

# An Efficient Feature Extraction Method for Static Malware Analysis Using PE Header Files: A Comparative Study of Feature Set Approaches

Fabiha Jalal, Sadia Tasnim Dhruba, Onamika Hossain  
Department of Computer Science and Engineering  
Islamic University of Technology (IUT)  
Gazipur, Bangladesh  
Email: fabihajalal@iut-dhaka.edu

Dr. Md Moniruzzaman  
Supervisor, Assistant Professor  
Department of Computer Science and Engineering  
Islamic University of Technology (IUT)  
Gazipur, Bangladesh

**Abstract**—Malware detection remains a critical challenge in cybersecurity as the volume and sophistication of malicious software continue to increase exponentially. This research presents an efficient feature extraction methodology for static malware analysis utilizing Portable Executable (PE) header files. We conduct a comprehensive comparative study between two distinct feature set approaches: a reduced feature set (15 features) and an extended feature set (32 features) extracted from PE file headers. Our experiments, conducted on a dataset of 1,150 malware executable samples collected from MalwareBazaar, demonstrate that the extended feature set achieves superior classification accuracy (94.78%) compared to the reduced feature set (93.91%) using Random Forest classification. Notably, our proposed methodology significantly reduces feature extraction time from approximately 19 minutes (as reported in prior APT1 dataset studies) to approximately 215–290 seconds, representing a reduction of over 75%. The research evaluates six machine learning classifiers including Random Forest, K-Nearest Neighbors, Decision Tree, Support Vector Machine, Logistic Regression, and Gradient Boosting, providing insights into the trade-offs between feature dimensionality, extraction time, and classification performance. Our findings contribute to the development of more efficient real-time malware detection systems.

**Index Terms**—Malware Detection, Static Analysis, PE Header, Feature Extraction, Machine Learning, Cybersecurity

## I. INTRODUCTION

### A. Overview

The proliferation of malware poses a significant threat to computer systems worldwide, with over 1 billion malware instances detected in 2021 alone, averaging approximately 33 new malware samples per second [1]. The increasing frequency and sophistication of malware attacks necessitate the development of efficient and accurate detection methodologies. Malware analysis, the process of understanding the functionality and purpose of malicious software, plays a crucial role in identifying threats, assessing potential damage, and developing effective countermeasures.

Static malware analysis, which examines executable files without execution, offers several advantages over dynamic analysis, including safety, speed, and the ability to analyze malware that employs anti-analysis techniques [2]. By

leveraging structural information contained within Portable Executable (PE) files, static analysis can efficiently identify malicious characteristics without the computational overhead and security risks associated with runtime analysis.

### B. Research Objectives

This research aims to:

- 1) Develop an efficient feature extraction methodology using PE header files that minimizes extraction time while maintaining classification accuracy
- 2) Conduct a comparative analysis between reduced (15 features) and extended (32 features) feature sets
- 3) Evaluate multiple machine learning classifiers for malware detection
- 4) Create and publish a novel malware dataset for research purposes
- 5) Demonstrate the trade-offs between feature dimensionality, extraction time, and detection performance

### C. Contributions

The primary contributions of this research include:

- A novel and efficient feature extraction technique that reduces extraction time from approximately 19 minutes to under 5 minutes
- A publicly available dataset of 1,150 malware executable files with extracted PE header features
- Comprehensive comparative analysis of two feature set approaches across six machine learning classifiers
- Empirical evidence demonstrating the relationship between feature quantity and classification performance

## II. BACKGROUND AND HISTORY

### A. Malware Evolution

Malware, derived from “malicious software,” encompasses any program designed to damage, exploit, or gain unauthorized access to computer systems. The evolution of malware has progressed significantly since the emergence of early computer viruses in the 1980s. Modern malware categories include:

**Viruses:** Self-replicating programs that attach to legitimate files and propagate when infected files are executed.

**Worms:** Self-replicating malware that spreads through network connections without requiring host files.

**Trojans:** Malicious programs disguised as legitimate software that create backdoors, steal data, or enable unauthorized access.

