Published online on: https://journal.uii.ac.id/jurnalsnati/
DOI: 10.20885/snati.v4.i2.38599

JURNAL SNATI
Sains, Nalar, dan Aplikasi Teknologi Informasi

Jurnal Sains, Nalar, dan Aplikasi Teknologi Informasi
Vol. 4 No. 2 (2025) 129 - 137 ISSN Media Electronic: 2807-5935

Penerapan Algoritma *K-Nearest Neighbors* untuk Deteksi Serangan *Network Flood* Berbasis *Supervised Learning*

Application of the K-Nearest Neighbors Algorithm for Supervised Learning-Based Network Flood Attack Detection

Roni Habibi[1], Naufal Dekha Widana[2]*
[1,2]Program Studi D4 Teknik Informatika, Sekolah Vokasi, Universitas Logistik dan Bisnis Internasional, Bandung, Indonesia
[1]roni.habibi@ulbi.ac.id, [2*]dekhawidananaufal@gmail.com

Abstract

Anomaly detection caused by flood attacks is a major challenge in modern network security management. This study proposes the application of the K-Nearest Neighbors (KNN) algorithm within a supervised learning framework to build a Network Flood Detection (NFD) model evaluated using comprehensive performance metrics, namely accuracy, precision, and recall. The model is developed using network features such as inbound bandwidth, outbound bandwidth, ping, and the classification of traffic into flood and normal categories. Data were collected from both real-time and historical network logs from an institutional environment and were processed through normalization, feature reduction, and noise removal. The evaluation results indicate that the model achieved up to 92.42% accuracy with balanced F1-scores across both classes. Additionally, a ROC curve with an AUC of 0.99 demonstrates a high discriminatory ability of the model in separating flood and normal traffic. These findings suggest that KNN, despite its simplicity, can be effectively applied in flood attack detection systems when supported by representative data and robust evaluation procedures.

Keywords: Network Security, Supervised Learning; KNN; Network Flood; Anomaly Detection

Abstrak

Deteksi anomali akibat serangan *flood* merupakan tantangan utama dalam pengelolaan keamanan jaringan modern. Penelitian ini mengusulkan penerapan algoritma *K-Nearest Neighbors* (KNN) dalam kerangka *supervised learning* untuk membangun model *Network Flood Detection* (NFD) yang dievaluasi menggunakan metrik performa yang lebih komprehensif, yaitu akurasi, presisi, dan *recall*. Model dikembangkan berdasarkan fitur jaringan seperti *bandwidth* masuk, *bandwidth* keluar, ping, serta distribusi trafik *flood* dan normal. Data diperoleh dari laporan jaringan instansi secara *real-time* dan historis, yang kemudian diproses melalui tahapan normalisasi, pengurangan fitur, dan penghapusan noise. Hasil evaluasi menunjukkan bahwa model mampu mencapai akurasi hingga 92,42% dengan skor F1 yang seimbang antar kelas. Selain itu, kurva ROC dengan AUC sebesar 0,99 menunjukkan bahwa model memiliki kemampuan diskriminasi yang tinggi dalam membedakan trafik *flood* dan normal. Temuan ini menunjukkan bahwa KNN, meskipun sederhana, dapat digunakan secara efektif dalam sistem deteksi serangan *flood* jika didukung oleh data yang representatif dan proses evaluasi yang tepat.

Kata kunci: Keamanan Jaringan; *Supervised Learning*; KNN; *Network Flood*; Deteksi Anomali

1. Pendahuluan

Jaringan komputer mendukung banyak aktivitas digital di bidang bisnis, pemerintahan, dan pendidikan [1], [2]. Kestabilan dan ketersediaan jaringan sangat penting untuk menjaga kontinuitas layanan dan produktivitas [3]. Namun, keadaan ini menimbulkan kerentanan baru terhadap ancaman siber, salah satunya adalah serangan *flood* yang memiliki kemampuan untuk melemahkan sistem dengan menggunakan lalu lintas data yang berlebihan secara tiba-tiba dan besar.

Serangan *Denial-of-service* (DoS) adalah ketika pelaku membanjiri sistem dengan trafik atau permintaan palsu sehingga mengganggu layanan yang legal [4]. Mengingat *volume* dan kecepatan data yang

Received: 22-01-2025 | Revised: 02-07-2025 | Accepted: 03-07-2025 | Published: 10-07-2025

tinggi, pola serangan seperti ini sering kali tidak dapat dideteksi secara manual [5]. Oleh karena itu, *machine learning* mesin mulai digunakan secara luas untuk membantu proses deteksi anomali secara otomatis dan efisien [6].

K-Nearest Neighbors (KNN) adalah salah satu algoritma pembelajaran mesin yang paling umum digunakan untuk klasifikasi data. Algoritma ini bekerja dengan menghitung jarak kemiripan antar data dan mengklasifikasikan data berdasarkan mayoritas pola persebaran kelas data mayoritas terdekat [7]. Karena implementasinya yang sederhana dan tidak memerlukan proses pelatihan model yang kompleks, penggunaannya dalam deteksi *flood attack* telah banyak dibahas. Sebagai contoh, Wang dkk [8] telah membahas pengaruh dari perbedaan metode *supervised*, *unsupervised*, *semi-supervised*, dan *reinforcement learning* pada suatu algoritma yang sama. Meskipun demikian, penelitian sebelumnya masih memiliki keterbatasan, terutama dalam hal kelengkapan evaluasi kinerja model.

