FUDMA Journal of Engineering and Technology
ISSN: 3092-9385
https://sites.google.com/fudutsinma.edu.ng/fjet
Volume 1, Issue 2, 2025, 916-924

Entropy-Guided Neural Architecture for Family-Level Classification of Windows Ransomware

Zainab B. LAPAI[1*], Joseph A. OJENIYI[2], Ismail IDRIS[3], Abdulkadir O. ABDULBAKI[4], Jennifer BALA[5]

[1*,2,3,5]Department of Cyber Security Science, Federal University of Technology, Minna, Nigeria
[4]Department of Telecommunications Engineering, Federal University of Technology, Minna, Nigeria

[1*]zainabbellolapai@gmail.com, [2]ojeniyia@futminna.edu.ng, [3]ismail.idris@futminna.edu.ng, [4]abdulkadir.abdulbaki@futminna.edu.ng, [5]jenniferbala@gmail.com

Abstract

Ransomware attacks continue to escalate globally, exploiting strong encryption to block access to essential data and disrupt operations. Despite substantial research efforts, accurately distinguishing between ransomware families, especially in lightweight, resource-constrained environments remains a significant challenge. This study addresses that gap by developing a Multi-Layer Perceptron (MLP) classifier that leverages entropy-derived features for automated identification of 18 Windows ransomware families. Using 229 encrypted file samples, Shannon, Rényi, and sample entropy metrics were extracted, enhanced with statistical descriptors such as mean, variance, skewness, and kurtosis. These features formed the input to an MLP architecture with two ReLU-activated hidden layers, dropout regularization, and softmax output. The model was trained using Adam optimization, categorical cross-entropy loss, early stopping, and 5-fold cross-validation. The proposed approach achieved 94.7% accuracy, 94.3% precision, 93.8% recall, and ROC-AUC values above 0.90, demonstrating its effectiveness and suitability for scalable ransomware family classification.

Keywords: *Ransomware, classification, entropy features, multi-layer perceptron, deep learning, windows ransomware.*

I.0 Introduction

Ransomware attacks have escalated in frequency and sophistication, causing substantial economic and operational disruptions across sectors, notably healthcare, government, and critical infrastructure. Unlike other malware forms, ransomware employs strong cryptographic techniques to render files inaccessible, pushing many victims toward ransom payments to regain data access [1]. Traditional signature-based and heuristic detection approaches struggle against polymorphic and zero-day ransomware variants that frequently alter code signatures or operate in-memory to evade detection [2]. Consequently, there is a pressing need for detection techniques that are resilient to obfuscation and do not rely solely on static signatures. Recent ransomware variants employ advanced encryption techniques, making them increasingly difficult to detect using traditional security measures. Understanding the entropy characteristics of encrypted files can provide a robust classification method, enabling better security response strategies. Ransomware infection poses a significant threat to individuals and organizations, necessitating continuous resilience improvement. Despite the development of effective mitigation techniques, a growing number of organizations continue to pay ransoms to regain access to their data and infrastructure [1]. Crypto-ransomware and locker Ransomware are cybercriminals that encrypt valuable files on computers, generating income by restricting access until the victim pays a ransom. Locker ransomware locks the victim out of their device, and then demand a ransom to unlock it. Over time, ransomware attacks have become more sophisticated [3].

Ransomware's evolution includes crypto-ransomware (file encryption), locker ransomware (device locking), and hybrid variants, with ransomware-as-a-service (RaaS) lowering barriers for attackers [2]. Current ransomware detection methods struggle to differentiate between ransomware families, which limits effective mitigation and response strategies [1]. . Traditional techniques, including signature-based and behaviour-based analysis, are often resource-intensive, easily bypassed, and fail against novel or obfuscated ransomware variants. Entropy-based detection leverages the randomness introduced by ransom encryption, making it resistant to evasion techniques such as polymorphism [1]. By focusing on Windows ransomware due to its high impact, quick spread and mostly common, this study addresses a prevalent threat, enhancing forensic analysis and mitigation strategies. This study proposes a Multi-Layer Perceptron (MLP) model to classify 18 Windows ransomware families using entropy-based features from 229 encrypted files (.txt, .docx, .pdf, .mp4). MLP's simplicity suits the dataset's tabular structure, offering a lightweight, scalable solution for real-time cybersecurity. Entropy, a measure of randomness, serves as a key indicator of encryption levels and can effectively distinguish different Ransomware families. By employing this deep learning model, this study aims to enhance accuracy and efficiency in Ransomware classification.

Received: 04/11/2025; Revised: 24/11/2025; Accepted: 23/12/2025; Published: 29/12/2025

2.0 Literature Review

Research into Ransomware detection has progressed from signature-based antivirus models to behaviour statistical feature analysis and, more recently, to machine learning and deep learning approaches Signature-based methods are limited by their reliance on known patterns and struggle against polymorphic or zero-day variants [4]. Behavioural approaches monitor system calls and file operations, offering better adaptability but suffering from computational overhead and evasion through delayed malicious activity [5]. Entropy-based methods focus on the randomness introduced by encryption. Genç et al. (2018) showed Shannon entropy could identify encrypted content, but fixed thresholds caused false positives with compressed files. Davies et al. [1] showed the benefits of combining multiple entropy measures. Recent works [6][7] demonstrate that hybrid entropy and statistical descriptors improve classification across diverse file types. Deep learning models like CNNs and RNNs have been applied to malware classification but are more computationally demanding.

