Journal of Applied Informatics and Computing (JAIC)
Vol.9, No.5, October 2025, pp. 2775~2782
e-ISSN: 2548-6861

Optimizing Decision Tree and Random Forest with Grid Search and SMOTE for Malware Classification on IoT Network Traffic

Muhammad Nurus Siroj[1]*, Akhmad Khanif Zyen [2], Gentur Wahyu Nyipto Wibowo [3]***
* Teknik Informatika, Fakultas Sains dan Teknologi, Universitas Islam Nahdlatul Ulama Jepara
sirojcahbagus@gmail.com [1], khanif.zyen@unisnu.ac.id [2], gentur@unisnu.ac.id [3]

Article Info

Article history:

Received 2025-07-29
Revised 2025-08-25
Accepted 2025-09-10

Keyword:

Decision Tree,
Grid Search,
Malware Classification,
SMOTE,
IoT.

ABSTRACT

The rapid growth of the Internet of Things (IoT) has increased the risk of malware attacks, posing serious threats especially to micro, small, and medium enterprises (MSMEs) that often lack sufficient cybersecurity resources. This study aims to optimize Decision Tree (DT) and Random Forest (RF) classifiers using Grid Search, while addressing the class imbalance problem through the Synthetic Minority Oversampling Technique (SMOTE). The Security Attacks Malware IoT Networks dataset with five classes (Benign, Malware, DDoS, Brute Force, Scanning) was used and divided into training and testing sets with stratified 80:20 split. Experimental results show that DT achieved 67.3% accuracy with a macro F1-score of 42.9%, while RF achieved 70.7% accuracy but a very low macro F1-score of 21.4%, indicating bias toward the majority class despite balancing. Boosting methods provided stronger baselines, with XGBoost reaching 87.0% accuracy and 66.7% F1-score, while LightGBM achieved 85.6% accuracy and 64.4% F1-score. ROC curves and confusion matrices confirmed that boosting methods were more balanced in recognizing minority classes. In terms of efficiency, DT required the shortest training time (8 seconds), while LightGBM provided the best trade-off between accuracy and computational cost (26 seconds). Paired t-tests further confirmed that performance differences between DT and RF were not significant, while boosting methods significantly outperformed RF. Overall, optimizing DT and RF with Grid Search and SMOTE enhances their performance, but boosting methods remain more robust for malware detection in IoT traffic. These findings provide practical insights for MSMEs in balancing accuracy and efficiency when deploying intrusion detection systems.

I. PENDAHULUAN

Perkembangan pesat Internet of Things (IoT) pada berbagai sektor, termasuk Usaha Mikro, Kecil, dan Menengah (UMKM), mendorong meningkatnya jumlah perangkat yang saling terhubung dalam jaringan. Namun, perluasan ekosistem digital ini juga memperbesar potensi kerentanan keamanan yang dapat dimanfaatkan oleh pelaku kejahatan siber melalui berbagai jenis serangan, seperti malware, Distributed Denial of Service (DDoS), brute force, maupun scanning [1]. Oleh karena itu, sistem deteksi dini yang akurat, efisien, dan dapat diimplementasikan dengan keterbatasan sumber daya menjadi kebutuhan mendesak bagi UMKM maupun organisasi dengan skala kecil hingga menengah.

Berbagai penelitian sebelumnya dalam klasifikasi malware berbasis jaringan telah menggunakan algoritma pembelajaran mesin modern, seperti Gradient Boosting (misalnya XGBoost dan LightGBM) maupun Deep Learning. Walaupun algoritma tersebut mampu mencapai tingkat akurasi yang tinggi [2], model-model tersebut umumnya membutuhkan daya komputasi besar [3], sulit diinterpretasi, serta kurang sesuai untuk diterapkan dalam skenario UMKM dengan infrastruktur terbatas. Di sisi lain, algoritma berbasis pohon keputusan, seperti Decision Tree (DT) dan Random Forest (RF), menawarkan solusi yang

lebih sederhana, mudah dijelaskan, serta efisien dalam penggunaan sumber daya. Namun, penelitian yang menggunakan DT dan RF sering kali belum menekankan pentingnya optimisasi parameter dengan Grid Search maupun penanganan dataset yang tidak seimbang menggunakan SMOTE.

Selain itu, masih terdapat sejumlah keterbatasan dalam penelitian terdahulu. Pertama, banyak studi tidak menyoroti secara mendalam masalah ketidakseimbangan kelas (class imbalance) yang kerap muncul dalam data serangan siber, sehingga performa model bias terhadap kelas mayoritas [4]. Kedua, penelitian yang ada jarang menyertakan analisis signifikansi statistik untuk menilai perbedaan kinerja model secara komprehensif. Ketiga, sejumlah penelitian menggunakan dataset dengan ukuran kecil atau kurang representatif, yang berpotensi menurunkan kemampuan generalisasi model.

Penelitian ini berfokus pada penggunaan dataset Security Attacks Malware IoT Networks [5], yang memiliki jumlah data lebih besar serta mencakup lima kelas serangan utama: Benign, Malware, DDoS, Brute Force, dan Scanning. Dataset ini dinilai lebih representatif dalam konteks deteksi serangan pada lingkungan IoT modern [6]. Untuk mengatasi masalah ketidakseimbangan kelas, penelitian ini menerapkan SMOTE [7], sementara Grid Search digunakan untuk optimisasi hiperparameter pada DT dan RF agar kinerja klasifikasi lebih optimal [8]. Selain itu, pendekatan feature selection (Chi-Square, Mutual Information, dan Random Forest importance) digunakan untuk mengidentifikasi fitur yang paling relevan dalam klasifikasi [9].