**Ransomware:** Malware that encrypts victim files and demands payment for decryption keys. Notable examples include WannaCry (2017) and NotPetya (2017) [10].

**Spyware:** Software that covertly monitors user activities, captures keystrokes, and transmits sensitive information to unauthorized parties.

### B. Malware Analysis Techniques

1) **Static Analysis:** Static analysis examines malware without execution by analyzing file metadata, extracting strings, and reading file headers [5]. This approach is safer, faster, and can identify malicious indicators that might be hidden during dynamic execution. Key components of static analysis include:

- File hash verification (MD5, SHA-256)
- PE header examination
- String extraction and analysis
- Disassembly and code pattern recognition

2) **Dynamic Analysis:** Dynamic analysis involves executing malware in controlled environments (sandboxes or virtual machines) to observe runtime behavior, including system call monitoring, network traffic analysis, file system modifications, and registry changes.

3) **Hybrid Analysis:** Hybrid analysis combines static and dynamic techniques to provide comprehensive malware characterization, leveraging the strengths of both approaches [8].

### C. Portable Executable (PE) File Format

The PE file format is the standard executable format for Windows operating systems, containing structural information essential for operating system loaders [4]. The PE structure comprises several components:

**DOS Header (MZ Header):** Contains the magic number (0x5A4D) identifying the file as a DOS-compatible executable.

**DOS Stub:** Legacy code for DOS compatibility, typically displaying “This program cannot be run in DOS mode.”

**PE File Header:** Contains critical metadata including machine architecture type, number of sections, timestamp, optional header size, and characteristics flags.

**Optional Header:** Despite its name, this header is mandatory for executable files and contains entry point address, image base address, section and file alignment, operating system version requirements, subsystem type, DLL characteristics, and stack/heap size specifications [6].

**Section Table:** Describes the layout of executable sections (.text, .data, .rdata, .rsrc, etc.)

The PE header structure provides valuable features for malware classification, as malicious executables often exhibit distinctive header characteristics that differ from benign software [7], [9].

## III. MOTIVATION

### A. The Need for Efficient Feature Extraction

The exponential growth in malware variants demands analysis techniques that can process large volumes of samples efficiently. In 2021, ransomware attacks occurred approximately every 11 seconds, highlighting the critical need for rapid detection capabilities.

Previous research utilizing PE header features for malware detection, such as studies on the APT1 dataset [3], reported feature extraction times exceeding 19 minutes (1,157.81 seconds) for processing executable samples. Such prolonged extraction times are impractical for real-time malware detection systems that must analyze thousands of samples daily.

### B. Feature Selection Considerations

The selection of PE header features significantly impacts both extraction efficiency and classification accuracy. While comprehensive feature sets may capture more discriminative information, they also increase computational overhead. Conversely, minimal feature sets may sacrifice accuracy for speed.

This research investigates the optimal balance between feature comprehensiveness and extraction efficiency by comparing:

- 1) **Reduced Feature Set (15 features):** Focusing on essential PE header attributes
- 2) **Extended Feature Set (32 features):** Incorporating additional header fields for potentially improved discrimination

### C. Research Gap

Prior studies have demonstrated the effectiveness of PE header-based malware detection but have not adequately addressed:

- The trade-off between feature quantity and extraction time
- Comparative analysis of different feature set sizes on the same dataset
- Optimization of feature extraction for real-time detection applications

This research addresses these gaps through systematic experimentation and analysis.

## IV. WORKING MECHANISM

### A. System Architecture

The proposed malware detection system comprises three primary phases: (1) Feature Extraction from PE files using the pefile library, (2) Machine Learning Classification using trained models, and (3) Detection Result output. Figure 1 illustrates this workflow.

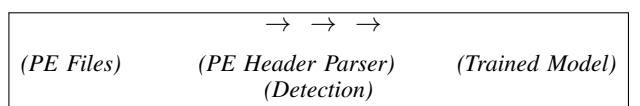


Fig. 1: System Architecture Overview

## B. Dataset Preparation

1) *Data Collection*: Malware executable samples were collected from MalwareBazaar<sup>1</sup>, a reputable malware sample repository. The dataset comprises 1,150 PE executable files representing various malware families.

