Minimax Theory and its Applications

Volume 11 (2026), No. 2s, 713 to 733

Adaptive Deep Representation Learning with Counterfactual Feature Attribution for Explainable Ransomware Detection in Class-Imbalanced Malware Environments

G Badrinath[1], Dr.Alampally Sreedevi[2]

[1]Research Scholar, Department of Computer Science and Engineering, Koneru Lakshmaiah Education Foundation, Hyderabad-500075, Telangana, India Email: badrinath.goteti@gmail.com
[2]Assistant Professor, Department of Computer Science and Engineering, Koneru Lakshmaiah Education Foundation, Moinabad Road, Aziznagar, Hyderabad, Telangana, India – 500075, Email: a.sreedevi@klh.edu.in

Abstract
The rapid evolution of ransomware has emerged as a critical cybersecurity challenge, demanding intelligent detection frameworks capable of accurately distinguishing malicious executables from benign software. Existing machine learning and deep learning approaches often exhibit degraded performance under class-imbalanced data, resulting in biased learning, increased false negatives, and limited model interpretability. To address these limitations, this study proposes an Adaptive Class-Sensitive Learning-enabled TabNet (ACSL–TabNet) framework that dynamically adjusts class weights during training while exploiting TabNet's sequential attention mechanism to learn highly discriminative representations from static Portable Executable (PE) header and behavioral features. The proposed model is implemented using Python 3.11 in the PyTorch framework with TabNet, Scikit-learn, and Captum libraries for explainability. Experiments were conducted on the Ransomware Dataset 2024 containing 21,752 executable samples from 26 malware families, including 10,876 ransomware and 10,876 benign files. The proposed framework achieved 98.8% accuracy, 98.84% recall, 98.78% F1-score, and 99.63% AUC, outperforming existing methods such as LGBM (91.0%), XGBoost (93.0%), Bi-LSTM with CNN (98.0%), and SB-BR-STM CNN (98.5%), yielding an accuracy improvement of 0.3% over the strongest baseline and 7.8% over conventional machine learning models. Furthermore, the integrated Counterfactual and Integrated Attribution-Based Explainability (CIAE) module provides both feature attribution and decision-boundary explanations, enhancing the transparency and trustworthiness of ransomware predictions. The experimental results demonstrate that the proposed ACSL–TabNet framework offers a robust, adaptive, and interpretable solution for accurate ransomware detection in imbalanced cybersecurity environments.

Keyword: Ransomware Detection, TabNet, Adaptive Class-Sensitive Learning, Class-Imbalanced Learning, Explainable Artificial Intelligence.

1. Introduction

The increasing proliferation of digital technology and interconnected computing environment has led to a marked rise in the prevalence of ransomware attacks that pose a grave threat to individuals, businesses[1], and infrastructure[2],[3]. In addition to encrypting data, ransomware has now adopted sophisticated methods to evade traditional signature-based protection mechanisms[4],[5]. Therefore, smart malware detection systems using artificial intelligence have received widespread focus due to their ability to detect complex patterns from PE and behavior features[6],[7]. Machine learning and deep learning models like decision trees ensemble[8], convolutional neural networks[9], recurrent neural networks[10], and combination of various models have shown potential in detecting such malwares[11]. However, their effectiveness is compromised by the presence of class imbalanced malware dataset in which most of the samples belong to the majority class[12],[13].

This drawback limits the detection accuracy, increases false negative rate, provides poor representation of features, and reduces model interpretability[14],[15].To overcome these challenges, this study proposes an adaptive deep representation learning framework with counterfactual feature attribution for explainable ransomware detection in class-imbalanced malware environments. The framework integrates complementary static PE header features and behavioral characteristics to construct a comprehensive malware representation capable of capturing diverse ransomware patterns. An adaptive class-sensitive learning strategy dynamically

optimizes the learning process by emphasizing difficult minority-class samples during training, thereby reducing class bias and improving detection robustness. Subsequently, a deep feature learning architecture extracts highly discriminative representations through sequential feature selection, enabling effective suppression of redundant attributes while preserving informative malware characteristics. Furthermore, a counterfactual and attribution-based explainability mechanism provide transparent prediction reasoning by identifying influential features and the minimal changes required to alter detection outcomes.

1.1Problem Statement

The evolution of ransomware attack techniques has revealed several drawbacks in the use of traditional malware detection systems, especially in those systems where the data distribution is class-imbalanced[16]. Current ransomware detection techniques may experience degraded performance when using class-imbalanced data sets, which causes low detection rates for minority ransomware samples and false negatives[17]. Moreover, traditional models are based on static class weighting approaches, and offer limited interpretability, which are less suited for the changing nature of ransomware attacks. The shortcomings require an adaptable and explainable detection system that can enhance detection accuracy in realistic cybersecurity environments.

1.2 Research Motivation

Majority of the conventional machine learning and deep learning models usually learn the majority class in their training process and consequently have poor detection rates for minority classes and high false negative rates. Many of these models adopt static class balancing mechanisms which cannot cope with learning dynamics, while insufficient utilization of static Portable Executable (PE) header and behavior attributes hinders effective extraction of malware representations. The other significant issue lies in the fact that the model lacks transparency and hence the decisions made by the system cannot be explained and validated by security experts. Thus, the need for an adaptive, interpretable and imbalance-sensitive ransomware detection system arises.

1.3 Significance of the Study

In this study, an adaptive and explainable framework for detecting ransomware is proposed, which is built on a dynamic class-sensitive learning approach, feature representation using TabNet, and enhanced explainability. The suggested method can improve ransomware detection in class-imbalanced scenarios, decrease false-negative rates and increase transparency of decisions. It can offer reliable and interpretable predictions which can be applied in real-world cybersecurity monitoring and intelligent threat analysis.

1.4 Key Contribution

- Proposes an Adaptive Class-Sensitive Learning (ACSL) module that dynamically adjusts class weights during training to mitigate class imbalance, thereby improving ransomware detection performance and reducing false negatives without relying on fixed weighting or sampling techniques.
- Develops a unified ransomware feature representation by integrating static PE header attributes with dynamic behavioral features, enabling comprehensive learning of both structural and runtime characteristics associated with ransomware attacks.
- Employs the TabNet architecture for deep feature learning, utilizing sequential attention and automatic feature selection to identify the most discriminative ransomware features while eliminating redundant information, thereby enhancing detection accuracy and model generalization.
- Introduces a Counterfactual and Integrated Attribution-Based Explainability (CIAE) module that combines feature attribution with counterfactual reasoning to provide transparent, interpretable, and actionable explanations for ransomware detection decisions, improving the framework's trustworthiness for cybersecurity applications.

The rest of this paper is organized as follows. Section 2 reviews recent studies on ransomware detection and their limitations. Section 3 presents the proposed methodology. Section 4 discusses the experimental results and comparative analysis. Finally, Section 5 concludes the study and outlines potential directions for future research.

2. Literature Review

Davidian et al.[18] suggested the use of a ransomware detection framework based on behavioural API call sequences, and conducted a systematic analysis of the API features, context-window sizes and machine learning models. The study succeeded in achieving better detection of zero-day ransomware, and the combination of the context-window size of seven and CNN–LSTM models with API operation results was the most effective one

for real-time classification. But the way it works is based on API behavior which makes it pretty weak when it comes to dealing with advanced evasion methods and varying real-world conditions. Herrera-Silva and Hernández-Álvarez[19] presented a dynamic ransomware detection system based on sandbox-based behavioural analysis and machine learning techniques with selected dynamic features. The study managed to achieve an accuracy of more than 99% in detection rate with Gradient Boosted Trees, Random Forest, and Neural Networks and released a public dynamic feature dataset. This evaluation, however, was based on a small number of samples of ransomware and good ware, which may limit the generalizability to new variants of ransomware in the wild.

Gulmez et al.[20] introduced XRan, which is an explainable ransomware detection framework that combines several dynamic behavioural sequences with a CNN model and XAI techniques (LIME and SHAP). The study reported 99.4% true positive rate and outperformed the state-of-the-art methods with the interpretable detection results. The framework, however, relies on dynamic analysis, which comes with execution overhead and could be susceptible to advanced evasion techniques used by malware.

Deng et al.[21] proposed a static ransomware detection framework that integrates features of the Portable Executable (PE) header and deep reinforcement learning to detect ransomware early. The study successfully identified hidden ransomware without having to open or execute malicious files, which not only decreases the likelihood of infection but also helps maximize the efficiency of the study. Using PE header information, however, may not always be accurate against obfuscated, packed, or very sophisticated ransomware variants.

Cen et al.[22] suggest an early ransomware detection framework named RansoGuard which detects ransomware by analysing pre-encryption API call sequences using natural language processing and an RNN classifier. The study has an impressive recall of 96.18% and an accuracy of 94.26%, which shows the effectiveness of the zero-day ransomware detection. The framework is based on dynamic behaviour analysis of APIs, though, which can add to computational burden and still be vulnerable to sophisticated ransomware evasion methods.

Hussain et al.[23] suggested a transfer learning approach with optimized hyperparameters that leverages BERT and RoBERTa for identifying and categorizing IoT ransomwares based on dynamic API call sequence analysis. The study was able to detect ransomware families with 95.60% accuracy using BERT and 94.39% accuracy using RoBERTa. The approach, however, requires dynamic analysis in the sandbox which adds to the computational burden and restricts deployment in resource-constrained IoT conditions.

Miraoui and Belgacem[24] presented a machine learning and deep learning approach to classify Portable Executable (PE) malware in two public datasets of binary and multiclass malware. The study was successful in effective malware classification using size of dataset and feature dimensions to identify appropriate algorithms. But the framework is more about general PE malware classification than ransomware detection, and doesn't cover real-time behavioural analysis.

