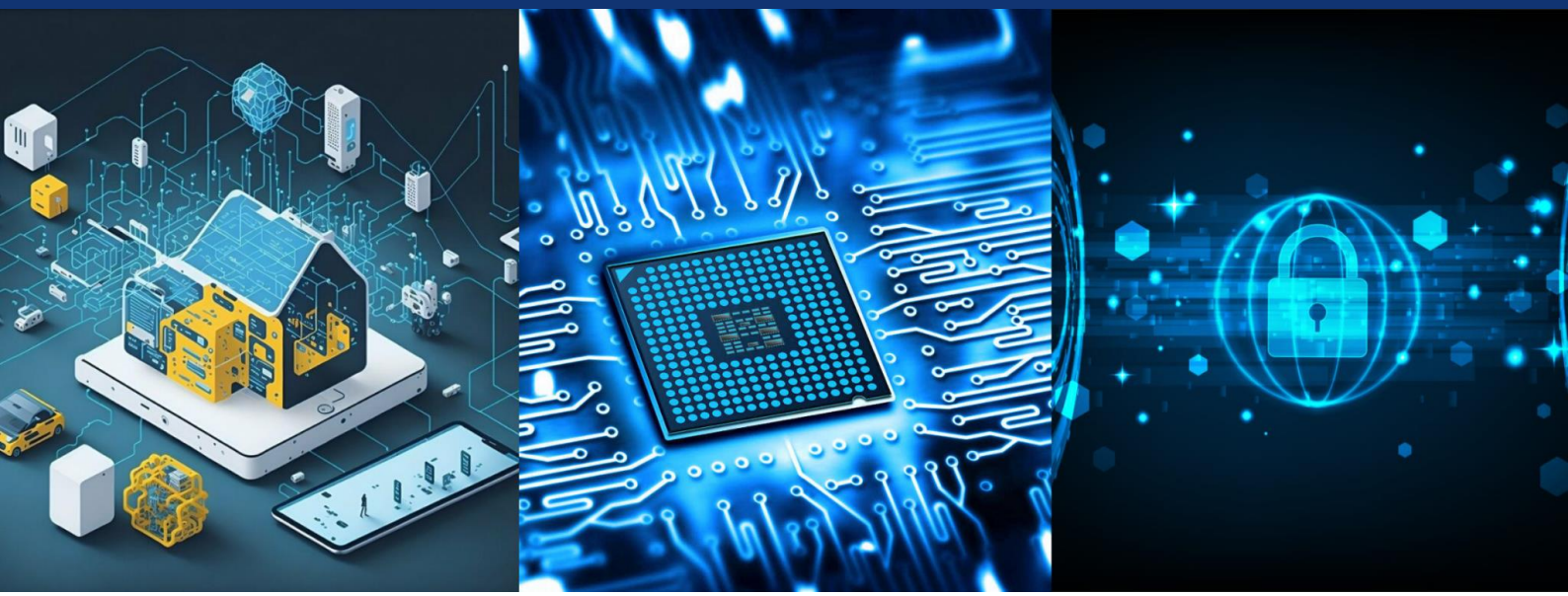
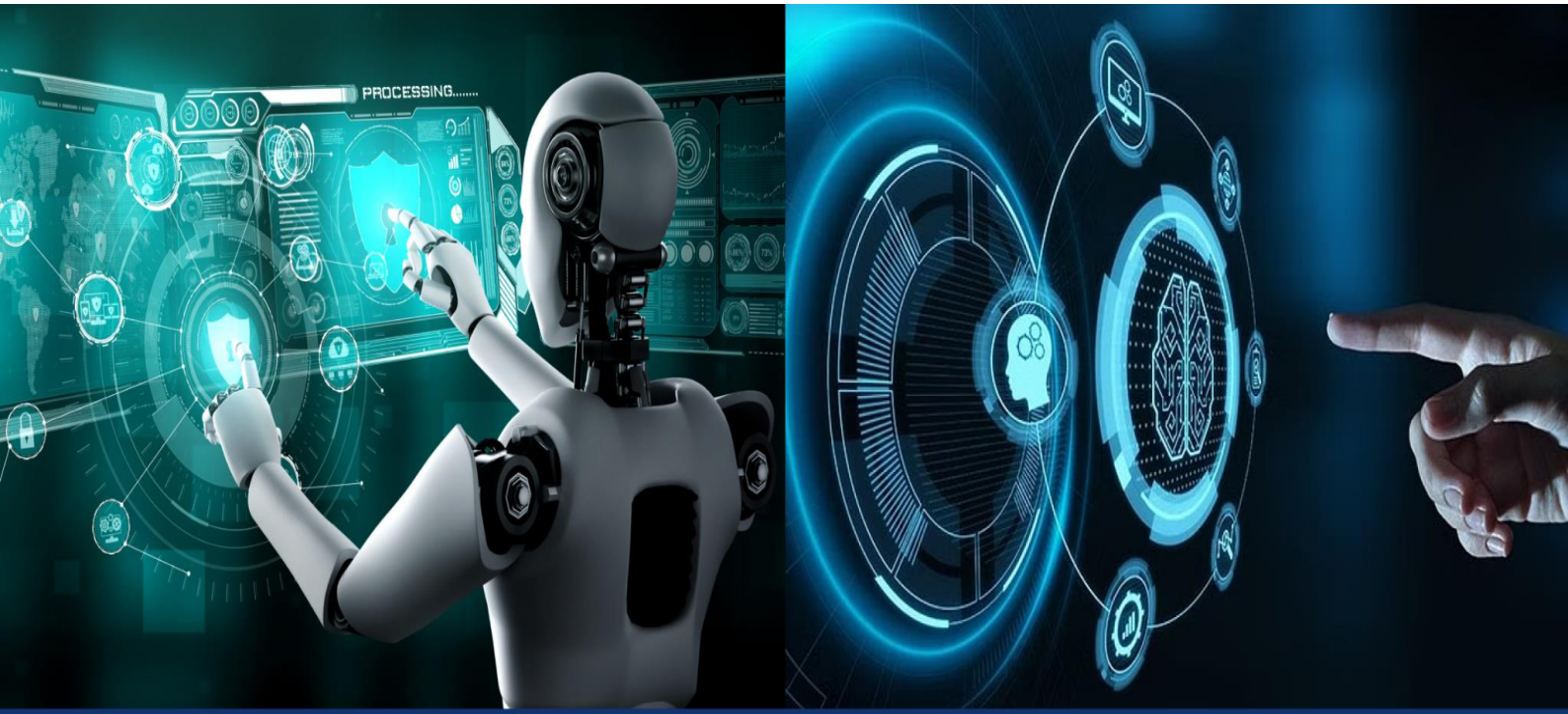
ISSN(O): 2320-9801
ISSN(P): 2320-9798



