Komputika: Jurnal Sistem Komputer
Volume 13, Nomor 2, Oktober 2024, hlm. 245 - 252
DOI: 10.34010/komputika.v13i2.12609

p-ISSN: 2252-9039
e-ISSN: 2655-3198

Perbandingan Algoritma Decision Tree dan K-Nearest Neighbor untuk Klasifikasi Serangan Jaringan IoT

Zishwa Muhammad Jauhar Nafis[1*], Rahmatun Nazilla[2], Rega Nugraha[3], Shofwatul 'Uyun[4]

[1, 2, 3, 4)] Magister Informatika, Fakultas Sains dan Teknologi, Universitas Islam Negeri Sunan Kalijaga
Jl. Marsda Adisucipto Yogyakarta 55281
**email: 23206051015@student.uin-suka.ac.id*

(Naskah masuk: *28 Maret 2024*; diterima untuk diterbitkan: *30 September 2024*)

ABSTRAK *– Seiring dengan perkembangan jumlah penggunaan Internet of Things yang terus meningkat dan meluas. Ancaman keamanan pada jaringan IoT juga meningkat. Terdapat beberapa teknik yang diterapkan untuk mengatasi ancaman keamanan ini. Salah satunya adalah teknik untuk mengklasifikasi suatu aktivitas yang termasuk dalam serangan atau bukan beserta jenis serangannya. Machine learning dapat dimanfaatkan untuk proses pengklasifikasian ini. Diantara algoritma machine learning yang dapat digunakan untuk penelitian ini adalah pendekatan algoritma Decision Tree dan K-Nearest Neighbor. Penelitian ini bertujuan untuk mendapatkan hasil klasifikasi terbaik untuk mendeteksi jenis serangan jaringan IoT baik dalam klasifikasi biner maupun klasifikasi multikleas. Dalam penelitian ini memanfaatkan Dataset Edge-IIoTset Cyber Security Dataset of IoT & IIoT. Hasil nilai evaluasi yang didapatkan menunjukkan bahwa performa algoritma Decision Tree lebih baik dibandingkan dengan Algoritma KNN. Dengan selisih nilai presisi, recall, F1-score, dan akurasi secara berurutan adalah 0.15, 0.18, 0.17 dan 0.08 dalam klasifikasi biner. Sedangkan dalam klasifikasi multikelas mendapatkan nilai selisih antar kedua algoritma sebesar 0.26, 0.20, 0.22, dan 0.23 secara berurutan untuk presisi, recall, F1-score, dan akurasi.*

Kata Kunci *– IoT; Serangan Jaringan; Klasifikasi; Decision Tree; K-Nearest Neighbor.*

Comparison of Decision Tree and K-Nearest Neighbour Algorithms for IoT Network Attack Classification

ABSTRACT *– As the number of uses of the Internet of Things continues to increase and expand. Security threats on IoT networks are also increasing. There are several techniques applied to overcome this security threat. One of them is a technique to classify an activity that is included in an attack or not along with the type of attack. Machine learning can be utilized for this classification process. Among the machine learning algorithms that can be used for this research are the Decision Tree and K-Nearest Neighbor algorithm approaches. This research aims to get the best classification results to detect the type of IoT network attack in both binary classification and multiclass classification. This research utilizes the Edge-IIoTset Cyber Security Dataset of IoT & IIoT. The results of the evaluation values obtained show that the performance of the Decision Tree algorithm is better than the KNN algorithm. With the difference in precision, recall, F1-score, and accuracy values are 0.15, 0.18, 0.17 and 0.08 in binary classification, respectively. While in multiclass classification, the difference value between the two algorithms is 0.26, 0.20, 0.22, and 0.23 respectively for precision, recall, F1-score, and accuracy.*

Keywords *– IoT; Network Attack; Classification; Decision Tree; K-Nearest Neighbor.*

1. PENDAHULUAN

Internet of Thing (IoT) merupakan koneksi unik perangkat komputasi yang dapat diidentifikasi, terintegrasi dan dapat mengirimkan data melalui jaringan tanpa interaksi antara pengguna atau perangkat [1]. IoT membuat banyak perangkat dapat tersambung bersama ke internet. Jumlah perkembangan perangkat yang menggunakan IoT saat ini semakin hari semaki pesat. Perkembangan

perangkat IoT membawa perubahan disetiap aspek kehidupan manusia saat ini, seperti *smart home*, *smart farming*, *smart city* dan lain-lain yang menggunakan internet guna memantau informasi yang diperlukan[2]. Kehadiran teknologi IoT sangat berharga dan sangat membantu dalam banyak aspek kehidupan manusia saat ini. Namun, kemudahan yang diberikan IoT tidak luput dari ancaman keamanan siber yang substansial bagi sistem yang lemah.