Gazzan et al.[25] introduced a multi-phase ransomware detection method that combines BGM-GAN to augment the data, IMIS to adaptively select features, and a Deep Belief Network optimized using UA-DES. The study's 98.6% detection accuracy was accompanied by a considerable decrease in false positives. The framework is however, very complex and demands extensive training, making it impractical for being deployed in resource constrained real-time applications.

Table 1. Summary of existing studies

Author	**Method**	**Advantage**	**Limitations**
Davidian et al.[18]	Behavioural ransomware detection using API call sequences with CNN and LSTM models	Improves zero-day ransomware detection; API operation results enhance classification accuracy; suitable for real-time detection	Relies on dynamic API behaviour; vulnerable to evasion techniques and unseen execution patterns
Herrera-Silva and Hernández-Álvarez[19]	Sandbox-based dynamic feature extraction with Gradient Boosted Trees, Random Forest, and Neural Networks	Achieves over 99% accuracy and provides a publicly available dynamic feature dataset	Evaluated on a limited number of ransomware and goodware samples, affecting generalizability
Gulmez et al.[20]	XRan framework combining multi-view dynamic analysis, CNN, LIME, and SHAP	High detection accuracy (99.4% TPR) with explainable ransomware	Dynamic analysis incurs computational overhead and may be bypassed by

		classification	advanced malware evasion
Deng et al.[21]	Static ransomware detection using Portable Executable (PE) header features with Deep Reinforcement Learning	Enables rapid early detection without executing ransomware, reducing infection risk	PE header features alone may be insufficient against obfuscated or packed ransomware
Cen et al.[22]	RansoGuard using API call sequences, NLP feature extraction, and RNN classifier	Provides early ransomware detection with strong recall (96.18%) and robustness against zero-day attacks	Requires dynamic execution and API monitoring, increasing computational cost
Hussain et al.[23]	Hyperparameter-optimized BERT and RoBERTa models using dynamic API call sequences	Effectively classifies ransomware families with high accuracy (95.60% using BERT) and supports IoT healthcare security	Depends on sandbox-based dynamic analysis and resource-intensive transformer models
Miraoui and Belgacem[24]	Binary and multiclass PE malware classification using machine learning and deep learning on multiple datasets	Compares multiple algorithms across datasets, improving malware classification reliability	Focuses on general PE malware rather than ransomware and lacks behavioural analysis
Gazzan et al.[25]	Multi-phase framework using BGM-GAN, IMIS, and Deep Belief Network with UA-DES optimization	Achieves 98.6% detection accuracy while reducing false positives	High computational complexity and extensive training requirements reduce real-time deployment feasibility

Table 1 shows that the current ransomware detection techniques are able to obtain high accuracy in static, dynamic, deep learning and transformer-based methods. Most, however, are based on computationally-intensive dynamic analysis, on small data sets or on only a single-feature representation, limiting scalability and generalisation. Moreover, explainability, timely detection, lightweight deployment, and robustness to future zero-day ransomware are under-studied, which calls for the creation of more effective and adaptive detection methods.

3.Proposed Adaptive Class-Sensitive TabNet Framework for Explainable Ransomware Detection

The proposed methodology introduces an adaptive deep learning framework that is able to detect ransomware attacks with high accuracy and explainability by combining both static Portable Executable (PE) header information and dynamic behavioural characteristics. The collected ransomware dataset is first pre-processed by eliminating duplicate records and irrelevant attributes, addressing missing data, encoding categorical data, normalizing numerical features, and dividing the data into training, validation, and testing sets. A single feature representation is then built by combining static and behavioural features and, finally, imbalance-aware training sets are created to emulate real cybersecurity scenarios. The proposed Adaptive Class-Sensitive Learning module is designed to dynamically change the class weights while training in order to improve minority ransomware detection. The optimized features are then learnt by the TabNet architecture to obtain highly discriminative features. Last but not least, a ransomware detection layer predicts benign or ransomware samples, while the Counterfactual and Integrated Attribution-Based Explainability module generates interpretable predictions by identifying influential features and decision-altering factors, making it more useful and applicable for cybersecurity analysis while increasing model transparency. The workflow of proposed Adaptive Class-Sensitive TabNet Framework for Explainable Ransomware Detection is shown in figure 1.

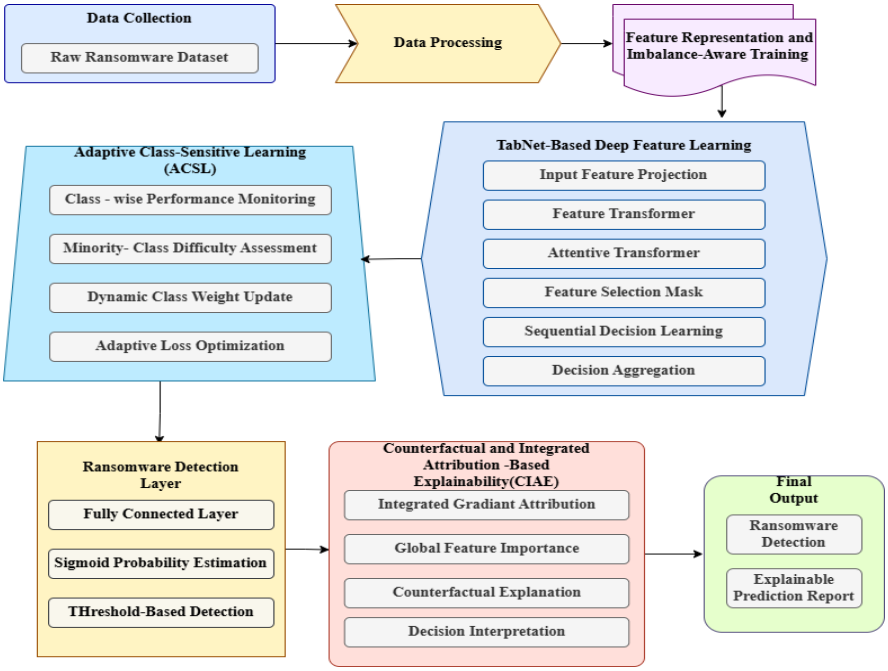


Figure 1. Workflow of proposed Adaptive Class-Sensitive TabNet Framework for Explainable Ransomware Detection

3.1 Data Collection

This proposed study involves using the Ransomware Dataset 2024, a benchmark dataset that is available in the public domain from Kaggle[26]. The dataset consists of 21,752 Portable Executable (PE) files, with an equal number of malicious and benign samples, which is 10,876 each, ensuring a balanced distribution to achieve robust model training. It contains 26 malware families divided into four main classes of malware, including 11 high-profile ransomware families: Cerber, DarkSide, Dharma, GandCrab, LockBit, Maze, Phobos, REvil, Ragnar Locker, Ryuk, Shade, and WannaCry. The static PE header attributes, behavioural indicators (registry, network, process, file, DLL and API activity), and metadata (Class, Category and Family labels) are all present in each sample. The proposed framework has rich structural and behavioral characteristics which allow it to learn discriminative ransomware patterns and accurately detect ransomware in various variants of ransomware.

3.2 Data Preprocessing

The pre-processing of the collected ransomware data is done to enhance the quality of the data, remove inconsistencies, and convert the features into a uniform format for deep learning. This stage is important for ensuring reliable learning, as it minimizes noise, retains meaningful information, and prepares the data for efficient feature extraction and ransomware detection.

3.2.1 Duplicate Record and Irrelevant Attribute Removal

To enhance data quality and model efficiency, the raw data is first cleaned by eliminating duplicate records and attributes that do not provide information. The redundant samples are removed by comparing all features in the data set to find the unique ones and discarding the duplicates, avoiding redundant learning, training bias and enhancing the generalization ability of the model without changing the original class distribution. Later, irrelevant attributes like MD5 and SHA1 hash values are dropped since they only inform about the uniqueness of the file and not about the ransomware behaviour or executable structure. By eliminating these attributes, feature dimensionality is reduced, computational complexity is lowered, and the proposed framework can be applied to informative static and behavioural attributes that can directly impact accurate ransomware detection.

3.2.2 Missing Value Analysis and Treatment

The data set is analyzed for any missing or null attributes in order to assure completeness of data. The numerical data is filled with the median of the corresponding feature as it is less affected by outliers, while the categorical data is filled with the most frequent category. This pre-processing step makes sure that there is no loss of information and that the proposed model is more robust.

3.2.3 Categorical Feature Encoding

Numerous attributes of the datasets, such as PEType, MachineType, Subsystem, Category, and Family, are categorical data types that are not easily useful to deep learning models. The label encoding maps each unique category to a different number, converting these categorical attributes into numbers.

3.2.4 Numerical Feature Normalization Using Min–Max Scaling

The numerical attributes have different value ranges, potentially impacting model convergence. The Min–Max normalization is therefore performed on all numerical features, so that each feature value falls between the range 0-1 and the relative distribution remains the same as in the original data. This normalization provides stability in training and faster convergence of the deep learning model. The Min–Max normalization is shown in equation (1).

$$x' = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \quad (1)$$

Where $x$is the original feature value, $x_{\min}$and $x_{\max}$denote the minimum and maximum values of the corresponding feature, and x'represents the normalized value.

3.2.5 Train–Validation–Test Split

The processed data is then split into three portions for model building and testing. The training set is used to learn feature representations; the validation set is used for hyperparameter tuning and to avoid overfitting and the testing set is used to evaluate the performance unbiasedly. The dataset is split into 70% training, 15% validation and 15% testing and each of these sets is equally distributed in terms of class distribution to ensure reliable ransomware detection.