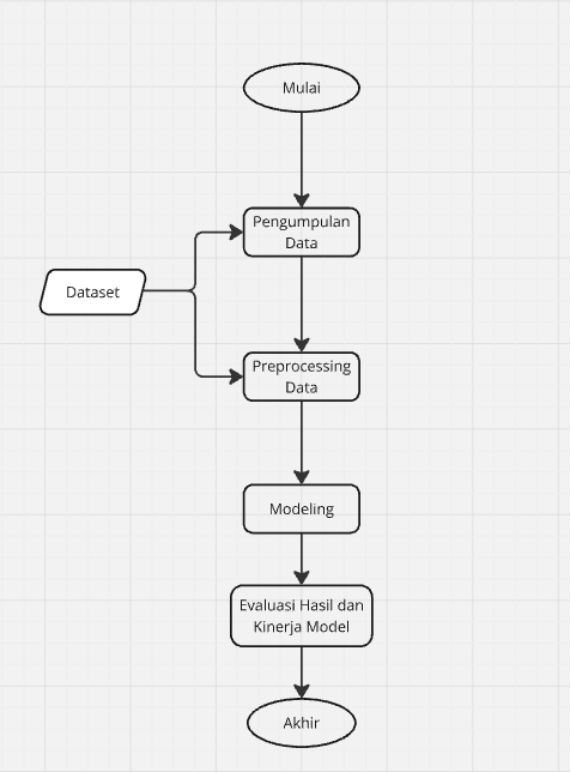
Beberapa penelitian terdahulu seperti penelitian dari Altayef dkk [9] yang membahas perbandingan akurasi dari 3 algoritma yaitu SVM, KNN dan DT, dan Cevik dan Akleylek [10] tentang penggunaan metode *Machine Learning* dan *Deep Learning* terhadap serangan anomali terhadap sinyal penyiaran radio di industri penerbangan, dan hanya mengukur keberhasilan model dengan menggunakan metrik akurasi, tetapi tidak memasukkan metrik lain seperti ketepatan dan *recall* yang penting untuk mendeteksi anomali. Kekurangan metrik ini dapat menyebabkan interpretasi keliru, terutama ketika data tidak seimbang antara kelas normal dan anomali, menurut Owusu-Adjei [11] yang membahas pengaruh ketidak seimbangan data pada algoritma yang dipakai. Apabila model seringkali tidak dapat menemukan serangan yang sebenarnya terjadi, akurasi tinggi belum selalu menunjukkan kemampuan deteksi yang baik.

Tujuan dari penelitian ini adalah untuk membuat ide tentang model NFD berbasis KNN yang dapat memperbaiki metode sebelumnya dengan melakukan evaluasi performa yang lebih menyeluruh. Model ini akan diuji berdasarkan tiga metrik utama, yaitu akurasi, ketepatan, dan *recall*. Dengan demikian, penelitian ini berharap dapat memberikan gambaran yang lebih realistis tentang efektivitas deteksi anomali. Selain itu, untuk menjaga kesederhanaan namun tetap relevan dengan fitur serangan *flood*, fitur jaringan yang umum seperti *bandwidth* masuk, *bandwidth* keluar, dan *ping* dipilih.

Penelitian ini diharapkan dapat memberikan kontribusi awal dalam pengembangan sistem deteksi anomali jaringan yang ringan, praktis, dan siap dikembangkan ke tahap implementasi nyata. Selain itu, gagasan ini dapat digunakan sebagai referensi atau landasan bagi penelitian lanjutan yang ingin mengevaluasi efektivitas model deteksi berbasis KNN secara lebih mendalam dalam berbagai skenario jaringan.

2. Metode Penelitian

Metodologi penelitian ini akan berfokus pada pendekatan utama, yaitu analisis data historis, untuk memberikan pemahaman yang mendalam tentang pola lalu lintas jaringan. Proses ini dirancang secara terintegrasi dan menggunakan hasil dari analisis data historis untuk membuat model awal, yang dibantu oleh data pemantauan *real-time* untuk memperbarui dan menyesuaikan model sesuai dengan kondisi jaringan saat ini.



Gambar 1. Diagram alur metodologi penelitian

Metode ini bertujuan untuk mengatasi keterbatasan masing-masing metode seperti pada gambar Gambar 1 yang merupakan urutan dari awal sampai akhir terkait pembuatan konsep model NFD ini. Misalnya, analisis historis tidak dapat mengidentifikasi perubahan pola terbaru dan analisis *real-time* menghadapi kesulitan saat menangani *volume* data yang besar. Metode ini diharapkan dapat membuat deteksi serangan *flood* lebih akurat dan responsif. Metode ini juga akan memastikan bahwa model pembelajaran mesin tetap relevan terhadap ancaman yang berkembang.

2.1. Pengumpulan Data

Data yang digunakan dalam penelitian ini diperoleh secara *real-time* dari laporan harian instansi Badan Usaha Milik Negara (BUMN). Laporan harian ini berisi informasi tentang lalu lintas jaringan, yang

mencakup aktivitas sehari-hari serta kemungkinan anomali yang terjadi dalam sistem jaringan instansi tersebut. Tim teknis dari instansi terkait bekerja sama untuk mengumpulkan data ini secara langsung, memastikan bahwa data itu akurat dan sesuai dengan ketentuan [12].