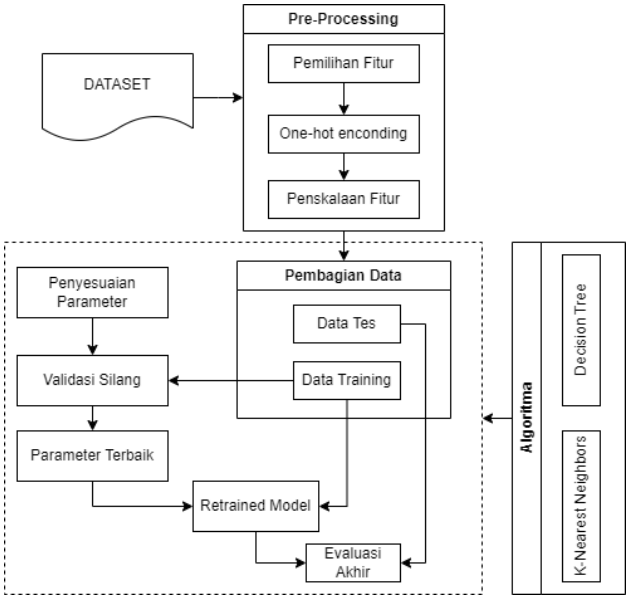
Dalam rangka mendorong pemanfaatan IoT secara luas, faktor keamanan harus diatasi. IoT adalah sistem yang kompleks. Hal ini disebabkan oleh banyaknya entitas yang berbeda, seperti data, perangkat, jalur komunikasi, sensor, dan lainnya, serta banyaknya peralatan yang memiliki berbagai kemampuan untuk berkomunikasi dan mengolah data. Karena banyaknya entitas dan data yang terlibat dalam IoT, ada ancaman keamanan yang dapat membahayakan konsumen. Ancaman ini dapat berupa kemampuan orang yang tidak berhak untuk mengakses data dan menyalahgunakan informasi pribadi, memfasilitasi serangan terhadap sistem lain, serta mengancam keselamatan pribadi penggunanya. Tergantung pada tujuan serangan, ancaman yang dapat memengaruhi entitas IoT sangat beragam [3].

Ari Sandriana dkk pada tahun 2022 melakukan penelitiain terkait klasifikasi biner serangan malware terhadap lalu lintas jaringan IoT menggunakan algoritma K-Nearest Neighbour (K-NN) dengan nilai akurasi yang didapat sebesar 0.94 [1]. Pada tahun 2021 Fery Antony melakukan penelitian deteksi serangan denial of service pada Internet of things menggunakan Finite-State Automata yang berhasil mengurangi serangan sebesar 0.6 [4]. Randi dkk menyebutkan bahwa algoritma CNN bekerja dengan sangat baik dalam pengklasifikasian serangan siber pada IoT dengan hasil akurasi 0.99, pada penelitian ini menggunakan 10 tipe jenis serangan yang berasal dari dataset publik di UCI *Repository*[5]. Nimisha pandey melakukan penelitian untuk mendeteksi serangan DDoS dalam jaringan IoT menggunakan dataset CICDDoS2019 dengan beberapa algoritma Machine Learning dan memberikan hasil model terbaik algoritma Random Forest [6].

Berdasarkan hasil penelitian yang dilakukan dalam klasifikasi deteksi serangan jaringan IoT menunjukan bahwa ada banyak penerapan algoritma yang menghasilkan akurasi baik. Sehingga, pada penelitian ini bertujuan untuk membandingkan performa dari Algoritma Decision Tree dan K-Nearest Neighbour dalam mengklasifikasikan serangan dalam jaringan IoT.

2. METODE DAN BAHAN

Dalam penelitian ini terdapat beberapa tahapan yang akan dilakukan, dimulai dari pemilihan dataset sampai tahapan akhir berupa evaluasi. Alur dari tahapan penelitian ini dapat dilihat pada Gambar 1. Detail dari setiap tahapan akan dijelaskan pada masing-masing sub bab berikut ini.



Gambar 1. Metode Penelitian

Dataset

Dalam penelitian ini dataset yang digunakan adalah *Edge-IIoTset Cyber Security Dataset of IoT & IIoT,* merupakan kumpulan data rekaman lalu lintas penggunaan jaringan IoT secara nyata yang memuat 14 jenis serangan yang terjadi pada jaringan IoT. Dataset ini bisa dimanfaatkan untuk pendeteksian serangan dengan *machine learning*. Dataset ini Terdiri dari 15.2196 data. Dengan distribusi dataset sebagaimana dalam Tabel 1.

Tabel 1. Distribusi Dataset *Edge-IIoTset Cyber Security Dataset of IoT & IIoT*

No	Jenis Serangan	Jumlah
1	Normal	24.101
2	DDoS_UDP	14.498
3	DDoS_ICMP	13.096
4	DDoS_HTTP	10.495
5	SQL_injection	10.282
6	DDoS_TCP	10.247
7	Uploading	10.214
8	Vulnerability_scanner	10.062
9	Password	9.972
10	Backdoor	9.865
11	Ransomware	9.689
12	XSS	9.543
13	Port_Scanning	8.921
14	Fingerprinting	853
15	MITM	358

Terdapat berbagai macam jenis serangan yang ada dalam dataset tersebut yang dapat dilihat pada

Tabel 2. Dalam dataset tersebut terdapat 61 kolom yang menunjukkan data *record* untuk atribut fitur lalu lintas jaringan IoT, 1 kolom (Attack_label) yang berisi tentang apakah record tersebut termasuk kategori serangan atau bukan, dan 1 kolom (Attack_type) yang menjelaskan suatu data masuk dalam kategori serangan jenis apa. Kolom Attack_label ini dapat dimanfaatkan untuk proses klasifikasi biner (serangan atau bukan) sedangkan kolom Attack_type untuk klasifikasi multikelas.