3.3 Feature Representation and Imbalance-Aware Training Data Construction

The preprocessed data is then cleaned and converted into a common feature representation that can be used for ransomware detection. The proposed framework combines complementary static and behavioural features to achieve a single discriminative feature space construction and generates imbalance aware training subsets. The study proposed in this study is designed to induce class imbalance in training data to enhance the ability of the detection model to be robust and adaptively learnable in a realistic environment of ransomware attack, and not simply by direct training with imbalanced datasets.

3.3.1 Static Feature Representation

Static analysis is an analysis of the actual code of the executable without running it. Portable Executable (PE) header attributes contain information that helps determine whether software is benign or ransomware, such as: Entry Point, Image Base, Size of Code, Section Alignment, DLL Characteristics, Machine Type, Subsystem and more. These features do not change during execution and are a good indicator of the variations in the organization of an executable introduced by the different ransomware families. It is represented statically as shown in equation (2).

$$F_{static} = [f_1, f_2, \dots, f_m] \quad (2)$$

Where F_{static}denotes the static feature vector, $f_i$represents the i^{th}static attribute extracted from the executable file, and $m$denotes the total number of selected static features.

3.3.2 Behavioural Feature Representation

Behavioural analysis is used to monitor the activities of executable files at runtime to detect malicious activities that are not evident from the static attributes of the file. The proposed framework is based on the registry operations, network communications statistics, frequency of API invocations, process behaviour, file system activities and DLL function calls present in the data-set. These attributes are dynamic and describe ransomware behavior like registry changes, unusual network activity, abnormal process creation and unauthorized file encryptions. The behaviour is represented as shown in equation (3).

$$F_{behavior} = [b_1, b_2, \dots, b_n] \quad (3)$$

Where $F_{behavior}$denotes the behavioural feature vector, $b_i$represents the i^{th}behavioural attribute, and $n$denotes the total number of behavioural features.

3.3.3 Unified Feature Representation

The proposed framework integrates the two kinds of information, structure and behavior, to get a complete description of each executable. Combining both feature groups allows the model to learn the characteristics of the executable as well as malicious activities during runtime, which enhances the ransomware detection capability against ransomware and other malware families. The representation of features in a unified way is given by equation (4).

$$F_u = F_{static} \oplus F_{behavior} \quad (4)$$

Where $F_u$denotes the unified feature vector, $\oplus$represents the concatenation operation, $F_{static}$is the static feature vector, and $F_{behavior}$is the behavioural feature vector.

3.3.4 Imbalance-Aware Training Data Construction

A big challenge in ransomware detection is that the number of benign or malicious samples in a real cyber security environment is unequal. The proposed study proposes a novel strategy, called Imbalance-Aware Training Data Construction, as a novel contribution other than relying on naturally balanced datasets. Controlled imbalance ratios are created only for the training set, while the validation and testing sets are kept as is, which means that they do not have controlled imbalance ratio.This imbalance ratio is shown in equation (5).

$$IR = \frac{N_{majority}}{N_{minority}} \quad (5)$$

Where IRdenotes the imbalance ratio, $N_{majority}$represents the number of majority-class samples, and $N_{minority}$represents the number of minority-class samples in the training dataset.

This approach allows the following Adaptive Class-Sensitive Learning module to adapt robust decision boundaries to the different prevalence of ransomware, thus enhancing the detection accuracy of minority ransomware samples and keeping high accuracy for benign applications. The proposed imbalance-aware training mechanism is more realistic in terms of cybersecurity deployments, as the existing studies mainly focus on a balanced dataset, improving the practical applicability of the proposed ransomware detection framework.

3.4 Adaptive Class-Sensitive Learning

The proposed Adaptive Class-Sensitive Learning module is introduced to remedy the learning bias which results from training ransomware on imbalanced ransomware data. ACSL adapts the learning priority dynamically, based on the performance of the model on each class in every training epoch, unlike the traditional fixed weight versions. The proposed framework can be used to enhance the ransomware detection capability especially for minority ransomware samples without having an impact on balanced learning for benign files and without increasing the false-negative predictions.

3.4.1 Dynamic Class Weight Adjustment

The key novelty of the proposed study is to dynamically update weights of the class during training rather than setting the weights before training the model. With the ransomware detection becoming more challenging, ACSL automatically enhances the ransomware sample contribution to the learning process. On the other hand, if ransomware recall improves, the weight slowly reduced to prevent too much bias towards the minority class and retain the model generalization. The class weight that is adaptive is represented by equation (6).

$$W_c^{(t+1)} = W_c^{(t)} + \lambda D_c \quad (6)$$

Where $W_c^{(t)}$denotes the class weight at the current training epoch, $W_c^{(t+1)}$represents the updated class weight, $D_c$is the learning difficulty of class c, and $\lambda$is the adaptive learning adjustment factor controlling the weight update.

3.4.2 Adaptive Loss Optimization

The updated class weights are now included in the loss function to make the most difficult ransomware samples have a larger impact during the optimization. As a result, the model focuses more on instances of ransomware that are often misclassified, and continues to learn without any changes to benign samples. This adaptive optimization reduces the false negative rate, and enhances the stability of ransomware detection in different imbalance levels. The adaptive weighted loss is given by the following equation (7).

$$L = -\frac{1}{N}\sum_{i=1}^{N} W_{y_i} \log\left(P_{y_i}\right) \quad (7)$$

Where Ldenotes the weighted classification loss, Nrepresents the total number of training samples, $W_{y_i}$is the adaptive weight assigned to the true class of the i^{th}sample, and P_{y_i}denotes the predicted probability of the corresponding true class.

The proposed ACSL module is the main methodological contribution of this study, as it facilitates dynamic class-sensitive optimization, which evolves and adapts to the changing learning difficulties over the training set. The proposed ransomware detection framework is different than the current approach of fixed class weighting and static sampling techniques, which have a high false-negative rate, by providing better minority ransomware detection, lower false-negative predictions, and a more practical and reliable detection framework in realistic imbalanced cybersecurity environments.

3.4.3TabNet-Based Deep Feature Learning

The feature matrix optimized for the ACSL is then passed to the TabNet-based Deep Feature Learning module for learning high-level discriminative representations of the features for ransomware detection. TabNet sequentially learns the most informative static and behavioural features based on an attention-guided learning mechanism instead of treating all these features equally. This allows the proposed framework to learn complex relationships between the executable characteristics while removing redundant information, which improves the detection accuracy and allows the learning of the features in an interpretable manner without manual feature engineering. The architecture of the TabNet-Based Deep Feature Learning is shown in figure 2.

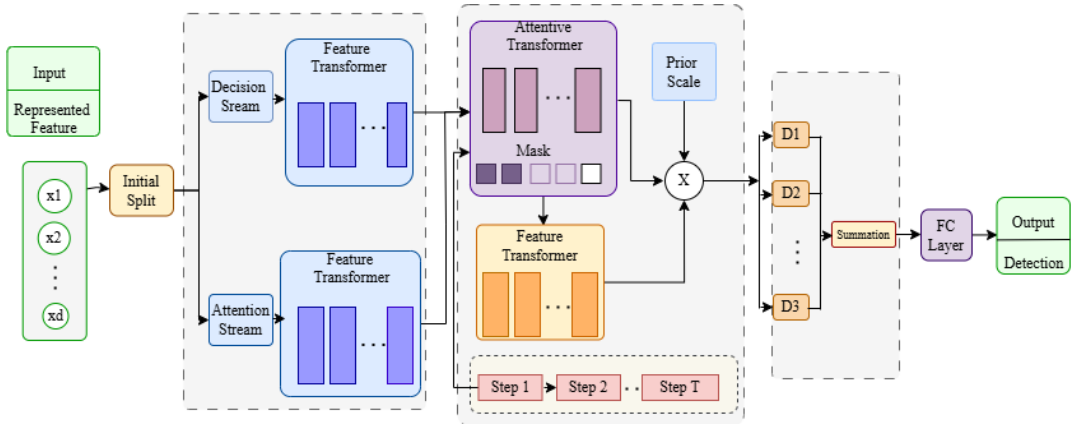


Figure 2. Architecture of the TabNet-Based Deep Feature Learning

The optimized feature vectors produced by the ACSL module are initially converted to the TabNet feature space. This projection maps out a variety of descriptive properties into a common latent space for sequential decision learning. The input feature projection is shown as the equation (8).

$$H_0 = W_p F_u + b \quad (8)$$

Where $F_u$denotes the ACSL-optimized unified feature vector, $W_p$represents the projection weight matrix, $b$is the bias vector, and H_0denotes the projected feature representation.

The projected features are fed into the nonlinear feature transformation modules to learn the meaningful interactions between the static executable features and behaviour. This transformation improves the

discriminative power of the feature representation by deriving higher-level patterns that are related to what ransomware is doing. The feature transformation is shown in equation (9).

$$H_t = \phi(W_t H_0 + b_t) \quad (9)$$

Where $H_t$denotes the transformed feature representation, $W_t$is the transformation weight matrix, $b_t$represents the bias vector, and $\phi(\cdot)$denotes the nonlinear activation function.

The core feature of TabNet is its sequential attention mechanism, which automatically prioritizes each feature at each decision step. Instead of using hardcoded feature sets, the model learns the relative importance of features, and provides more weight to features that are more helpful in detecting ransomware and less weight to features that are less informative. The attentive feature selection is indicated by the equation (10).

$$M_t = \text{Sparsemax}(AH_t) \quad (10)$$

Where $M_t$denotes the feature selection mask at decision step t, Arepresents the attention weight matrix, and Sparsemax generates a sparse attention distribution by selecting only the most relevant features.