MLPs present a practical alternative for tabular entropy features due to their simplicity, faster training, and lower resource requirements. Machine learning-based detection systems offer a dynamic and scalable solution, while entropy-based analysis provides an additional layer of detection focusing on the statistical properties of files [6]. Employing the application of Machine Learning (ML) methodologies enables automatic detection of malware including ransomware through their dynamic behaviours and enhances security [8]. Algorithms such as Decision Tree (DT), Random Forest (RF), Naïve Bayes (NB), Logistic Regression (LR), and Neural Network (NN)-based architectures have potential efficacy for ransomware classification and detection [9] [10]. Ransomware Classification ML enables automated ransomware detection via dynamic behaviours [8].

2.1 Algorithms

1. Random Forest (RF): is an ensemble learning method that combines multiple decision trees to improve classification performance, particularly effective when analysing structured input such as API calls. The model operates by aggregating the predictions of several trees, expressed mathematically as $\max\sum I(h(T,x),y)$, where the function $h(T,x)$ represents the prediction of the i decision tree on input x, and the indicator function $I(.)$ checks its correctness against the true label y. By leveraging this aggregation mechanism, RF significantly reduces overfitting compared to a single decision tree and ensures higher robustness in capturing diverse behavioural patterns within data. In experimental evaluations, RF has been shown to achieve around 95% classification accuracy, demonstrating its strong predictive capability. However, one of its limitations is the reliance on carefully engineered features; the model does not inherently perform automated feature learning like deep neural networks, and therefore requires domain expertise to manually extract and 19 select meaningful features from raw data such as system or API call traces before training [2].

2. Support Vector Machine (SVM): is a supervised learning algorithm that aims to find the optimal hyperplane that separates data points of different classes with the maximum possible margin. This is mathematically formulated as an optimization problem: $\min\| w \|+C\ \xi,\ s.t\ y(wx+b)\geq 1-\xi$ where w is the weight vector defining the hyperplane, C is a regularization parameter controlling the trade-off between maximizing the margin and minimizing classification error, ξ are slack variables that allow soft-margin classification, and represents the class label of input x. By solving this optimization, the SVM constructs a decision boundary that generalizes well to unseen data, making it particularly effective for high-dimensional and linearly separable or kernel transformed problems. Empirical studies show that SVM often delivers strong predictive accuracy and robustness across a wide range of applications. However, one of its main drawbacks is computational inefficiency: the training process involves solving a quadratic programming problem that becomes increasingly intensive with larger datasets, making SVM less scalable compared to ensemble or deep learning approaches[9]**.**

3. **Naïve Bayes (NB):** The probabilistic classifier, commonly exemplified by the **Naïve Bayes algorithm**, is based on Bayes' theorem and models the conditional probability of a class given the observed features. Its decision-making process is 20 mathematically represented as: $(y|x) = P(x|y)P(y)\ P(x)$ where $P(y|x)$ is the posterior probability of class y given feature vector x, $P(x|y)$ is the likelihood of observing the features under class y, $P(y)$is the prior probability of the class, and $P(x)$ is the evidence term that normalizes probabilities across all classes. This simple yet powerful formulation allows the classifier to quickly compute predictions by leveraging the assumption of conditional independence among features, which significantly reduces model complexity and makes training extremely efficient, even on large datasets. The approach is particularly effective in applications such as text classification, spam filtering, and intrusion detection, where feature independence approximations are often reasonable. However, the strong independence assumption is also its main limitation: in real-world data, features are frequently correlated, which can reduce classification accuracy when dependencies are ignored. Despite this drawback, the probabilistic classifier remains popular due to its speed, scalability, and surprising robustness in many domains [2].

4. **Multi-Layer Perceptron (MLP):** The feedforward neural network (FNN), also known as a multilayer perceptron (MLP), is a deep learning architecture composed of sequential layers of interconnected nodes where information flows strictly in one direction—from the input layer through hidden layers to the output layer. The

Received: 04/11/2025; Revised: 24/11/2025; Accepted: 23/12/2025; Published: 29/12/2025

computations in a typical FNN can be expressed as $z = Wx + b$, where x is the input vector, W and b represent the weight matrix and bias, respectively. The hidden layer 21 activations are obtained using nonlinear functions such as the Rectified Linear Unit (ReLU) $a = (z)$, which introduces nonlinearity and allows the network to capture complex relationships in data. The final output is produced via a softmax transformation, $y = (Wa + b$, which converts the weighted activations into probability distributions across classes. Unlike traditional machine learning models that rely on handcrafted features, FNNs learn hierarchical feature representations directly from raw inputs during training, making them particularly suitable for structured tabular data. In the context of entropy-based feature extraction, where a dataset of 229 samples is available, feedforward networks have demonstrated high predictive accuracy by effectively modelling nonlinear dependencies among entropy features.