Instansi tersebut menggunakan sistem monitoring jaringan yang secara otomatis mencatat *volume bandwidth* masuk dan keluar, kecepatan paket per detik, dan protokol jaringan yang digunakan, jumlah rata-rata paket, jumlah penggunaan *port* dan jumlah dari penggunaan CPU. Lalu *log* jaringan harian mencatat data ini, yang kemudian disimpan dalam basis data internal untuk analisis tambahan [13].

Penelitian ini juga menggunakan data historis yang telah dikumpulkan sebelumnya oleh sistem monitoring yang sama. Data historis ini mencakup aktivitas jaringan dalam jangka waktu yang lebih lama dan sangat berguna untuk memahami pola-pola trafik jaringan secara keseluruhan. Tujuan integrasi data historis ini adalah untuk memperbaiki proses pelatihan model dan memberikan konteks terhadap pola lalu lintas yang biasa mengalami fluktuasi atau perubahan [14].

Pendekatan segmentasi waktu digunakan untuk mengintegrasikan data historis dan aktual. Data historis digunakan sebagai data latihan (*training*) untuk membentuk pola klasifikasi dasar dalam model KNN, dan data aktual digunakan sebagai data uji (*testing*) untuk mengukur performa model terhadap trafik yang sedang berlangsung. Metode ini tidak hanya memastikan kelengkapan konteks proses klasifikasi, tetapi juga menguji kemampuan model untuk menyesuaikan dengan kondisi jaringan terbaru. Proses validasi dilakukan untuk memastikan bahwa data historis dan *real-time* memiliki struktur dan cakupan fitur yang seragam. Untuk memastikan bahwa tidak ada *noise* atau inkonsistensi yang signifikan, tim IT instansi secara berkala melakukan pengujian kualitas data. Ini mendukung upaya untuk membuat model NFD yang bergantung pada data yang bersih, akurat, dan representatif dari situasi nyata [15] .

2.2. Preprocessing Data

Untuk memastikan kualitas dan konsistensi data sebelum digunakan dalam proses pelatihan dan pengujian model, dilakukan beberapa tahap pra-pemrosesan. Tahap ini sangat penting karena data dari sistem pemantauan historis dan *real-time* mungkin mengandung duplikasi, nilai kosong (nilai yang tidak ada), atau anomali yang tidak terkait dengan serangan jaringan (seperti *spike* alami yang disebabkan oleh *backup internal*). Oleh karena itu, pembersihan data juga dikenal sebagai pembersihan data dilakukan terlebih dahulu untuk menghilangkan entri duplikat dan menangani nilai kosong dengan menggunakan teknik sederhana seperti interpolasi atau penghapusan baris yang tidak relevan [16].

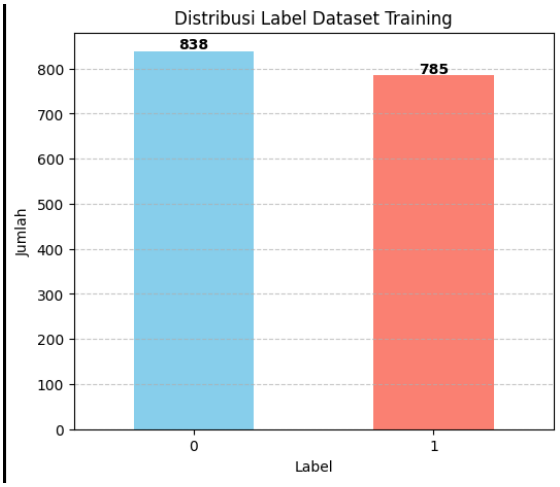
	timestamp	bandwidth_in (kb)	bandwidth_out (kb)	jenis_probo	ping	label
0	2024-12-01 00:00:00	1924	509	ftp	28.23	Normal
1	2024-12-01 00:00:15	2386	1779	ftp	74.33	Normal
2	2024-12-01 00:01:10	1524	9774	http	4.15	Flood
3	2024-12-01 00:00:18	3682	3911	https	60.60	Normal
4	2024-12-01 00:02:24	3357	11831	ftp	70.43	Flood
5	2024-12-01 00:02:15	3711	7459	https	28.54	Normal
6	2024-12-01 00:05:36	206	2715	ftp	42.84	Normal
7	2024-12-01 00:02:06	2647	3627	dns	34.32	Normal
8	2024-12-01 00:00:48	6324	1684	icmp	84.90	Normal
9	2024-12-01 00:05:51	4433	811	ftp	46.48	Normal

Gambar 2. Data training

```
Data columns (total 6 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   timestamp           8115 non-null   object
 1   bandwidth_in (kb)   8115 non-null   int64
 2   bandwidth_out (kb)  8115 non-null   int64
 3   jenis_probo         8115 non-null   object
 4   ping                8115 non-null   float64
 5   label               8115 non-null   object
dtypes: float64(1), int64(2), object(3)
memory usage: 380.5+ KB
```

Gambar 3. Kolom keterangan data

Selanjutnya, data seperti kecepatan paket per detik, *bandwidth* masuk, dan *bandwidth* keluar dinormalisasi. Algoritma *K-Nearest Neighbors* (KNN) sangat bergantung pada perhitungan jarak antar data, sehingga hasil klasifikasi dapat sangat dipengaruhi oleh perbedaan skala antar fitur. Oleh karena itu, untuk memastikan bahwa semua fitur berada dalam rentang nilai yang sama selama proses perhitungan jarak, teknik *Min-Max Scaling* digunakan untuk menyeragamkan skala data.