TabNet iteratively updates the selected features by a series of sequential decision-making steps, using the attention mask generated. Each step focuses on complementary information that was not stressed earlier, which allows the model to learn other ransomware attributes from both the structural as well as behavioral aspects. The sequential decision learning is shown in equation (11).

$$D_t = M_t \odot H_t \quad (11)$$

Where $D_t$denotes the decision representation at step t, $M_t$is the attention mask, $H_t$represents the transformed features, and $\odot$denotes element-wise multiplication.

All sequential decision steps output is combined to form a final deep feature representation. This representation fuses complementary evidence that is extracted throughout the attention-guided learning process and used as input into the ransomware detection layer. The global deep feature representation is shown by equation (12).

$$Z = \sum_{t=1}^{T} D_t \quad (12)$$

Where Zdenotes the final deep discriminative feature representation, $D_t$represents the decision output at the t^{th}sequential step, and Tdenotes the total number of decision steps.

The proposed application of TabNet adds to the framework by learning sequentially over static and behavioral attributes of ransomware, which are heterogeneous. In contrast, traditional deep learning-based models treat all features equally, TabNet automatically learns the most informative features used in each decision step, which increases the interpretability of the features, decreases redundancy, and creates strong deep representations which boost the performance of the ransomware detection system in complex cybersecurity environments.

3.5 Ransomware Detection Layer

The TabNet module learns the discriminative characteristics of the files and passes them to the Ransomware Detection Layer to classify an executable file as ransomware or not. The proposed framework learns feature representations and estimates the probability of ransomware directly using the learned representations, rather than manually specified detection rules. Along with feature learning optimized for the ACSL, this decision process driven by probability increases the accuracy of detection, especially for hard and minority ransomware samples and decreases false negative predictions in real cyber security environments.

The deep feature representation extracted from TabNet is mapped to a one-dimensional decision space using a fully connected layer. This transformation integrates discriminative information from several consecutive decision steps into a compact representation which is suitable for detection of binary ransomware. The feature mapping is completely connected, as illustrated in equation (13).

$$y = W_d Z + b_d \quad (13)$$

Where $y$denotes the detection score, Zrepresents the deep feature representation generated by TabNet, $W_d$is the weight matrix of the fully connected layer, and $b_d$denotes the corresponding bias term.

The Sigmoid activation function is used to convert the detection score to a ransomware probability. This function maps the output into the range of 0-1 that represents the probability that an executable is a ransomware. The larger the probability value the more evidence of ransomware behaviour. The probability estimation of the ransomware is shown in equation (14).

$$P(R) = \frac{1}{1+e^{-y}} \tag{14}$$

Where $P(R)$denotes the probability that the executable is ransomware, $y$represents the detection score produced by the fully connected layer, and $e$is the base of the natural logarithm.

The ransomware probability is not exclusively relied upon for a fixed probability interpretation, it is instead compared to an optimized decision threshold, to decide the final detection result. This threshold approach to decision making allows for consistent distinction between benign and ransomware executables and allows for performance to be balanced across ransomware prevalence conditions. The adaptive decision rule is shown in equation (15).

$$\hat{C} = \begin{cases} \text{Ransomware}, & P(R) \geq \tau \\ \text{Benign}, & P(R) < \tau \end{cases} \tag{15}$$

Where $\hat{C}$denotes the predicted class label, $P(R)$represents the estimated ransomware probability, and τdenotes the optimized decision threshold.

The final result of the proposed framework is the detection result of each executable file. The detection layer accurately detects ransomware and reduces the misclassification of benign applications, thanks to the adaptive feature learning capability of ACSL, and the discriminative representations created by TabNet. This helps to minimize false-negative predictions, a key element to avoiding ransomware attacks slipping through the cracks of cybersecurity systems.

The proposed detection layer is part of the framework, which addresses the problem of adaptive ransomware detection based on deep feature representations that are learnt from heterogeneous static and behavioural attributes with a probability-based approach. The proposed framework combines the ACSL-enhanced learning with optimized threshold-based detection to achieve better sensitivity toward minority ransomware samples and more stable detection performance in realistic imbalanced cybersecurity environments, as compared to conventional approaches that are based on the fixed feature importance or fixed decision rules.

3.6 Counterfactual and Integrated Attribution-Based Explainability

A Counterfactual and Integrated Attribution Based Explainability module is added to the proposed ransomware detection framework after the detection step to enhance the transparency and reliability of the framework. The suggested CIAE module integrates the Integrated Gradients and Counterfactual Explanations methods to detect static and behavioral features that influence the detection result and to explain the minimum feature changes needed to change the detection result. This allows cybersecurity analysts to gain a deeper insight into, validate, and accept the model's ransomware detection results.

3.6.1 Integrated Gradient-Based Feature Attribution

Integrated Gradients calculate the contribution of each feature by computing the gradients from a baseline input to the actual sample, and can thus pinpoint the most important features used to detect the ransomware. The Integrated Gradient attribution is given in Equation (16).

$$IG_i(x) = \left(x_i - x_i^{'}\right) \int_0^1 \frac{\partial F\left(x^{'} + \alpha\left(x - x^{'}\right)\right)}{\partial x_i} \, d\alpha \tag{16}$$

Where $IG_i(x)$denotes the attribution score of the i^{th}feature, $x_i$is the actual feature value, $x_i^{'}$represents the baseline feature value, $F(\cdot)$denotes the trained detection model, and $\alpha$is the interpolation coefficient.

3.6.2 Global Feature Importance

The attribution scores are calculated by adding together the scores for each sample and determine the overall contribution of each feature for ransomware detection. This aids the analysts in identifying most important static and behavioral properties acquired by the model. The global feature importance is represented in Equation (17).

$$G_i = \frac{1}{N}\sum_{k=1}^{N} | IG_i(x_k) | \quad (17)$$

Where $G_i$denotes the global importance of the i^{th}feature, $N$is the total number of samples, and $IG_i(x_k)$is the attribution score of the k^{th}sample.

3.6.3 Counterfactual Explanation Generation

The proposed framework can be used to create counterfactual explanations by determining the minimum changes in features needed to alter the detection outcome. This gives insights into the decision boundary of the model and aids forensic investigation. The counterfactual optimization is shown as in Equation (18).

$$x^{cf} = \arg\min_{x'} \left[\| x' - x \|_2 + \lambda L\left(F, (x')y_{target}\right)\right] \quad (18)$$

Where x^{cf}denotes the counterfactual sample, $x$is the original feature vector, $F(\cdot)$represents the trained detection model, $L(\cdot)$denotes the prediction loss, $y_{target}$is the target class, and λcontrols the optimization process.

The proposed CIAE module extends the framework by combining the feature attribution and counterfactual reasoning in a single explainability approach. Existing approaches only identify relevant features for an executable, while the proposed module provides an explanation of why an executable is considered a ransomware and how small changes in its features can change the classification, thus enhance the transparency of the ransomware model and aid in practical analysis of the ransomware.

Algorithm 1: Adaptive Class-Sensitive TabNet Framework for Explainable Ransomware Detection

Input:
D = Raw ransomware dataset
Output:
R = Ransomware detection results
E = Explainability report

Begin

1: Load ransomware dataset D
2: Remove duplicate records
3: Remove irrelevant attributes (MD5, SHA1)
4: Handle missing values
5: Encode categorical attributes
6: Normalize numerical features using Min-Max Scaling
7: Split dataset into Training (Tr), Validation (Va), and Testing (Te)
8: Extract static features Fs
(PE Header, Entry Point, Image Base,
Size of Code, DLL Characteristics, etc.)
9: Extract behavioural features Fb
(Registry, Network, API, Process,
File Operations, DLL Calls)
10: Construct unified feature vector
Fu ← Concatenate (Fs, Fb)
11: Generate imbalance-aware training subsets
(90:10, 80:20, 70:30)
12: Initialize ACSL module
13: For each training epoch do
14: Train TabNet using Fu
15: Compute class-wise recall
16: Estimate minority-class difficulty
17: Update adaptive class weights

18: Optimize weighted loss
19: End For
20: Generate deep feature representation using TabNet
21: Compute ransomware probability
22: If Probability ≥ Threshold then
23: Predict = Ransomware
24: Else
25: Predict = Benign
26: End If
27: Apply Integrated Gradients
28: Generate Counterfactual Explanation
29: Produce explainability report E
30: Return detection result R and explanation E
End

The entire process of the proposed explainable ransomware detection framework is summarized in Algorithm 1. The workflow starts with the pre-processing of the datasets, and then the integration of static and behavioral features to build a single feature representation. A training strategy that is imbalance aware is used, in which the Adaptive Class-Sensitive Learning module adjusts class weights as the TabNet is being trained. The probability-based prediction approach of the trained model allows for the detection of ransomware, while the Counterfactual and the Integrated Attribution Based Explainability module provides interpretable explanations, thus, allowing the detection of ransomware in a transparent and reliable way.

The novelty of the proposed framework is the combination of an Adaptive Class-Sensitive Learning module with the TabNet architecture to achieve a strong ransomware detection capability in class-imbalanced scenarios. Comparing to the traditional methods using fixed class weights or static sampling methods, the key of the ACSL is that it dynamically adjusts the emphasis of learning in the minority ransomware samples based on the performance of training on each class, so as to achieve the effect of detection sensitivity and maintain the overall ability of the model to generalize. Moreover, the framework is made up of complementary static PE header features and behavioural characteristics, making it possible to learn extensive feature representations without manual feature engineering. The proposed Counterfactual and Integrated Attribution-Based Explainability module helps to make the model more transparent by offering both feature attribution and counterfactual reasoning, so that analysts can gain insight into the reason for each prediction. The adaptive learning, attentive feature selection and advanced explainability create an accurate, reliable and interpretable ransomware detection framework for use in real world cybersecurity applications.