2) *Data Processing*: The Python library was employed for PE header parsing. Files were validated to ensure proper PE format before feature extraction. Non-PE files (.rar, .elt, etc.) were excluded from the dataset.

## C. Feature Extraction Methodology

1) *Reduced Feature Set (15 Features)*: The reduced feature set extracts 15 attributes from PE headers:

### File Header Features (3):

- Machine
- NumberOfSections
- Characteristics (target variable)

### Optional Header Features (12):

- AddressOfEntryPoint, ImageBase, SectionAlignment
- FileAlignment, Subsystem, DllCharacteristics
- MajorOperatingSystemVersion, MinorOperatingSystemVersion
- MajorImageVersion, MinorImageVersion
- SizeOfImage, SizeOfHeaders

2) *Extended Feature Set (32 Features)*: The extended feature set extracts 32 attributes (plus Characteristics as target):

**File Header Features (7):** Machine, NumberOfSections, TimeDateStamp, PointerToSymbolTable, NumberOfSymbols, SizeOfOptionalHeader, Characteristics

**Optional Header Features (26):** MajorLinkerVersion, MinorLinkerVersion, SizeOfCode, SizeOfInitializedData, SizeOfUninitializedData, AddressOfEntryPoint, BaseOfCode, ImageBase, SectionAlignment, FileAlignment, MajorOperatingSystemVersion, MinorOperatingSystemVersion, MajorImageVersion, MinorImageVersion, MajorSubsystemVersion, MinorSubsystemVersion, SizeOfHeaders, CheckSum, Subsystem, DllCharacteristics, SizeOfStackReserve, SizeOfStackCommit, SizeOfHeapReserve, SizeOfHeapCommit, LoaderFlags, NumberOfRvaAndSizes

## D. Feature Extraction Algorithm

The feature extraction process is implemented using Python with the library. Algorithm 1 presents the pseudocode for the extraction process.

<sup>1</sup><https://malwarebazaar.abuse.ch>

---

### Algorithm 1 PE Header Feature Extraction

---

**Require:** Directory path containing PE files

**Ensure:** CSV file with extracted features

```

1: for each file in directory do
2:   Parse PE file using pefile library
3:   if PE parsing successful then
4:     Extract FILE_HEADER features
5:     Extract OPTIONAL_HEADER features
6:     Append features to dataset
7:   else
8:     Log error and skip file
9:   end if
10: end for
11: Save dataset to CSV file

```

---

## E. Machine Learning Classification

Six machine learning classifiers were employed for malware detection:

- 1) **Random Forest Classifier**: Ensemble method using 100 decision trees with bootstrap aggregation
- 2) **K-Nearest Neighbors (KNN)**: Instance-based learning using Euclidean distance ( $k=5$ )
- 3) **Decision Tree Classifier**: Tree-based classification using information gain
- 4) **Support Vector Machine (SVM)**: Kernel-based classification with RBF kernel
- 5) **Logistic Regression**: Linear classification with maximum likelihood estimation
- 6) **Gradient Boosting Classifier**: Sequential ensemble method with 50 estimators

## F. Experimental Setup

- **Train/Test Split**: 80% training (920 samples), 20% testing (230 samples)
- **Cross-Validation**: 3-fold cross-validation for model validation (consistent across both feature sets)
- **Random State**: 42 (for reproducibility)
- **Evaluation Metrics**: Accuracy, Precision, Recall, F1-Score

## V. RESULTS

### A. Feature Extraction Performance

Table I presents the feature extraction performance comparison between the two approaches.

