

Model Deteksi *Malware* pada Sistem Operasi Windows Menggunakan Teknik *Machine Learning*

Malware Detection Model on Windows Operating System Using Machine Learning Techniques

Farhan Naufal Satya Chery^{a,1}, Lili Tanti^{b,2}, Bob Subhan Riza^{c,3}

^{a,b}Fakultas Ilmu Komputer, Program Studi Informatika, Universitas Potensi Utama, JL. KL. Yos Sudarso No. 3A, Tanjung Mulia, Kota Medan, Sumatera Utara 20241, Indonesia
fnaufal200902@gmail.com¹, lilitanti82@gmail.com², bob.potensi@gmail.com³

ABSTRAK

Malware masih menjadi salah satu ancaman terbesar bagi keamanan siber, terutama pada sistem operasi Windows, yang masih banyak digunakan di berbagai perangkat komputer. Teknik Random Forest akan digunakan dalam proyek ini untuk menciptakan sistem deteksi virus berbasis pembelajaran mesin. Dataset Malware yang Dioptimalkan, yang digunakan dalam studi ini, menjalani proses praproses menyeluruh yang mencakup normalisasi data menggunakan pendekatan Standard Scaler, mitigasi outlier menggunakan kombinasi teknik penghapusan dan pembatasan, serta penanganan nilai yang hilang menggunakan metode imputasi rata-rata. Pendekatan matriks korelasi digunakan dalam proses pemilihan fitur untuk menentukan atribut mana yang paling berdampak pada kelas target. SMOTE (Synthetic Minority Over-sampling Technique) digunakan untuk mengatasi ketidakseimbangan kelas. Model kemudian dilatih dengan konfigurasi algoritma Random Forest yang telah disempurnakan setelah dataset dibagi menggunakan metode pemisahan train-test. Laporan klasifikasi, matriks konfusi, dan validasi silang 5 kali lipat digunakan untuk mengevaluasi kinerja model secara menyeluruh. Temuan menunjukkan bahwa metode yang disarankan memiliki kemampuan yang kuat untuk mendeteksi dan mengkategorikan berkas berbahaya, terbukti dari nilai F1-score, akurasi, dan presisinya yang tinggi. Hasil ini menegaskan bahwa sistem deteksi malware yang adaptif dan andal dapat dihasilkan dengan menggabungkan teknik prapemrosesan yang efisien dengan algoritma Random Forest, yang menawarkan dasar yang kuat untuk pengembangan dan penguatan kerangka kerja keamanan siber selanjutnya.

Kata Kunci: Deteksi Malware, Machine Learning, Random Forest, SMOTE, Correlation Matrix, Sistem Operasi Windows.

ABSTRACT

Malware is still one of the biggest threats to cybersecurity, especially when it comes to the Windows operating system, which is still widely used in a variety of computer settings. The Random Forest technique will be used in this project to create a machine learning-based virus detection system. The Optimized Malware Dataset, which was used in this study, underwent a thorough preprocessing pipeline that included data normalization using the Standard Scaler approach, outlier mitigation using a combination of removal and capping techniques, and treatment of missing values using the mean imputation method. The correlation matrix approach is used in the feature selection process to determine which attributes have the most impact on the target class. The SMOTE (Synthetic Minority Over-sampling Technique) was used to alleviate class imbalance. The model was then trained under an improved Random Forest algorithm configuration after the dataset was divided using a train–test split method. A classification report, confusion matrix, and 5-fold cross-validation were used to thoroughly evaluate the model's performance. The findings show that the suggested method has a strong ability to detect and categorize harmful files, as seen by its high F1-score values, accuracy, and precision. These results confirm that an adaptable and dependable malware detection system can be produced by combining efficient preprocessing techniques with the Random Forest algorithm, offering a strong basis for the development and strengthening of next cybersecurity frameworks.

Keywords: Malware Detection, Machine Learning, Random Forest, SMOTE, Correlation Matrix, Windows Operating System.

Info Artikel

Disubmit

Direview

Diterima

Copyright © 2025 - Journal UPU. All rights reserved.

1. PENDAHULUAN

Perkeembangan teknologi informasi yang begitu pesat membawa dampak yang bersifat dua sisi dalam dunia digital. Di satu sisi, kemajuan ini memberikan banyak kemudahan dan manfaat bagi aktivitas manusia; namun di sisi lain, juga menghadirkan ancaman serius terhadap keamanan data serta sistem komputer. Salah satu ancaman yang terus berevolusi seiring waktu adalah *malware*, yaitu *software* asing yang dirancang untuk mengacaukan sistem, menyusup ke jaringan, mencuri informasi penting, atau mengganggu kinerja sistem operasi [9]. Ragam *malware* yang muncul pun semakin luas, mencakup *virus*, *worm*, *trojan*, *spyware*, *adware*, hingga *ransomware*, dengan tingkat kompleksitas yang kian meningkat sejalan dengan kemajuan teknologi digital dan teknik serangan siber yang semakin canggih [9].

Sistem operasi *Windows* merupakan target paling dominan bagi serangan *malware*. Hal ini disebabkan oleh tingginya tingkat penggunaan *Windows* di berbagai sektor, baik personal, organisasi, maupun institusi pemerintah [31]. Data dari lembaga keamanan siber menunjukkan bahwa sebagian besar serangan siber global memanfaatkan celah pada sistem operasi *Windows* karena ekosistem penggunaannya yang luas dan variatif [32]. Akibat serangan *malware*, kerugian yang ditimbulkan tidak hanya berupa kerusakan sistem dan kehilangan data, tetapi juga dapat menyebabkan kebocoran informasi penting yang berimplikasi pada aspek finansial maupun reputasi organisasi [33].

Dalam penelitian ini diterapkan algoritma *Decision Tree*, yang dikenal memiliki keunggulan dalam hal kemudahan interpretasi, efisiensi proses, serta kemampuan klasifikasi yang cukup baik [11]. Algoritma ini menyajikan hasil dalam bentuk struktur pohon keputusan yang mudah dipahami, sehingga pengguna dapat dengan jelas melihat pola serta aturan klasifikasi yang terbentuk. Selain itu, *Decision Tree* memiliki kelebihan pada kecepatan proses pelatihan dan tidak membutuhkan penyesuaian parameter yang rumit [10].

Untuk meningkatkan performa model, dilakukan pula tahap seleksi fitur menggunakan metode *Mutual Information*, dengan tujuan memilih hanya atribut yang memiliki tingkat relevansi tinggi terhadap variabel target [38]. Pendekatan ini membantu mengurangi kompleksitas model serta meningkatkan akurasi dalam proses klasifikasi.