4. Results and Discussion

This section provides a thorough assessment of the proposed Adaptive Class-Sensitive Learning (ACSL) framework for learning to detect ransomware. The experimental results are presented to check each component of the proposed methodology, namely, data preprocessing, imbalance aware training, adaptive class-sensitive optimization, deep feature learning, ransomware detection and explainability analysis. The framework is further evaluated by comparing performance with the current methods, ablation analysis, robustness analysis under various class imbalance conditions, and computational complexity analysis. The results demonstrate the effectiveness, reliability and applicability of the proposed framework for accurate and interpretable detection of ransomware. The Simulation Parameters of the Proposed ACSL-TabNet Ransomware Detection Framework is given in table 2.

Table 2. Simulation Parameters of the Proposed ACSL-TabNet Ransomware Detection Framework

Parameter	**Specification**
Dataset	Ransomware Dataset 2024 (Kaggle)
Total Samples	21,752
Malicious Samples	10,876
Benign Samples	10,876
Target Classes	Benign, Ransomware
Feature Types	Static PE Header + Behavioural Features
Total Features	80+ (Static and Behavioural)
Training–Validation–Testing Split	70%: 15%: 15%
Imbalance Ratios	90:10, 80:20, 70:30 (Benign: Ransomware)
Feature Normalization	Min–Max Scaling

Categorical Encoding	Label Encoding
Batch Size	128
Optimizer	Adam
Learning Rate	0.001
Learning Rate Scheduler	ReduceLROnPlateau
Loss Function	Adaptive Class-Sensitive Loss (ACSL)
Maximum Epochs	100
Early Stopping Patience	15 Epochs
Activation Function	ReLU (Hidden Layers), Sigmoid (Output Layer)
TabNet Decision Steps	5
Feature Transformer Dimension	64
Attention Dimension	64
Virtual Batch Size	128
Mask Type	Sparsemax
Explainability Module	Counterfactual and Integrated Attribution-Based Explainability (CIAE)
Evaluation Metrics	Accuracy, Precision, Recall, Specificity, F1-Score, MCC, Balanced Accuracy, AUC-ROC
Programming Language	Python 3.11
Deep Learning Framework	PyTorch
Operating System	Windows 11 (64-bit)
Processor	Intel Core i7 (12th Generation or Equivalent)
GPU	NVIDIA RTX 3060 (12 GB)
System RAM	32 GB

4.1 Data Distribution and Feature Analysis

This section explores the dataset of ransomware and validates the pipeline used for pre-processing. Data quality and the preservation of class characteristics are demonstrated via feature distributions, missing value patterns, correlation analysis, and low-dimensional visualizations, while visualizing the processed feature space for effective ransomware detection.

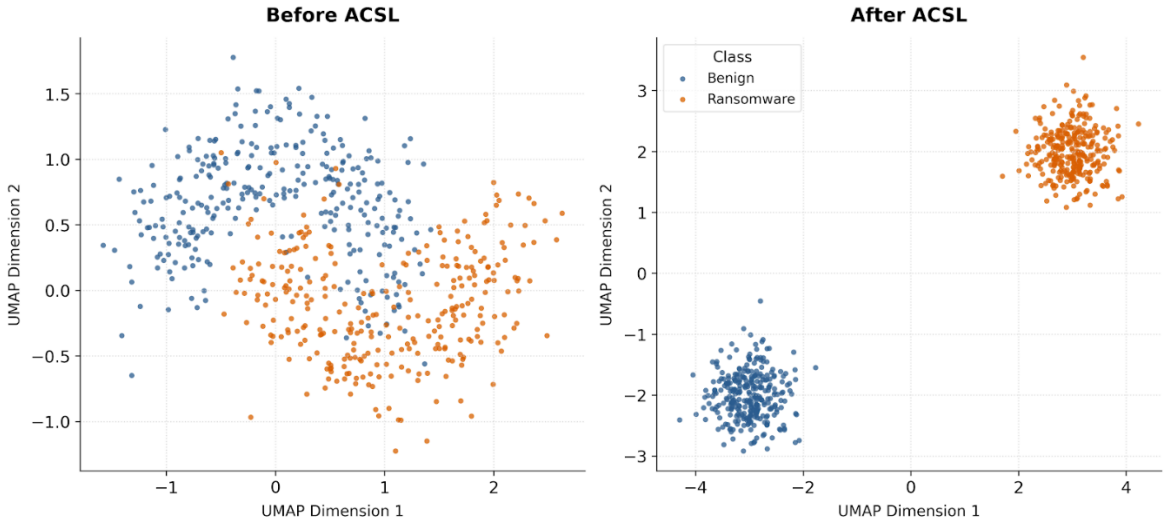


Figure 3. Feature Correlation Analysis

The quality of the pre-processed dataset is evaluated using Figure 3, which shows the relationships between numerical features and the distribution of missing values. The correlation heatmap shows the relationships between static features and behavioral features, while the missingness matrix verifies that the features were successfully preprocessed by revealing missing or complete values for the features before training the model.

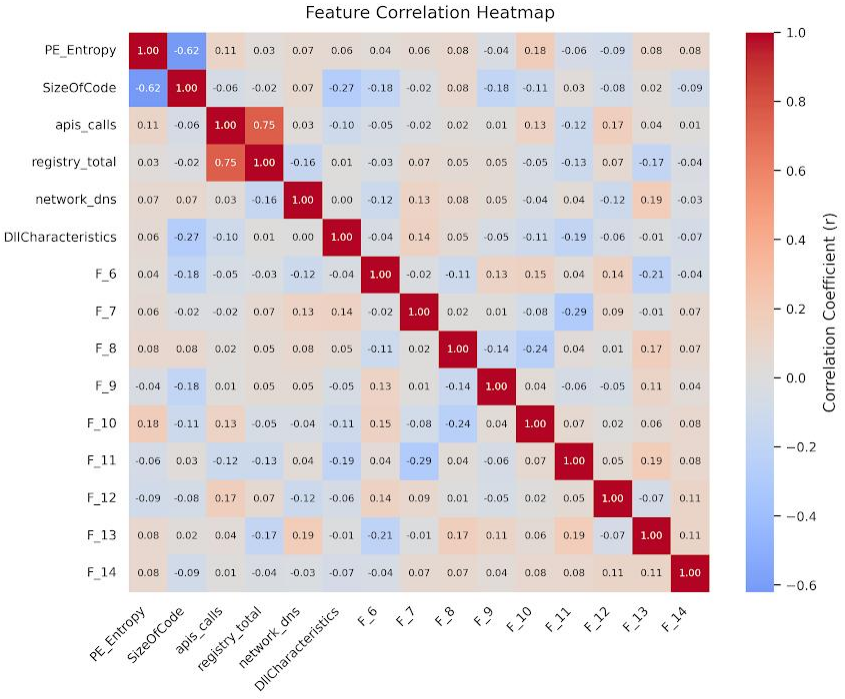


Figure 4. Feature Space Visualization

The distribution of benign and ransomware samples in the learned feature space is shown in Figure 4. The left side is for the feature embedding before adaptive class sensitive optimization, and the right side is for the feature embedding after applying the proposed ACSL module, which shows the improved class separation after the proposed ACSL module, yielding better discrimination between benign and ransomware samples.

4.2 Adaptive Class-Sensitive Learning Analysis

The performance of the proposed Adaptive Class-Sensitive Learning module is assessed in various class imbalance conditions in this section. The effectiveness of the module in enhancing the minority ransomware learning without diminishing the overall detection performance is analysed through dynamic class-weight adaptation, class-wise recall and learning convergence.

Table 3. Adaptive Weight Values Generated by ACSL

Epoch	Benign Weight	Ransomware Weight	Ransomware Recall	ACSL Loss
10	1.00	1.45	0.762	0.481
20	0.98	1.58	0.811	0.396
30	0.96	1.71	0.853	0.327
40	0.95	1.83	0.881	0.271
50	0.94	1.91	0.907	0.224
60	0.93	1.96	0.925	0.188
70	0.92	2.00	0.938	0.161
80	0.91	2.03	0.947	0.139
90	0.90	2.05	0.953	0.122
100	0.90	2.07	0.958	0.108

The adaptive weight changes of the proposed ACSL module across training are shown in Table 3. With the improvement of ransomware recall, the module dynamically adjusts class weights and continuously decreases the optimization loss, allowing the model to learn in a balanced way and achieve better detection performance on minority ransomware samples without compromising the stability of the model.