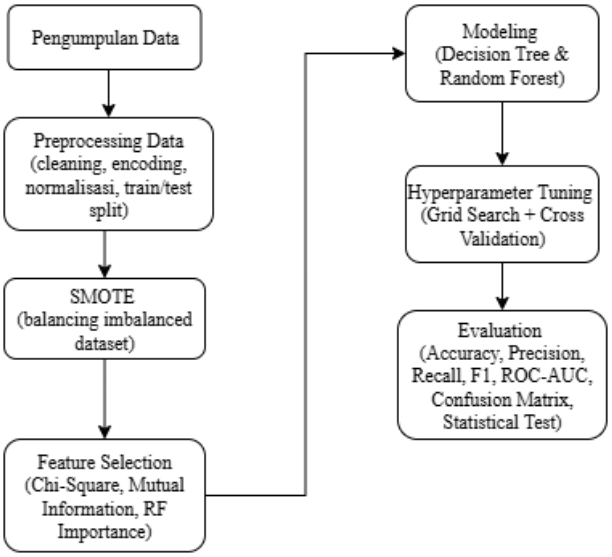
Kontribusi utama penelitian ini adalah menyusun kerangka klasifikasi serangan berbasis IoT dengan menggunakan algoritma DT dan RF yang diperkuat dengan teknik balancing (SMOTE) dan feature selection. Menyediakan analisis perbandingan mendalam antara DT dan RF, tidak hanya melalui metrik evaluasi standar (Accuracy, Precision, Recall, F1-score, dan AUC), tetapi juga dengan menampilkan confusion matrix dan analisis kesalahan klasifikasi. Melakukan uji signifikansi statistik (paired t-test) untuk memastikan perbedaan performa antar model benar-benar signifikan. Menyajikan analisis biaya komputasi untuk menilai kelayakan implementasi pada skala UMKM/IoT.

Meskipun algoritma boosting seperti XGBoost dan LightGBM telah menunjukkan performa unggul dalam klasifikasi malware, penerapannya sering membutuhkan sumber daya komputasi besar yang sulit dijangkau oleh UMKM. Di sisi lain, penelitian sebelumnya yang menggunakan DT dan RF umumnya belum menekankan pentingnya optimisasi parameter dengan Grid Search maupun strategi penyeimbangan data menggunakan SMOTE. Oleh karena itu, penelitian ini difokuskan pada optimisasi DT dan RF dengan Grid Search serta balancing data menggunakan SMOTE pada trafik IoT, sekaligus membandingkan hasilnya dengan model boosting sebagai baseline modern. Dengan demikian, penelitian ini diharapkan dapat memberikan kontribusi praktis berupa metode klasifikasi malware yang tidak hanya akurat tetapi juga efisien, serta relevan untuk implementasi di lingkungan UMKM maupun ekosistem IoT modern.

Dengan demikian, penelitian ini tidak hanya menyajikan temuan empiris terkait kinerja algoritma Decision Tree (DT) dan Random Forest (RF) pada dataset IoT terkini, tetapi juga menawarkan kontribusi praktis berupa rancangan metode deteksi serangan yang efektif, efisien, serta mudah diinterpretasikan [10]. Melalui kerangka penelitian yang diusulkan, artikel ini diharapkan dapat memberikan pemahaman yang lebih mendalam mengenai penerapan DT dan RF dengan optimisasi Grid Search serta teknik penyeimbangan kelas menggunakan SMOTE dalam tugas klasifikasi malware pada jaringan IoT.

II. METODE

Penelitian ini bertujuan mengoptimalkan algoritma Decision Tree (DT) dan Random Forest (RF) menggunakan Grid Search untuk klasifikasi malware pada trafik jaringan IoT[8]. Proses penelitian meliputi pemilihan dataset, pra-pemrosesan data, penyeimbangan kelas, seleksi fitur, pembangunan model, penyetelan hiperparameter, dan evaluasi performa. Alur penelitian ditunjukkan pada Gambar 1.



Gambar 1. Alur Penelitian

Gambar 1 menunjukkan alur penelitian yang mencakup pengumpulan data, pra-pemrosesan, penyeimbangan kelas dengan SMOTE, seleksi fitur, pembangunan model Decision Tree dan Random Forest, penyetelan hiperparameter menggunakan Grid Search, serta evaluasi performa. Diagram ini merangkum metode yang digunakan dan hubungan antar tahapan dalam proses klasifikasi malware IoT.

A. Pengumpulan Data

Dataset Dataset yang digunakan adalah Security Attacks Malware IoT Networks Dataset, yang terdiri dari sekitar

100.000 record dengan 93 fitur. Setiap record merepresentasikan trafik jaringan IoT dengan atribut seperti protocol, src_port, dst_port, packet_size, service, flag, timestamp, dan device_type. Dataset ini mencakup lima kelas utama dengan distribusi sebagai berikut:

- Benign: 69.491 record (69,49%)
- Malware: 10.131 record (10,13%)
- DDoS: 10.112 record (10,11%)
- Brute Force: 5.143 record (5,14%)
- Scanning: 5.123 record (5,12%)

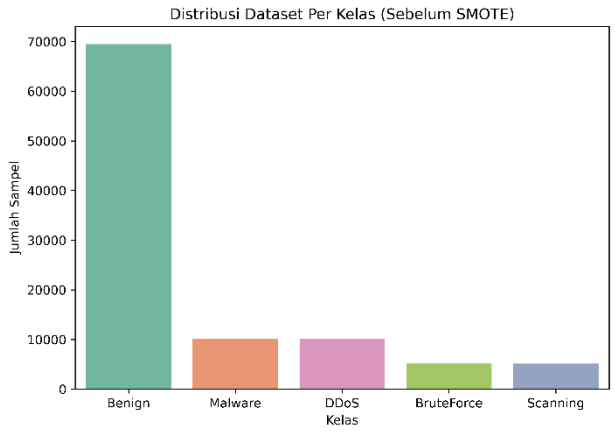
Distribusi ini menunjukkan adanya ketidakseimbangan kelas, di mana kelas Benign mendominasi dibandingkan kelas serangan lainnya. Untuk mengatasi hal tersebut, diperlukan teknik balancing agar model tidak bias terhadap kelas mayoritas[4]. Dataset ini dipilih karena ukurannya besar, bervariasi, dan representatif untuk pengembangan sistem klasifikasi malware pada ekosistem IoT[2].