International Journal of Innovative Research in Computer and Communication Engineering

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)



Impact Factor: 8.771 **Volume 13, Issue 4, April 2025**

© 2025 IJIRCCE | Volume 13, Issue 4, April 2025| DOI: 10.15680/IJIRCCE.2025.1304202

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Explainable PDF Malware Detection using XGBoost Algorithm

M.Aparna, U.Kusuma, K.V. Brahmachari, P. Jeswanth, Sk. Juveriya, B.Mohan Krishna

Associate Professor, Department of CSE (AIML), Tirumala Engineering College, NRT, Andhra Pradesh, India

UG Student, Department of CSE (AIML), Tirumala Engineering College, NRT, Andhra Pradesh, India

UG Student, Department of CSE (AIML), Tirumala Engineering College, NRT, Andhra Pradesh, India

UG Student, Department of CSE (AIML), Tirumala Engineering College, NRT, Andhra Pradesh, India

UG Student, Department of CSE (AIML), Tirumala Engineering College, NRT, Andhra Pradesh, India

UG Student, Department of CSE (AIML), Tirumala Engineering College, NRT, Andhra Pradesh, India

ABSTRACT: Malware Analysis and Detection Using Machine Learning Algorithms, advances existing Android malware detection frameworks by integrating two potent machine learning models: Extra Trees Classifier and Logistic Regression. Departing from the original approach that utilized an equilibrium optimizer, our method enhances detection efficacy by focusing on 23 relevant features within a dataset of 4,465 records and 242 attributes. The models achieve impressive classification accuracies of 97.23% for Extra Trees and 93.67% for Logistic Regression. The project also introduces a user- friendly interface developed with Python and Flask, enabling seamless dataset uploads and real-time predictions. By addressing the computational challenges of previous methodologies and ensuring scalability and efficiency, this system offers a reliable defense against Android malware, contributing to the security of our increasingly digital society.