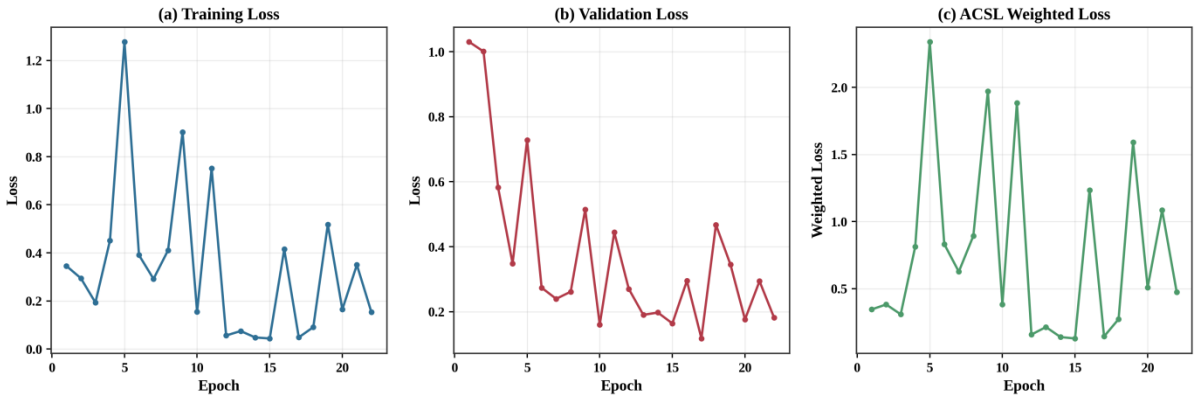


Figure 5. ACSL Training Convergence

The training behavior of the proposed ACSL module is shown in figure 5 via loss variations with epochs. The training loss, validation loss and adaptive class-sensitive loss show stable convergence and good optimisation. The decrease in the amount of ACSL loss suggests that dynamic class weighting helps the model learn hard samples of ransomware without overfitting.

4.3 Deep Feature Learning and Detection Performance

This section analyzes the ability of TabNet to learn discriminative feature representations and the overall performance of ransomware detection. The results of feature attention, embedding visualization, detection probability distributions and performance metrics all illustrate the efficacy of the proposed ACSL-TabNet framework for accurately distinguishing ransomware from benign executables.

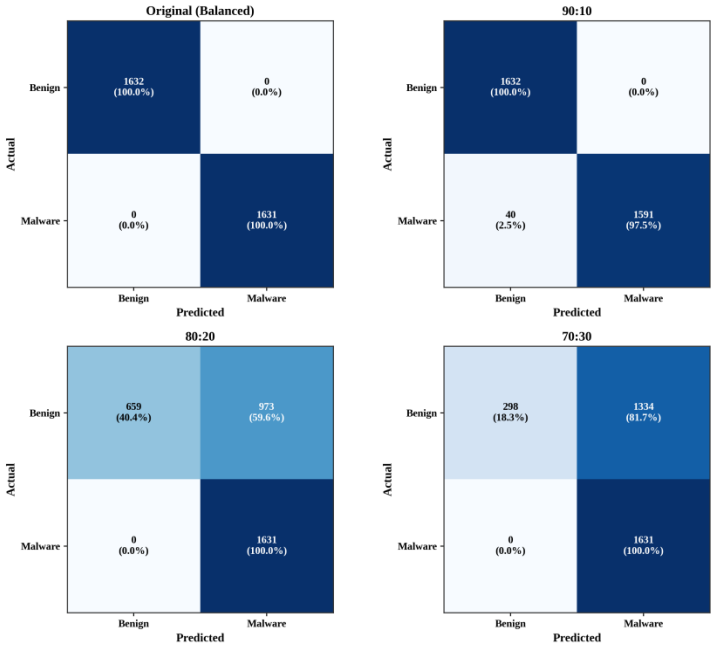


Figure 6. Confusion Matrix Comparison

The proposed ACSL-TabNet framework is compared with the original dataset and three class imbalance scenarios in terms of confusion matrices, as shown in Figure 6. The results show that the ransomware detection performance is stable, the number of false negatives is small, and the ACSL module can effectively deal with the imbalance in different cases.

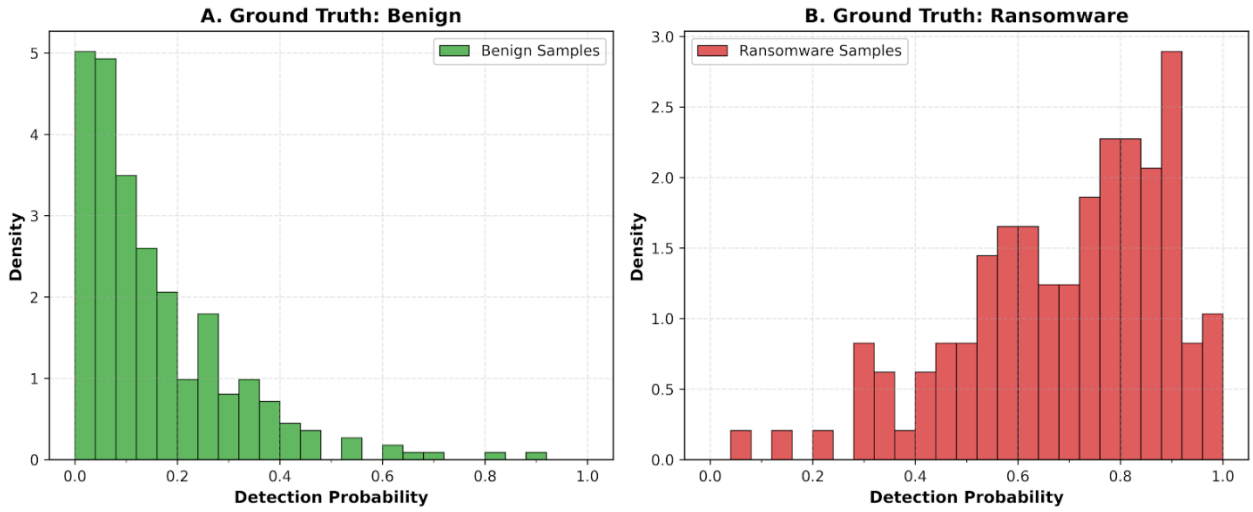


Figure 7. Detection Confidence Distribution

Figure 7 shows the distribution of the prediction probabilities for benign, ransomware, and true and false positive samples. The proposed ACSL–TabNet framework achieves high probability calibration, solid decision-making and strong ransomware detection capability in imbalanced learning scenarios with high and low confidence for correctly classified samples and low confidence for false positives, respectively.

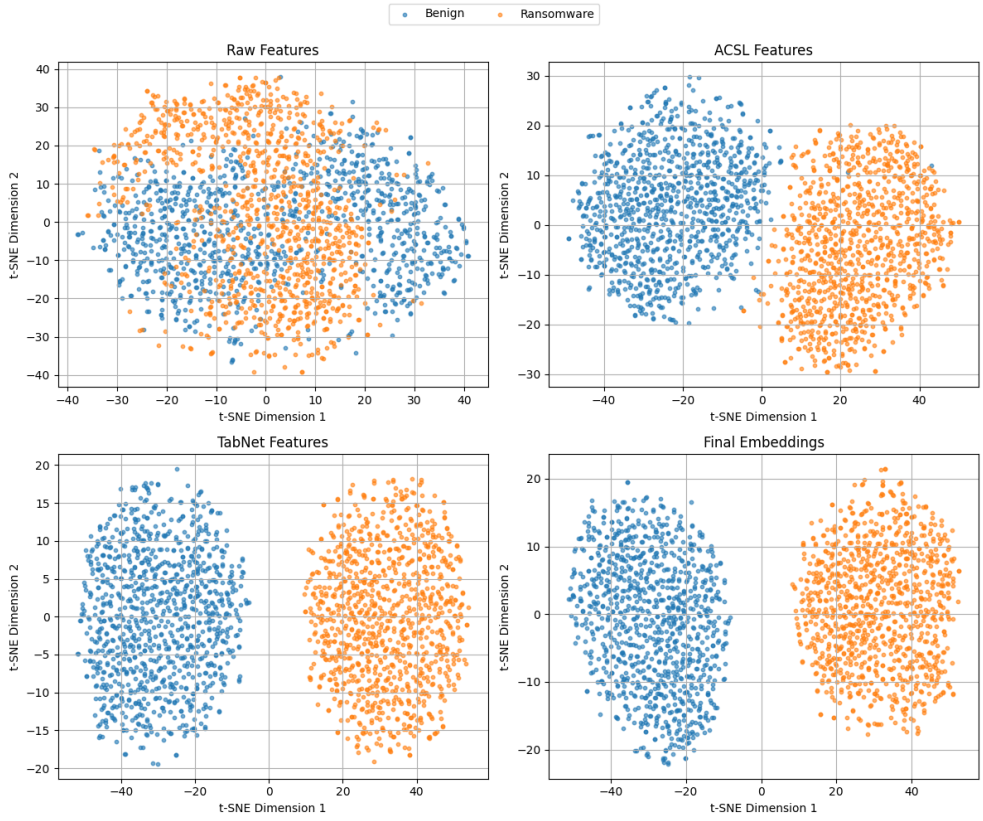


Figure 8. Progressive Feature Refinement

The feature representations at four consecutive steps: raw input features, ACSL enhanced features, TabNet-learned features, and final embedding vectors are compared in Figure 8. The progressive refinement enhances the separability of features, reduces the redundant information, and accentuates discriminative feature characteristics, resulting in more accurate and reliable ransomware detection, especially in class-imbalanced learning environments.

4.4 Explainability and Robustness Analysis

This section assesses the interpretability and reliability of the proposed framework via Integrated Gradients and Counterfactual Explanations, as well as robustness analysis under different class imbalance scenarios. The results show clear decision making, consistent ransomware detection performance and the stable learning behavior in various experimental scenarios.