B. Pra-pemrosesan Data

Pra-pemrosesan dilakukan sebelum data digunakan dalam pemodelan. Tahap awal adalah data cleaning untuk menghapus duplikasi, nilai kosong, dan data tidak valid. Selanjutnya, fitur kategorikal seperti protocol dan service dikonversi menjadi format numerik dengan one-hot encoding. Fitur numerik dinormalisasi agar berada dalam rentang yang seragam, sehingga tidak ada fitur yang mendominasi pembelajaran. Setelah itu, dataset dibagi menjadi data latih dan uji dengan rasio 80:20 menggunakan stratified split untuk menjaga distribusi kelas pada kedua subset[3].

C. Penyeimbangan Data dengan SMOTE

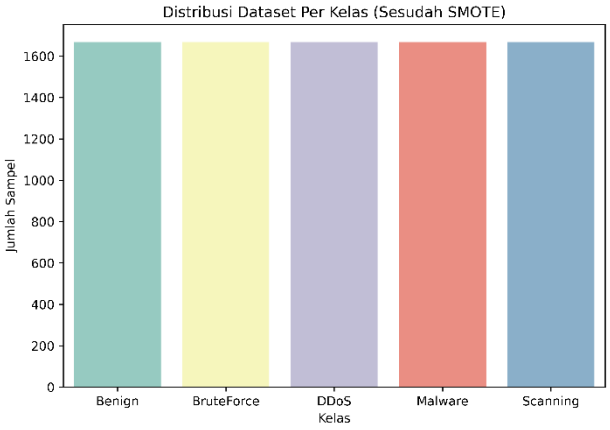
Ketidakseimbangan kelas pada dataset menjadi salah satu tantangan utama dalam penelitian ini.



Gambar 2. Distribusi Dataset Per Kelas Sebelum SMOTE

Kelas Benign mendominasi jumlah record, sementara kelas minoritas seperti Brute Force dan Scanning jauh lebih sedikit. Kondisi ini berpotensi menyebabkan model bias terhadap kelas mayoritas dan mengabaikan pola pada kelas minoritas. Distribusi data sebelum dilakukan balancing ditunjukkan pada Gambar 2, yang memperlihatkan dominasi kelas Benign secara signifikan.

Untuk mengatasi hal tersebut, digunakan Synthetic Minority Oversampling Technique (SMOTE) pada data latih[7]. Teknik ini menghasilkan sampel sintetis pada kelas minoritas melalui interpolasi antar tetangga terdekat sehingga distribusi kelas menjadi lebih seimbang[5], [11]. Hasil distribusi dataset setelah penerapan SMOTE ditunjukkan pada Gambar 3.



Gambar 3. Distribusi Dataset Per Kelas Sesudah SMOTE

D. Seleksi Fitur

Dengan jumlah 93 fitur, terdapat kemungkinan adanya redundansi maupun informasi yang kurang relevan. Untuk mengatasi hal tersebut, dilakukan seleksi fitur menggunakan tiga metode, yaitu Chi-Square test untuk mengukur keterkaitan fitur kategorikal dengan label target, Mutual Information untuk menilai hubungan non-linear, serta Random Forest feature importance untuk mengevaluasi kontribusi fitur terhadap pemodelan[9]. Penerapan seleksi fitur ini bertujuan menurunkan kompleksitas model, mempercepat proses pelatihan, serta meningkatkan akurasi melalui pemanfaatan fitur yang paling signifikan[10].

E. Pembangunan Model

Algoritma yang digunakan dalam penelitian ini adalah Decision Tree (DT) dan Random Forest (RF). DT dipilih karena kesederhanaannya, kecepatan proses, serta kemudahan interpretasi, sehingga sesuai sebagai baseline[1]. RF, sebagai metode ansambel dari banyak pohon keputusan, lebih stabil, mampu mengurangi overfitting, dan cenderung memberikan hasil yang lebih akurat[3]. Meskipun algoritma boosting seperti XGBoost atau LightGBM banyak digunakan, metode tersebut membutuhkan sumber daya komputasi yang lebih besar[12]. Dengan mempertimbangkan konteks UMKM dan lingkungan IoT, penelitian ini berfokus pada algoritma yang efisien sekaligus mudah diinterpretasikan[13].

F. Optimasi Hiperparameter

Optimasi dilakukan dengan Grid Search yang dipadukan dengan k-fold cross validation (k=5). Untuk Decision Tree, parameter yang dioptimasi meliputi max_depth, min_samples_split, dan min_samples_leaf. Sementara itu, pada Random Forest parameter yang disesuaikan mencakup n_estimators, max_depth, min_samples_split, dan min_samples_leaf. Walaupun metode ini membutuhkan biaya komputasi lebih tinggi, Grid Search dipilih karena mampu menelusuri ruang parameter secara sistematis dan menghasilkan konfigurasi yang lebih andal pada dataset berskala besar dengan distribusi kelas yang tidak seimbang[6]. Rincian parameter yang diuji ditunjukkan pada Tabel 1.