Tujuan dari penelitian ini adalah:

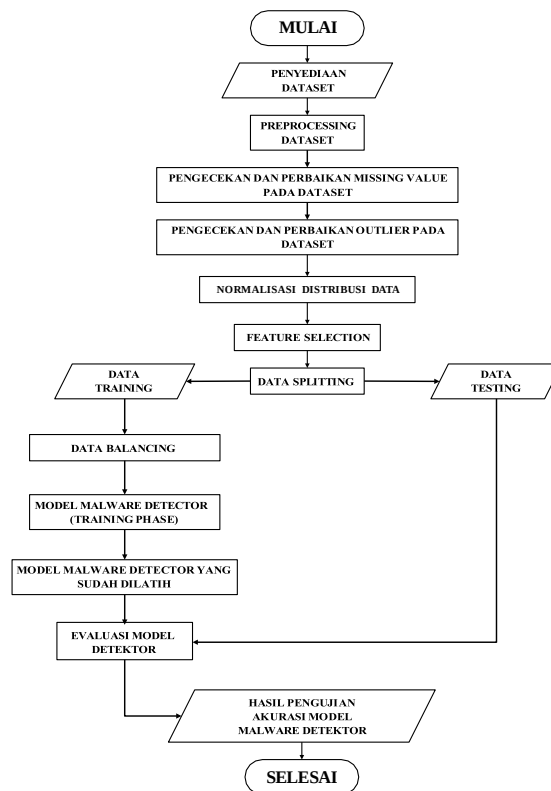
1. Melakukan *preprocessing* data *malware* dan benign agar siap digunakan dalam pemodelan *Machine Learning*.
2. Melakukan pemilihan fitur menggunakan metode *mutual information* untuk menentukan fitur-fitur yang paling berpengaruh.
3. Membangun model deteksi *malware* menggunakan algoritma *Decision Tree*.
4. Melakukan evaluasi kinerja model menggunakan berbagai metrik pengukuran, meliputi akurasi, presisi, *Recall*, nilai-*f1*, serta *Confusion Matrix*, dengan tujuan untuk menilai sampai mana model yang dibuat mampu mengklasifikasikan data secara akurat dan andal.

Penelitian ini diharapkan dapat memberikan kontribusi yang signifikan dalam bidang keamanan komputer, khususnya dalam aspek deteksi *malware* pada sistem operasi *Windows*. Model yang dihasilkan ditujukan untuk menjadi referensi pengembangan sistem deteksi *malware* yang sederhana, mudah dipahami, namun tetap memiliki tingkat efektivitas tinggi dalam mengenali berbagai potensi ancaman. Selain itu, penelitian ini juga memberikan sumbangan akademik melalui penekanan pada pentingnya penerapan *feature selection* sebagai strategi untuk meningkatkan kinerja algoritma *machine learning* di ranah keamanan siber. Pendekatan ini diharapkan mampu memperkuat kemampuan deteksi dan pencegahan serangan siber di masa yang akan datang.

2. METODE

<https://upu-journal.potensi-utama.org/index.php/itjournal/index> | redaksijurnalupu@gmail.com

Metode penelitian ini disusun secara sistematis agar dapat membangun model deteksi malware pada sistem operasi Windows dengan menggunakan teknik machine learning. Tahapan yang dilakukan meliputi pengumpulan dataset, preprocessing data, seleksi fitur, pembagian data, pelatihan model, dan evaluasi kinerja. Rangkaian tahapan tersebut ditunjukkan pada Gambar 1:



Gambar 1. FlowChart Penelitian

A. Dataset

Dattaset yang diigunakan pada penellitian ini merupakan kumpullan data file yangg dikelompokkan ke dalam dua kategori utama, yaitu malware dan benign. Setiap file direpresentasikan oleh 25 fitur utama yang merepresentasikan berbagai karakteristik penting, seperti entropi, frekuensi opcode, jumlah pemanggilan API, ukuran resource, jumlah section, serta keberadaan Dynamic Link Library (DLL) [2]. Dataset ini diperoleh dari repositori publik yang sering digunakan dalam penelitian terkait keamanan komputer. Oleh karena itu, dataset tersebut dapat diakses, diverifikasi, dan direplikasi oleh peneliti lain sebagai dasar dalam pengembangan model deteksi malware yang serupa [2].

B. Preprocessing Data

Sebelum digunakan dalam proses pemodelan, dataset terlebih dahulu melalui serangkaian tahapan preprocessing untuk memastikan kualitas data. Adapun serangkaian tahapan itu adalah:

1. Data-data yang dipakai untuk penelitian ini dibuat dari sekumpulan file yang diklasifikasikan ke dalam dua jenis utama, yakni malware dan benign. Masing-masing file direpresentasikan oleh 25 fitur awal yang mencerminkan berbagai karakteristik penting, seperti entropi, frekuensi opcode, jumlah pemanggilan API, ukuran resource, jumlah section, serta keberadaan Dynamic Link Library (DLL). Dataset ini bersumber dari repositori publik yang umum digunakan dalam

penelitian keamanan komputer, sehingga memungkinkan proses replikasi dan pengujian ulang oleh penelitian lain. Adapun rincian fitur yang digunakan dapat dilihat pada tabel 1 dibawah:

Table 1. Fitur yang Terdapat pada Dataset

Nama Fitur	File_Size, Entropy, Opcode_Frequency, API_Calls, Imports_Count, Sections_Count, Dll_Imports, Resources_Size, TLS_Callbacks, Packed, Sections_Entropy_Mean, Signature_Match, Network_Connections, Registry_Modifications, Thread_Count, Handles_Open, AV_Detected, Mutex_Creation, Process_Hollowing, File_Extension_DLL, Suspicious_PE_Sections, Code_Obfuscation, Suspicious_API, Anti_VM_Checks, Execution_Time, Malware_Type
-------------------	---