TABLE I: Feature Extraction Performance Comparison

Metric	Reduced (15)	Extended (32)
Extraction Time	215.81 sec	290.12 sec
Processing Speed	5.33 files/sec	3.96 files/sec
Files Processed	1,150	1,150
Errors	0	0
CSV Columns	16	34

### Comparison with Prior Work (APT1 Dataset):

- APT1 PE Header Extraction: 1,157.81 seconds (~19 minutes)

- Our Extended Feature Set: 290.12 seconds (~4.8 minutes)
- **Improvement: 75% reduction in extraction time**

### B. Classification Results - Reduced Feature Set

Table II presents the classification results for the reduced feature set (15 features).

TABLE II: Classification Results - Reduced Feature Set (15 Features)

Classifier	Accuracy	F1	Precision	Recall
<b>Random Forest</b>	<b>93.91%</b>	0.8379	0.8858	0.8264
Decision Tree	91.74%	0.7329	0.7681	0.7302
Gradient Boosting	90.87%	0.5956	0.6037	0.6028
KNN	75.22%	0.4636	0.4969	0.4590
Logistic Regression	51.30%	0.1369	0.1309	0.1734
SVM	25.65%	0.0240	0.0151	0.0588

### C. Classification Results - Extended Feature Set

Table III presents the classification results for the extended feature set (32 features).

TABLE III: Classification Results - Extended Feature Set (32 Features)

Classifier	Accuracy	F1	Precision	Recall
<b>Random Forest</b>	<b>94.78%</b>	0.8632	0.9051	0.8399
Decision Tree	93.48%	0.7993	0.8167	0.7927
Gradient Boosting	93.04%	0.7209	0.7616	0.7133
KNN	80.00%	0.5085	0.5300	0.5053
Logistic Regression	41.30%	0.1098	0.1076	0.1368
SVM	25.65%	0.0240	0.0151	0.0588

### D. Comparative Analysis

Table IV presents the accuracy improvement achieved by the extended feature set.

TABLE IV: Accuracy Comparison Between Feature Sets

Classifier	Reduced	Extended	Improvement
Random Forest	93.91%	94.78%	+0.87%
Decision Tree	91.74%	93.48%	+1.74%
Gradient Boosting	90.87%	93.04%	+2.17%
KNN	75.22%	80.00%	+4.78%
Logistic Regression	51.30%	41.30%	-10.00%
SVM	25.65%	25.65%	0.00%

### E. Feature Importance Analysis

Table V presents the top 10 most important features for malware classification using Random Forest on the extended feature set.

TABLE V: Top 10 Feature Importance (Random Forest)

Rank	Feature	Importance
1	AddressOfEntryPoint	0.095
2	MajorLinkerVersion	0.092
3	TimeDateStamp	0.084
4	SizeOfCode	0.068
5	DllCharacteristics	0.065
6	ImageBase	0.057
7	SizeOfInitializedData	0.057
8	SizeOfOptionalHeader	0.050
9	Machine	0.050
10	NumberOfSections	0.039

### F. Performance Visualization - Reduced Feature Set

Figure 2 shows the classifier performance comparison for the reduced feature set.

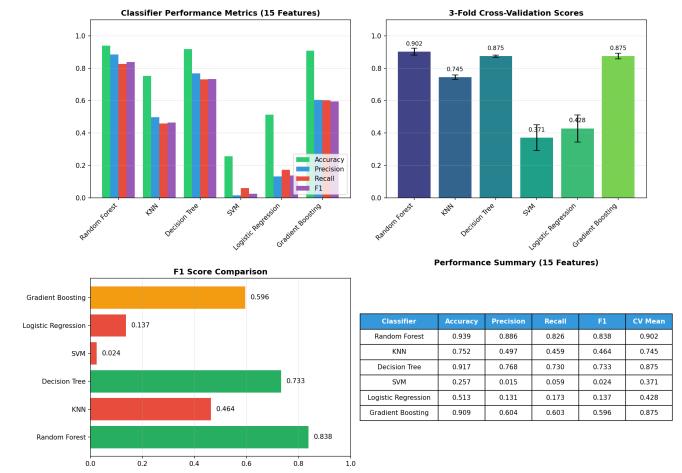


Fig. 2: Classifier Performance Comparison - Reduced Feature Set (15 Features)