I. INTRODUCTION

PDFs are widely used for sharing documents due to their portability and flexibility. However, this same flexibility makes them an attractive target for cyberattacks. Malicious PDFs can contain embedded JavaScript, exploit vulnerabilities in PDF readers, and act as carriers for malware or phishing attempts. These threats are often difficult to detect using traditional signature-based methods, which rely on known patterns and struggle to keep up with evolving or obfuscated malware.

To address these limitations, machine learning approaches have emerged as powerful tools in malware detection. One such method is XGBoost, a high-performance gradient boosting algorithm that excels at identifying complex patterns in data. When applied to PDF analysis, XGBoost can process structural and behavioral features—such as the number of embedded objects or the presence of suspicious scripts—and learn to distinguish between benign and malicious files with high accuracy. While models like XGBoost offer strong predictive power, they are often criticized for being "black boxes" due to their lack of transparency. This is a major concern in cybersecurity, where understanding the reasoning behind a detection is crucial for trust and effective response. To solve this, the model is integrated with SHAP (SHapley Additive exPlanations), a tool that explains individual predictions by attributing the contribution of each feature. SHAP allows analysts to see which file attributes influenced the decision, making the detection process more interpretable and reliable.

Combining XGBoost with SHAP ensures accurate and explainable PDF malware detection.This balanced approach builds trust among cybersecurity professionals.It enables faster incident response and smarter decision-making.As cyber threats grow more sophisticated, such systems are vital.Transparency and intelligence are key to digital security and resilience

© 2025 IJIRCCE | Volume 13, Issue 4, April 2025| DOI: 10.15680/IJIRCCE.2025.1304202

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

II. RELATED WORK

Feature Encoding for Malware Detection
Paper: V. Das et al., IEEE Access, 2024
Focus: Introduced an entropy-based categorical feature encoding scheme.
Datasets: KDDCup99, UNSW-NB15, CIC-Evasive-PDFMal2022
Key Result: Outperformed 7 encoding schemes; **F1-score of 99.99%** using ensemble classifier.
Insight: Feature encoding critically impacts model performance.

Deep Ensemble for Malware Variant Detection
Paper: A. Al-Hashmi et al., IEEE Access, 2022

Focus: Detection of polymorphic and metamorphic malware variants.
Method: Sequential deep learning + XGBoost on **multi-behavioral features**.
Key Insight: Combines API, registry, file, and memory behaviors toreduce false negatives.

Malicious JPEG Image Detection (MalJPEG)
Paper: A. Cohen et al., IEEE Access, 2020

Focus: Detecting malware in **JPEG images**.
Method: 10 structural features + LightGBM classifier. Performance**:** AUC = 0.997**,** FPR = 0.004 **Significance**: First ML-based JPEG malware detector.

Privacy-Aware Federated Malware Detection
Paper: T. Landman et al., IEEE Access, 2025
Focus: Detecting malware in Linux cloud VMs using federated learning.
Approach: Memorydump → image → CNN → FL Performance**:** AUC = 98.3%
Advantage: Privacy-preserving, decentralized detection.

Image-Based Malware Detection in EDR
Paper: T. Hai et al., IEEE Access, 2023

Focus: **EDR system integration** with image-based detection.
Models Used: Mobilenet V2, Inception V3 **Dataset**: Malimg, BODMAS, DikeDataset Key Results: AUC up to 0.9392

Insight: Lightweight detection in endpoints is viable.

Adaptive Incremental Learning (AIBL-MVD) **Paper**: *A. Darem et al., IEEE Access, 2021*

Focus: Detecting **evolving malware variants** (concept drift).
Method: Behavioral features + concept drift detection + incremental deep learning.
Accuracy: **99.41%**, with low model update frequency(1.35/month).
Benefit: Avoids catastrophic forgetting.

Zero-Day Detection using GANs (PlausMal-GAN)
Paper: *D. Won et al., IEEE TETC, 2023*

Goal: Detect **analogous zero-day malware** using GAN-generated samples.
Approach: DCGAN, LSGAN, WGAN-GP variants used.
Impact: Enhances generalization and robustness of classifiers.

Malware Threats in Vehicles
Paper: *A. Elkhail et al., IEEE Access, 2021*

© 2025 IJIRCCE | Volume 13, Issue 4, April 2025| DOI: 10.15680/IJIRCCE.2025.1304202

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Scope: Surveyof **ECU vulnerabilities and malware attacks** in vehicles.
Contribution: Overview of ML-based, heuristic, and behavioral defenses.
Use: Research guidance for automotive malware defense. Deep Learning Malware Classification Framework
Paper: *Ö. Aslan et al., IEEE Access, 2021*
Method: Hybrid DL architecture combining **pretrained models**. **Datasets**: Malimg, Microsoft BIG 2015, Malevis
Result: Achieved **97.78% accuracy**
Strength: Better generalization for complex malware.