2. Preprocessing Dataset. Tahap Awal dalam pembangunan model ini adalah preprocessing data. Pada tahap ini dataset mentah yang sudah disediakan akan diproses dan diolah terlebih dahulu sedemikian rupa dengan serangkaian proses pengolahan.
3. Pengecekan dan Perbaikan Missing Value pada Dataset. Dataset mentah pertama-tama akan diperiksa apakah terdapat didalamnya missing value menggunakan metode Exploratory Data Analysis (EDA) dan Pandas built-in function yang akan mengecek setiap field yang ada pada dataset apakah terdapat field yang kosong (NaN). Setelah pengecekan missing value selesai, program akan menggunakan teknik Mean Imputation untuk menangani missing value yang sudah ditemukan.
4. Pengecekan dan Perbaikan Outlier pada Dataset. Dataset yang telah dipastikan bersih dan bebas dari missing value, selanjutnya akan memasuki pemeriksaan dan pembersihan outlier secara umum. Dalam proses ini penulis menggunakan gabungan dari beberapa metode untuk pengecekan dan penanganannya. Teknik pertama yang digunakan adalah teknik Inter Quartile Range (IQR) untuk menemukan outlier pada dataset dengan menghitung batas standar dan batas maksimal identifikasi outlier.
5. Normalisasi Data. Meski sudah melewati tahap pengecekan dan penangan *missing value* dan *outlier*, biasanya dataset masih memiliki kelemahan. Kelemahan yang dimaksud disini adalah ketidakseimbangan distribusi data antar kelas dimana distribusi data antar fitur tidak seimbang dan jika tetap dilanjutkan ke proses selanjutnya, bisa saja menimbulkan bias pada model karena distribusi datanya tidak seimbang yang akan mengakibatkan menurunnya variasi pengetahuan model tentang pola data dan menurunkan akurasi.
6. Feature Selection. Tahapan ini bertujuan untuk memilih fitur yang paling relevan terhadap variabel target agar model dapat belajar dengan lebih efisien. Teknik Correlation Matrix digunakan untuk mengidentifikasi tingkat hubungan antar-fitur dengan label *Malware_Type*, dan fitur yang memiliki korelasi terkuat akan dipertahankan untuk proses pelatihan model. Proses ini membantu mengurangi kompleksitas model dan mencegah overfitting:

Table 2. Fitur Terpilih Setelah Proses *feature selection*

Nama Fitur	Packed, Sections_Count, Entropy, API_Calls, Opcode_Frequency, Resources_Size, File_Extension_DLL, Suspicious_API, AntVir_Detected, Registry_Modifications, Sections_Entropy_Mean, Dll_Imports
-------------------	---

7. Data Splitting. Tahap data splitting dilakukan setelah seluruh proses preprocessing selesai, mencakup pembersihan data dari missing value dan outlier, penyesuaian tipe data, serta pemilihan fitur yang paling relevan. Pada tahap ini, dataset kemudian dibagi kedalam dua komponen utama yakni data train dan data test. Rasio pembagiannya adalah 80:20. 80% data akan digunakan untuk melatih model dalam mengenali pola data, dan 20% sisanya akan dipakai untuk menguji kemampuan model dalam membedakan data yang belum pernah

dipelajari sebelumnya. Data training berfungsi untuk memberikan pembelajaran kepada model, agar sistem dapat mengenali pola dan menentukan apakah suatu file tergolong aman atau berbahaya. Sementara itu, data testing digunakan untuk mengevaluasi akurasi dan kemampuan generalisasi model dalam membedakan file malware (berbahaya) dari file benign (aman).

8. Data Balancing. Setelah proses pembagian data, dilakukan penyeimbangan kelas menggunakan teknik Synthetic Minority Oversampling Technique/SMOTE untuk mengatasi ketimpangan antara data malware dan benign. Proses ini menghasilkan jumlah sampel yang seimbang pada kedua kelas, sehingga model tidak bias terhadap kelas mayoritas.\.
9. Model Malware Detector (Training Phase). Dalam training phase atau tahap pelatihan model ini, akan digunakan algoritma ensemble learning random forest classifier yang akan mengambil keputusan berdasarkan perhitungan pohon keputusan. Pohon keputusan yang nantinya akan dibuat tentu akan ditentukan spesifikasinya dan parameter pengukurnya agar output yang dihasilkan bisa lebih efektif, efisien dan terukur. Dengan spesifikasi dan parameter yang telah ditentukan, pohon keputusan yang akan dibuat oleh program tidak akan melewati ketentuan tersebut sehingga pengambilan keputusan bisa lebih efisien dan bisa memotong waktu pelatihan model.
10. Model Malware Detector yang Sudah Dilatih (Trained Model). sistem telah mendapatkan dan menyimpan model terlatih yang mampu mendeteksi pola dari aktivitas perilaku suatu file. Dengan program ini yang berisikan serangkaian kode program yang telah terlatih dan memahami serangkaian tahapan dan prosedur yang bisa digunakan untuk melakukan deteksi dan pemindaian malware.
11. Evaluasi Model Detector. Evaluasi ini akan melibatkan gabungan dari beberapa teknik evaluasi metrik yang umum seperti Accuracy, Precision, Recall dan F-1 Score. Teknik evaluasi tersebut akan memeriksa keakuratan dan jumlah poin benar dalam proses akurasi berdasarkan proses dari gabungan data testing dan data training sebelumnya. kemudian menghitung jumlah poin benar yang didapat lalu mengubahnya menjadi bentuk persen yang nantinya akan disajikan dalam beberapa bentuk seperti, table persentasi ataupun graf-graf yang memaprkan tingkat akurasi dari model detector dan semua hasil penghitungan itu dikumpulkan dan akan disajikan kedalam bentuk graph.

3. HASIL PEMBAHASAN DAN

Penelitian ini bertujuan untuk membangun sebuah model deteksi malware pada sistem operasi Windows dengan memanfaatkan teknik machine learning, khususnya algoritma Random Forest. Pembahasan pada bab ini mencakup analisis data, fase pre-processing data, model training, serta evaluasi model deteksi malware yang diperoleh dari proses tersebut. Semua langkah dilakukan secara sistematis untuk memastikan akurasi dan keandalan model dalam mendeteksi file malware secara otomatis. Setiap langkah akan dijelaskan secara sistematis untuk menggambarkan bagaimana model dibangun dan bagaimana performanya serta cara kerjanya dalam mendeteksi data berbahaya (malware) dan normal (benign).