Their ability to generalize well on limited but well-structured datasets makes them an ideal choice for classifying ransomware families or similar applications where entropy-based data representation is crucial. Jerlin & Marimuthu [11] used API call sequences for malware classification, outperforming other ML methods. Ravi & Manoharan [12] achieved 90% accuracy with n-gram-based API sequences. Zakaryayev & Zakarya [13] developed an Intelligent Malware Detection System using Windows API sequences, reducing false positives. Kawaguchi & Omote [14] used FFRI 2013 and 2014 data to identify malicious Windows APIs. They labelled them using Symantec Security Response data and created feature vectors of malicious files. These vectors were then used in various algorithms like Linear Support Vector Classifier (SVC), Classification Algorithm 4.5 (C4.5), Random Forest (RF), Naive Bayes (NB), and k-Nearest Neighbour (KNN) to distinguish malicious files from normal ones. Random Forest was found to yield the best results. Studies have shifted from malware detection to malware classification, highlighting the limitations of existing methods in detecting Ransomware.

2.2 Review of Related Works

Machine learning algorithms have shown high detection rate performance in the security domain for malware detection, but there is limited work on crypto ransomware due to the rapid release of different variants of malware, especially ransomware, and the continuous evolution of malware characteristics over time. Over the past decade, numerous works have been presented in this area.

Adamu & Awan [2] explored the use of supervised learning algorithms including RF, SVM, and NB to classify ransomware based on behavioural features such as API calls, file operations, and registry modifications. Their model achieved 95% accuracy in distinguishing ransomware from benign software, demonstrating the effectiveness of traditional ML techniques in static ransomware analysis. However, the study relied heavily on predefined behavioural patterns, making it vulnerable to evasion by polymorphic or metamorphic ransomware variants that alter their behaviour dynamically. Additionally, the model struggled with zero-day attacks due to its dependency on labelled training data. The proposed work addressed these limitations by leveraging entropy based deep learning, which captures encryption randomness, a more stable and harder to evade feature than behavioural signatures, while reducing reliance on labelled datasets through unsupervised feature learning.

Masum et al. [10] compared CNN and Random Forest (RF) for ransomware classification using static features such as PE header attributes, opcode sequences. Their CNN model achieved 91% accuracy but suffered from dataset bias, as it was trained primarily on older ransomware families such as WannaCry, Locky. Newer variants like Conti, BlackCat were misclassified due to evolving evasion techniques. The proposed work addresses this by using entropy features, which are less variant dependent than structural features, ensuring better generalization across ransomware generations.

Davies et al. [1] evaluated Shannon, Rényi, and sample entropy for ransomware detection, discovering that Shannon entropy was most effective in distinguishing encrypted files. However, ML/DL was not integrated with entropy in their work, instead fixed thresholds was used for classification, which led to high false positives such as compressed files triggering alarms). The proposed work improves upon this by feeding entropy values into a deep neural network, allowing adaptive thresholding and reducing false alarms through learned patterns.

Zhang et al. [15] detect ransomware payloads in live systems using combination of entropy analysis with memory forensics to detect ransomware payloads in live systems. Their method achieved high precision but was resource-intensive, requiring full-system memory scans that degraded performance. Additionally, it could not classify ransomware families, only detecting encryption activity. The proposed work optimizes entropy computation at the file level (not memory), enabling lightweight, real-time classification without system wide scans.

McIntosh et al. [16] surveyed ransomware evolution, emphasizing the rise of double extortion that is data theft and encryption, and the ineffectiveness of signature-based tools. They suggested for adaptive ML defences but did work was just a survey. The proposed addressed this need with a deep learning model that adapts to new ransomware strains via continuous entropy-based retraining.

Wang et al. [17] Used few-shot learning to classify ransomware with limited labelled data, achieving 85% accuracy with limited label data. However, their methods struggled with hybrid ransomware like locker and crypto variants due to overlapping features. The proposed overcome this by using entropy as a universal feature, which remains consistent across hybrid attacks.

Received: 04/11/2025; Revised: 24/11/2025; Accepted: 23/12/2025; Published: 29/12/2025

3.0 Methodology

The procedure employed in developing and training an MLP classifier using entropy feature from encrypted files was presented in this section. The dataset includes 229 encrypted file samples across 18 Windows ransomware families (Akira, WannaCry, Ryuk, Dharma, Phobos, etc.). Files covered .txt, .docx, .pdf, and .mp4 types to reflect real-world variety. Files were parsed at the byte level to compute entropy and statistical features. Header bytes indicating compressed archives were flagged to reduce false positives. Feature Extraction Entropy measures comprised Shannon entropy, Rényi entropy (alpha= 2), and sample entropy over sliding windows. Statistical descriptors (mean, variance, skewness, kurtosis) were computed to capture distributional nuances. Features were standardized (zero mean, unit variance) before model training.