Tabel 2. Jenis Serangan pada Dataset *Edge-IIoTset Cyber Security Dataset of IoT & IIoT*

Kategori Serangan	**Jenis Serangan**	**Deskripsi**
Dos/ DDos	TCP SYN Flood	Membuat server *IoT edge* tidak tersedia untuk permintaan yang sah
	UDP flood	Membanjiri pemrosesan dan respon perangkat Iot
	HTTP flood	Mengeksploitasi HTTP GET atau POST yang tampaknya sah untuk menyerang aplikasi Iot
	ICMP Flood	Server Iot edge menjadi tidak dapat diakses oleh lalu lintas normal dengan membanjiri mereka dengan paket permintaan
Information gathering	Port Scanning	Mencari pintu terbuka atau titik lemah di Jaringan Iot edge
	OS Fingerprinting	Menganalisis paket data Iot untuk menemukan kelemahan perangkat Iot serta server edge
	Vulnerability scanning attack	Mengidentifikasi kerentanan keamanan jaringan Iot
Man in the Middle Attacks	DNS Spooling attack	Penyadapan komunikasi antara perangkat Iot dan server DNS
	ARP Spoofing attack	Menghubungkan alamat MAC penyerang dengan alamat IP perangkat IoT atau server
Injection Attacks	Cross-site Scripting (XSS) attack	Mengirimkan skrip berbahaya kepada pengguna yang tidak menaruh curiga, yang dapat mengakses informasi sensitif, token sesi, cookie, dll.
	SQL Injection	Mengolah data sensitif dari database IoT dengan injeksi kueri SQL
	Uploading attack	Mengunggah file yang berisi perintah dan data kontrol malware
Malware Attacks	Backdoor attack	Memasang backdoor untuk mengendalikan komponen jaringan IoT
	Password cracking attack	Mengidentifikasi kata sandi yang tidak diketahui atau dilupakan ke perangkat IoT untuk mendapatkan akses tidak sah ke sumber daya IoT

Preprocessing

Dalam tahapan ini dilakukan pengolahan data sehingga data siap untuk digunakan sebagai masukan model *machine learning* yang akan digunakan. Tahapan pertama untuk teknik preprocessing ini adalah membuang fitur yang tidak relevan dalam penentuan jenis serangan jaringan IoT, seperti *IP address, ports, timestamp dan payload information*. Teknik *preprocessing* selanjutnya adalah merubah fitur yang berupa kategori menjadi representasi biner dengan menggunakan *one-hot encoding* agar fitur tersebut dapat diolah oleh model. Setelah itu dilakukan penskalaan fitur dengan memanfaatkan *Standarscaler* dari *sklearn.preprocessing* untuk menstandarisasi fitur sehingga hasil dari proses tersebut data akan memiliki nilai rata-rata 0 dan simpangan baku bernilai 1. Adapun persamaan dari proses tersebut adalah sebagaimana pada Persamaan 1 [7].

$$x_{i.scaled} = \frac{x_i - \bar{x}}{s} \quad (1)$$

$x_{i.scaled} = Data\ ke-i\ yang\ diskalakan$
$x_i = Data\ asli\ ke-i$
$\bar{x} = Rata-rata\ data$
$s = Simpangan\ baku\ data$

Algoritma K-NN

Algoritma ini termasuk *supervised learning*. Algoritma ini digunakan untuk mengklasifikasikan data berdasarkan contoh data training dengan tetangga terdekat di wilayah tertentu. Pada suatu data baru, K tetangga terdekat akan dihitung dan mayoritas data tetangga akan menentukan klasifikasi untuk data baru tersebut. Meskipun pengklasifikasi ini sederhana, nilai K memiliki peran penting dalam mengklasifikasikan data yang tidak berlabel [8]. Kelas yang paling banyak muncul dalam algoritma ini akan menjadi kelas hasil klasifikasi[9]. Ada beberapa jenis teknik yang digunakan dalam menentukan jarak antara data baru dengan data training, seperti Minkowski, Euclidean, Manhattan, Cosine dan lain sebagainya [10].

Algoritma Decision Tree

Decision Tree (DT) merupakan sebuah algoritma untuk membuat keputusan berdasarkan nilai dari beberapa parameter input yang disebut fitur. Hal ini dapat direpresentasikan dengan struktur pohon sehingga disebut sebagai pohon keputusan (*Decision Tree*) [11]. Sebuah pohon (*Tree*) memiliki struktur data yang terdiri dari simpul (*node*) dan rusuk (*edge*). Ada tiga jenis simpul pada pohon: simpul akar (*root/node*), simpul percabangan/internal (*branch/internal node*), dan simpul daun (*leaf node*).[12].