A. Hasil Proses Preprocessing Data

Dataset mentah yang sudah disiapkan akan dianalisis dan diperiksa apakah ada missing value dan outlier didalamnya lalu dibersihkan dan dioptimalkan menggunakan teknik Isolation Forest.

```

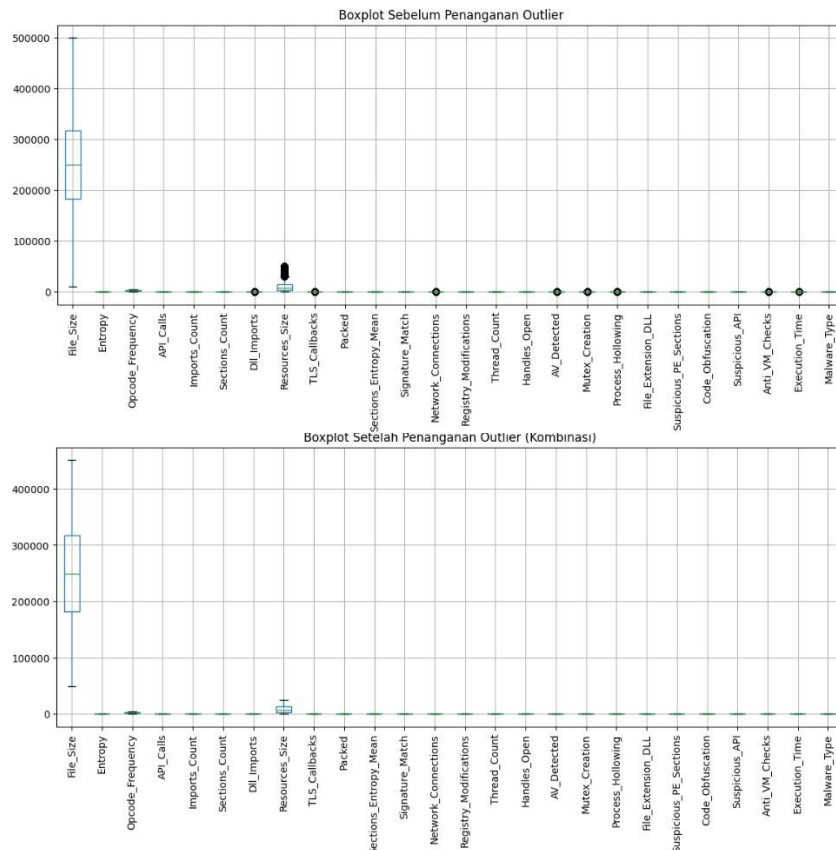
Percentase missing values per kolom:
File_Size          0.0
Entropy            0.0
Opcode_Frequency   0.0
API_Calls          0.0
Imports_Count      0.0
Sections_Count     0.0
Dll_Imports        0.0
Resources_Size     0.0
TLS_Callbacks      0.0
Packed             0.0
Sections_Entropy_Mean 0.0
Signature_Match    0.0
Network_Connections 0.0
Registry_Modifications 0.0
Thread_Count       0.0
Handles_Open       0.0
AV_Detected        0.0
Mutex_Creation     0.0
Process_Hollowing  0.0
File_Extension_DLL 0.0
Suspicious_PE_Sections 0.0
Code_Obfuscation   0.0
Suspicious_API     0.0
Anti_VM_Checks     0.0
Execution_Time     0.0
Malware_Type       0.0
dtype: float64

Total missing values di seluruh dataset: 0

Total baris yang memiliki missing values: 0

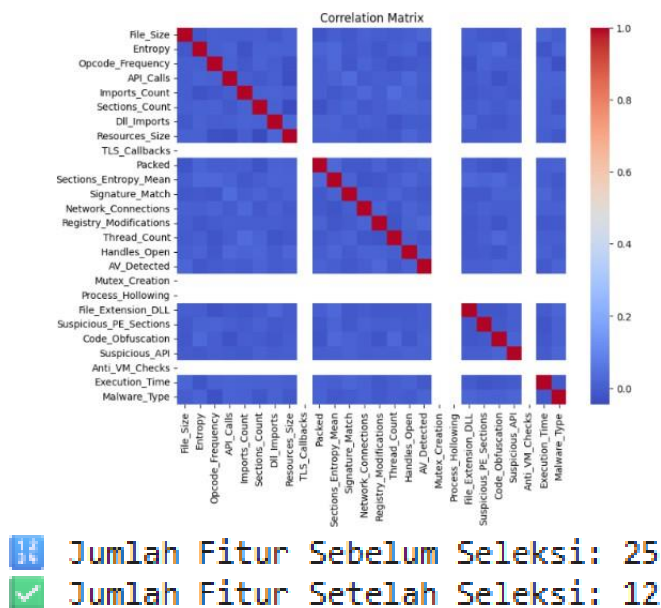
```

Gambar 3. Hasil Pengecekan dan Penangan *Missing Value*



Gambar 4. Hasil Pengecekan (atas) dan Penangan (bawah) *Outlier*

Kemudian dilakukan pemilihan fitur (Feature Selection) yang relevan dan memiliki korelasi serta mendukung untuk meningkatkan akurasi dan performa model selama proses pelatihan.

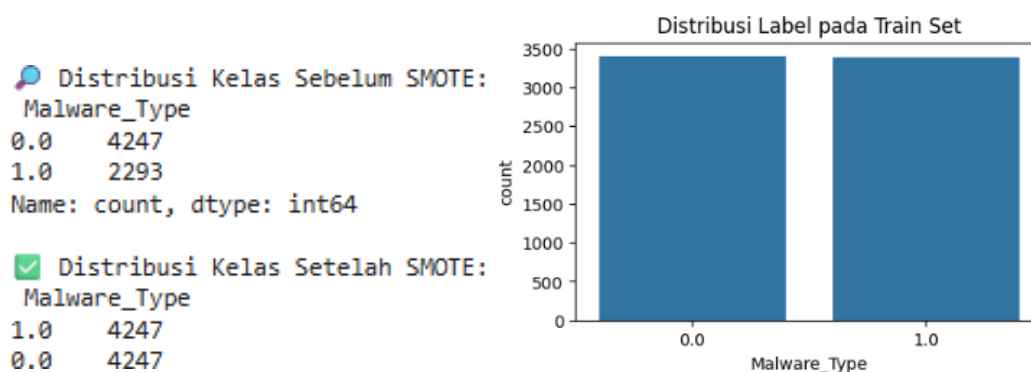


Gambar 2. Hasil Proses Feature Selection

Selanjutnya, dilakukan data splitting menjadi 2 jenis yaitu test dan train data yang nantinya akan digunakan untuk melakukan pelatihan pada model dan juga untuk testing serta evaluasi model.

B. Hasil Tahapan Data Balancing

Pada tahap ini dilakukan penyeimbangan proporsi data guna mengatasi masalah kelas yang tidak seimbang (imbalance), dilakukan teknik SMOTE atau Random Over Sampler sehingga jumlah data antara kelas “normal” dan “malware” menjadi seimbang. Hal ini cukup penting agar model tidak bias terhadap salah satu kelas saja dan bisa meningkatkan akurasi dan performa model.