Gambar 4. Distribusi data training

Dalam konteks pengenalan serangan *flood* nirkabel, ini menjadi keunggulan karena serangan *flood* seringkali memiliki pola numerik yang sama atau berulang dan dapat diidentifikasi melalui pendekatan berbasis kedekatan antar instance. Prinsip kerja KNN sangat sesuai dengan karakteristik data jaringan yang digunakan dalam penelitian ini [17]. Data yang sudah distandarisasi secara skala memungkinkan perhitungan

jarak secara adil antar fitur. Selain itu, fitur seperti kecepatan paket per detik dan *bandwidth* memiliki pola distribusi yang cukup konsisten dalam kondisi normal, tetapi mereka berubah secara signifikan setelah serangan. Dengan sedikit pelatihan, KNN dapat secara langsung menangkap perubahan dalam nilai.

Meskipun distribusi data 51:49 tetapi proses pra-pemrosesan juga melibatkan pembagian dataset menjadi dua bagian utama: data latih dan data uji. Kedua kategori ini biasanya berbagi rasio 80 banding 20 yang mana ini rasio yang umum untuk melakukan *splitting dataset* [18]. Karena sifat KNN yang tidak melakukan proses pembelajaran yang jelas, keberhasilan klasifikasi sangat bergantung pada kualitas data latih, yang telah dikurasi dan dibersihkan secara menyeluruh dalam tahapan sebelumnya.

Data latih digunakan sebagai basis referensi KNN untuk mencari tetangga terdekat dari data uji. Dalam proses pra-pemrosesan ini, data dari dua sumber waktu, baik historis maupun aktual, disiapkan untuk mendukung pembentukan model NFD berbasis KNN. Kombinasi data yang telah dinormalisasi, dilabeli, dan dikodekan secara seragam memungkinkan algoritma KNN bekerja secara optimal dalam mengidentifikasi lalu lintas jaringan yang menyimpang. Dengan metode ini, diharapkan model akan menghasilkan hasil klasifikasi yang lebih presisi, dan juga lebih optimal.

2.3. Pelatihan Model KKN

Dataset yang digunakan dalam penelitian ini terdiri dari 8.116 sampel data jaringan yang dikumpulkan merupakan data historis maupun data real-time, yang mencakup berbagai metrik numerik seperti *bandwidth*, ukuran paket (*packet size*), dan *latency*. Tujuan dari penggunaan *dataset* skala desimal adalah agar model dapat mempelajari pola trafik dengan cara yang mirip dengan kondisi nyata di lingkungan jaringan, sehingga model yang dibangun memiliki kemampuan umum untuk mempelajari pola trafik.

Tabel 1. Distribusi data latih

Label	Jumlah Data	Persentase (%)
Normal	4,139	51%
Flood	3,977	49%
Total	8,116	100%

Distribusi data 51:49 yang mana 51% data Normal dan 49% *Flood*, lalu data dibagi menjadi dua subset: data latihan (*training*) dan data uji (*test*). Skema pembagian random sebesar 80% untuk *training* dan 20% untuk *test* digunakan untuk memastikan bahwa variasi data di masing-masing subset menggambarkan distribusi awalnya. Untuk menghindari bias model yang cenderung memprioritaskan prediksi untuk satu kelas, distribusi label pada masing-masing subset cukup seimbang [19].

Label untuk *dataset* dipilih berdasarkan kondisi trafik jaringan. Label Normal diberikan pada data yang menunjukkan aktivitas jaringan yang stabil tanpa indikasi serangan, tetapi label *Flood* diberikan pada data yang menunjukkan indikasi serangan, seperti *bandwidth* yang melebihi ambang batas, jumlah paket yang sangat tinggi, atau pola trafik yang tidak normal. *Labeling* ini menggunakan *log* hasil pemantauan jaringan serta batas teknis yang ditetapkan melalui analisis data trafik sebelumnya.

Tabel 2. Scalling label

	Kondisi	Label
Bandwidth	>8000 Mbps	*Flood*
Bandwidth	<8000 Mbps	Normal

Sebelum data digunakan untuk pelatihan model, proses penghilangan suara dilakukan untuk mengurangi gangguan yang dapat mempengaruhi kualitas model. *Interquartile Range* (IQR) digunakan; data yang berada di luar rentang Q1–1.5 × IQR dan Q3+1.5 × IQR dianggap sebagai *outlier*. Setelah prosedur ini selesai, sejumlah data dihapus untuk mencegah model terdistraksi oleh jumlah data yang berlebihan, yang dapat menyebabkan *overfitting*. Jumlah total *outlier* yang ditemukan dan dihapus adalah sekitar 1,8% dari total data [20].

Tabel 3. *Noice Removal*

Feature	Jumlah *Outlier*	Persantase *Outlier* (%)
Bandwidth in	45	0,6%
Bandwidth out	38	0.5%
Ping	52	
Packet Per Second	30	0.2%
Average Packet Size	45	0.5%
Port Usage	35	0.7%
CPU Usage	27	0.4%
Error Rate	22	0.2%
Total	294	3.8%

Selain menghilangkan *noise*, tahap pengurangan fitur juga dilakukan untuk meningkatkan akurasi dan efisiensi model. Untuk mengidentifikasi multikolinearitas, *feature reduction* dilakukan melalui analisis korelasi antar *feature*. Karena berpotensi menyebabkan redundansi informasi, fitur yang memiliki korelasi yang sangat besar dengan fitur lain dihapus. *Latency*, misalnya, dihapus karena memiliki korelasi yang sangat besar dengan *bandwidth*, sehingga keberadaannya tidak memberikan informasi tambahan yang signifikan bagi model [21].