DT membagi dataset menjadi subset berdasarkan pengukuran *impurity* (ketidakmurnian) data, hal ini

bisa dilakukan menggunakan beberapa teknik seperti entropi dan gini [13]. Secara umum, DT terdiri dari node induk dan node anak, dima node induk memecah data menjadi node anak yang lebih kecil dan lebih kecil lagi (*subset*) menggunakan variabel tertentu yang dipilih dengan kriteria pemisahan. Algoritma DT melakukan iterasi pada pemisahan terbaik dari data training sampai kriteria penghentian tercapai.

Biaya komputasi dari DT relatif rendah, dan aturan keputusan berbasis pohon untuk dataset yang besar dapat dihasilkan dengan relatif cepat[13].

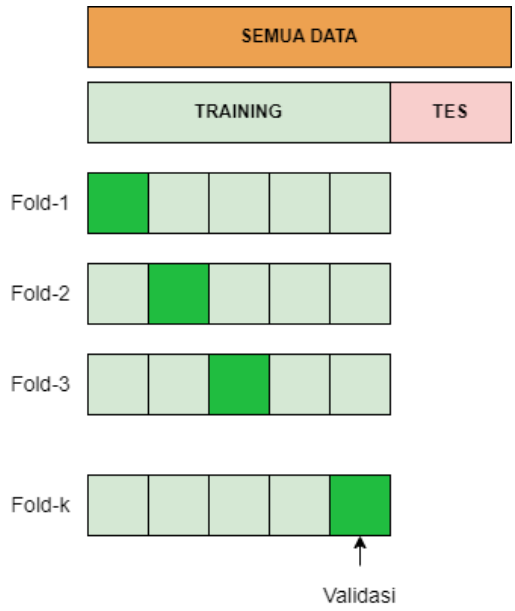
Pembagian Data

Pembagian dataset untuk penelitian ini digunakan perbandingan 80 : 20. Dimana 80% dataset sebagai data training dan validasi sedangkan 20% dataset sebagai data tes. Data training dan validasi secara otomatis terbagi oleh teknik validasi silang k-fold. Data tersebut akan digunakan dalam proses validasi silang dengan penyesuaian parameter terbaik untuk mencari parameter terbaik model *machine learning* yang digunakan. Kemudian model akan diuji dengan data tes untuk dievaluasi performanya.

Validasi Silang Dan Penyesuaian Parameter

Pada setiap percobaan akan dilakukan validasi silang dan penyesuaian parameter terbaik. Teknik validasi model yang digunakan adalah *k-fold cross validation* dimana dalam setiap pengujian data akan terbagi menjadi bagian sebanyak *k*. Sehingga semua data pernah menjadi data training dan data validasi. Proses validasi silang untuk k-fold bisa dilihat pada Gambar 2. Untuk penelitian ini digunakan nilai k = 5. Pada setiap pengujian validasi silang digunakan parameter berbeda dan akan diambil parameter terbaik untuk membuat model terakhir.

Menyempurnakan parameter model akan memaksimalkan performa model. Dalam konteks pembelajaran mesin, nilai parameter yang ditetapkan sebelum proses pembelajaran disebut *hyperparameter*. Di sisi lain, nilai parameter model diperoleh dari data *training* [14]. Parameter model mengacu pada bobot dan koefisien pada suatu algoritma. Setiap algoritma memiliki sekumpulan parameter yang dapat ditentukan, misalnya untuk K-NN kita bisa mengatur nilai dari K, jumlah tetangga terdekat. Dalam melakukan penyesuian parameter terbaik penelitian ini memanfaatkan fungsi *RandomizedSearchCV* dari *sklearn.model_selection*. Metode *RandomSearchCV* adalah metode pencarian model terbaik dengan menspesifikasikan nilai parameter dari algoritma pembelajaran secara manual. *RandomSearchCV* bekerja dengan cara mengkombinasikan parameter secara acak yang akan dipilih dan digunakan untuk melatih model [15].



Gambar 2. Proses Validasi Silang K-fold

Evaluasi Model

Mengevaluasi performa model dari pembelajaran *supervised* maupun *unsupervised* adalah aspek paling mendasar dari pembelajaran mesin. Hal ini sangat penting untuk memilih model yang paling tepat dari sekumpulan model yang diberikan. Ketika metode evaluasi yang tidak tepat diterapkan, model yang berkinerja baik dapat diukur secara tidak tepat berdasarkan informasi yang tersedia [16].

Performa dari model akan diukur dengan nilai akurasi, presisi, recall, dan f1 score. Sedangkan visualisasi banyak data tes yang diprediksi secara benar ataupun salah akan ditampilkan dalam bentuk *Confusion Matrix*.

3. HASIL DAN PEMBAHASAN

Tahap *Preprocessing*

Dari 61 atribut dalam dataset dilakukan pemilihan atribut fitur yang berguna dalam proses klasifikasi. Didapatkan ada 15 kolom yang tidak relevan, sehingga tersisa 46 kolom atribut yang akan digunakan. Dari 46 kolom tersebut terdapat beberapa kolom yang termasuk dalam fitur kategori, sehingga perlu dilakukan proses *one-hot encoding*. Proses mengubah fitur kategori menjadi representasi biner dinyatakan dalam Tabel 3.