Gambar 3. Hasil Proses Data Balancing

C. Pelatihan dan Penyimpanan Model (Training Phase)

Setelah dataset melalui tahap penyeimbangan data dan dianggap siap serta layak untuk digunakan, maka tahapan selanjutnya adalah pelatihan model. Pada tahap ini, model deteksi akan dilatih dan diperkenalkan dengan pola-pola file yang mengindikasikan identitas file tersebut apakah aman (benign)

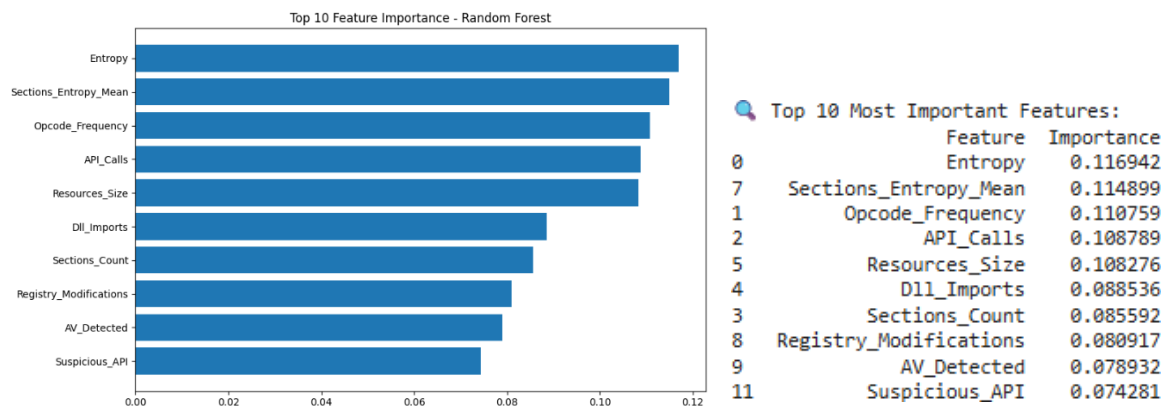
atau berbahaya (malware). Adapun spesifikasi dan parameter yang akan digunakan untuk pohon keputusan pada algoritma ini adalah:

1. `n_estimators = 200` (Jumlah kedalaman pohon keputusan).
2. `max_depth = 15` (Kedalaman maksimal setiap pohon).
3. `class_weight = "balanced"` (Tipe class yang telah diseimbangkan).

Dengan spesifikasi dan parameter yang telah ditentukan itu, pohon keputusan yang akan dibuat oleh program tidak akan melewati ketentuan tersebut sehingga pengambilan keputusan bisa lebih efisien dan bisa memotong waktu pelatihan model. Setelah program dijalankan, sistem akan memulai membaca, mempelajari dan memahami dataset yang diberikan. Sistem akan memberikan pengetahuan kepada model deteksi tentang pola-pola dari setiap data pada dataset agar bisa membedakan mana pola yang menyatakan bahwa suatu data adalah data benign (aman) dan mana pola yang menyatakan bahwa suatu data adalah malware (berbahaya).

D. Analisis Feature Importance pada Model Random Forest

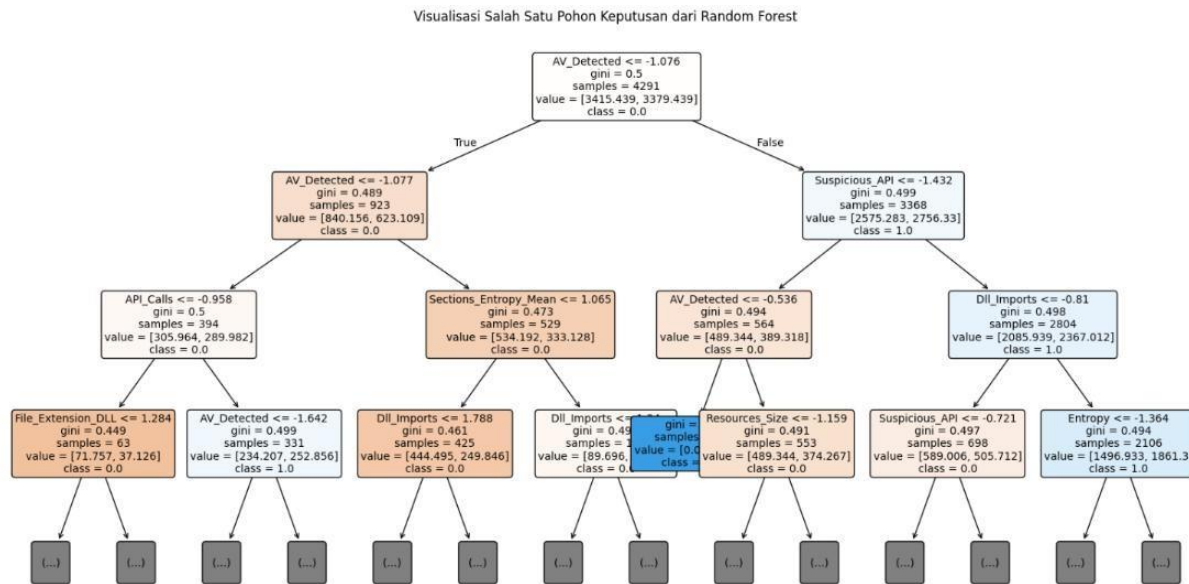
Pada model Random Forest, fitur yang paling berpengaruh terhadap hasil klasifikasi diidentifikasi dengan menggunakan analisis fitur penting. Nilai penting menunjukkan seberapa besar kontribusi setiap fitur dalam proses pemisahan data dan penurunan impurity pada seluruh pohon keputusan. Berdasarkan hasil perhitungan, fitur-fitur seperti Entropy, Section_Entropy_Mean dan Opcode_Frequency memiliki pengaruh terbesar terhadap akurasi model dalam membedakan data malware dan non-malware. Visualisasi feature importance pada Gambar 4 memperlihatkan sepuluh fitur utama dengan kontribusi tertinggi, yang dapat dijadikan acuan untuk optimasi model dan seleksi fitur pada penelitian selanjutnya.



Gambar 4. Hasil Analisa Feature Importance

E. Visualisasi Pohon Keputusan

Untuk memberikan gambaran logika keputusan dari model Random Forest yang telah dibuat, tentu harus melihat sendiri visualisasi dari pohon keputusan yang dibuat sistem dalam mengambil keputusan. Maka dari itu salah satu pohon dari model harus divisualisasikan dalam bentuk diagram pohon keputusan agar dapat dilihat dengan jelas. Visualisasi ini akan membantu menjelaskan bagaimana model memproses fitur-fitur dari input data untuk menentukan apakah suatu file termasuk kategori file berbahaya (malware) atau aman (benign).



Gambar 5. Sample Salah Satu Pohon Keputusan Untuk Fase Training Model

F. Hasil Evaluasi

Evaluasi performa model disajikan dalam 2 bentuk yaitu dalam bentuk *Classification Report* dan *Confusion Matrix*. Adapun hasil dari evaluasi performa model dipaparkan sebagai berikut:

1. Classification Report

Hasil awal dari proses evaluasi disajikan dalam bentuk *Classification Report*, yang menampilkan metrik kinerja model secara keseluruhan. Laporan ini dapat dilihat pada Gambar 5, yang memperlihatkan hasil pengukuran dari setiap kelas berdasarkan nilai precision, recall, f1-score, serta support.