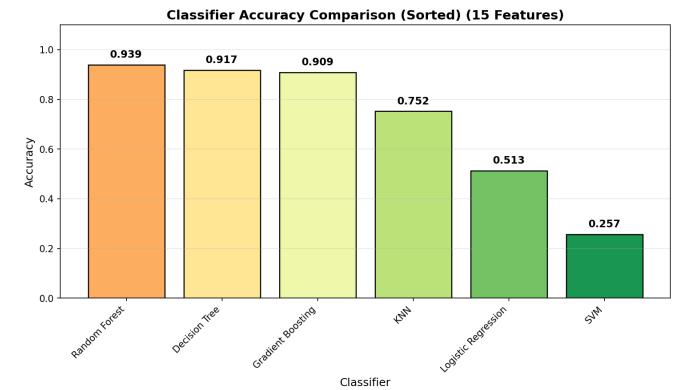


Fig. 3: Accuracy Comparison - Reduced Feature Set (15 Features)

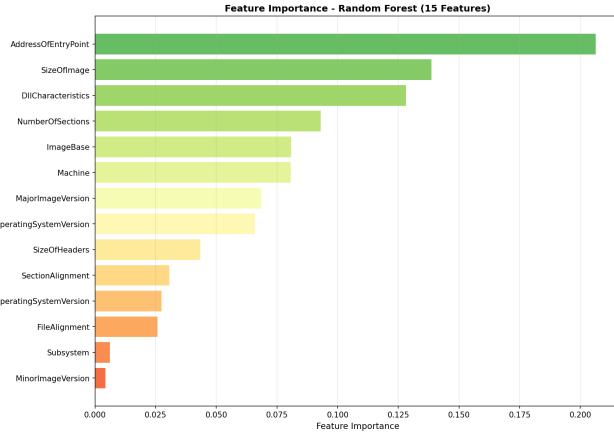


Fig. 4: Feature Importance - Reduced Feature Set (15 Features)

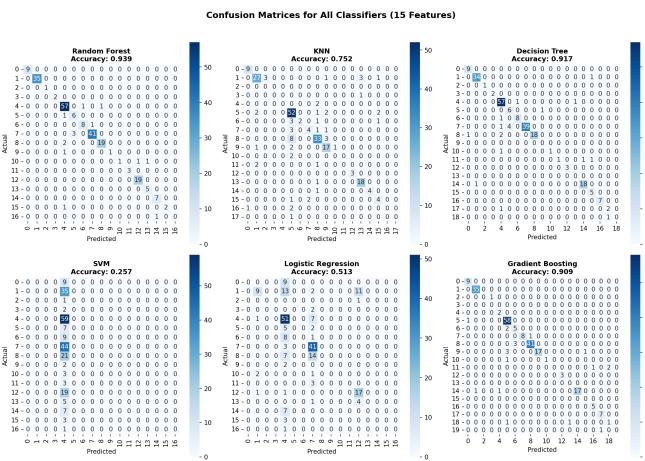


Fig. 5: Confusion Matrices - Reduced Feature Set (15 Features)

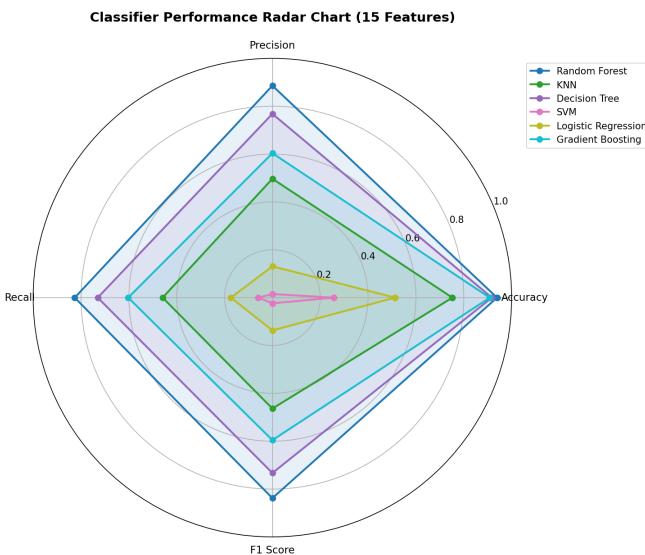


Fig. 6: Performance Radar Chart - Reduced Feature Set (15 Features)

## G. Performance Visualization - Extended Feature Set

Figure 7 shows the classifier performance comparison for the extended feature set.