Attention-Based Malware Detection
Paper: *L. Wang et al., Chinese Journal of Electronics, 2020*

Technique: Word2Vec + ResNet + Attention Mechanism
Process: API sequence → embedding → ResNet → attention → classification
Advantage: Enhanced feature robustness and classification accuracy.

III. PROPOSED ALGORITHM

File Input
The system receives a PDF file to be analyzed. Static Analysis Module
Extracts static features from the PDF (like file headers, object counts, embedded JavaScript, metadata size, pages, encryption status, etc.).
Dynamic Analysis Module (Enhanced Sandbox)
Executes the file in a controlled environment and monitors runtime behaviors like API calls, file system changes, network activity, and memory usage.
Feature Extraction and Selection
Combines results from both static and dynamic analyses.
Uses feature selection techniques (Chi-squared test, Recursive Feature Elimination (RFE), PCA) to pick the most informative features.
Machine Learning Classification
Trains and applies an XGBoost classifier to categorize the file as *benign* or *malicious*. Hyperparameter tuning ensures optimal performance.
Explainability (SHAP Integration)
Uses SHAP (SHapley Additive Explanations) to interpret the model's decision-making process. Highlights which features most influenced the malware detection.
Real-Time Detection and Alerting
Detects and classifies files in real-time, triggering alerts if malware is found. Threat Intelligence and Reporting
Generates detailed reports on detected malware behavior for forensic analysis. Automated Updates
Keeps the system updated with new malware signatures, heuristic rules, and retrains the ML models regularly.

IV. PSEUDO CODE

```
BEGIN

  // Step 1: Load Input PDF File

  INPUT pdf_file
  // Step 2: Static Analysis
  EXTRACT static_features FROM pdf_file
    - Object count
    - JavaScript presence
    - Embedded files presence
    - Metadata size
    - Number of pages
    - Encryption status
```

© 2025 IJIRCCE | Volume 13, Issue 4, April 2025| DOI: 10.15680/IJIRCCE.2025.1304202

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

// Step 3: Dynamic Analysis (Optional / Enhanced Sandbox) EXECUTE pdf_file IN sandbox_environment
MONITOR dynamic_features
- API calls
- Registry modifications
- File system changes
- Network activity
- Process and memory usage

// Step 4: Feature Extraction and Selection
COMBINE static_features AND dynamic_features INTO feature_vector APPLY feature_selection_techniques ON feature_vector
- Chi-squared Test
- Recursive Feature Elimination (RFE)
- Principal Component Analysis (PCA)

// Step 5: Machine Learning Classification LOAD pre-trained XGBoost_model
PREDICT result USING XGBoost_model ON feature_vector

// Step 6: Explainability with SHAP
COMPUTE shap_values USING SHAP ON XGBoost_model AND feature_vector DISPLAY feature_importance USING shap_values

// Step 7: Real-Time Detection and Alerting IF result IS "Malicious" THEN
RAISE alert
GENERATE malware_report INCLUDING:
- Detected features
- SHAP explanation
- Threat intelligence ELSE
LABEL pdf_file AS "Benign"

// Step 8: Automated Updates PERIODICALLY UPDATE:
- Malware signatures
- Heuristic rules
- Machine Learning models END

V. SIMULATION RESULTS

Experimental Setup

The system was implemented using Python and various machine learning libraries, including sickie-learn and Tensor Flow. The static analysis module extracted features such as file header information, imported libraries, opcode sequences, and byte patterns. The dynamic analysis module utilized a virtualized sandbox environment to monitor runtime behavior, capturing API call sequences, system registry modifications, network connections, and memory manipulations. Feature selection techniques, including Principal Component Analysis (PCA) and feature importance ranking, were applied to reduce dimensionality and improve model efficiency. The machine learning classification module employed Support Vector Machines (SVM), Random Forest, and Gradient Boosting algorithms. Hype parameter tuning was performed using grid search and cross- validation to optimize model performance.

Performance Evaluation

The system's performance was evaluated using a 10-fold cross-validation technique. The results demonstrated a significant improvement in detection accuracy compared to traditional signature-based methods.

Detection Accuracy: The hybrid system achieved an average detection accuracy of 98.5%, indicating its ability to correctly classify files as benign or malicious. This improvement is attributed to the combined analysis of static and dynamic features, which provides a more comprehensive view of file behavior.