Tabel 4. *Feature* yang digunakan

Feature	Status	Alasan
Bandwidth in	Digunakan	Kontribusi Tinggi
Bandwidth out	Digunakan	Kontribusi Tinggi

Ping	Digunakan	Informasi Tambahan
Packet Per Second	Digunakan	Kontribusi Tinggi
Average Packet Size	Digunakan	Informasi Tambahan
Port Usage	Digunakan	Kontribusi Tinggi
CPU Usage	Digunakan	Kontribusi Tinggi
Error Rate	Digunakan	Kontribusi Tinggi

Setelah proses *preprocessing*, yang mencakup penghapusan suara dan pengurangan fitur, data dinormalisasi menggunakan metode *StandardScaler*. Tujuan normalisasi ini adalah untuk memastikan bahwa semua fitur memiliki skala yang sama, sehingga model *K-Nearest Neighbors* (KNN) dapat menghitung jarak antar titik data secara adil. Dengan normalisasi ini, model menjadi lebih sensitif terhadap pola yang relevan dan menghindari dominasi fitur dengan rentang nilai yang lebih besar. Akhir dari proses *preprocessing* adalah menilai distribusi data dan kinerja model menggunakan berbagai metrik seperti akurasi, presisi, *recall*, dan skor F1 [22].

Tabel 5. *Classification report Training* K 150

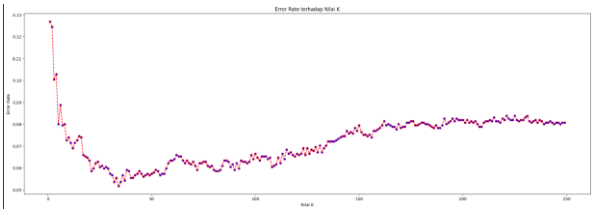
Label	*Precision*	*Recall*	F1-Score
Flood	92%	92%	92%
Normal	92%	92%	92%

Hasilnya menunjukkan bahwa model memiliki akurasi rata-rata sekitar 92 persen, dengan skor presisi, *recall*, dan F1 yang sebanding. Karena skor pada data *train* dan test relatif seimbang, hal ini menunjukkan bahwa model sudah mampu mendeteksi serangan dan menghindari *overfitting*. Model deteksi *flood attack* yang kuat dan siap digunakan pada sistem jaringan nyata dibuat dengan dukungan dari skema preprocessing yang komprehensif ini.

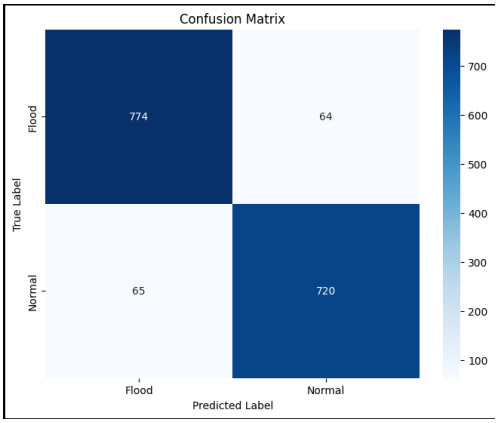
2.4. Evaluasi Kinerja Model

Tahap akhir yang sangat penting, evaluasi kinerja model, dilakukan untuk mengetahui seberapa baik model yang dibangun dapat memenuhi tujuan penelitian. Dalam penelitian ini, berbagai metrik utama akurasi, presisi, *recall*, dan skor F1 digunakan untuk evaluasi. Fokus utama evaluasi adalah pada metrik presisi dan *recall* karena keberhasilannya dalam mendeteksi serangan *flood* sebanyak mungkin sangat penting untuk keandalan sistem deteksi.

Karena jumlah data yang digunakan dalam penelitian ini cukup besar, nilai K = 150 dipilih karena jarak nilai K hanya sekitar 1 hingga 3 persen dari total data. Metode ini digunakan untuk mendapatkan hasil akurasi yang optimal sekaligus menghindari risiko *overfitting* atau *underfitting*, yang memastikan bahwa model tetap memiliki kemampuan generalisasi yang baik saat digunakan pada data nyata.



Gambar 5. Grafik *error rate* terhadap nilai K *data training*



Gambar 6. *Confusion matrix data training*

Selain itu, *confusion matrix*, juga dikenal sebagai matriks kebingungan, memberikan gambaran detail tentang distribusi prediksi model; matriks ini mencakup jumlah positif benar, positif salah, dan positif salah.

Dari total data uji, 774 data *Flood* berhasil diprediksi dengan benar sebagai *Flood* (*true positive*), dan 720 data Normal berhasil diprediksi dengan benar sebagai Normal (*true negative*). Namun, 64 data *Flood* yang salah diklasifikasikan sebagai Normal (*false negative*) dan 65 data Normal yang salah diklasifikasikan sebagai *flood* (*false positive*). Secara keseluruhan, evaluasi performa model menggunakan *confusion matrix* menunjukkan bahwa hasilnya mendukung temuan evaluasi sebelumnya, di mana akurasi, presisi, *recall*, dan skor F1 rata-rata mencapai 92 persen.

Oleh karena itu, model yang telah dilatih dengan skema *preprocessing* (pengurangan fitur, normalisasi, dan penghapusan suara) dapat mendeteksi trafik jaringan secara akurat dan seimbang antara kedua kelas. Ini juga memperkuat bukti bahwa model tidak mengalami *overfitting* atau *underfitting* [23].