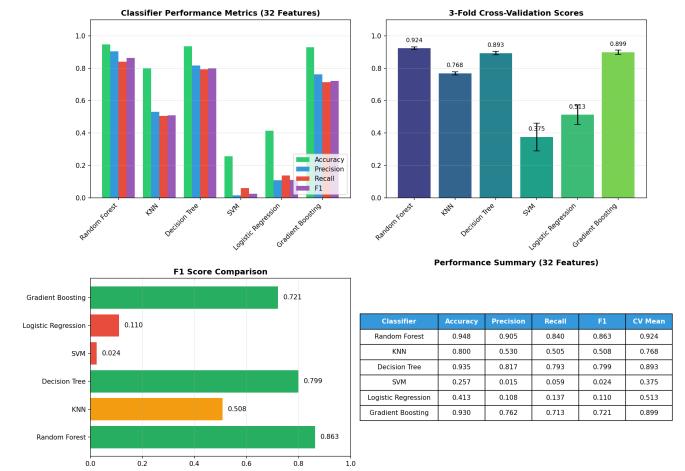


Fig. 7: Classifier Performance Comparison - Extended Feature Set (32 Features)

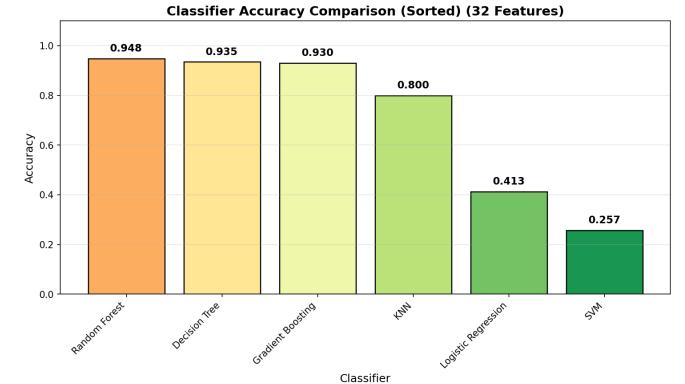


Fig. 8: Accuracy Comparison - Extended Feature Set (32 Features)

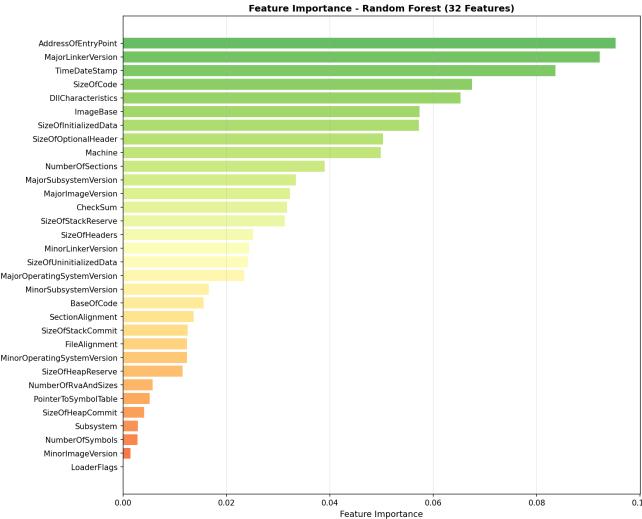


Fig. 9: Feature Importance - Extended Feature Set (32 Features)