Tabel 3. Mengubah Fitur Kategori dengan *One-hot Encoding*

Asli	One-hot Encoding				
http.request	**null**	**GET**	**POST**	**OPTIONS**	**TRACE**
null	1	0	0	0	0
GET	0	1	0	0	0
POST	0	0	1	0	0
OPTIONS	0	0	0	1	0
TRACE	0	0	0	0	1

Selanjutnya adalah melakukan penskalaan fitur dikarenakan beberapa fitur memiliki nilai yang cukup besar dan jangkauan yang bervariasi.

Pengujian Klasifikasi Biner

Pengujian dalam klasifikasi biner ini bertujuan untuk mengetahui bagaimana performa dari masing-masing algoritma ketika digunakan untuk mendeteksi apakah ada serangan atau tidak dalam jaringan IoT. Dimana data yang normal diberi label 0 dan data serangan diberi label 1.

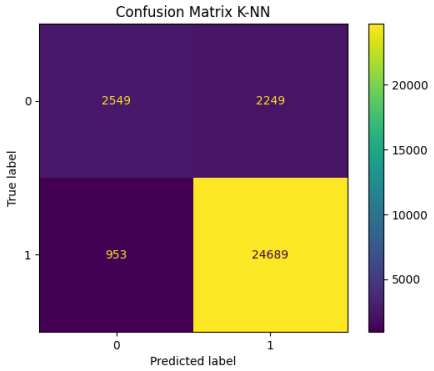
a. Algoritma KNN

Pada percobaan dengan menggunakan algoritma KNN ini untuk melakukan penyesuain parameter terbaik dilakukan dengan pencarian kombinasi terbaik menggunakan *RandomizedSearchCV* dari beberapa parameter yang dapat diubah. Parameter tersebut dapat dilihat pada Tabel 4.

Tabel 4. Penyesuain Paremeter untuk KNN

Parameter	Nilai
n_neighbors	[1, 3, 5, 7]
weights	[uniform, distance]
metric	[minkowski, euclidean, manhattan]

Dalam proses pencarian parameter terbaik didapatkan kombinasi yang menghasilkan nilai performa paling bagus dengan penyesuain parameter n_neighbors = 7, weight = uniform, dan metric = manhattan. Dan didapatkan hasil performa seperti pada Tabel 5. Hasil prediksi data tes ditampilkan dalam bentuk *confusion matrix* pada Gambar 3.



Gambar 3. *Confusion Matrix KNN* klasifikasi biner

Tabel 5. Hasil Pemodelan KNN klasifikasi Biner

Kelas	Presisi	Recall	F1-score	Acc.
Normal	0.77	0.52	0.62	0.90
Attack	0.92	0.97	0.94	
macro avg.	0.84	0.74	0.78	

b. Algoritma *Decision Tree*

Beberapa kombinasi parameter yang digunakan untuk penyesuaian parameter dalam pengujian klasifikasi biner pada algoritma *Decision Tree* dapat dilihat pada Tabel 6.

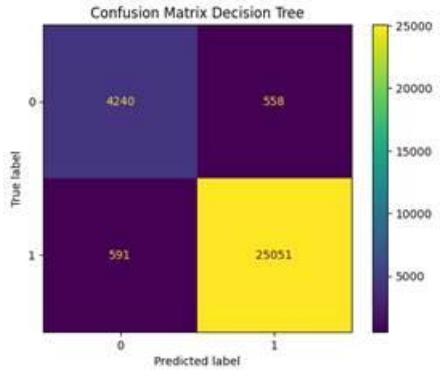
Tabel 6. Penyesuain Paremeter untuk Decision Tree

Parameter	Nilai
Criterion	[gini, entropy]
Min samples leaf	[1, 2, 4]
Max depth	[none, 5, 10, 15]
Min samples split	[2, 5, 10]

Nilai parameter terbaik pada percobaan ini adalah criterion = entropy, min samples leaf = 4, max depth = 10, dan min sample split =2. Dengan nilai performa yang dihasilkan seperti yang ada pada Tabel 7. Hasil prediksi data tes yang ditampilkan dalam bentuk *confusion matrix* pada Gambar 4.

Tabel 7. Hasil Pemodelan Decision Tree klasifikasi Biner

Kelas	Presisi	Recall	F1-score	Acc.
Normal	1.0	0.85	0.92	0.98
Attack	0.97	1.0	0.99	
macro avg.	0.99	0.92	0.95	

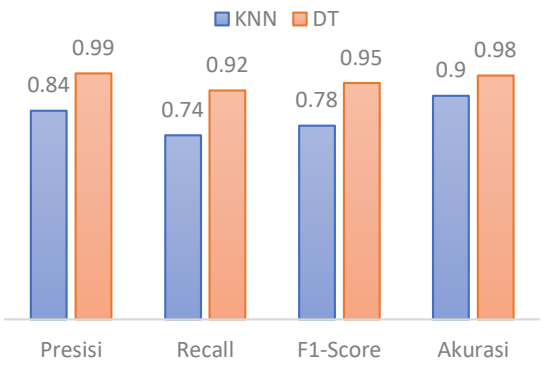


Gambar 4. *Confusion Matrix* DT Klasifikasi Biner

c. Perbandingan Model

Setelah hasil dari masing-masing algoritma didapatkan, maka dilakukan perbandingan performa model antara algoritma *K-NN* dan *Decision Tree* dalam pengklasifikasian biner.