Matriks konfusi dan metrik seperti skor F1, presisi, dan *recall* digunakan untuk mengevaluasi performa model. Di mana TP adalah nilai positif asli, TN adalah nilai negatif asli, FP adalah nilai positif palsu, dan FN adalah nilai negatif palsu, metrik ini dihitung dengan menggunakan rumus berikut:

$$Accuracy = \frac{TN + TP}{TN+TP+FN+FP}$$

$$Precision = \frac{TP}{TP+FP}$$

$$Recall = \frac{TP}{TP+FN}$$

$$F1\text{-}Score = 2 \times \frac{Presisi \times Recall}{Presisi + Recall}$$

3. Hasil dan Pembahasan

3.1. Hasil

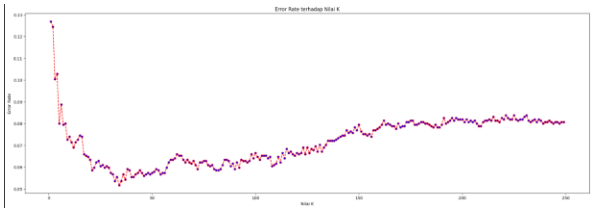
Hasil uji coba yang dilakukan terhadap model NFD yang didasarkan pada algoritma *K-Nearest Neighbors* (KNN) cukup menjanjikan. 80% data digunakan untuk pelatihan model, dan 20% sisanya digunakan untuk pengujian. Untuk memastikan penilaian yang objektif terhadap kinerja klasifikasi model, pembagian ini dilakukan secara acak tetapi seimbang antara dua kelas, trafik normal dan *flood*.

	precision	recall	f1-score	support
Flood	0.93	0.92	0.93	838
Normal	0.92	0.93	0.92	785

Gambar 7. Hasil akurasi data *testing*

Hasil pengujian menunjukkan akurasi model sebesar 92 persen hingga 93 persen, dengan parameter nilai k 150 dan variasi data uji, nilai K 150 digunakan bertujuan apakah dengan nilai K yang sama antara data latih dan uji akan mengalami perubahan dan dengan nilai K yang sama selisih akurasi sebesar 1%. Nilai ini menunjukkan kemampuan model untuk mengklasifikasikan data trafik jaringan dengan tingkat kesalahan yang relatif rendah.

Model dapat secara konsisten membedakan pola perilaku antara trafik *flood* dan normal, menunjukkan bahwa fitur-fitur jaringan seperti kecepatan paket per detik, *bandwidth* masuk, dan *bandwidth* keluar cukup representatif untuk menangkap karakteristik serangan *flood*.



Gambar 8. Grafik *error rate* terhadap *data testing*

Selain akurasi, hasil evaluasi menunjukkan bahwa kinerja model tidak berubah secara signifikan dalam berbagai skenario pengujian. Hal ini memperkuat gagasan bahwa model KNN dapat digunakan dengan cukup percaya diri dalam lingkungan jaringan nyata yang dinamis selama kualitas dan kelengkapan data pelatihan tetap terjaga. Hasil ini memberikan dasar yang kuat untuk tahap diskusi lebih lanjut tentang kelebihan, keterbatasan, dan kemungkinan pengembangan model NFD yang diusulkan.

3.2. Nilai Probabilitas Data Testing

Pada titik ini, pengujian dilakukan pada model yang telah dilatih menggunakan data uji untuk mengevaluasi kinerja dan akurasi prediksinya. Nilai probabilitas pada data uji menunjukkan tingkat kepercayaan model dalam menentukan klasifikasi setiap sampel data. Nilai probabilitas ini diperoleh dengan menghitung jarak antara sampel data uji dan data latihan menggunakan algoritma *K-Nearest Neighbors* (KNN).

Dengan mempertimbangkan parameter K yang optimal, model menentukan hasil klasifikasi berdasarkan mayoritas kelas dari tetangga terdekat yang telah ditentukan sebelumnya [24].

Tabel 6. Hasil Probabilitas Prediksi Data Training

Probabilitas Prediksi Data Training		
Data ke-	Nilai Prediksi *Flood*	Nilai Prediksi Normal
1	100%	0%
2	26%	73,91%
3	0%	100%
4	0%	100%
5	0%	100%

Dalam kebanyakan kasus, nilai probabilitas prediksi pada data pelatihan cenderung lebih ekstrem sekitar nol persen (0%) atau seratus persen (100%) dibandingkan dengan data pengujian. Hal ini disebabkan oleh fakta bahwa model telah belajar langsung dari data pelatihan, sehingga memiliki kepercayaan yang kuat terhadap prediksi yang dihasilkannya. Sebagai contoh, pada data pertama dari pelatihan, model memprediksi *Flood* dengan kemungkinan penuh seratus persen (100%), dan pada data ketiga hingga lima, memprediksi Normal dengan kemungkinan penuh seratus persen. Nilai-nilai yang sangat tinggi ini menunjukkan bahwa model mampu mengenali pola yang sudah dilatih dengan sangat baik, tetapi juga menunjukkan kecenderungan yang sangat tinggi untuk "keyakinan"[25].