Classification Report:				
	precision	recall	f1-score	support
0.0	0.66	0.86	0.75	838
1.0	0.81	0.58	0.67	861
accuracy			0.72	1699
macro avg	0.73	0.72	0.71	1699
weighted avg	0.74	0.72	0.71	1699

Gambar 6. Classification Report Hasil dari Evaluasi Performa Model

Dapat dilihat akurasi keseluruhan mencapai 72% dengan perolehan f1 score di kelas 0 sebesar 75% dan pada kelas 1 sebesar 65%. Hal ini menunjukkan bahwa model mamapu membedakan dan menyaring file antara kelas malware dan benign dengan baik dan dengan peroforma yang cukup baik pula.

Berdasarkan hasil evaluasi, model menunjukkan nilai precision dan recall yang lebih rendah pada kelas malware dibandingkan dengan kelas benign. Kondisi ini menunjukkan bahwa model masih mengalami kesulitan dalam mengenali beberapa sampel malware secara tepat. Salah satu penyebabnya

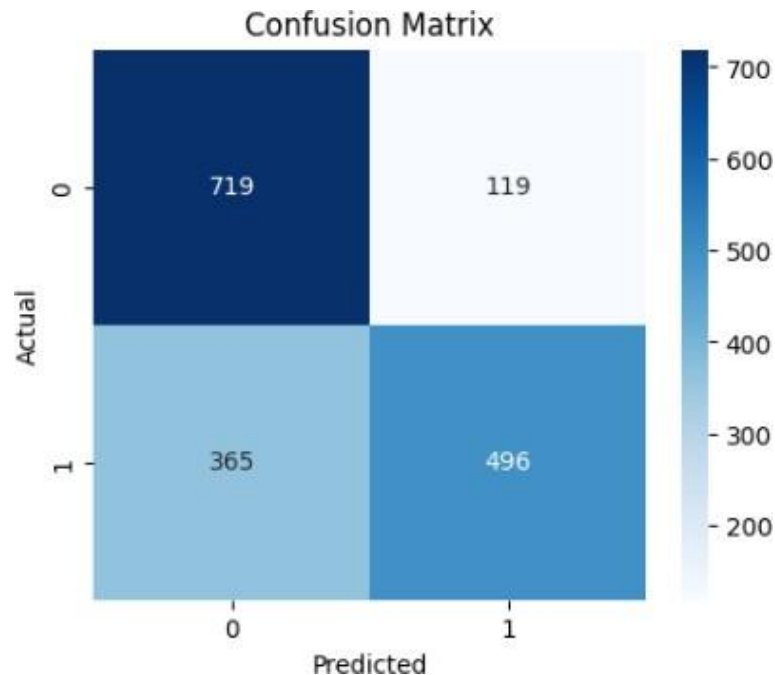
adalah karakteristik malware yang cenderung kompleks dan bervariasi, baik dari sisi struktur file maupun pola perilaku yang ditunjukkan pada fitur sistem. Selain itu, tumpang tindih (overlap) nilai antar fitur antara file benign dan malware tertentu dapat menyebabkan ambiguitas dalam proses klasifikasi.

Meskipun metode balancing data dengan SMOTE telah diterapkan, distribusi fitur sintetis yang dihasilkan masih mungkin belum sepenuhnya merepresentasikan karakteristik sebenarnya dari malware di dunia nyata. Fenomena serupa juga ditemukan pada penelitian sebelumnya oleh Kumar et al. (2021) dan Santos et al. (2020), yang melaporkan bahwa ketidakseimbangan dataset dan kompleksitas pola perilaku malware merupakan salah satu tantangan utama dalam penerapan machine learning untuk deteksi malware.

Berdasarkan hal tersebut, penelitian selanjutnya disarankan untuk memperbanyak variasi data malware, serta mengombinasikan pendekatan feature engineering berbasis analisis perilaku dinamis agar performa deteksi terhadap kelas malware dapat meningkat.

2. Confussion Matrix

Hasil selanjutnya yang didapatkan adalah laporan dalam bentuk grafik Confussion matrix tentang jumlah dan keterangan tentang posisi file setelah melewati model deteksi seperti pada gambar 6



Gambar 7. Grafik Confussion Matrix Hasil Evaluasi Performa Model

Pada grafik confusion matrix dapat dilihat hasil prediksi menunjukkan angka yang cukup baik. Dengan perolehan angka True Positive 0 (koordinat 0,0) dan True Positive 1 (koordinat 1,1) yang menunjukkan angka yang lebih tinggi yang artinya model berhasil menyaring dan membedakan serta mengidentifikasi file pada dataset dengan yakin dan benar. Dapat dilihat bahwa model berhasil membedakan mayoritas file malware dan benign dengan baik, namun tetap masih ada sedikit celah yang mengakibatkan model menganggap benign sebagai malware (koordinat 1,0) dan sebaliknya menganggap file malware sebagai benign (koordinat 0,1). Meski demikian, secara keseluruhan akurasi model sudah terbilang baik dan layak untuk digunakan karena angka deteksi benar jauh lebih besar dari pada angka deteksi salah.

4. KESIMPULAN

Penelitian ini telah berhasil mengembangkan model deteksi malware pada sistem operasi Windows dengan menggunakan algoritma Random Forest sebagai pendekatan utama dari teknik machine learning. Penelitian ini dirancang untuk menjawab permasalahan deteksi malware modern (termasuk zero-day malware) yang semakin kompleks dan sulit terdeteksi oleh metode konvensional seperti signature-based detection. Berdasarkan rangkaian proses yang telah dilakukan, berikut adalah beberapa kesimpulan utama dari penelitian ini:

1. Preprocessing data secara menyeluruh memberikan dampak besar terhadap akurasi model. Tahapan preprocessing yang mencakup pemeriksaan dan penanganan missing value, outlier, normalisasi data, serta feature selection telah dilakukan secara sistematis. Kombinasi teknik removal dan winsorization untuk menangani outlier, serta mean imputation untuk mengisi missing value, terbukti berhasil membersihkan dan menstabilkan dataset.
2. Pemilihan fitur menggunakan Correlation Matrix efektif dalam mengurangi kompleksitas model tanpa mengurangi performa. Dari total 26 fitur awal, hanya 12 fitur terpilih berdasarkan korelasi terhadap label target. Fitur-fitur seperti Entropy, Opcode_Frequency, Sections_Count, dan DLL_Imports memiliki peran dominan dalam proses klasifikasi antara malware dan benign.
3. SMOTE berhasil menyeimbangkan distribusi kelas dalam dataset. Kelas minoritas (malware) yang awalnya jauh lebih sedikit dibandingkan kelas mayoritas (benign) diseimbangkan menggunakan Synthetic Minority Oversampling Technique (SMOTE). Hal ini terbukti meningkatkan kinerja model dengan mengurangi bias terhadap kelas mayoritas.
4. Model Random Forest memberikan performa yang cukup baik dengan akurasi sebesar 72%. Model berhasil membedakan file malware dan benign dengan F1-score sebesar 75% untuk kelas benign dan 65% untuk kelas malware. Visualisasi pohon keputusan juga menunjukkan bahwa model mampu menjelaskan logika klasifikasinya dengan jelas dan interpretatif.