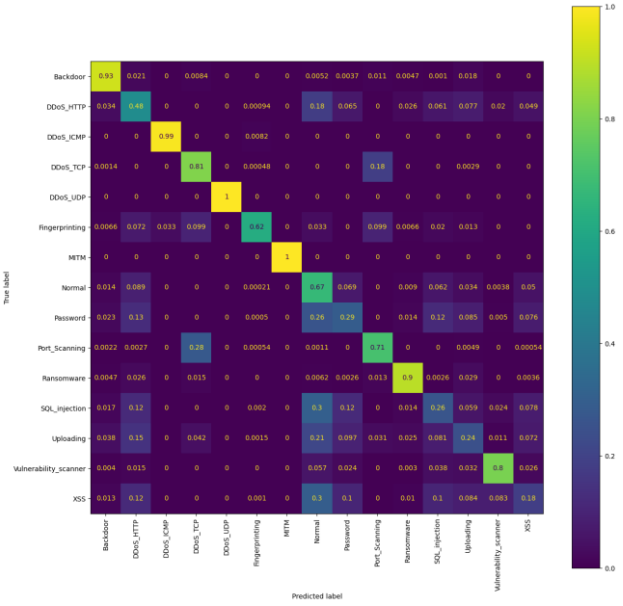
Dari Gambar 5, terlihat grafik perbandingan antara kedua algoritma. Decision Tree memiliki nilai performa yang lebih tinggi jika dibandingkan dengan KNN, baik dalam nilai presisi, recall, F1-score, maupun akurasi. Selisih nilai performa tersebut secara berurutan adalah 0.15, 0.18, 0.17 dan 0.08. Dari masing-masing *confusion matrix* untuk KNN terdapat 2549 data dengan label normal yang benar diprediksi dan 24689 data dengan label serangan. Sedangkan untuk DT terdapat 4240 data dengan label normal yang benar diprediksi dan 25051 data dengan label serangan.



Gambar 5. Grafik Perbandingan Klasifikasi Biner

Pengujian Klasifikasi Multikelas

Pengujian ini bertujuan untuk mengetahui kemampuan algoritma KNN dan Decision Tree dalam mengklasifikasikan jenis serangan yang terjadi dengan jumlah label kelas sebanyak 15 (14 tipe serangan dan 1 bukan serangan).



Gambar 6. *Confusion Matrix* KNN Klasifikasi Multikelas

a. Algoritma K-NN

Pencarian parameter terbaik yang digunakan dalam percobaan ini sama dengan paramater yang ada di pengujian klasifikasi biner. Hasil parameter terbaik yang didapatkan adalah n_neighbors = 5, weight = distance, dan metric = minkowski. Dengan parameter tersebut performa dari algoritma ini seperti pada Tabel 8. Dan memiliki hasil prediksi pada data tes yang diperlihatkan dalam confusion matrix pada Gambar 7.

Tabel 8. Hasil Pemodelan KNN klasifikasi multi kelas

Kelas	Presisi	Recall	F1-score	Acc.
Backdoor	0.88	0.92	0.9	
DDoS_HTTP	0.49	0.43	0.46	
DDoS_ICMP	1	0.99	0.99	
DDoS_TCP	0.75	0.78	0.76	
DDoS_UDP	1	1	1	
Fingerprinting	0.64	0.61	0.62	
MITM	1	1	1	
Normal	0.59	0.63	0.61	0.66
Password	0.33	0.3	0.32	
Port_Scanning	0.71	0.74	0.73	
Ransomware	0.86	0.9	0.88	
SQL_injection	0.32	0.31	0.32	
Uploading	0.32	0.31	0.32	
Vulnerability scanner	0.81	0.81	0.81	
XSS	0.33	0.32	0.32	
macro avg.	0.67	0.67	0.67	

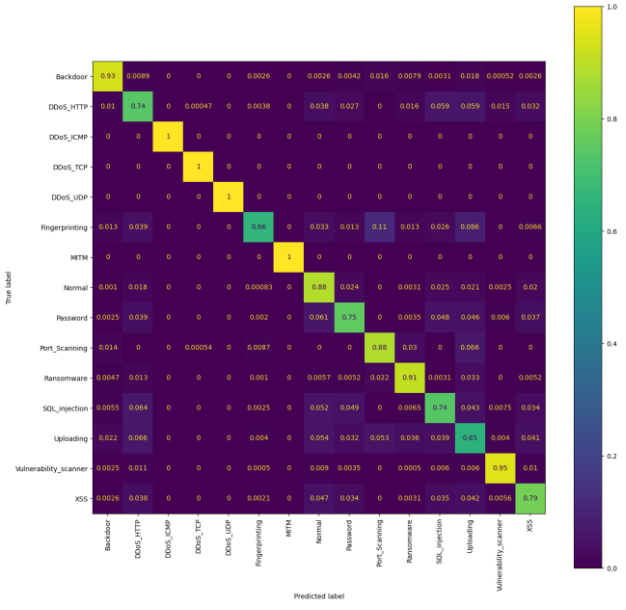
Jika dibandingkan dengan hasil performa dari klasifikasi biner, maka KNN mengalami penurunan performa. Performa terburuk ditemukan dalam mendeteksi tipe serangan password, SQL injection dan Uploading. Sedangkan memiliki performa terbaik dalam menentukan serangan MITM dan DdoS UDP.