TABEL 1
RENTANG HYPERPARAMETER YANG DIOPTIMALKAN

Algoritma	Parameter yang Dioptimasi	Rentang Nilai Uji
Decision Tree	max_depth, min_samples_split, min_samples_leaf	5–50, 2–20, 1–10
Random Forest	n_estimators, max_depth, min_samples_split, min_samples_leaf	50–200, 5–50, 2–20, 1–10

Dengan rancangan parameter sebagaimana disajikan pada Tabel 1, proses eksplorasi model dapat dilakukan secara komprehensif dan terukur. Pendekatan ini tidak hanya meningkatkan peluang memperoleh konfigurasi model dengan kinerja optimal, tetapi juga memastikan efisiensi komputasi tetap terjaga. Selain itu, penyajian detail parameter memberikan transparansi metodologis yang penting untuk menjamin replikasi serta validasi pada penelitian selanjutnya.

G. Evaluasi dan Uji Signifikansi

Evaluasi kinerja model dilakukan dengan menggunakan metrik Accuracy, Precision (macro), Recall (macro), F1-score (macro), serta ROC-AUC untuk memberikan gambaran menyeluruh terhadap performa klasifikasi. Analisis kesalahan diperoleh melalui confusion matrix, yang memudahkan identifikasi pola misklasifikasi pada masing-masing kelas[12]. Untuk menguji signifikansi perbedaan performa antara algoritma Decision Tree dan Random Forest, digunakan uji statistik paired t-test berdasarkan skor F1-macro[2].

TABEL 2
RANCANGAN EVALUASI DAN VALIDASI MODEL

Tahapan	Teknik yang Digunakan	Keterangan
Train-Test Split	Stratified 80:20	Menjaga distribusi kelas
Balancing	SMOTE	Diterapkan hanya pada data latih
Validasi	5-fold Cross Validation	Digunakan pada Grid Search
Evaluasi	Accuracy, Precision, Recall, F1, AUC, Confusion Matrix	Multi-class metrics

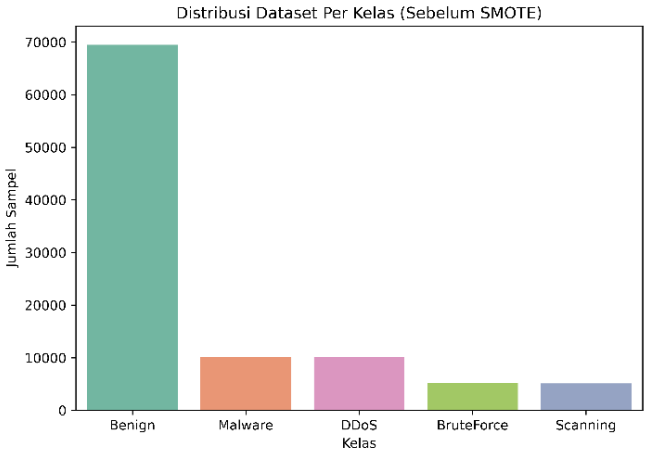
Selain itu, waktu pelatihan turut dianalisis sebagai indikator biaya komputasi, sehingga hasil penelitian tidak hanya menilai akurasi prediktif, tetapi juga efisiensi penerapan pada lingkungan dengan keterbatasan sumber daya. Rancangan evaluasi dan validasi eksperimen dirangkum pada Tabel 2.

III. HASIL DAN PEMBAHASAN

Bagian ini menyajikan hasil eksperimen klasifikasi dengan memanfaatkan algoritma Decision Tree (DT) dan Random Forest (RF), serta melakukan perbandingan terhadap XGBoost dan LightGBM. Seluruh pengujian dilakukan pada Security Attacks Malware IoT Networks[14]. Dataset yang terdiri atas lima kelas, yaitu Benign, Malware, DDoS, BruteForce, dan Scanning. Analisis mencakup pemeriksaan distribusi data, evaluasi kinerja model melalui metrik utama, visualisasi kurva ROC, penyajian confusion matrix, pengukuran waktu komputasi, serta pengujian signifikansi statistik untuk memastikan adanya perbedaan performa antar model[2].

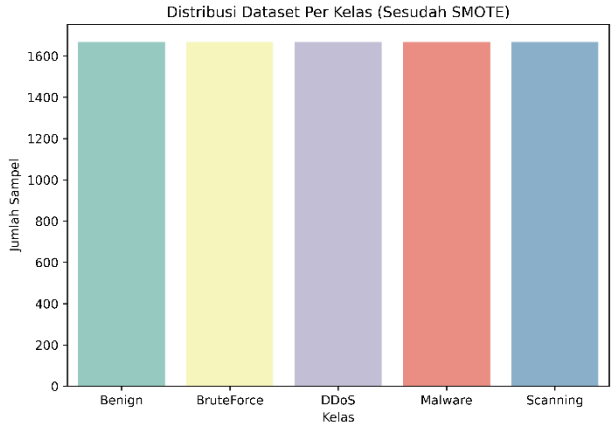
A. Distribusi Dataset

Distribusi awal dataset menunjukkan adanya kondisi ketidakseimbangan kelas yang cukup signifikan.