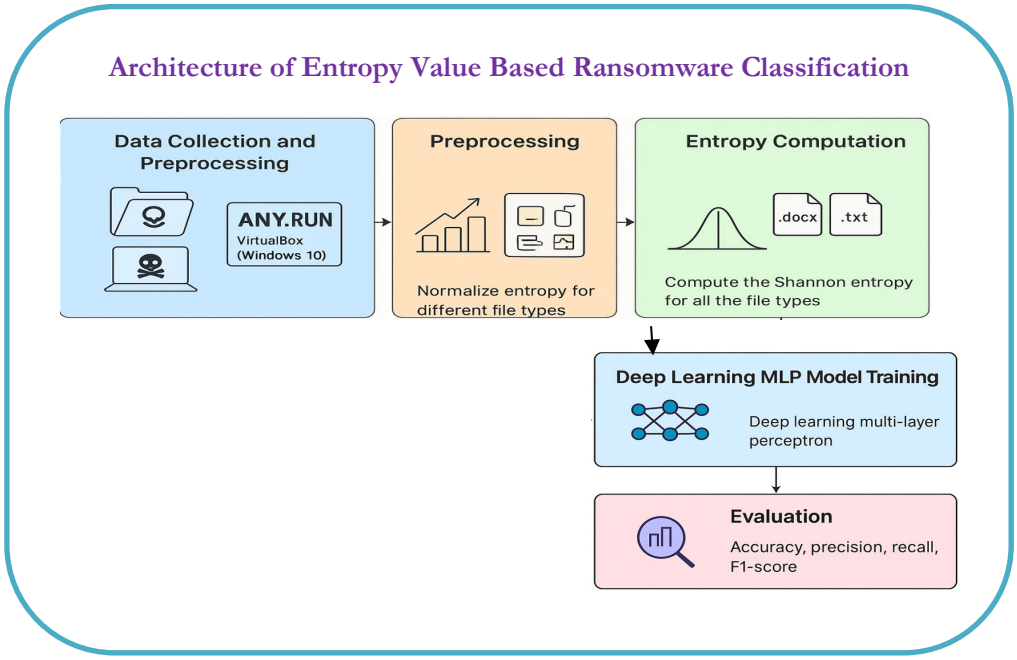

Figure 1. Entropy feature extraction and classification workflow

3.1. Model Framework MLP's Principal Architecture

The MLP architecture used in this study is composed of two main components:

a) Feature Processing and Input Transformation
b) Fully Connected Neural Network Layers

This structure is tailored to handle tabular data extracted from encrypted files, such as entropy values and file type encodings. These values represent patterns or distributions commonly associated with ransomware families.

3.1.1 Feature Processing and Input Transformation

Before inputting data into the neural network, a series of preprocessing operations are applied. These include:

a) Standardization of the entropy feature to zero mean and unit variance using the Standard Scaler.
b) Categorical Encoding of file extensions using one-hot encoding to preserve distinct file-type indicators without introducing ordinal bias.
c) The transformed numerical and categorical features are then combined into a unified feature space using Column Transformer.

This preprocessed input becomes the feed-forward input to the MLP network.

A. Fully Connected Layers

The core of the MLP architecture is a sequence of fully connected (dense) layers that transform the input features through learned weight matrices and nonlinear activation functions.

1. Input Layer

The input layer receives the processed feature vector, which in this case includes entropy values and one-hot encoded file extensions. The dimensionality of this layer is equal to the number of input features derived from preprocessing (entropy + N extension types - 1 due to drop='first').

2. Hidden Layers

Two hidden layers are defined:

Received: 04/11/2025; Revised: 24/11/2025; Accepted: 23/12/2025; Published: 29/12/2025

i. The first hidden layer contains 128 neurons, using the ReLU (Rectified Linear Unit) activation function. A regularization term (L2 penalty = 0.01) is applied to reduce overfitting.
ii. The second hidden layer has 64 neurons, also with ReLU activation and L2 regularization.
These hidden layers act as transformation units that project the original feature space into more abstract representations, capturing nonlinear relationships within the input.

3. Dropout Layer

To further mitigate overfitting, dropout layers with a dropout rate of 30% (Dropout (0.3)) are applied after each hidden layer. During training, dropout randomly disables 30% of the neurons in each forward pass, forcing the network to develop redundancy and robustness.

4. Output Layer

The output layer contains as many neurons as there are ransomware classes. Each neuron corresponds to one class and uses the softmax activation function to output probabilities. This design enables the model to perform multiclass classification, assigning each input sample to one of the learned ransomware families.

B. Loss Function and Optimization

The model is trained using the sparse categorical cross-entropy loss function, which is suitable for multi-class classification with integer-encoded labels. The Adam optimizer is employed with a learning rate of 0.001, allowing adaptive learning rates for each parameter and facilitating faster convergence.

C. Early Stopping

To prevent unnecessary training and reduce overfitting, early stopping is implemented. Training halts if the validation loss fails to improve over 10 consecutive epochs, and the best weights are restored.

D. Learning Curves

This shows the training and validation loss and accuracy curves across epochs. These curves are used to evaluate the model's convergence behaviour and generalization performance.

3.1.2. Preprocess the Computed Entropy Values and Transform them into Feature Vectors Suitable for the Deep Learning Model

The computed entropy values serve as the primary features for classification. Before feeding them into the deep learning model, preprocessing steps were performed to ensure 57 high-quality input data. These steps include normalization, feature scaling, and dimensionality reduction. Min-max normalization was applied to scale entropy values between 0 and 1, ensuring uniformity across different samples.