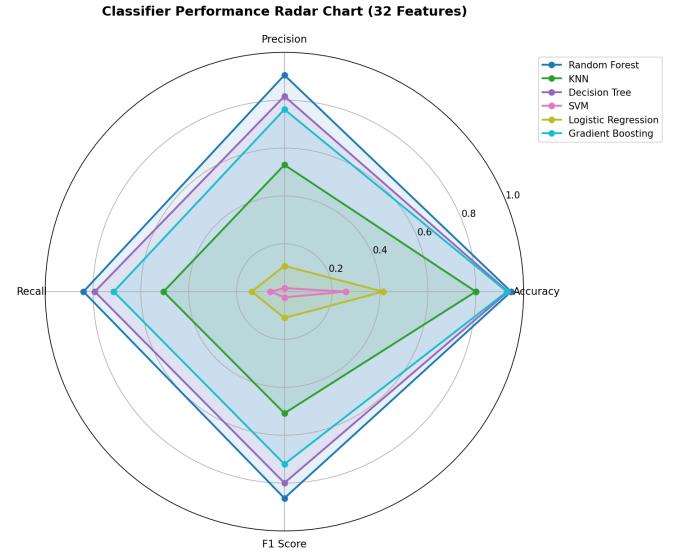


Fig. 11: Performance Radar Chart - Extended Feature Set (32 Features)

## VI. CONCLUSION

This research presents a comprehensive comparative study of PE header-based feature extraction for static malware analysis. Our key findings demonstrate:

### A. Feature Extraction Efficiency

The proposed methodology achieves significant improvements in feature extraction time compared to prior work. Processing 1,150 malware samples requires only 215–290 seconds (depending on feature set size), representing a 75% reduction compared to the 19-minute extraction time reported for the APT1 dataset. This improvement enables more practical deployment in real-time malware detection scenarios.

### B. Classification Performance

The extended feature set (32 features) outperforms the reduced feature set (15 features) for tree-based and instance-based classifiers:

- **Random Forest** achieved the highest accuracy at **94.78%** with the extended feature set
- Tree-based methods (Random Forest, Decision Tree, Gradient Boosting) and KNN improved accuracy by 0.87% to 4.78% with the extended feature set
- Logistic Regression performed worse with the extended set (41.30% vs. 51.30%), likely due to multicollinearity among the additional features
- SVM and Logistic Regression exhibited poor performance overall, likely due to the multi-class nature of the problem and the lack of feature scaling

### C. Trade-off Analysis

While the extended feature set provides improved accuracy, it requires approximately 34% more extraction time (290 seconds vs. 215 seconds). For applications prioritizing speed over marginal accuracy improvements, the reduced feature set offers an effective compromise.

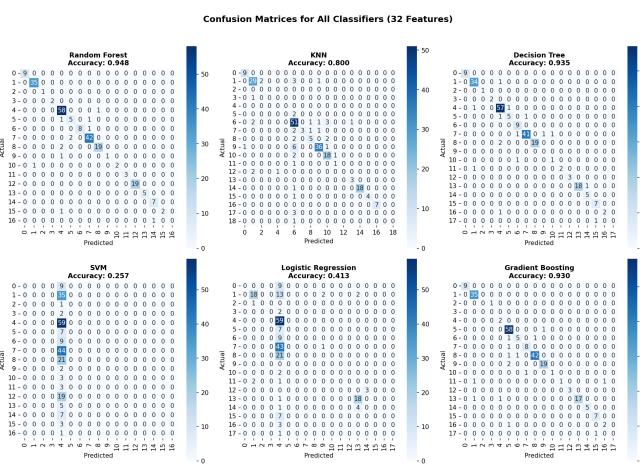


Fig. 10: Confusion Matrices - Extended Feature Set (32 Features)

#### D. Practical Implications

The research demonstrates that efficient malware detection is achievable using only PE header features without requiring computationally expensive dynamic analysis. The methodology is suitable for real-time malware scanning applications, large-scale malware dataset analysis, integration with existing antivirus systems, and automated malware triage systems.