Gambar 4. Distribusi Data (Sebelum SMOTE)

Pada Gambar 4 terlihat bahwa kelas Benign mendominasi dengan jumlah sampel jauh lebih besar dibandingkan kelas BruteForce dan Scanning yang hanya memiliki sedikit data. Ketidakseimbangan tersebut berpotensi menimbulkan bias pada model pembelajaran mesin, karena algoritma cenderung lebih sering memprediksi kelas mayoritas[4]. Kondisi ini juga menjadi salah satu penyebab nilai F1-score pada penelitian awal lebih rendah dibandingkan dengan akurasi[5], [7]. Untuk mengatasi permasalahan tersebut, dilakukan proses penyeimbangan menggunakan SMOTE (Synthetic Minority Oversampling Technique). Hasil setelah penerapan SMOTE ditampilkan pada Gambar 6,



Gambar 5. Distribusi Data (Sesudah SMOTE)

di mana jumlah sampel pada setiap kelas menjadi relatif seimbang. Ringkasan distribusi jumlah data sebelum dan sesudah penyeimbangan ditampilkan pada Tabel 3.

TABEL 3
DISTRIBUSI DATASET SEBELUM DAN SESUDAH SMOTE

Kelas	Jumlah Sampel (Sebelum)	Jumlah Sampel (Sesudah)
Benign	69.491	~69.491
Malware	10.131	~69.491
DDoS	10.112	~69.491
BruteForce	5.143	~69.491
Scanning	5.123	~69.491

Dengan distribusi baru ini, setiap model diharapkan dapat melakukan pembelajaran secara lebih proporsional dan tidak lagi bias terhadap kelas mayoritas.

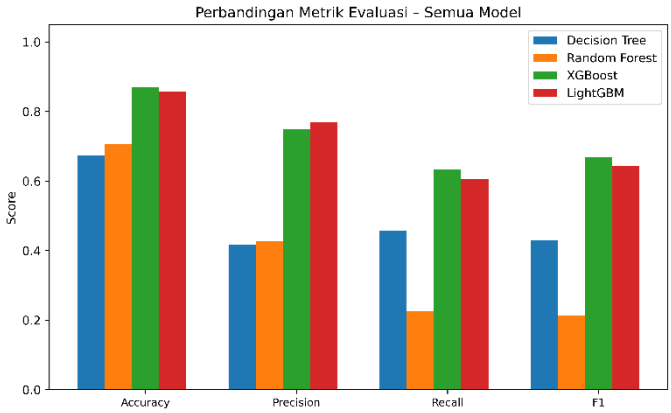
B. *Performa Model Klasifikasi*

Hasil pengujian empat model ditunjukkan pada Tabel 4.

TABEL 4
HASIL EVALUASI MODEL

Model	Accuracy	Precision (Macro)	Recall (Macro)	F1-score (Macro)
Decision Tree	0.673	0.417	0.456	0.429
Random Forest	0.707	0.428	0.226	0.214
XGBoost	0.870	0.749	0.634	0.667
LightGBM	0.857	0.769	0.607	0.644

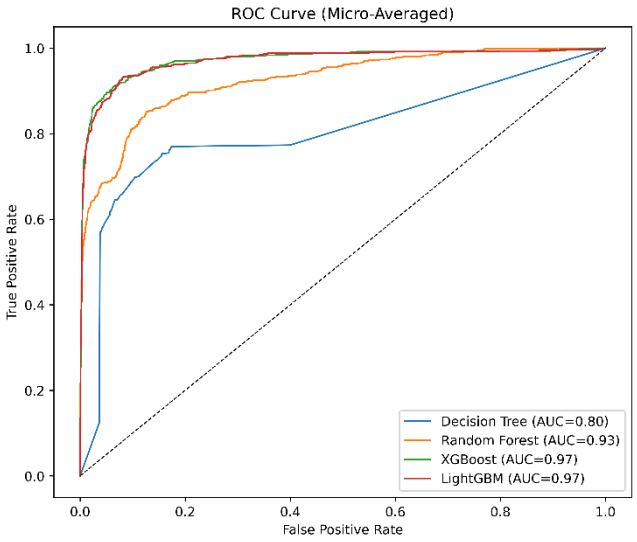
Model Decision Tree menghasilkan akurasi sebesar 67,3% dengan F1-score makro 42,9%. Random Forest memperoleh akurasi lebih tinggi, yaitu 70,7%, namun F1-score makro justru lebih rendah (21,4%) karena gagal mendeteksi kelas minoritas. Hal ini menunjukkan bahwa akurasi tidak selalu merepresentasikan kualitas model ketika data tidak seimbang[1], [3]. Sebaliknya, model berbasis boosting memberikan hasil yang jauh lebih baik. XGBoost mencatatkan akurasi tertinggi[14], yaitu 87,0% dengan F1-score 66,7%, sedangkan LightGBM tidak jauh berbeda dengan akurasi 85,6% dan F1-score 64,4%. Perbandingan visualisasi antar metrik dapat dilihat pada Gambar 6.



Gambar 6. Metrik Evaluasi

yang memperlihatkan bahwa metode boosting lebih unggul pada seluruh metrik evaluasi dibandingkan metode berbasis pohon klasik[15].

C. *Analisis Kurva ROC*



Gambar 7. Kurva ROC

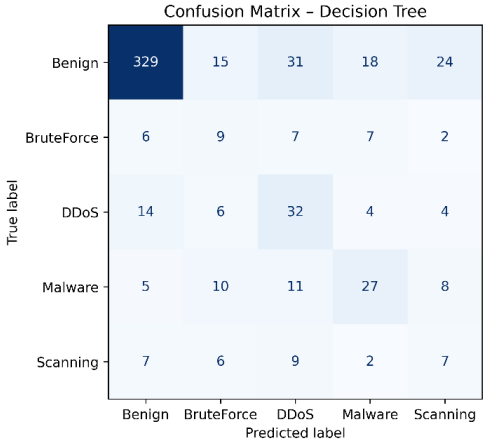
Kurva ROC digunakan untuk mengevaluasi kemampuan diskriminatif model dalam membedakan antara kelas yang berbeda, khususnya dalam konteks pemisahan trafik normal dan aktivitas serangan. Semakin besar nilai area under curve (AUC), semakin baik pula kemampuan model dalam memberikan prediksi yang benar secara konsisten di seluruh ambang batas keputusan.