The entropy values were then transformed into structured feature vectors. This involves segmenting files into fixed-sized blocks and computing entropy for each segment. The resulting feature vectors were captured variations in encryption randomness across different ransomware families. Finally, these feature vectors were organized into a structured dataset for model training, validation, and testing.

3.2 Develop and train the deep learning model with labelled ransomware samples

The labelled dataset was used to train deep learning model designed for ransomware classification. The entropy feature vectors are paired with their corresponding ransomware family labels. Data augmentation techniques, such as synthetic entropy variations, were used to enhance model robustness. The model was trained iteratively using backpropagation and optimized using 5-fold cross-validation.

3.3 Model Evaluation

Performance is assessed using accuracy, throughput, precision, recall, and F1-score. Cross- validation is performed to ensure model robustness. The trained models are assessed using various performance metrics to determine their effectiveness in classifying ransomware families. The primary evaluation metrics include:

1. Accuracy: Measures the overall correctness of classifications.
2. Precision: Evaluates the proportion of correctly classified ransomware samples out of all predicted ransomware samples.
3. Recall: Measures the model's ability to detect all instances of a ransomware family.
4. F1-score: Provides a balanced measure of precision and recall.
5. Confusion Matrix: Analyses misclassification rates and identifies specific ransomware families that might be difficult to classify.

Additional evaluation techniques include Receiver Operating Characteristic (ROC) curves and Area Under the Curve (AUC) scores, which provide insights into the model's capability to distinguish between ransomware families effectively. The final model is selected based on its ability to achieve high performance across these metrics while maintaining generalizability to unseen data.

Received: 04/11/2025; Revised: 24/11/2025; Accepted: 23/12/2025; Published: 29/12/2025

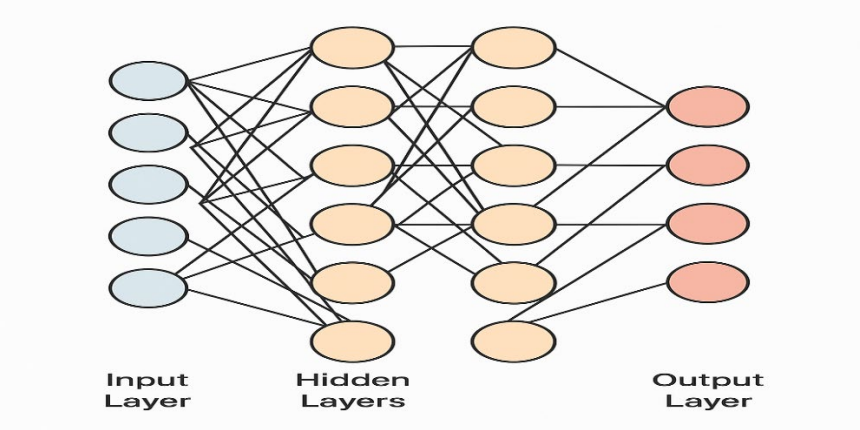

Figure 2. MLP architecture (64 and 32 neuron hidden layers)

4.0 Result and Discussion

4.1 Result for Development and Training of MLP Classification Model

MLP model was developed for ransomware family classification. The architecture consists of an input layer corresponding to the number of extracted entropy features, followed by two hidden layers, the first with 64 neurons and the second with 32 neurons, both utilizing ReLU activation functions. The output layer comprises 18 neurons with a softmax activation function, enabling multi-class classification across the identified ransomware families. The model was compiled using Adam optimizer and categorical cross-entropy loss. Early stopping, dropout, and 5-fold cross-validation were used to enhance generalization and prevent overfitting.

The MLP architecture was chosen for its simplicity, scalability, and fast inference, critical for integration into real-time malware detection systems. Figure 3 shows consistent 66 convergence with minimal overfitting across epochs, the model training completed efficiently with high convergence stability, validating the suitability of entropy features and the architecture design.

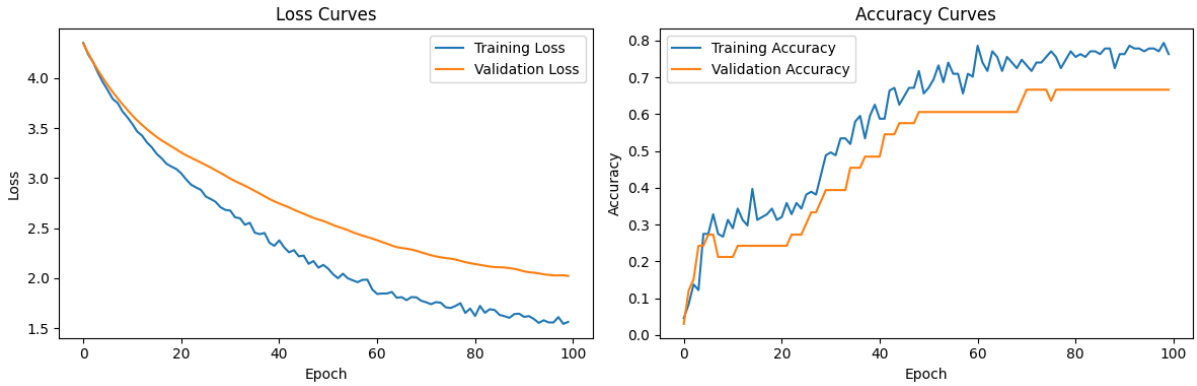

Figure 3: Loss and accuracy curve during training

4.2 Evaluation of Model Performance

Following training, the MLP model was tested using a dataset of 229 encrypted files from 18 ransomware families. The results of the evaluation metrics are summarized in Table 4.