b. Algoritma Decision Tree

Pencarian parameter terbaik yang digunakan dalam percobaan ini sama dengan paramater yang ada di pengujian klasifikasi biner. Hasil parameter terbaik yang didapatkan adalah criterion = entropy, min samples leaf = 1, max depth = 15, dan min sample split =10. Dengan parameter tersebut performa dari algoritma ini seperti pada Tabel 9. Dan memiliki hasil prediksi pada data tes yang diperlihatkan dalam confusion matrix pada Gambar 5.

Tabel 9. Hasil Pemodelan DT klasifikasi multikelas

Kelas	Presisi	Recall	F1-score	Acc.
Backdoor	0.99	0.93	0.96	
DDoS_HTTP	0.58	0.92	0.71	
DDoS_ICMP	1	1	1	
DDoS_TCP	1	1	1	
DDoS_UDP	1	1	1	
Fingerprinting	0.92	0.64	0.76	
MITM	1	1	1	
Normal	0.76	0.95	0.85	0.89
Password	0.99	0.71	0.83	
Port_Scanning	0.87	0.99	0.93	
Ransomware	0.99	0.89	0.94	
SQL_injection	0.96	0.7	0.81	
Uploading	0.92	0.62	0.74	
Vulnerability scanner	0.98	0.95	0.97	
XSS	0.99	0.76	0.86	
macro avg.	0.93	0.87	0.89	

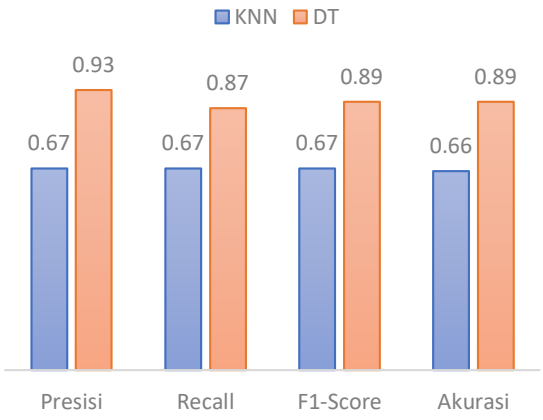


Gambar 7. *Confusion Matrix* DT Klasifikasi Multikelas

Meskipun dalam klasifikasi multikelas DT mengalami penurunan performa, tetapi tergolong masih cukup bagus. Nilai performa terbaik didapatkan Ketika model melakukan klasifikasi DDoS_ICMP, DDoS_TCP, DDoS_UDP, dan MITM.

c. Perbandingan Model

Perbandingan model antara KNN dan DT dalam melakukan klasifikasi multikelas dapat dilihat pada grafik di Gambar 8.



Gambar 8. Grafik Perbandingan Klasifikasi Multikelas

Sama dengan klasifikasi biner, dalam klasifikasi multikelas DT lebih unggul dari pada KNN di semua nilai performa, baik di presisi, recall, F1-score dan akurasi. Selisih nilai performa secara berurutan adalah 0.26, 0.20, 0.22, dan 0.23. Nilai selisih ini lebih besar dibandingkan dengan nilai selisih yang ada pada klasifikasi biner. Hal ini dikarenakan KNN mengalami penurunan yang signifikan ketika melakukan klasifikasi multikelas.

4. KESIMPULAN

Dari beberapa percobaan yang dilakukan, KNN masih cukup bagus untuk klasifikasi biner sedangkan untuk klasifikasi multikelas KNN mengalami penurunan nilai performa yang cukup besar. Algoritma DT untuk kedua kasus klasifikasi (biner dan multikelas) memiliki performa yang bagus. Hasil percobaan klasifikasi biner untuk kedua algoritma ini memiliki nilai performa yang lebih baik jika dibandingkan dengan klasifikasi multikelas. Hal ini dikarenakan perbedaan jumlah kelas untuk klasifikasi biner dan klasifikasi multikelas terdapat perbedaan yang jauh. DT memiliki performa yang lebih baik jika dibandingkan algoritma KNN dalam kasus klasifikasi serangan pada jaringan IoT, baik dalam klasifikasi biner maupun klasifikasi multikelas. Khususnya jika DT digunakan dalam klasifikasi multikelas maka memiliki perbedaan nilai performa yang cukup signifikan jika dibandingkan dengan KNN.

DAFTAR PUSTAKA

[1] A. Sandriana, Rianto, and F. Maulana, "Klasifikasi serangan Malware terhadap Lalu Lintas Jaringan Internet of Things menggunakan Algoritma K-Nearest Neighbour (K-NN)," *E-JOINT (Electronica and Electrical Journal Of Innovation Technology)*, vol. 3, no. 1, pp. 12–22, Jun. 2022, doi: 10.35970/e-joint.v3i1.1559.