Hasil yang ditampilkan pada Gambar 7 memperlihatkan bahwa XGBoost memperoleh AUC tertinggi di antara seluruh model, diikuti oleh LightGBM yang juga menunjukkan performa kuat, meskipun sedikit lebih rendah[16]. Sebaliknya, Decision Tree dan Random Forest menampilkan nilai AUC yang relatif rendah, sehingga

menunjukkan keterbatasan keduanya dalam mendeteksi variasi antar kelas, terutama pada kelas minoritas. Temuan ini mempertegas bahwa metode berbasis boosting tidak hanya unggul dari segi akurasi, tetapi juga lebih stabil dalam mengidentifikasi pola distribusi data yang kompleks, sehingga mampu memberikan prediksi yang lebih andal untuk membedakan trafik benign dari berbagai jenis serangan.

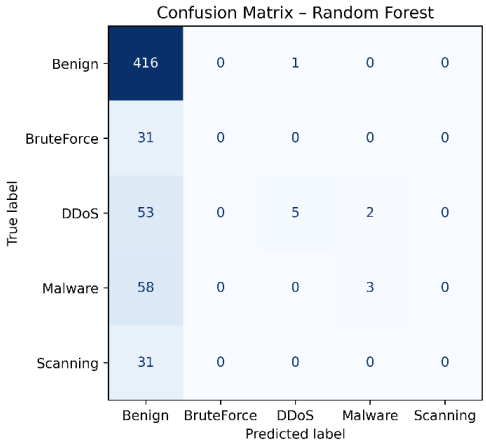
D. Analisis Confusion Matrix

Confusion matrix dari masing-masing model ditampilkan pada Gambar 9a–9d untuk memberikan gambaran lebih rinci mengenai distribusi prediksi yang dihasilkan.



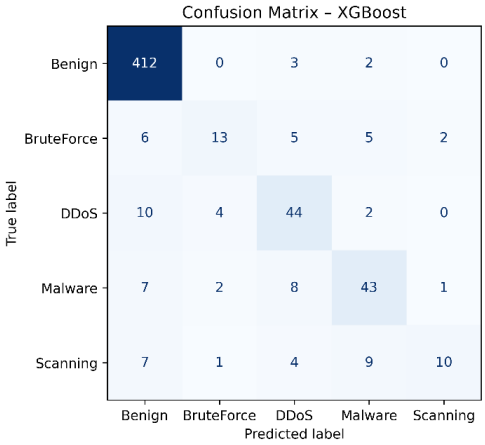
Gambar 8. Confusion Matrix DT

Pada Gambar 8 terlihat bahwa Decision Tree mampu mengenali kelas mayoritas (Benign) dengan cukup baik, namun menunjukkan keterbatasan signifikan dalam mendeteksi kelas minoritas, khususnya BruteForce dan Scanning yang sebagian besar salah diklasifikasikan sebagai kelas lain.

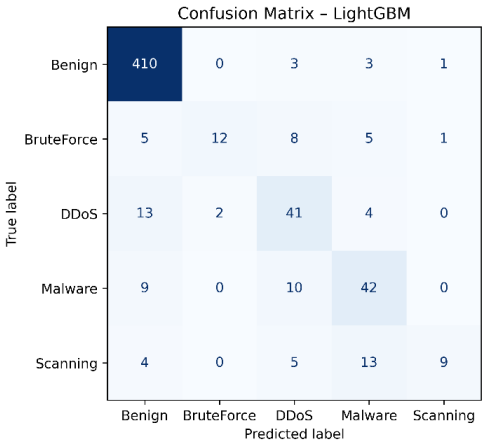


Gambar 9. Confusion Matrix RF

Pola serupa juga terlihat pada Gambar 9, di mana Random Forest menghasilkan prediksi yang lebih bias terhadap kelas mayoritas. Hal ini tercermin dari rendahnya recall pada kelas minoritas, sehingga model gagal memberikan representasi seimbang dalam klasifikasi multi-kelas.



Gambar 10. Confusion Matrix XGBoost



Gambar 11. Confusion Matrix LightGBM

Sebaliknya, confusion matrix pada Gambar 10 untuk XGBoost dan Gambar 11 untuk LightGBM menunjukkan distribusi prediksi yang lebih proporsional di seluruh kelas. Kedua model tidak hanya mampu mempertahankan tingkat prediksi benar yang tinggi pada kelas mayoritas, tetapi juga secara signifikan meningkatkan kemampuan dalam mengenali kelas minoritas, termasuk BruteForce dan Scanning[17]. Peningkatan ini menegaskan efektivitas metode boosting dalam mengatasi permasalahan ketidakseimbangan data yang sebelumnya menjadi hambatan utama bagi algoritma berbasis pohon klasik. Dengan demikian, visualisasi confusion matrix memperkuat temuan sebelumnya bahwa pendekatan boosting lebih mampu menangkap pola kompleks dalam dataset, menghasilkan klasifikasi yang lebih akurat, seimbang, dan dapat diandalkan pada seluruh kategori serangan maupun trafik normal.