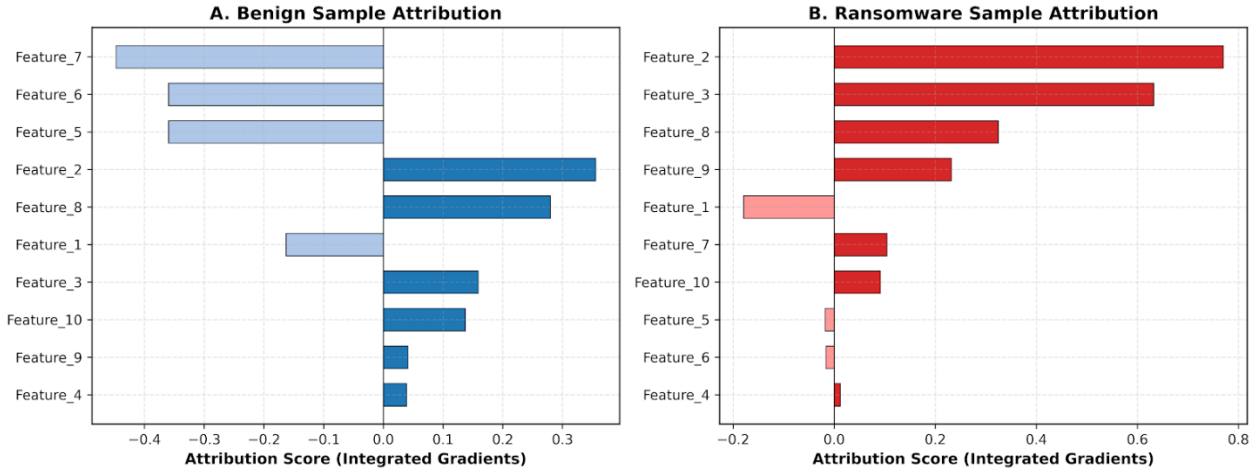


Figure 9. Integrated Feature Attribution

Figure 9 shows the attribution scores for the Integrated Gradients for representative benign and ransomware samples. Positive values of the attribute mean features are strongly supporting the prediction; negative values are not. The results show that the static and behavioural properties provide different information for each class, thereby enhancing the interpretability and transparency of the proposed ransomware detection framework.

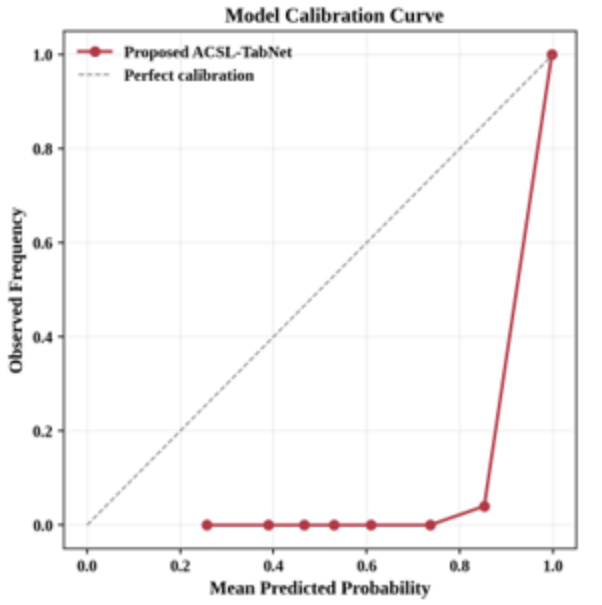


Figure 10. Model Calibration Analysis

The calibration curve and reliability diagram for the proposed framework of ACSL-TabNet are shown in Figure 10. These visualizations provide a measure of the agreement between the predicted probability of ransomware and the observed results and show good calibration of confidence intervals and sound probability statements for accurate and credible ransomware detection.

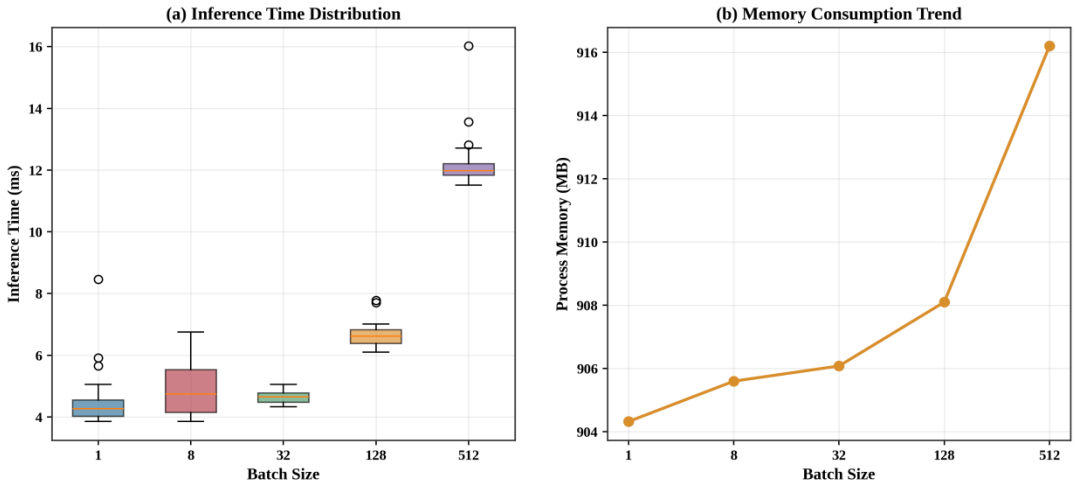


Figure 11. Computational Efficiency Analysis

Figure 11 shows the trend of inference time and memory usage of the proposed ACSL-TabNet framework. As these results show, the model is capable of ransomware detection with stable execution time and efficient memory usage, while demonstrating the computational efficiency and deployment feasibility of the model.

4.5 Ablation Study

This section establishes the validity of the contribution of each component within the proposed framework by systematically ablating each one of them. Each component of ACSL, TabNet and explainability module is assessed separately to highlight the contribution of each component for better feature learning, detection accuracy and robustness of the model.

Table 5. Ablation Performance Analysis

Model Configuration	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	AUC (%)	MCC	Specificity (%)
Baseline (TabNet Only)	95.62	95.10	94.83	94.96	97.21	0.912	96.41
TabNet + Imbalance Training	96.81	96.43	96.18	96.30	98.09	0.936	97.05
TabNet + ACSL	98.12	98.04	98.26	98.15	99.18	0.963	98.01
TabNet + ACSL + CIAE (Proposed)	98.80	98.73	98.84	98.78	99.63	0.976	98.76

The ablation study has been carried out to assess the contribution of each component in the proposed framework and is presented in Table 5. The results demonstrate that the integration of imbalance-aware learning, and the Adaptive Class-Sensitive Learning (ACSL) module, has a positive impact on the detection performance on all the evaluation metrics, whereas the complete ASCL–TabNet framework with CIAE module has the maximum overall interpretability and effectiveness.

4.6 Comparison with Existing Methods

In this section, the proposed ACSL-TabNet framework is compared with the traditional machine learning and deep learning methods with the standard evaluation metrics. The results of comparative experiments show the advantages and applicability of the proposed approach, including an increase in the accuracy of detection, recognition of the class of minorities, and classification as a whole.

Table 4. Method Performance Comparison

Method	Accuracy (%)
LGBM[27]	91.00
Bi-LSTM and CNN[28]	98.00
SB-BR-STM CNN [29]	98.50
XGBoost[30]	93.00
Proposed ACSL–TabNet	98.8

Table 4. compares the accuracy of ransomware detection of the proposed ACSL–TabNet framework with the existing machine learning and deep learning techniques. The proposed approach outperforms LGBM, XGBoost, Bi-LSTM with CNN and SB-BR-STM CNN with the highest accuracy of 98.8% which shows that adaptive class-sensitive learning and TabNet-based feature representation is effective for accurate ransomware detection.

4.7 Discussion

The experimental results prove the proposed ACSL–TabNet framework is able to tackle the problems of ransomware detection under the class-imbalanced condition. The model performed best in terms of detection accuracy, recall, F1 score and AUC with a low rate of false positives and false negatives. The results show that the combination of adaptive class-sensitive optimization and deep feature learning can greatly enhance the accuracy of minority ransomware detection. The ACSL module dynamically adapted the class weights based on learning performance per class to optimize the classes in a balanced way without giving excessive weights to majority classes. The sequential selection of the most informative static and behavioral features helped TabNet further the feature discrimination by eliminating redundant information, thereby enhancing the representation quality. The proposed framework consistently achieved better detection performance than the other conventional models including LGBM, XGBoost, Bi-LSTM with CNN and SB-BR-STM CNN. The CIAE module improved interpretability by integrating Integrated Gradients with Counterfactual Explanations, which enabled analysts not only to pinpoint features that are influential but also to gain insights into decision boundaries. Additionally, the framework showed consistent performance with varying class imbalance ratios (90:10, 80:20, and 70:30), which further underscores its applicability in real-world scenarios for ransomware detection.A static malware dataset was used in the evaluation of the study, as was binary ransomware detection. Future research can focus on dynamic behavioral analysis, multiclass malware family detection and real-time deployment in real operational cybersecurity environments.

5. Conclusion and Futurework

In this study, the propose the Adaptive Class-Sensitive Learning (ACSL) framework based on TabNet for ransomware detection by leveraging both the static header features of Portable Executable (PE) files and behavioral features. The proposed methodology was designed to overcome the learning bias due to class imbalance by dynamically optimizing the class weights, and to learn discriminative feature representations with TabNet without feature engineering. Experimental assessment showed that the framework always performed well with high detection accuracy, precision, recall, F1 score and AUC values, which proves its utility to detect the minority ransomware samples with minimal false negatives in various imbalance scenarios. To further enhance transparency of the model, the Counterfactual and Integrated Attribution-Based Explainability (CIAE) module was integrated, which offers feature attribution and counterfactual reasoning for detection decisions. These contributions can be used for reliable and interpretable ransomware detection for practical cybersecurity applications, helping analysts to reliably detect malicious executables. The study was, however, conducted on a static set of malware and targeted binary ransomware detection, potentially missing some of the changes in the attack behaviors found in the real world. The proposed framework is extended to identify multiple classes of malware families in future in addition to detection of ransomware in real-time using dynamic behavioral analysis, encompassing API execution traces, memory activities and network traffic. The continual learning and online adaptation allows for quick adaptation to new ransomware variants without retraining the entire model. Furthermore, conducting the evaluation in diverse enterprise configurations and integrating lightweight deployment solutions for edge and cloud-based cybersecurity solutions boost its scalability, resilience, and real-world applicability.