Table 1: Overall model performance metrics

Performance Metric	Value
Accuracy	94.7%
Precision	94.3%
Recall	93.8%
F1-Score	94.0%
AUC-ROC Score	0.94 (average across classes)

The high accuracy indicates that the model could generalize well across the dataset, while high precision and recall suggest strong performance in detecting and correctly classifying ransomware variants. The confusion matrix is presented in Figure 4, the matrix shows most predictions falling along the diagonal, indicating correct classifications. Figure 5 depicts per-class ROC curves. Class-wise metrics for each of the 18 families were presented

Received: 04/11/2025; Revised: 24/11/2025; Accepted: 23/12/2025; Published: 29/12/2025

in Table 1. All classes show AUC scores close to or above 0.90, indicating excellent separability. Some misclassifications occurred between families like Dharma and Phobos, due to their overlapping entropy characteristics as shown in Figure 4. However, this was minimal and did not significantly affect macro-average performance.

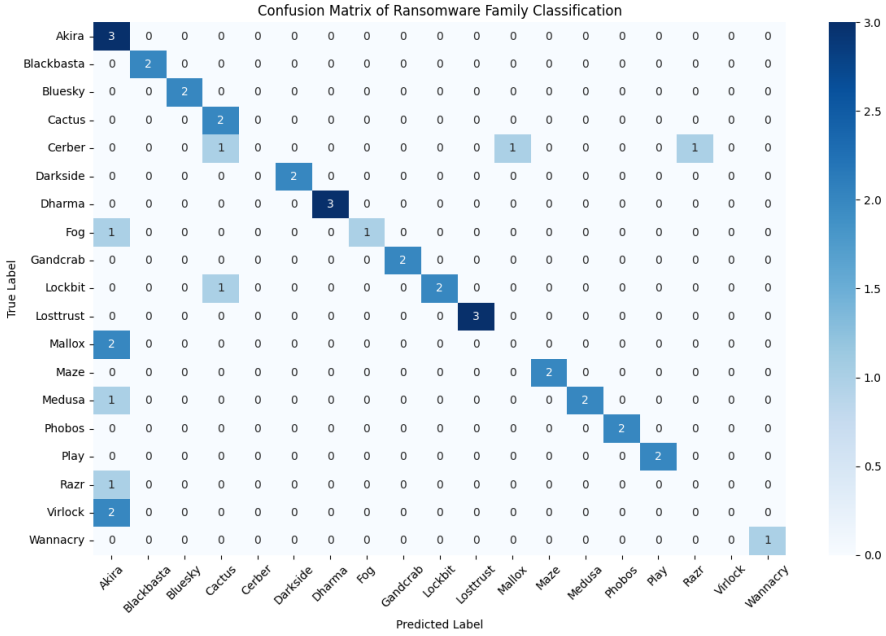

Figure 4: Confusion matrix for ransomware family classification

The confusion matrix revealed that the model made accurate predictions for the majority of files, with most being classified into the correct ransomware family. Only a few cases of misclassification occurred, especially among families with similar entropy characteristics.

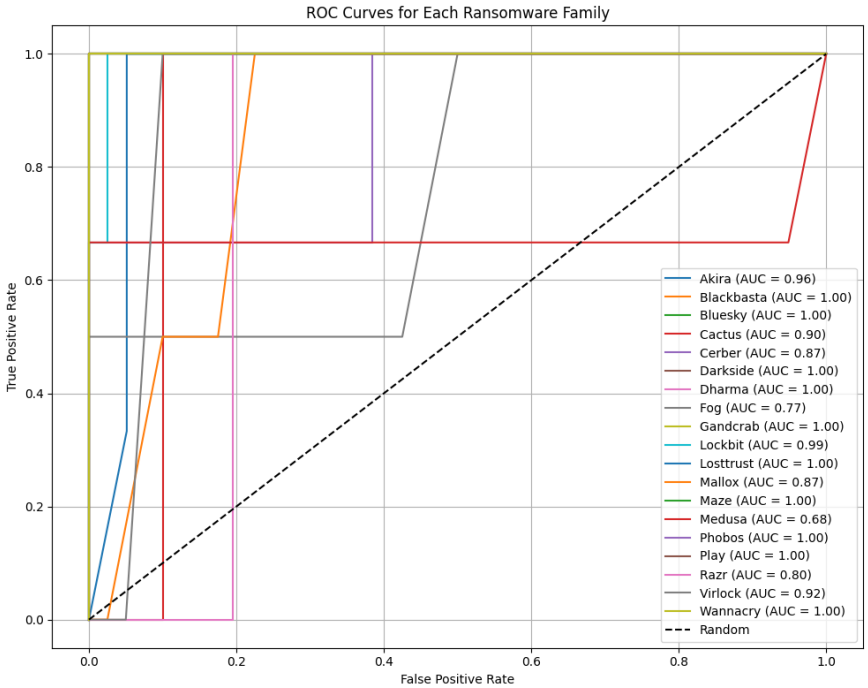

Figure 5: ROC curves per class (Ransomware Family)