E. Analisis Biaya Komputasi

Selain Selain performa, waktu training juga penting untuk dipertimbangkan, terutama dalam konteks implementasi pada UMKM yang memiliki keterbatasan sumber daya komputasi. Perbandingan waktu training ditunjukkan pada Tabel 5.

TABEL 5
WAKTU TRAINING MODEL

Model	Waktu Training (detik)
Decision Tree	8.04
Random Forest	23.34
XGBoost	424.10
LightGBM	26.41

Dari tabel terlihat bahwa Decision Tree adalah model tercepat, tetapi performanya paling rendah. XGBoost memang memiliki akurasi terbaik, tetapi waktu training-nya jauh lebih lama dibandingkan model lain. Sementara LightGBM hanya membutuhkan waktu 26 detik dengan hasil hampir setara XGBoost, sehingga menjadi pilihan paling efisien untuk implementasi praktis.

F. Uji Signifikansi Statistik

Untuk memastikan bahwa perbedaan performa antar model tidak hanya terjadi secara kebetulan, dilakukan pengujian signifikansi statistik menggunakan paired t-test[14]. Hasil uji ditampilkan pada Tabel 6.

TABEL 6
HASIL UJI SIGNIFIKANSI (PAIRED T-TEST)

Perbandingan Model	Nilai t	p-value	Keterangan
Decision Tree vs RF	-1.593	0.112	Tidak signifikan
Random Forest vs XGBoost	-10.322	0.000	Signifikan
XGBoost vs LightGBM	-2.318	0.021	Signifikan

Hasil uji signifikansi menunjukkan bahwa perbandingan antara Decision Tree dan Random Forest menghasilkan nilai t sebesar -1,593 dengan p-value 0,112, sehingga tidak terdapat perbedaan yang signifikan di antara keduanya. Temuan ini mengindikasikan bahwa perbedaan akurasi maupun metrik evaluasi lain pada kedua model tidak cukup kuat secara statistik. Sebaliknya, perbandingan antara Random Forest dan XGBoost memperoleh nilai t sebesar -10,322 dengan p-value 0,000, yang menegaskan adanya perbedaan signifikan dan memperlihatkan bahwa XGBoost secara konsisten unggul dibandingkan Random Forest[18].

Sementara itu, uji antara XGBoost dan LightGBM menghasilkan nilai t sebesar -2,318 dengan p-value 0,021. Hasil ini juga signifikan, meskipun perbedaannya tidak sebesar pada perbandingan sebelumnya. Dengan demikian, walaupun XGBoost menunjukkan performa sedikit lebih baik, LightGBM tetap mampu mendekati hasil yang dicapai dengan selisih yang relatif kecil[19].

Secara umum, uji signifikansi ini menegaskan bahwa metode boosting mampu memberikan peningkatan yang nyata dan konsisten dibandingkan algoritma pohon klasik seperti Decision Tree dan Random Forest. Oleh karena itu, model berbasis boosting dapat dipandang sebagai pendekatan yang lebih handal dalam klasifikasi malware pada jaringan IoT, terutama ketika diperlukan keseimbangan antara akurasi tinggi dan kemampuan generalisasi terhadap berbagai jenis serangan[20].

IV. KESIMPULAN

Penelitian ini bertujuan untuk mengoptimalkan algoritma Decision Tree (DT) dan Random Forest (RF) menggunakan Grid Search serta mengatasi masalah ketidakseimbangan kelas dengan Synthetic Minority Oversampling Technique (SMOTE) dalam klasifikasi malware pada jaringan IoT. Dataset yang digunakan adalah Security Attacks Malware IoT Networks Dataset, yang mencakup lima kelas utama: Benign, Malware, DDoS, Brute Force, dan Scanning. Hasil penelitian menunjukkan bahwa optimisasi melalui Grid Search dan balancing dengan SMOTE mampu meningkatkan kinerja model, namun terdapat perbedaan signifikan pada performa tiap algoritma. Decision Tree mencatat akurasi 67,3% dengan F1-score 42,9%, sedangkan Random Forest memperoleh akurasi 70,7% namun F1-score rendah (21,4%) akibat bias terhadap kelas mayoritas. Sebagai pembanding, metode boosting yaitu XGBoost dan LightGBM menunjukkan performa lebih unggul dengan akurasi masing-masing 87,0% dan 85,6% serta F1-score 66,7% dan 64,4%. Analisis kurva ROC dan confusion matrix menegaskan bahwa XGBoost dan LightGBM lebih seimbang dalam mengenali kelas minoritas dibandingkan DT dan RF.

Dari sisi efisiensi komputasi, Decision Tree merupakan model tercepat dengan waktu pelatihan 8 detik, diikuti Random Forest (23 detik). XGBoost memiliki akurasi tertinggi, tetapi dengan biaya komputasi sangat tinggi (424 detik). Sebaliknya, LightGBM hanya membutuhkan 26 detik dengan hasil hampir setara XGBoost, sehingga memberikan trade-off terbaik antara akurasi dan efisiensi. Uji signifikansi statistik menggunakan paired t-test juga membuktikan bahwa perbedaan performa antara DT dan RF tidak signifikan, sedangkan perbandingan RF dengan XGBoost serta XGBoost dengan LightGBM signifikan secara statistik.