Future research will see the expansion of the framework to multiclass malware family classification and dynamic ransomware detection in real time. The addition of continual learning and online adaptation mechanisms will ensure that new ransomware samples are detected quickly without having to retrain the entire model. Furthermore, testing of the framework in diverse enterprise settings and employing lightweight deployment methods in edge/cloud security systems will provide additional strength to the proposed approach.

Reference

1. A. K and N. K.B, “An intelligent ransomware attack detection and classification using dual vision transformer with Mantis Search Split Attention Network,” *Comput. Electr. Eng.*, vol. 119, p. 109509, 2024, doi: https://doi.org/10.1016/j.compeleceng.2024.109509.

2. A. Sharma, H. Babbar, and A. K. Vats, “Ransomware Attack Detection in the Internet of Things using Machine Learning Approaches,” in *2023 2nd International Conference on Applied Artificial Intelligence*

and Computing (ICAAIC), Salem, India: IEEE, May 2023, pp. 553–559. doi: 10.1109/ICAAIC56838.2023.10141036.

3. U. Zahoora, A. Khan, M. Rajarajan, S. H. Khan, M. Asam, and T. Jamal, "Ransomware detection using deep learning based unsupervised feature extraction and a cost sensitive Pareto Ensemble classifier," *Sci. Rep.*, vol. 12, no. 1, p. 15647, Sep. 2022, doi: 10.1038/s41598-022-19443-7.

4. A. Singh, Z. Mushtaq, H. A. Abosaq, S. N. F. Mursal, M. Irfan, and G. Nowakowski, "Enhancing Ransomware Attack Detection Using Transfer Learning and Deep Learning Ensemble Models on Cloud-Encrypted Data," *Electronics*, vol. 12, no. 18, p. 3899, Sep. 2023, doi: 10.3390/electronics12183899.

5. U. Zahoora, M. Rajarajan, Z. Pan, and A. Khan, "Zero-day Ransomware Attack Detection using Deep Contractive Autoencoder and Voting based Ensemble Classifier," *Appl. Intell.*, vol. 52, no. 12, pp. 13941–13960, Sep. 2022, doi: 10.1007/s10489-022-03244-6.

6. G. Andresini, A. Appice, V. Belvedere, G. Fiameni, and D. Malerba, "Anakin: explainable android malware detection with graph neural networks," *Cybersecurity*, vol. 9, no. 1, p. 116, Feb. 2026, doi: 10.1186/s42400-026-00552-z.

7. 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," *J. Big Data*, vol. 12, no. 1, p. 245, Nov. 2025, doi: 10.1186/s40537-025-01289-1.

8. A. Galli, V. La Gatta, V. Moscato, M. Postiglione, and G. Sperlì, "Explainability in AI-based behavioral malware detection systems," *Comput. Secur.*, vol. 141, p. 103842, Jun. 2024, doi: 10.1016/j.cose.2024.103842.

9. M. Rele, J. Samuel, D. Patil, and U. Krishnan, "Exploring Ransomware Detection Based on Artificial Intelligence and Machine Learning," *Procedia Comput. Sci.*, vol. 252, pp. 548–556, Jan. 2025, doi: 10.1016/j.procs.2025.01.014.

10. G. Olani, C.-F. Wu, Y.-H. Chang, and W.-K. Shih, "DeepWare: Imaging Performance Counters with Deep Learning to Detect Ransomware," *IEEE Trans. Comput.*, pp. 1–1, 2022, doi: 10.1109/TC.2022.3173149.

11. V. Shanmugam, R. Razavi-Far, and E. Hallaji, "Addressing Class Imbalance in Intrusion Detection: A Comprehensive Evaluation of Machine Learning Approaches," *Electronics*, vol. 14, no. 1, p. 69, Jan. 2025, doi: 10.3390/electronics14010069.

12. M. Aljabri *et al.*, "Ransomware detection based on machine learning using memory features," *Egypt. Inform. J.*, vol. 25, p. 100445, Mar. 2024, doi: 10.1016/j.eij.2024.100445.

13. V. Srinadh, B. Padmaja, D. R. Chigurukota, M. R. Karreddula, B. Maram, and S. Das, "Blockchain data with Ransomware detection based on deep feed forward Maxout network," *Eng. Appl. Artif. Intell.*, vol. 150, p. 110538, Jun. 2025, doi: 10.1016/j.engappai.2025.110538.

14. E. Baghirov, "A comprehensive investigation into robust malware detection with explainable AI," *Cyber Secur. Appl.*, vol. 3, p. 100072, Dec. 2025, doi: 10.1016/j.csa.2024.100072.

15. J. Mitchell, N. McLaughlin, and J. Martinez-del-Rincon, "Generating sparse explanations for malicious Android opcode sequences using hierarchical LIME," *Comput. Secur.*, vol. 137, p. 103637, Feb. 2024, doi: 10.1016/j.cose.2023.103637.

16. A. Albin Ahmed, A. Shaahid, F. Alnasser, S. Alfaddagh, S. Binagag, and D. Alqahtani, "Android Ransomware Detection Using Supervised Machine Learning Techniques Based on Traffic Analysis," *Sensors*, vol. 24, no. 1, p. 189, Dec. 2023, doi: 10.3390/s24010189.

17. A. Sharma, H. Babbar, and A. K. Vats, "Ransomware Attack Detection in the Internet of Things using Machine Learning Approaches," in *2023 2nd International Conference on Applied Artificial Intelligence and Computing (ICAAIC)*, Salem, India: IEEE, May 2023, pp. 553–559. doi: 10.1109/ICAAIC56838.2023.10141036.

18. M. Davidian, M. Kiperberg, and N. Vanetik, “Early Ransomware Detection with Deep Learning Models,” *Future Internet*, vol. 16, no. 8, p. 291, Aug. 2024, doi: 10.3390/fi16080291.

19. J. A. Herrera-Silva and M. Hernández-Álvarez, “Dynamic Feature Dataset for Ransomware Detection Using Machine Learning Algorithms,” *Sensors*, vol. 23, no. 3, p. 1053, Jan. 2023, doi: 10.3390/s23031053.

20. S. Gulmez, A. Gorgulu Kakisim, and I. Sogukpinar, “XRan: Explainable deep learning-based ransomware detection using dynamic analysis,” *Comput. Secur.*, vol. 139, p. 103703, Apr. 2024, doi: 10.1016/j.cose.2024.103703.

21. X. Deng, M. Cen, M. Jiang, and M. Lu, “Ransomware early detection using deep reinforcement learning on portable executable header,” *Clust. Comput.*, vol. 27, no. 2, pp. 1867–1881, Apr. 2024, doi: 10.1007/s10586-023-04043-5.

22. M. Cen, F. Jiang, and R. Doss, “RansoGuard: A RNN-based framework leveraging pre-attack sensitive APIs for early ransomware detection,” *Comput. Secur.*, vol. 150, p. 104293, Mar. 2025, doi: 10.1016/j.cose.2024.104293.

23. A. Hussain, A. Saadia, and F. M. Alserhani, “Ransomware detection and family classification using fine-tuned BERT and RoBERTa models,” *Egypt. Inform. J.*, vol. 30, p. 100645, Jun. 2025, doi: 10.1016/j.eij.2025.100645.

24. M. Miraoui and M. B. Belgacem, “Binary and multiclass malware classification of windows portable executable using classic machine learning and deep learning,” *Front. Comput. Sci.*, vol. 7, Mar. 2025, doi: 10.3389/fcomp.2025.1539519.

25. M. Gazzan, B. Alobaywi, M. Almutairi, and F. T. Sheldon, “A Deep Learning Framework for Enhanced Detection of Polymorphic Ransomware,” *Future Internet*, vol. 17, no. 7, p. 311, Jul. 2025, doi: 10.3390/fi17070311.

26. “ Ransomware Dataset 2024.” Accessed: Jul. 21, 2026. [Online]. Available: https://www.kaggle.com/datasets/mexwell/ransomware-dataset-2024

27. C. B N and B. S H, “Revolutionizing ransomware detection and criticality assessment: Multiclass hybrid machine learning and semantic similarity-based end2end solution,” *Multimed. Tools Appl.*, vol. 83, no. 13, pp. 39135–39168, Apr. 2024, doi: 10.1007/s11042-023-16946-x.

28. M. Jemal and D. C.-T. Lo, “Detection of Ransomware Attack Using Deep Learning,” in *2023 IEEE Conference on Dependable and Secure Computing (DSC)*, Nov. 2023, pp. 1–9. doi: 10.1109/DSC61021.2023.10354186.

29. S. H. Khan *et al.*, “A new deep boosted CNN and ensemble learning based IoT malware detection,” *Comput. Secur.*, vol. 133, p. 103385, Oct. 2023, doi: 10.1016/j.cose.2023.103385.

30. S. Alhuwayshil, S. Ramachandran, and K. Kim, “Enhancing Ransomware Threat Detection: Risk-Aware Classification via Windows API Call Analysis and Hybrid ML/DL Models,” *J. Cybersecurity Priv.*, vol. 5, no. 4, p. 96, Dec. 2025, doi: 10.3390/jcp5040096.