[2] R. A. Khairulah, R. Herdianto, and M. A. Setiawan, "Klasifikasi Serangan Pada Jaringan Internet of Thing (IoT): Tinjauan Literatur Komparatif," *Jurnal Inovasi Teknik dan Edukasi Teknologi*, vol. 3, no. 1, pp. 47–53, doi: 10.17977/um068v3i12023p47-53.

[3] W. Najib, S. Sulistyo, and Widyawan, "Tinjauan Ancaman dan Solusi Keamanan pada Teknologi Internet of Things," *Jurnal Nasional Teknik Elektro dan Teknologi Informasi*, vol. 9, no. 4, pp. 375–384, Dec. 2020, doi: 10.22146/jnteti.v9i4.539.

[4] F. Antony and R. Gustriansyah, "Deteksi Serangan Denial of Service pada Internet of Things Menggunakan Finite-State Automata," *MATRIK : Jurnal Manajemen, Teknik Informatika dan Rekayasa Komputer*, vol. 21, no. 1, pp. 43–52, Nov. 2021, doi: 10.30812/matrik.v21i1.1078.

[5] R. Rizal, N. Widiyasono, and S. Yuliyanti, "Kecerdasan Buatan untuk Klasifikasi Serangan Siber pada Internet of Things Network Traffic." JUMANJI (*Jurnal Masyarakat Informatika Unjani*), vol. 7, no. 2, pp. 61-71, nov. 2023, doi: 10.26874/jumanji.v7i2.325.

[6] N. Pandey and P. K. Mishra, "Detection of DDoS attack in IoT traffic using ensemble

machine learning techniques," *Networks and Heterogeneous Media*, vol. 18, no. 4, pp. 1393–1409, 2023, doi: 10.3934/nhm.2023061.

[7] L. B. V. de Amorim, G. D. C. Cavalcanti, and R. M. O. Cruz, "The choice of scaling technique matters for classification performance," *Appl Soft Comput*, vol. 133, p. 109924, Jan. 2023, doi: 10.1016/j.asoc.2022.109924.

[8] K. Taunk, S. De, S. Verma, and A. Swetapadma, "A Brief Review of Nearest Neighbor Algorithm for Learning and Classification," in *2019 International Conference on Intelligent Computing and Control Systems (ICCS)*, IEEE, May 2019, pp. 1255–1260. doi: 10.1109/ICCS45141.2019.9065747.

[9] A. Muzakir, A. Desiani, and A. Amran, "Klasifikasi Penyakit Kanker Prostat Menggunakan Algoritma Naïve Bayes dan K-Nearest Neighbor," *Komputika : Jurnal Sistem Komputer*, vol. 12, no. 1, pp. 73–79, May 2023, doi: 10.34010/komputika.v12i1.9629.

[10] S. Nayak, M. Bhat, N. V Subba Reddy, and B. Ashwath Rao, "Study of Distance Metrics on K-Nearest Neighbors Algorithm for Star Categorization," *J Phys Conf Ser*, vol. 2161, no. 1, p. 012004, Jan. 2022, doi: 10.1088/1742-6596/2161/1/012004.

[11] T. Xiao, O. N. Timo, F. Avellaneda, Y. Malik, and S. Bruda, "An Approach to Evaluating Learning Algorithms for Decision Trees," Oct. 2020, [Online]. Available: http://arxiv.org/abs/2010.13665

[12] A. H. Nasrullah, "Implementasi Algoritma Decision Tree untuk Klasifikasi Produk Laris," *Jurnal Ilmiah Ilmu Komputer*, vol. 7, no. 2, pp. 45–51, Sep. 2021, doi: 10.35329/jiik.v7i2.203.

[13] S. Lee, C. Lee, K. G. Mun, and D. Kim, "Decision Tree Algorithm Considering Distances Between Classes," *IEEE Access*, vol. 10, pp. 69750–69756, 2022, doi: 10.1109/ACCESS.2022.3187172.

[14] E. Elgeldawi, A. Sayed, A. R. Galal, and A. M. Zaki, "Hyperparameter Tuning for Machine Learning Algorithms Used for Arabic Sentiment Analysis," *Informatics*, vol. 8, no. 4, p. 79, Nov. 2021, doi: 10.3390/informatics8040079.

[15] T. A. E. Putri, T. Widiharih, and R. Santoso, "Penerapan Tuning Hyperparameter Randomsearchcv Pada Adaptive Boosting untuk Prediksi Kelangsungan Hidup Pasien Gagal Jantung," *Jurnal Gaussian*, vol. 11, no. 3, pp. 397–406, Jan. 2023, doi: 10.14710/j.gauss.11.3.397-406.

[16] Gs. Suneetha, "Classification Evaluation Metrics in Machine learning," JETIR: *Journal of Emerging Technologies and Innovative Research*, vol. 9, Issue 10, pp. c684-c687, Oct 2022, doi: 10.1729/Journal.31871.