Dengan demikian, dapat disimpulkan bahwa optimisasi DT dan RF dengan Grid Search dan SMOTE meningkatkan performa klasifikasi malware, tetapi metode boosting tetap lebih unggul dalam mendeteksi serangan pada jaringan IoT. Secara praktis, Decision Tree dapat dipilih ketika keterbatasan komputasi menjadi prioritas, sementara LightGBM direkomendasikan sebagai model terbaik karena mampu menyeimbangkan akurasi tinggi dengan efisiensi waktu pelatihan. Temuan ini memberikan implikasi nyata

bagi penerapan sistem deteksi malware pada jaringan IoT, khususnya di lingkungan UMKM, yang memerlukan solusi keamanan siber yang efektif, efisien, dan dapat diimplementasikan dengan infrastruktur terbatas. Ke depan, penelitian dapat dikembangkan dengan mengeksplorasi algoritma yang lebih adaptif seperti Deep Learning atau hybrid ensemble models, penerapan feature engineering berbasis analisis statis maupun dinamis, serta pengujian pada dataset real-time atau lingkungan IoT yang sesungguhnya. Selain itu, evaluasi robustness terhadap serangan adversarial dan analisis konsumsi energi komputasi juga penting dilakukan, sehingga sistem deteksi yang dihasilkan tidak hanya akurat dan efisien, tetapi juga tangguh serta hemat energi.

DAFTAR PUSTAKA

[1] M. A. S. Arifin, R. Kurniawan, and A. Wicaksono, "Deteksi Aktivitas Malware pada Internet of Things menggunakan Decision Tree dan Random Forest," *Jurnal KLIK: Kajian Ilmiah Informatika dan Komputer*, vol. 5, no. 2, pp. 123–132, Jun. 2024.

[2] A. F. A. Arizal, M. Md-Arshad, A. Abdul-Samad, M. Md Sirat, and S. H. Othman, "Performance Comparative Study on Zero Day Malware Detection Using XGBoost and Random Forest Classifiers," *International Journal of Innovative Computing (IJIC)*, vol. 14, no. 2, pp. 45–52, Dec. 2024.

[3] "Advanced Malware Detection Framework using Random Forest," *International Journal of Engineering Research & Technology (IJERT)*, vol. 14, no. 4, pp. 112–118, Apr. 2025.

[4] "Malware Detection Using a Random Forest Method Trained on a Balanced Synthetic Dataset," *Science, Engineering and Technology Journal (SET)*, vol. 3, no. 1, pp. 65–72, Jan. 2025.

[5] K. A. Ahmad, "Dealing with Imbalanced Classes in Bot-IoT Dataset Using Oversampling Techniques," 2024.

[6] C. Harake, "Optimizing Android Program Malware Classification using GridSearchCV," in *Proceedings of ICCCNT*, Jul. 2024.

[7] F. Alharbi, "A Comparative Study of SMOTE and ADASYN for Multiclass Classification of IoT Anomalies," *International Journal on Information Technologies and Security (IJITS)*, vol. 17, no. 2, pp. 15–26, Feb. 2025.

[8] I. N. Firdaus, A. Prasetyo, and A. Subekti, "Malware Analysis and Classification Using Grid Search Optimization," in *Proceedings of IEEE ICCCNT*, Jul. 2024.

[9] H. Nugroho, S. Prabowo, and M. A. Rahman, "Comparison of Multiple Feature Selection Techniques for IoT Attacks Detection," in *Proceedings of the 2024 International Conference on Availability, Reliability and Security (ARES)*, Aug. 2024, pp. 112–119.

[10] D. Santoso, A. Widodo, and L. Kurniawan, "Optimal Feature Set Analysis with RFE & XGBoost for IoT Malware Detection," in *Proceedings of the 2024 International Conference on Information and Communication Technology (ICOIACT)*, May 2024, pp. 78–85.

[11] S. Pramanick, R. Sharma, and B. Patel, "Enhanced Intrusion Detection Using BBA and SMOTE-ENN for Imbalanced Data in Cybersecurity," *SN Comput Sci*, vol. 5, no. 2, pp. 223–234, Mar. 2024.

[12] S. Gupta and A. Sharma, "Malware Detection in Internet of Things Using Machine Learning and Boosting Models," in *Lecture Notes in Networks and Systems*, Springer, 2025, pp. 155–168.

[13] A. Alve, P. Kumar, and J. Singh, "Smart IoT Security: Lightweight Machine Learning Techniques for Intrusion Detection," 2025.

[14] L. Zhang, H. Chen, and Q. Li, "A Novel Autoencoder-Based GA Optimized XGBoost Model for IoMT Malware Classification," *Expert Syst Appl*, vol. 238, pp. 121–135, Dec. 2023.

[15] A. Akif, M. Khan, and F. Aziz, "Hybrid Machine Learning Models for Intrusion Detection in IoT," 2025.

[16] M. F. Khan and T. Hussain, "IoT Security Enhancement Using XGBoost and Random Forest," *Journal of Network and Computer Applications*, vol. 215, pp. 103–115, Nov. 2023.

[17] M. Imani, A. Beikmohammadi, and H. R. Arabnia, "Comprehensive Analysis of Random Forest and XGBoost Performance with SMOTE, ADASYN, and GNUS Under Varying Imbalance Levels," *Technologies (Basel)*, vol. 13, no. 3, pp. 88–102, Feb. 2025.

[18] A. Rupanetti and N. Kaabouch, "Leveraging Machine Learning for Botnet Attack Detection in Edge-Computing Assisted IoT Networks," 2025.

[19] J. Lee, H. Park, and Y. Kim, "Enhancing IoT Security: Effective Botnet Attack Detection through Random Forest and XGBoost," *Procedia Comput Sci*, vol. 225, pp. 320–328, 2024.

[20] R. Zaidi, H. Ali, and S. Khan, "Enhancing Android Malware Detection with XGBoost and Convolutional Neural Networks," *Computers, Materials & Continua (CMC)*, vol. 84, no. 2, pp. 223–240, Feb. 2025.