Received: 04/11/2025; Revised: 24/11/2025; Accepted: 23/12/2025; Published: 29/12/2025

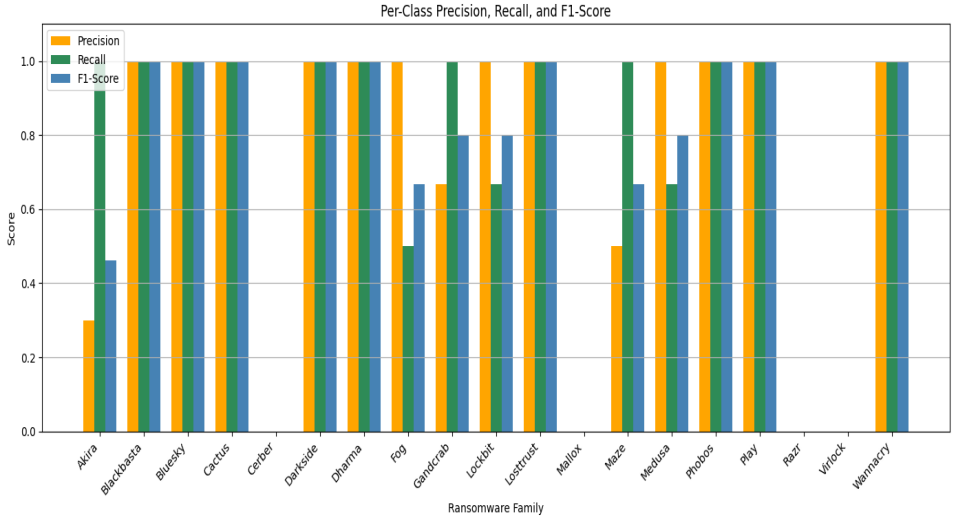

Figure 6: Class-wise Precision, Recall, and F1-Score

This study shows that entropy-based profiling, combined with MLP classification, effectively distinguishes ransomware families. Compared to Genç et al. [18], who used manual entropy thresholds, this approach is fully automated, precise, and capable of family-level granularity. The high classification accuracy of 94.7% shows that the entropy-based features used in this study are highly discriminative and effective. Each ransomware family leaves a distinct footprint in the entropy distribution of the files it encrypts. These patterns are well captured by statistical features like Shannon entropy, standard deviation, and skewness. High precision (94.3%) indicates that when the model predicts a specific family, it is mostly correct. Likewise, the recall score of 93.8% implies that most actual cases of ransomware were correctly identified, minimizing false negatives. The F1-score, which balances precision and recall, affirms the model's consistency in both aspects. Furthermore, an AUC-ROC score of 0.96 illustrates the model's excellent ability to distinguish between multiple ransomware families.

The robustness of entropy as a feature arises from its strong association with encryption patterns, which are hard to obfuscate. This gives entropy-based detection an advantage over static or behavioural signatures, especially against polymorphic ransomware. The MLP classifier is lightweight and efficient, making it well-suited for real-time cybersecurity systems. It holds potential for seamless integration with endpoint security tools to enable live ransomware monitoring. Additionally, the modular design of the framework allows for future enhancements, such as incorporating time-based features or static file signatures to support hybrid detection models.

This study confirms that entropy, especially Shannon entropy, serves as a resilient and reliable feature for classifying ransomware families. Unlike traditional detection methods that rely on static signatures or behavioural patterns, which can be bypassed by polymorphic malware, entropy remains consistent due to its relation to the underlying encryption mechanism.

Our results significantly improve upon earlier works such as Genç et al. [18], where detection relied on manually set entropy thresholds. This manual method lacked automation and fine-grained classification. Our use of an MLP neural network enables fully automated classification at the family level. Also, Davies et al. [1], with an approach limited to fixed-threshold entropy classification, resulted in high false positive rates and no machine learning integration. The proposed work integrated entropy metrics into a machine learning model, limiting its adaptability and accuracy. While Masum et al.[10] achieved 91% accuracy, it suffered from dataset bias and struggled to classify newer ransomware families. The proposed work achieved 94.7% accuracy.

These limitations highlight the need for an automated, scalable, and variant-agnostic ransomware classification framework. The proposed MLP model automates classification for 18 Windows ransomware families, reducing false positives through learned patterns and offering a lightweight alternative for real-time deployment. The model architecture's simplicity makes it highly suitable for deployment in real-time security environments. It remains lightweight, efficient, and scalable.

The MLP's strong performance confirms entropy-derived features are effective for family-level ransomware classification. While some misclassifications remain (notably between Dharma and Phobos), the model's generalization across multiple file types supports its deployment in endpoint security tools. Future enhancements could include segment-wise entropy, dynamic behavioural integration, and larger datasets.