Secara keseluruhan, penelitian ini menunjukkan bahwa Teknik machine learning, khususnya algoritma Random Forest, dapat digunakan untuk membuat sistem deteksi malware kontemporer yang akurat dan dapat disesuaikan pada sistem operasi Windows.

Selain itu, penelitian ini memiliki unsur kebaruan (novelty) pada pendekatan yang diterapkan. Kebaruan penelitian ini terletak pada penerapan kombinasi tahapan preprocessing yang komprehensif dan terintegrasi, meliputi imputasi nilai hilang menggunakan Mean Imputation, penanganan outlier melalui kombinasi metode removal dan capping, normalisasi dengan StandardScaler, seleksi fitur menggunakan Correlation Matrix, serta penyeimbangan data dengan SMOTE.

Kombinasi metode tersebut belum banyak diterapkan secara menyeluruh dalam penelitian deteksi malware berbasis sistem operasi Windows, khususnya pada klasifikasi multi-kelas. Pendekatan ini berhasil meningkatkan stabilitas data dan akurasi model, sekaligus memperkuat kemampuan sistem dalam mengenali berbagai jenis malware secara lebih seimbang. Dengan demikian, penelitian ini berkontribusi dalam memberikan alternatif pendekatan machine learning pipeline yang lebih efektif dan dapat direplikasi untuk penelitian lanjutan di bidang keamanan siber.

Meskipun model yang diusulkan telah memberikan hasil yang cukup memuaskan, masih terdapat sejumlah aspek yang berpotensi untuk ditingkatkan maupun dikembangkan lebih lanjut. Oleh karena itu, beberapa rekomendasi untuk penelitian berikutnya dapat dipertimbangkan sebagai berikut:

1. Integrasi antara analisis statis dan dinamis. Penelitian ini masih berfokus pada fitur statis yang diekstraksi dari file executable. Untuk meningkatkan efektivitas deteksi, disarankan agar penelitian selanjutnya mengombinasikan analisis statis dengan analisis dinamis, seperti pengamatan terhadap perilaku proses selama runtime. Pendekatan ini berpotensi memperkuat

kemampuan sistem dalam mendeteksi jenis malware yang menerapkan teknik obfuscation maupun polimorfisme tingkat lanjut.

2. Eksplorasi algoritma machine learning lain dan ensemble model. Selain Random Forest, penelitian selanjutnya dapat mempertimbangkan penggunaan algoritma lain seperti XGBoost, LightGBM atau deep learning (CNN) untuk membandingkan performa dan menemukan pendekatan yang lebih optimal.
3. Peningkatan kualitas serta keragaman dataset. Penggunaan dataset yang lebih luas dan beragam direkomendasikan untuk penelitian selanjutnya, guna memperluas kemampuan model dalam mengenali pola serta karakteristik berbagai jenis malware. Dengan memperkaya sumber data, model akan memiliki potensi yang lebih baik dalam mengidentifikasi varian baru, termasuk jenis zero-day malware yang terus berevolusi seiring perkembangan teknologi.
4. Penelitian di masa mendatang disarankan untuk mengimplementasikan pendekatan interpretabilitas model yang lebih mendalam. Penggunaan metode seperti *shapley additive explanations*/SHAP dan *local interpretable model-agnostic explanations*/LIME dapat membantu menjelaskan secara lebih detail bagaimana model mengambil keputusan terhadap suatu data. Dengan penerapan teknik interpretabilitas lanjutan ini, hasil analisis yang dihasilkan akan menjadi lebih transparan, mudah dipahami, serta dapat meningkatkan tingkat kepercayaan dan akuntabilitas terhadap sistem deteksi yang dikembangkan.
5. Pengujian langsung pada lingkungan nyata. Untuk meningkatkan validitas dan kegunaan praktis, pengujian terhadap file-file real-world dari sistem operasional Windows yang aktif akan memberikan gambaran kinerja model yang lebih nyata dan aplikatif.

Dengan saran-saran tersebut, diharapkan pengembangan model deteksi malware di masa mendatang akan semakin kuat, adaptif, dan mampu menjawab tantangan keamanan siber yang terus berkembang.

UCAPAN TERIMA KASIH

Penulis ingiin menyampaikan terima kasih banyak kepada Ibu Dr. Lili Tanti, M.Kom, atas peran beliau sebagai dosen pembimbing. Mereka juga berterima kasih atas bimbingan, arahan, dan dukungan yang mereka berikan selama fase penelitian hingga karya tulis ini selesai dengan baik dan tepat pada waktunya. Selain daripada itu, penulis juga ingin mengucapkan penghargaan yang tulus kepada ibunda tercinta atas doa, inspirasi, dan bantuan terus menerus sepanjang proses penyusunan karya ilmiah ini. Harapan kami adalah temuan penelitian ini akan membantu kemajuan sains, khususnya di bidang teknologi dan keamanan digital.

REFERENSI

- [1] H. Almajed, A. Alsaqer, and M. Frikha, "Imbalance datasets in malware detection: A review of current solutions and future directions," *Int. J. Adv. Comput. Sci. Appl.*, vol. 16, no. 1, pp. 1–10, 2025.
- [2] Al Maqousi, A., & Alauthman, M. (2025). RoboGuard: Enhancing robotic system security with ensemble learning. *International Arab Journal of Information Technology*, 20(6), 965–976.
- [3] Anderson, H. S., & Roth, P. (2018). EMBER: An open dataset for training static PE malware machine learning models. *arXiv preprint arXiv:1804.04637*.
- [4] Arifin, M. A. S. (2024). Deteksi aktivitas malware pada Internet of Things menggunakan algoritma decision tree dan random forest. *KLIK - Kumpulan Jurnal Ilmu Komputer*, 11(2), 3070–3080.
- [5] Aziz, T. A. (2025). Klasifikasi malware Android dengan menggunakan metode XGBoost algoritma. *Repositor: Jurnal Ilmiah*, 25(3).