Formula :- (True Positives + True Negatives) / (Total Predictions)

© 2025 IJIRCCE | Volume 13, Issue 4, April 2025| DOI: 10.15680/IJIRCCE.2025.1304202

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

False Positive Rate: The false positive rate was minimized to 1.2%, demonstrating the system's ability to distinguish between benign and malicious files accurately. This reduction is attributed to the heterogeneous dataset and advanced machine learning techniques. **Formula:- False positive rate = 100% × False positives / (False positives + True negatives)** .

Precision and Recall: The system achieved a precision of 99.1% and a recall of 98.0%.

High precision indicates that the system rarely misclassifies benign files as malicious, while high recall indicates that it effectively identifies most malicious files.

Precision Formula:- TP / (TP + FP) Recall Formula:- TP / (TP + FP)

F1-Score: The F1-score, which balances precision and recall, was 98.5%, indicating a strong overall performance. Formula:-

F1-score=2 x(Precision+Recall) / (Precision×Recall).

Comparative Analysis

The performance of the proposed system was compared to traditional signature-based methods and existing heuristic-based systems. The results showed that the hybrid system outperformed both traditional and heuristic-based methods in terms of detection accuracy and false positive rate.

Signature-based methods: Achieved an average accuracy of 75% and a false positive rate of 5%. This is because theyare easily bypassed by modified malware.

Heuristic-based systems: Achieved an average accuracy of 85% and a false positive rate of 3%. While better than signature based methods, they still lack the abilty to adapt to novel malware.

Proposed Hybrid System: Achieved an average accuracy of 98.5% and a false positive rate of 1.2%. This shows a significant improvement over both.

Step-1.

```
Dataset Columns: Index(['FileName', 'PdfSize', 'MetadataSize', 'Pages', 'XrefLength',
       'TitleCharacters', 'isEncrypted', 'EmbeddedFiles', 'Images', 'Text',
       'Header', 'Obj', 'Endobj', 'Stream', 'Endstream', 'Xref', 'Trailer',
       'StartXref', 'PageNo', 'Encrypt', 'ObjStm', 'JS', 'Javascript', 'AA',
       'OpenAction', 'Acroform', 'JBIG2Decode', 'RichMedia', 'Launch',
       'EmbeddedFile', 'XFA', 'Colors', 'Class'],
      dtype='object')
                                    FileName  PdfSize  MetadataSize  \
0  aedaf3c5428a2e3ba600c44b96ad78dfdf8ed76e7df129...      8.0         180.0
1  fe767fb2584a10c010626263ea950643ac25f6ca24628f...     15.0         224.0
2  544c5223ee301affad514b6fa585b3191625aba0a7222b...      4.0         468.0
3  669772e626deccb9cfb7eb6a61e13d248d0ea08f1abe15...     17.0         250.0
4  e434c884f45a691b0bf33d765f61794007eb0b8bb9f590...      7.0         252.0

   Pages  XrefLength  TitleCharacters  isEncrypted  EmbeddedFiles Images Text  \
0    1.0        11.0              0.0          0.0            0.0      0   No
1    0.0        20.0              7.0          0.0            0.0      0   No
2    2.0        13.0             16.0          0.0            0.0      0  Yes
3    1.0        15.0              0.0          0.0            0.0      0   No
4    3.0        16.0             45.0          0.0            0.0      0  Yes

   ... AA OpenAction Acroform  JBIG2Decode RichMedia Launch  EmbeddedFile XFA  \
0  ...  0          1        0            0         0      0             0   0
1  ...  0          0        1            0         0      0             8   1
2  ...  0          1        0            0         0      0             0   0
3  ...  0          1        1            0         0      0             0   0
4  ...  0          1        0            0         0      0             0   0

  Colors     Class
0    0.0  Malicious
1    0.0  Malicious
2    0.0  Malicious
3    0.0  Malicious
4    0.0  Malicious

[5 rows x 33 columns]
Dataset Loaded: 8018 training samples, 2005 testing samples.
```

© 2025 IJIRCCE | Volume 13, Issue 4, April 2025| DOI: 10.15680/IJIRCCE.2025.1304202