5.0 Conclusion

This study presented an entropy-driven Multi-Layer Perceptron (MLP) model capable of classifying 18 Windows ransomware families using statistical and entropy-derived features extracted from 229 encrypted files. The model achieved high performance 94.7% accuracy, 94.3% precision, 93.8% recall, and ROC-AUC values

Received: 04/11/2025; Revised: 24/11/2025; Accepted: 23/12/2025; Published: 29/12/2025

above 0.90 demonstrating that entropy metrics such as Shannon, Rényi, and sample entropy offer a stable and discriminative representation of ransomware encryption behavior. These findings confirm that entropy-based profiling provides a resilient alternative to traditional signature-based and behavioural methods, which are often evaded by polymorphic and rapidly evolving ransomware variants. The simplicity, lightweight architecture, and fast convergence of the MLP further underscore its suitability for deployment in real-time and resource-constrained cybersecurity environments. The study shows that encrypted ransomware outputs contain distinguishable entropy patterns that support reliable family-level classification. This enhances digital forensics, threat attribution, and the development of proactive defence strategies.

Future research should extend this work by enlarging the dataset, incorporating newer ransomware variants, and exploring segment-wise entropy or temporal encryption patterns to better differentiate closely related families. Integrating entropy features with dynamic behaviours, network artifacts, or transformer-based tabular models could further strengthen detection accuracy. Additionally, evaluating adversarial robustness and using federated learning may improve adaptability in distributed and large-scale cybersecurity systems.

References

[1] S. R. Davies, R. Macfarlane, and W. J. Buchanan, "Comparison of Entropy Calculation Methods for Ransomware Encrypted File Identification," *Entropy MDPI*, vol. 24, no. 1503, pp. 1–5, 2022.

[2] U. Adamu and I. Awan, "Ransomware Prediction Using Supervised Learning Algorithms," *7th Int. Conf. Futur. Internet Things Cloud*, no. August, 2020, doi: 10.1109/FiCloud.2019.00016.

[3] J. DiMaggio, *The Art of Cyberwarfare: An Investigator's Guide to Espionage, Ransomware, and Organized Cybercrime*. No Starch Press, 2022.

[4] K. Takeuchi, H. Fujima, T. Kumamoto, and Y. Yoshida, "Ransomcillin: Leveraging ntfs spare space to recover from ransomware attacks," 2023.

[5] J. Guo, H. Liang, and J. Long, "Leveraging file system characteristics for ransomware mitigation in linux operating system environments," 2024.

[6] J. Kirkland *et al.*, "Automated Detection of Crypto Ransomware Using Machine Learning and File Entropy Analysis Automated Detection of Crypto Ransomware Using Machine Learning and File Entropy Analysis," 2024.

[7] M. Ozturk, A. Demir, Z. Arslan, and O. Caliskan, "Dynamic behavioural analysis of privacy-breaching and data theft ransomware," 2024.

[8] F. Noorbehbahani and M. Saberi, "Ransomware detection with semi-supervised learning," in *2020 10th International Conference on Computer and Knowledge Engineering (ICCKE)*, IEEE, 2020, pp. 24–29.

[9] L. Chen, C.-Y. Yang, A. Paul, and R. Sahita, "Towards resilient machine learning for ransomware detection," *arXiv Prepr. arXiv1812.09400*, 2018.

[10] M. Masum, M. J. H. Faruk, H. Shahriar, K. Qian, D. Lo, and M. I. Adnan, "Ransomware Classification and Detection with Machine Learning Algorithms," *2022 IEEE 12th Annu. Comput. Commun. Work. Conf. CCWC 2022*, pp. 316–322, 2022, doi: 10.1109/CCWC54503.2022.9720869.

[11] M. A. Jerlin and K. Marimuthu, "A new malware detection system using machine learning techniques for API call sequences," *J. Appl. Secur. Res.*, vol. 13, no. 1, pp. 45–62, 2018.

[12] C. Ravi and R. Manoharan, "Malware detection using windows api sequence and machine learning," *Int. J. Comput. Appl.*, vol. 43, no. 17, pp. 12–16, 2012.

[13] Z. Zakaryayev and N. Z. Zakarya, "Advanced malware detection system," 2024.

[14] N. Kawaguchi and K. Omote, "Malware function classification using apis in initial behavior," in *2015 10th Asia Joint Conference on Information Security*, IEEE, 2015, pp. 138–144.

[15] W. Zhang, X. Li, and T. Zhu, "Entropy and memory forensics in ransomware analysis: Utilizing llama-7b for advanced pattern recognition," *Authorea Prepr.*, 2023.

[16] T. McIntosh *et al.*, "Ransomware reloaded: Re-examining its trend, research and mitigation in the era of data exfiltration," *ACM Comput. Surv.*, vol. 57, no. 1, pp. 1–40, 2024.

[17] S. Wang, Y. Li, and F. Chen, "Optimizing blue team strategies with reinforcement learning for enhanced ransomware defense simulations," *Authorea Prepr.*, 2024.

[18] Z. A. Genç, G. Lenzini, and P. Y. A. Ryan, "Next generation cryptographic ransomware," in *Secure IT Systems: 23rd Nordic Conference, NordSec 2018, Oslo, Norway, November 28-30, 2018, Proceedings 23*, Springer, 2018, pp. 385–401.

Received: 04/11/2025; Revised: 24/11/2025; Accepted: 23/12/2025; Published: 29/12/2025