Tabel 7. Hasil Probabilitas Prediksi Data Training

Data Ke-	Nilai Prediksi *Flood*	Nilai Prediksi Normal
1	87,33%%	12,67%
2	48%	52%
3	25,33%	74,67%
4	10,67%	89,33%
5	10%	90%

Lalu pada data pengujian, model cenderung menghasilkan probabilitas yang lebih seimbang atau moderat. Sebagai contoh, pada data uji kedua, model memprediksi *flood* dengan probabilitas 48% dan Normal dengan probabilitas 52%, yang menunjukkan ketidakpastian. Nilai probabilitas yang tidak terlalu tinggi ini menunjukkan bahwa model berhadapan dengan data yang belum pernah dilihat sebelumnya atau *unseen* data. Akibatnya, model harus "mempertimbangkan" dua kelas dengan lebih hati-hati. Karena tidak langsung "*overconfident*" pada prediksi di data tes, fenomena ini wajar terjadi dan menunjukkan bahwa model memiliki generalisasi yang baik.

Perbedaan pola probabilitas ini menunjukkan bahwa model tidak hanya menyesuaikan (*fit*) pada data pelatihan tetapi juga mampu menilai secara kritis data tes. Hal ini mendukung kesimpulan bahwa model telah dioptimasi untuk menghindari *overfitting*: model yang *overfit* biasanya akan tetap menunjukkan probabilitas ekstrem pada data testing, sedangkan model yang umum (dalam kasus ini) menunjukkan probabilitas yang lebih seimbang, selaras, dan selaras dengan data tes.

3.3 Evaluasi Akurasi per-*Fold* Menggunakan *Cross-Validation*

Untuk mengevaluasi keandalan model dalam berbagai skenario pembagian data, metode *cross-validation* dengan jumlah fold sebanyak 5 digunakan. Dalam penelitian ini, dataset dibagi secara merata menjadi lima subset yang tidak tumpang tindih. Empat *subset* digunakan sebagai data latihan (training data), sementara satu subset lagi digunakan sebagai data uji (*testing data*) dalam setiap iterasi. Proses ini diulang lima kali, sehingga setiap subset berfungsi sebagai data uji satu kali.

Tabel 8. Hasil *Cross-validation* 5 *Fold* pada Data *Training*

CV	*Cross-validation Accuracy*	*Mean Cross-validation Score*
5	[0.9963054187, 0.995073891, 0.993824535, 0.993842335, 0.9950678175]	0.9956

Dengan menggunakan *cross-validation* dengan jumlah fold sebanyak 5, hasil evaluasi menunjukkan akurasi rata-rata 99,56% pada data pelatihan dan 92,59% pada data pengujian. Perbedaan ini menunjukkan bahwa model sangat baik dalam mempelajari pola pada data pelatihan karena langsung belajar dari data tersebut. Akibatnya, nilai *cross-validation* sangat tinggi dan stabil di atas 9.

Tabel 9. Hasil *Cross-validation* 5 *Fold* pada Data *Testing*

CV	*Cross-validation Accuracy*	*Mean Cross-validation Score*
5	[0.933078911, 0.918488170, 0.9331717098, 0.923709187, 0.9214677809]	0.9259

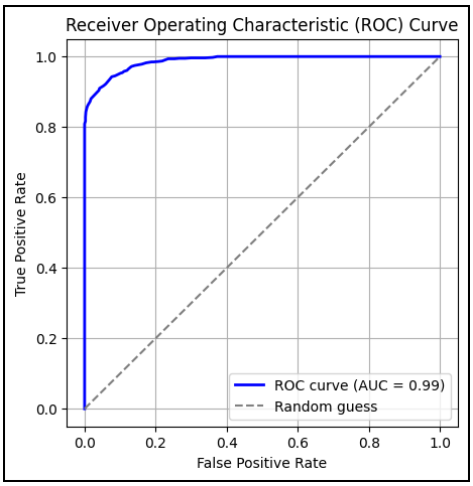
Namun, nilai *cross-validation* yang lebih rendah pada pengujian data pelatihan (sekitar 92 %) menunjukkan bahwa model dihadapkan pada data yang belum pernah dilihat sebelumnya. Nilai-nilai ini tetap tinggi dan relatif stabil di setiap *fold* (sekitar 91 hingga 93%), menunjukkan bahwa model masih mampu menjaga performa generalisasi meskipun tidak setinggi saat diuji pada data pelatihan. Penurunan akurasi pada data pengujian ini justru menunjukkan bahwa model tidak mengalami *overfitting*; jika terjadi *overfitting*, akurasi pelatihan akan tinggi, tetapi akurasi pengujian akan sangat rendah. Nilai *cross-validation* yang lebih tinggi pada pelatihan data daripada penilaian data adalah wajar. Dalam hal ini, ini menunjukkan bahwa model dapat mempertahankan generalisasi yang baik dengan perbedaan akurasi yang masih dalam batas yang masih bisa ditoleransi.

3.4. *Receiver Operating Characteristic Curve (ROC Curve*

Pada penelitian ini, kurva karakteristik operasi penerima (ROC) digunakan untuk menilai kinerja model pembelajaran mesin dalam mendeteksi *flood* pada jaringan. ROC *Curve* menunjukkan hubungan antara *True Positive Rate* (TPR) pada sumbu Y dan *False Positive Rate* (FPR) pada sumbu X. Ini menunjukkan kemampuan model untuk membedakan antara kelas *flood* positif dan kelas *flood* negatif.