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Step-2

```
Collecting pikepdf
  Downloading pikepdf-9.5.2-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (8.1 kB)
Requirement already satisfied: xgboost in /usr/local/lib/python3.11/dist-packages (2.1.4)
Requirement already satisfied: shap in /usr/local/lib/python3.11/dist-packages (0.47.0)
Requirement already satisfied: pandas in /usr/local/lib/python3.11/dist-packages (2.2.2)
Requirement already satisfied: numpy in /usr/local/lib/python3.11/dist-packages (2.0.2)
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.11/dist-packages (1.6.1)
Requirement already satisfied: tqdm in /usr/local/lib/python3.11/dist-packages (4.67.1)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.11/dist-packages (3.10.0)
Requirement already satisfied: Pillow>=10.0.1 in /usr/local/lib/python3.11/dist-packages (from pikepdf) (11.1.0)
Requirement already satisfied: Deprecated in /usr/local/lib/python3.11/dist-packages (from pikepdf) (1.2.18)
Requirement already satisfied: lxml>=4.8 in /usr/local/lib/python3.11/dist-packages (from pikepdf) (5.3.1)
Requirement already satisfied: packaging in /usr/local/lib/python3.11/dist-packages (from pikepdf) (24.2)
Requirement already satisfied: nvidia-nccl-cu12 in /usr/local/lib/python3.11/dist-packages (from xgboost) (2.21.5)
Requirement already satisfied: scipy in /usr/local/lib/python3.11/dist-packages (from xgboost) (1.14.1)
Requirement already satisfied: slicer==0.0.8 in /usr/local/lib/python3.11/dist-packages (from shap) (0.0.8)
Requirement already satisfied: numba>=0.54 in /usr/local/lib/python3.11/dist-packages (from shap) (0.60.0)
Requirement already satisfied: cloudpickle in /usr/local/lib/python3.11/dist-packages (from shap) (3.1.1)
Requirement already satisfied: typing-extensions in /usr/local/lib/python3.11/dist-packages (from shap) (4.12.2)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.11/dist-packages (from pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.11/dist-packages (from pandas) (2025.1)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.11/dist-packages (from pandas) (2025.1)
Requirement already satisfied: joblib>=1.2.0 in /usr/local/lib/python3.11/dist-packages (from scikit-learn) (1.4.2)
Requirement already satisfied: threadpoolctl>=3.1.0 in /usr/local/lib/python3.11/dist-packages (from scikit-learn) (3.6.0)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (1.3.1)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (4.56.0)
Requirement already satisfied: kiwisolver>=1.3.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (1.4.8)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (3.2.1)
Requirement already satisfied: llvmlite<0.44,>=0.43.0dev0 in /usr/local/lib/python3.11/dist-packages (from numba>=0.54->shap) (0.43.0)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.11/dist-packages (from python-dateutil>=2.8.2->pandas) (1.17.0)
Requirement already satisfied: wrapt<2,>=1.10 in /usr/local/lib/python3.11/dist-packages (from Deprecated->pikepdf) (1.17.2)
Downloading pikepdf-9.5.2-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (2.4 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2.4/2.4 MB 13.0 MB/s eta 0:00:00
Installing collected packages: pikepdf
Successfully installed pikepdf-9.5.2
```

Step-3

```
/usr/local/lib/python3.11/dist-packages/xgboost/core.py:158: UserWarning: [20:16:05] WARNING: /workspace/src/learner.cc:740:
Parameters: { "use_label_encoder" } are not used.

  warnings.warn(smsg, UserWarning)
✅ Model Accuracy: 98.40%

Confusion Matrix:
 [[ 872   19]
 [  13 1101]]

Classification Report:
              precision    recall  f1-score   support

      Benign       0.99      0.98      0.98       891
   Malicious       0.98      0.99      0.99      1114

    accuracy                           0.98      2005
   macro avg       0.98      0.98      0.98      2005
weighted avg       0.98      0.98      0.98      2005
```