## VII. LIMITATIONS

#### A. Dataset Constraints

- The dataset comprises 1,150 samples, which may not fully represent the diversity of real-world malware
- Samples were collected from a single source (MalwareBazaar), potentially introducing selection bias
- The dataset lacks benign samples for binary classification evaluation

#### B. Methodological Limitations

- Static analysis cannot detect malware that employs runtime obfuscation or packing techniques
- PE header features alone may not capture behavioral characteristics of sophisticated malware
- The methodology is specific to Windows PE executables and does not apply to other platforms

#### C. Classifier Limitations

- SVM and Logistic Regression exhibited poor performance, possibly due to the multi-class nature of the classification problem
- Hyperparameter tuning was limited; extensive optimization may yield improved results
- Cross-validation was performed with only 3 folds due to computational constraints

## VIII. FUTURE WORK

#### A. Dataset Enhancement

Expand the dataset to include a larger and more diverse sample set (10,000+ samples), incorporate benign samples for binary classification experiments, and include samples from multiple malware families for family classification tasks.

#### B. Feature Engineering

Explore the combination of dense (file header, optional header) and sparse (section, import table) features. Implement feature selection techniques (PCA, t-SNE) to identify optimal feature subsets. Investigate the correlation between different PE header fields.

#### C. Algorithmic Improvements

Implement parallel feature extraction using multi-core processors or distributed computing. Optimize the pefile parsing module for improved extraction speed. Explore deep learning approaches (CNN, LSTM) for automatic feature learning [8].

#### D. Ensemble Methods

Develop ensemble models combining multiple classifiers for improved robustness. Implement stacking and boosting techniques with PE header features. Explore hybrid approaches combining static and dynamic features.

#### E. Real-world Deployment

Develop a production-ready malware detection system based on the proposed methodology. Evaluate performance on streaming malware samples in real-time scenarios. Integrate with existing security infrastructure and antivirus solutions.

## REFERENCES

- [1] R. Kalakuntla, A. B. Vanamala, and R. R. Kolipyaka, "Cyber security," *HOLISTICA-Journal of Business and Public Administration*, vol. 10, no. 2, pp. 115–128, 2019.
- [2] P. Black and J. Opacki, "Anti-analysis trends in banking malware," in *2016 11th International Conference on Malicious and Unwanted Software (MALWARE)*, pp. 1–7, IEEE, 2016.
- [3] N. Balram, G. Hsieh, and C. McFall, "Static malware analysis using machine learning algorithms on APT1 dataset with string and PE header features," in *2019 International Conference on Computational Science and Computational Intelligence (CSCI)*, pp. 90–95, IEEE, 2019.
- [4] T. Rezaei and A. Hamze, "An efficient approach for malware detection using PE header specifications," in *2020 6th International Conference on Web Research (ICWR)*, pp. 234–239, IEEE, 2020.
- [5] R. S. Santos and E. D. Festijo, "Generating features of windows portable executable files for static analysis using portable executable reader module (pefile)," in *2021 4th International Conference of Computer and Informatics Engineering (IC2IE)*, pp. 283–288, IEEE, 2021.
- [6] N. Maleki and H. Rastegari, "An improved method for packed malware detection using PE header and section table information," *International Journal of Computer Network & Information Security*, vol. 11, no. 9, 2019.
- [7] H. H. Al-Khshali, M. Ilyas, and O. N. Ucan, "Effect of PE file header features on accuracy," in *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 1115–1120, IEEE, 2020.
- [8] S. Kumar *et al.*, "MCFT-CNN: Malware classification with fine-tune convolution neural networks using traditional and transfer learning in internet of things," *Future Generation Computer Systems*, vol. 125, pp. 334–351, 2021.
- [9] C. K. Yuk and C. J. Seo, "Static analysis and machine learning-based malware detection system using PE header feature values," *International Journal of Innovative Research and Scientific Studies*, vol. 5, no. 4, pp. 281–288, 2022.
- [10] K. K. R. Choo and A. Dehghanianha, *Handbook of Big Data Analytics and Forensics*. Springer, 2022.