetic side channel-based mechanism," *IEEE Transactions on Information Forensics and Security*, vol. 20, pp. 4874–4889, 2025, doi: 10.1109/TIFS.2025.3563069.
[11] O. M. K. Alhawi, J. Baldwin, and A. Dehghantanha, "Leveraging machine learning techniques for windows ransomware network traffic detection," in *Advances in Information Security*, vol. 70, 2018, pp. 93–106.
[12] S. H. Khan *et al.*, "A new deep boosted CNN and ensemble learning based IoT malware detection," *Computers and Security*, vol. 133, p. 103385, Oct. 2023, doi: 10.1016/j.cose.2023.103385.
[13] Q. V. Liao and K. R. Varshney, "Human-centered explainable AI (XAI): from algorithms to user experiences," *arXiv preprint arXiv:2110.10790*, 2021, [Online]. Available: http://arxiv.org/abs/2110.10790.
[14] S. M. Kasongo, "A deep learning technique for intrusion detection system using a recurrent neural networks based framework," *Computer Communications*, vol. 199, pp. 113–125, Feb. 2023, doi: 10.1016/j.comcom.2022.12.010.
[15] N. Capuano, G. Fenza, V. Loia, and C. Stanzione, "Explainable artificial intelligence in cybersecurity: a survey," *IEEE Access*, vol. 10, pp. 93575–93600, 2022, doi: 10.1109/ACCESS.2022.3204171.
[16] K. Zhang, Y. Wang, U. A. Bhatti, Y. Zhou, and M. Jin, "Enhanced ransomware attacks detection using feature selection, sensitivity analysis, and optimized hybrid model," *Journal of Big Data*, vol. 12, no. 1, p. 245, Nov. 2025, doi: 10.1186/s40537-025-01289-1.
[17] V. Mokoma and A. Singh, "RanViz: ransomware visualization and classification based on time-series categorical representation of API calls," *IEEE Access*, vol. 13, pp. 56237–56254, 2025, doi: 10.1109/ACCESS.2025.3555163.
[18] H. Baek, M. Lee, and H. Kim, "CryptoLLM: harnessing the power of LLMs to detect cryptographic API misuse," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 14982 LNCS, 2024, pp. 353–373.
[19] A. Sagheer and M. Kotb, "Unsupervised pre-training of a Deep LSTM-based stacked autoencoder for multivariate time series forecasting problems," *Scientific Reports*, vol. 9, no. 1, p. 19038, Dec. 2019, doi: 10.1038/s41598-019-55320-6.
[20] A. Afkari-Fahandari, E. Shabaninia, F. Asadi-Zeydabadi, and H. Nezamabadi-Pour, "A comprehensive survey of transformers in text recognition: techniques, challenges, and future directions," *ACM Computing Surveys*, vol. 58, no. 5, pp. 1–42, Apr. 2026, doi: 10.1145/3771273.
[21] S. Mahendru and T. Pandit, "SecureNet: a comparative study of DeBERTa and large language models for phishing detection," in *2024 IEEE 7th International Conference on Big Data and Artificial Intelligence (BDAI)*, Jul. 2024, pp. 160–169, doi: 10.1109/BDAI62182.2024.10692765.
[22] J. Nicholls, A. Kuppa, and N. A. Le-Khac, "Large language model XAI approach for illicit activity investigation in bitcoin," *Neural Computing and Applications*, vol. 37, no. 30, pp. 24869–24881, 2025, doi: 10.1007/s00521-024-10510-w.
[23] N. Nayyer, N. Javaid, M. Akbar, A. Aldegheishem, N. Alrajeh, and M. Jamil, "A new framework for fraud detection in bitcoin transactions through ensemble stacking model in smart cities," *IEEE Access*, vol. 11, pp. 90916–90938, 2023, doi: 10.1109/ACCESS.2023.3308298.
[24] F. Bourebaa and M. Benmohammed, "Evaluating lightweight transformers with local explainability for android malware detection," *IEEE Access*, vol. 13, pp. 101005–101026, 2025, doi: 10.1109/ACCESS.2025.3577775.
[25] H. H. Kao, "Accelerating Multilingual Cryptocurrency Forensics: An NLP-driven approach for efficient mnemonic identification," *IEEE Access*, vol. 13, pp. 10513–10526, 2025, doi: 10.1109/ACCESS.2025.3528829.
[26] R. Ma, Y. Zhan, T. Yan, Y. Xia, and Y. Ali, "Interless: interference-aware deep resource prediction for serverless computing," in *Proceedings of the 36th Chinese Control and Decision Conference, CCDC 2024*, May 2024, pp. 3783–3788, doi: 10.1109/CCDC62350.2024.10588201.
[27] R. Li *et al.*, "SeIoT: detecting anomalous semantics in smart homes via knowledge graph," *IEEE Transactions on Information Forensics and Security*, vol. 19, pp. 7005–7018, 2024, doi: 10.1109/TIFS.2024.3428856.
[28] F. Olayah *et al.*, "An efficient lightweight crypto security module for protecting data transmission through IoT based electronic sensors," *Journal of Nanoelectronics and Optoelectronics*, vol. 19, no. 6, pp. 646–657, Jun. 2024, doi: 10.1166/jno.2024.3609.
[29] M. H. K. Abbasi, S. Ullah, T. Ahmad, and A. Buriro, "A real-time hybrid approach to combat in-browser cryptojacking Malware," *Applied Sciences (Switzerland)*, vol. 13, no. 4, p. 2039, Feb. 2023, doi: 10.3390/app13042039.
[30] N. Nayyer, N. Javaid, M. Akbar, A. Aldegheishem, N. Alrajeh, and M. Jamil, "A new framework for fraud detection in bitcoin transactions through ensemble stacking model in smart cities," *IEEE Access*, vol. 11, pp. 90916–90938, 2023, doi: 10.1109/ACCESS.2023.3308298.
[31] N. Gaidamakin, "Sequential analysis of cryptojacker sings based on the wald criterion," in *2024 IEEE Ural-Siberian Conference on Biomedical Engineering, Radioelectronics and Information Technology (USBEREIT)*, May 2024, vol. 11, pp. 84–87, doi: 10.1109/USBEREIT61901.2024.10584011.
[32] Y. Wang *et al.*, "A detection method against selfish mining-like attacks based on ensemble deep learning in IoT," *IEEE Internet of Things Journal*, vol. 11, no. 11, pp. 19564–19574, Jun. 2024, doi: 10.1109/JIOT.2024.3367689.
[33] M. Caprolu, S. Raponi, G. Oligeri, and R. Di Pietro, "Cryptomining makes noise: detecting cryptojacking via machine learning," *Computer Communications*, vol. 171, pp. 126–139, Apr. 2021, doi: 10.1016/j.comcom.2021.02.016.