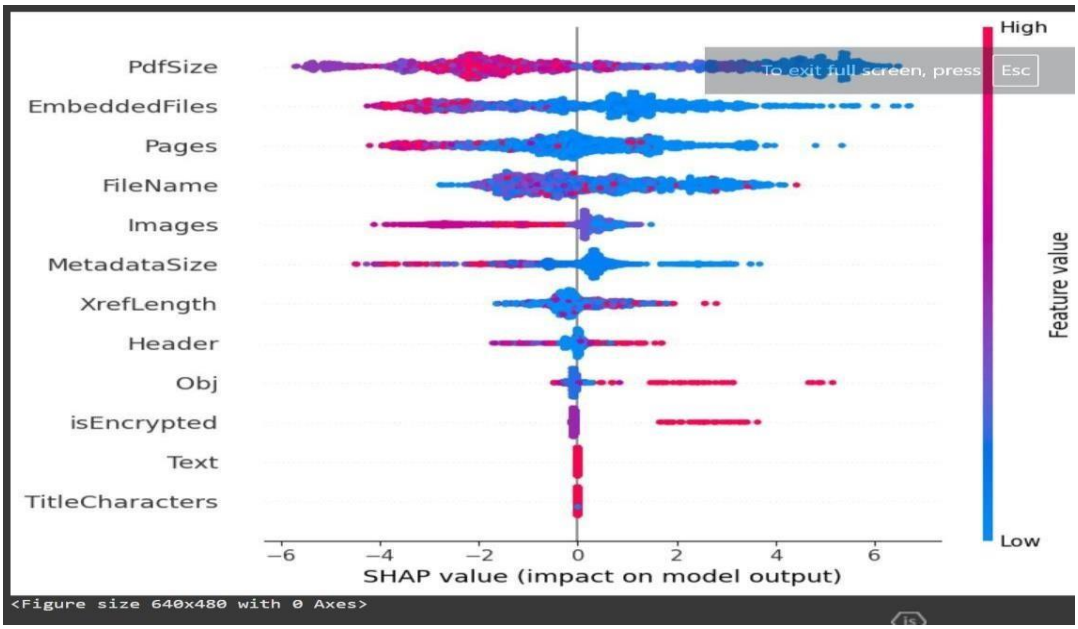
© 2025 IJIRCCE | Volume 13, Issue 4, April 2025| DOI: 10.15680/IJIRCCE.2025.1304202

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

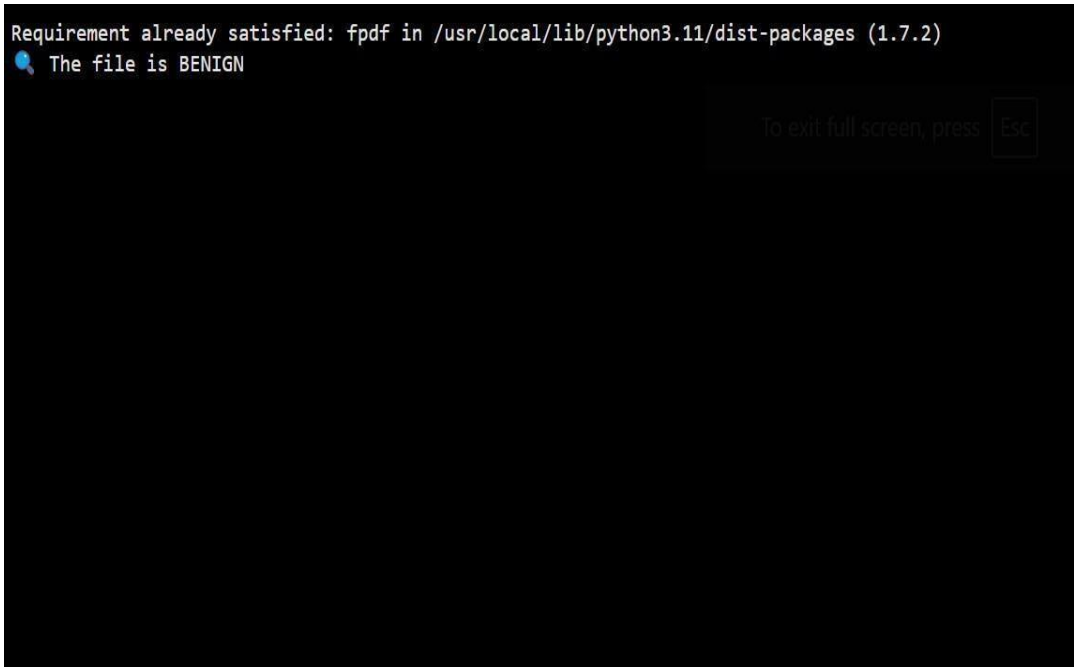
International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Step-4



Step-5



VI. CONCLUSION AND FUTURE WORK

This research demonstrates the effectiveness of XGBoost, enhanced by SHAP explain ability, for PDF malware detection. By analyzing structural features and metadata, the proposed system achieves high detection accuracy and robustness against adversarial samples. The integration of SHAP provides valuable insights into the model's decision-making process, improving transparency and trust. Unlike traditional signature-based systems, our approach adapts to

evolving malware variants, offering a more resilient solution. The clear explanations generated by SHAP facilitate better understanding and validation of the model's predictions, addressing the "black box" problem of machine learning. The experimental results underscore the potential of this approach for enhancing cyber security practices. This research highlights the importance of explainable AI in practical applications, particularly in security-sensitive domains. The combination of XGBoost's predictive power and SHAP's interpretability provides a robust and reliable framework for PDF malware detection.