- [6] Chen, Y., et al. (2023). Android malware detection with unbiased confidence guarantees. arXiv:2312.11559.
- [7] Thapa, D., Yadav, H. N. R., & Dhakal, M. (2025). Malware detection using the concept of random forest algorithm. *Journal of Science & Technology*, 10(5), 38–49.
- [8] Agu, E. O., & Ebelogu, C. U. (2023). Malware mitigation through detection using support vector machine and random forest algorithm. *International Journal of Advances in Scientific Research and Engineering*, 9(7), 18–24.
- [9] El Attaoui, A., El Hami, N., & Koulou, Y. (2023). Android malware detection using the random forest algorithm. *Indonesian Journal of Electrical Engineering and Computer Science*, 36(3), 1876–1883.
- [10] Gibert, D., Mateu, C., Planes, J., & Vicens, R. (2020). The rise of machine learning for detection and classification of malware: Research developments, trends, and challenges. *Journal of Network and Computer Applications*, 153, 102526.
- [11] Giffin, M., & Zhang, L. (2007). A survey on malware detection and classification. *Journal of Computer Science and Technology*, 22(3), 323–335.
- [12] Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press.
- [13] Ibrahim, A. M., & Sallow, A. B. (2023). Feature selection for Android malware detection with random forest on smartphones. *Revista de Ingeniería e Innovación Aplicada*.
- [14] D. Illes, “On the impact of dataset size and class imbalance in evaluating machine-learning- based Windows malware detection techniques,” arXiv preprint arXiv:2206.06256, 2022.
- [15] Manokaran, J., & Vairavel, G. (2023). GIWRF-SMOTE: Gini impurity-based weighted random forest with SMOTE for effective malware attack and anomaly detection in IoT-Edge. *Smart Science*, 11(2), 276–292.
- [16] Jumaah, M., Yassin, A., Abduljabbar, Z. A., Jawad, M., Nyangaresi, V. O., & Ali, A. H. (2025). Amalgamating ensemble machine learning soft voting classifier, SMOTE, and Pearson’s correlation coefficient for enhanced malware detection. *Engineering, Technology & Applied Science Research*, 15(3), 22746–22752.
- [17] Kikelomo, A. A., Asafe, N. N., & Oludayo, I. O. (2023). Malware detection system using mathematics of random forest classifier. *International Journal of Advances in Scientific Research and Engineering*, 9(3), 45–53.
- [18] Kunku, K., Zaman, A. N., & Roy, K. (2023). Ransomware detection and classification using machine learning. arXiv:2311.16143.
- [19] Lee, H., et al. (2025). MAD-ANET: Malware detection using attention-based deep learning + SMOTE. *Computers & Security*. Elsevier.
- [20] Li, M., et al. (2023). Investigating feature and model importance in Android malware detection. arXiv:2301.12778.
- [21] Lubis, H. T., Roslina, R., & Tanti, L. (2025). Anomaly detection in computer networks using isolation forest in data mining. *Jurnal Teknik Informatika*, 18(1), 77–86.
- [22] Matin, I. M. M., Agustin, M., Sugiarto, B., & Asri, A. (2023). Deteksi malware menggunakan machine learning dengan metode ensemble. *Prosiding Seminar Nasional Sains dan Teknologi*, 13(1).

- [23] Reddy, M. H., & Gayathri, A. (2025). Predicting the efficient malware detection: Random forest algorithm accuracy compared with linear regression algorithm. *AIP Conference Proceedings*, 3270.
- [24] Poltek Harber. (2024). Performance improvement of random forest algorithm for malware detection on imbalanced dataset using random under-sampling. *Jurnal Informatika*, 7(1).
- [25] Raff, E., Barker, J., Sylvester, J., Brandon, R., Catanzaro, B., & Nicholas, C. K. (2020). Malware detection by eating a whole EXE. *Proceedings of the AAAI Conference on Artificial Intelligence*.
- [26] Rafrastaraa, F. A., Pramunendar, R. A., Prabowo, D. P., & Sudibyo, U. (2023). Optimasi algoritma random forest menggunakan principal component analysis untuk deteksi malware. *Jurnal Teknologi dan Sistem Informasi Bisnis*, 5(2), 123–130.
- [27] Rkhouya, S., & Chougali, K. (2024). Malware detection using a machine-learning based approach. *International Journal of Computer Science and Network Security*, 24(5), 15–22.
- [28] Shirley, S. C. P., et al. (2025). A unified machine learning approach for real-time intrusion, malware, and URL detection using random forest and advanced algorithms. *Journal of Information Systems Engineering and Management*, 10(37s).
- [29] Sampoerna University. (2025). Optimizing malware detection and prevention on proxy servers through RF + RFE. *International Journal of Computer Engineering and Data Science*, 9(2), 47– 55.
- [30] Saxe, J., & Berlin, J. (2015). Deep neural network-based malware detection with Google's TensorFlow. *Proceedings of the 2015 Workshop on Machine Learning for Cyber Security*.
- [31] Shiravi, H., Shiravi, A., & Ghorbani, A. A. (2012). A survey of network traffic analysis and intrusion detection systems. *International Journal of Computer Applications*, 44(9), 16–21.
- [32] Springer. (2023). Framework for detection of malware using random forest classifier. In *Lecture Notes in Electrical Engineering* (Vol. 1073, pp. 725–738). Springer.
- [33] Statista. (2023). Market share held by PC operating systems worldwide.
- [34] Stoneburner, G., Goguen, J., & Feringa, A. (2008). Malicious code and malware. In *Handbook of Information Security: Threats, Vulnerabilities, and Countermeasures*.
- [35] Subuhanto, D., & Tanti, L. (2024). Model deteksi anomali jaringan komputer menggunakan teknik machine learning. In *Prosiding Seminar Nasional Multi Disiplin Ilmu (SENADIMU)* (Vol. 1, No. 1).
- [36] Syamsul Arifin, M. A. (2024). Deteksi aktivitas malware pada Internet of Things menggunakan algoritma decision tree dan random forest. *KLIK - Kumpulan Jurnal Ilmu Komputer*, 11(2), 3070–3080.
- [37] Tanti, L., et al. (2024). Detection and classification model for respiratory diseases using machine learning techniques. In *2024 6th International Conference on Cybernetics and Intelligent System (ICORIS)*. IEEE.
- [38] Tran, T. D., & Vo, H. Q. (2023). Enhancing malware detection by integrating machine learning via CNN/RNN vs RF/KNN/XGB. *arXiv:2311.04372*.
- [39] Wang, Y., et al. (2023). A feature set of small size for the PDF malware detection. *arXiv:2308.04704*.

- [40] Yao, D., & Guo, L. (2017). Feature selection and extraction techniques for malware detection. *Computer Networks and Communications*, 210–221.
- [41] Zhang, H., Lu, C., & Wang, X. (2020). Adversarial machine learning for malware detection: Challenges and opportunities. *International Journal of Information Security*, 19(4), 503–516.
- [42] Zhang, X., Wei, Z., & Zeng, J. (2019). A deep learning approach for malware detection based on file feature analysis. In *Proceedings of the 2019 International Conference on Computer Communication and Network Security*.
- [43] Zhou, Y., Yang, X., & Guo, L. (2018). Dynamic feature extraction for malware detection using machine learning techniques. *Journal of Computer Security*, 26(5), 735–762.