BIOGRAPHIES OF AUTHORS

Elodie Ngoie Mutombo is currently pursuing her master's degree in Computer Science at the University of Pretoria, South Africa, under the supervision of Dr. Mike Wa Nkongolo. She holds a B.Sc. Honors degree in Computer Science from the University of Pretoria. Her research areas include blockchain security, XAI, machine learning, and data science. She can be contacted at email: u22608754@tuks.co.za.

Mike Wa Nkongolo is a senior lecturer in the Department of Informatics at the University of Pretoria. He received his HDip, B.Sc. Honors, and M.Sc. degrees in Computer Science from the University of the Witwatersrand, and the Ph.D. degree in Information Technology from the University of Pretoria, in South Africa. He has authored or co-authored various publications and serves as a reviewer for several international journals. Dr. Mike Nkongolo Wa Nkongolo is a member of IEEE and the South African Institute of Computer Scientists and Information Technologists (SAICSIT) with research interests in cybersecurity, AI, machine learning, and natural language processing. He can be contacted at email: mike.wankongolo@up.ac.za.

Mahmut Tokmak is an associate professor in the Department of Management Information Systems at Bucak Zeliha Tolunay School of Applied Technology and Management, Burdur Mehmet Akif Ersoy University, Turkey. He holds a Ph.D. degree in Computer Engineering. His research interests include AI, machine learning, and malware analysis. He can be contacted at email: mahmuttokmak@mehmetakif.edu.tr.