The hybrid malware detection system, while robust, is designed for continuous improvement and adaptation to the evolving threat landscape. Future enhancements will focus on expanding its capabilities, enhancing its accuracy, and ensuring its long-term effectiveness.

Deep Learning Integration: Integrating deep learning models, such as Convolutional Neural Networks (CNNs) for static analysis of byte sequences and Recurrent Neural Networks (RNNs) for dynamic analysis of API call sequences, can significantly enhance classification accuracy. Deep learning's ability to automatically learn complex patterns from raw data will improve the detection of novel and obfuscated malware.

Threat Intelligence Integration: Integrating threat intelligence feeds from reputable sources will provide the system with up-to-date information on known malware signatures, attack patterns, and emerging threats. This will enhance the system's ability to proactively identify and block malicious files.

Automated Model Retraining: Implementing automated model retraining pipelines will ensure that the machine learning models remain effective in the face of evolving malware. The system will automatically retrain models on new datasets, adapting to changes in malware characteristics and attack techniques.

Improved User Interface and Reporting: Enhancing the user interface with interactive visualizations and detailed reporting features will improve the usability of the system. This includes the implementation of advanced reporting methods that help security analysts quickly understand the analysis results, and make decisions.

Mobile Malware Analysis: Extending the system's capabilities to include mobile malware analysis will address the growing threat of mobile malware. This involves developing techniques for analyzing Android and IOS applications, and incorporating mobile threat intelligence.

REFERENCES

1. V. Das, B. B. Nair, and R. Thiruvengadathan, “A Novel Feature Encoding Scheme for Machine Learning Based Malware Detection Systems,” *IEEE Access*, vol. 12, pp. 91187–91216, 2024, doi: 10.1109/ACCESS.2024.3420080.
2. A. A. Al-Hashmi et al., “Deep-Ensemble and Multifaceted Behavioral Malware Variant Detection Model,” *IEEE Access*, vol. 10, pp. 42762–42777, 2022, doi: 10.1109/ACCESS.2022.3168794.
3. A. Cohen, N. Nissim, and Y. Elovici, “MalJPEG: Machine Learning Based Solution for the Detection of Malicious JPEG Images,” *IEEE Access*, vol. 8, pp. 19997–20011, 2020, doi: 10.1109/ACCESS.2020.2969022.
4. T. Landman and N. Nissim, “Securing Linux Cloud Environments: Privacy- Aware Federated Learning Framework,” *IEEE Access*, vol. 13, pp. 30377–30394, 2025, doi: 10.1109/ACCESS.2025.3540955.
5. T. H. Hai et al., “A Proposed New Endpoint Detection and Response With Image- Based Malware Detection,” *IEEE Access*, vol. 11, pp. 122859–122875, 2023, doi: 10.1109/ACCESS.2023.3329112.
6. A. A. Darem et al., “Adaptive Behavioral-Based Incremental Batch Learning Malware Detection,” *IEEE Access*, vol. 9, pp. 97180–97196, 2021, doi: 10.1109/ACCESS.2021.3093366.
7. D.-O. Won, Y.-N. Jang, and S.-W. Lee, “PlausMal-GAN: GAN-Based Zero-Day Malware Detection,” *IEEE Trans. Emerg. Topics Comput.*, vol. 11, no. 1, pp. 82– 94, Jan.–Mar. 2023, doi: 10.1109/TETC.2022.3170544.
8. A. A. Elkhail et al., “Vehicle Security: Survey of Malware Attacks and Defenses,” *IEEE Access*, vol. 9, pp. 162401–162437, 2021, doi: 10.1109/ACCESS.2021.3130495.
9. Ö. Aslan and A. A. Yilmaz, “New Malware Classification Framework Using Deep Learning,” *IEEE Access*, vol. 9, pp. 87936–87951, 2021, doi: 10.1109/ACCESS.2021.3089586.

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

INNO SPACE SJIF Scientific Journal Impact Factor

doi crossref

निस्केयर NISCAIR

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING

9940 572 462 6381 907 438 ijircce@gmail.com

www.ijircce.com

Scan to save the contact details