Pada data tes, diperoleh *Area Under Curve* (AUC) sebesar 0,99, berdasarkan hasil evaluasi menggunakan kurva *Receiver Operating Characteristic* (ROC). Nilai AUC yang mendekati nilai 1 menunjukkan bahwa model memiliki kemampuan diskriminasi yang sangat baik untuk membedakan antara kelas *flood* dan Normal. Ini ditunjukkan oleh kurva ROC yang hampir menempel di sisi kiri atas grafik (nilai positif benar mendekati 1 dengan nilai positif salah sangat rendah), yang menunjukkan bahwa model memiliki kemampuan diskriminasi yang sangat. Selain itu, kurva ROC menunjukkan bahwa model memiliki kemampuan untuk mempertahankan *trade-off* yang ideal antara tingkat sensitivitas, yang dikenal sebagai

tingkat positif asli, dan tingkat spesifisitas, yang dikenal sebagai tingkat positif palsu. Dengan nilai AUC yang tinggi, model tidak hanya memiliki akurasi yang tinggi, tetapi juga dapat digeneralisasi dengan baik terhadap data yang belum pernah dilihat (unseen data), khususnya dalam konteks pengujian data. Hal ini mendukung skor F1, akurasi, presisi, dan *recall*, yang menunjukkan nilai tinggi dan seimbang di kedua kelas.



Gambar 9. Kurva ROC pada model KNN

Secara keseluruhan, hasil evaluasi menggunakan kurva ROC mendukung kesimpulan bahwa model deteksi *flood attack* yang dibangun sudah berada pada kondisi ideal, tidak mengalami *overfitting*, dan siap untuk digunakan pada data real di lingkungan jaringan sebenarnya.

3.5. Pembahasan

Hasil penelitian menunjukkan bahwa metode pembelajaran yang diawasi dapat digunakan untuk membuat model *machine learning* untuk mendeteksi *flood* pada jaringan. Namun, untuk memastikan bahwa model dapat belajar secara optimal, metode ini memerlukan data dalam skala besar. Selain itu, parameter tertentu diperlukan yang dapat mensimulasikan atau memberikan label yang jelas terhadap pola *flood* pada jaringan. Dengan adanya parameter yang sesuai, model dapat melakukan klasifikasi dengan lebih baik.

```
Classification Report setelah tuning threshold:
              precision    recall  f1-score   support

       Flood       0.93      0.92      0.93       838
      Normal       0.92      0.93      0.92       785

    accuracy                           0.92      1623
   macro avg       0.92      0.92      0.92      1623
weighted avg       0.92      0.92      0.92      1623

Akurasi setelah tuning threshold (threshold = 0.5): 92.42%
```

Gambar 10. Akurasi *testing* setelah proses *tuning*

Proses tuning *threshold* pada model digunakan untuk mengevaluasi perbedaan antara sensitivitas (*recall*) dan spesifisitas (*precision*), dengan *threshold* akhir sebesar 0,5. Hasilnya menunjukkan akurasi sebesar 92,42%. Pilihan ambang sangat memengaruhi kinerja model, terutama dalam kasus deteksi angkatan air. Di sini, keseimbangan antara mendeteksi serangan dan menghindari *false alarm* pada trafik biasa sangat penting.

Menurut laporan klasifikasi setelah penyesuaian, nilai ketepatan kelas *flood* sebesar 93% dan nilai ketepatan kelas Normal sebesar 93%, dengan nilai *recall* masing-masing sebesar 93% dan 93%. Nilai f1-*score* yang konsisten antara 92 dan 93 persen menunjukkan bahwa model tidak hanya mampu mendeteksi serangan dengan baik tetapi juga mampu mendeteksi trafik normal dengan optimal.

Threshold yang telah dioptimasi memungkinkan model untuk mengubah tingkat kepercayaan dalam menentukan *instance* sebagai *flood* atau Normal. Ini membuat hasil prediksi lebih seimbang dan tidak berat sebelah, mendukung hasil kurva ROC sebelumnya yang menunjukkan area di bawah kurva (AUC) yang sangat besar dan mendukung kesimpulan bahwa model telah berada pada kondisi idealnya tanpa *overfitting*.

Secara keseluruhan, akurasi sebesar 92,42% setelah ambang penyesuaian menunjukkan bahwa model mampu bekerja dengan sangat baik dalam pengujian data. Selain itu, proses penyesuaian ambang ini meningkatkan keandalan model saat digunakan pada sistem real karena memungkinkan kontrol yang lebih akurat terhadap kesalahan *trade-off* yang mungkin terjadi di dunia nyata.

4. Kesimpulan

Penelitian ini menemukan bahwa model *Network Flood Detection* (NFD) yang menggunakan algoritma pembelajaran *Supervised K-Nearest Neighbor* (KNN) memiliki kinerja yang sangat baik dalam mendeteksi *flood* network. Dengan nilai akurasi tinggi, model yang dikembangkan menghasilkan nilai *cross-validation* rata-rata sebesar 0,9259 dan AUC sebesar 0,99 juga target awal yaitu presisi di 92%-93% dan *recall* 92%-93% dengan total akurasi keseluruhan 92,42% setelah tuning *treshold*, yang menunjukkan bahwa model memiliki kemampuan untuk membedakan lalu lintas jaringan *flood* dan normal secara optimal. Hasil ini menunjukkan bahwa metode pembelajaran yang diawasi, yang sebelumnya dianggap memiliki keterbatasan karena ketergantungannya pada data latihan, juga mampu menghasilkan model yang efektif dan akurat jika diterapkan dengan benar.

Konsep Model *Network Flood Detection* (NFD) yang telah dikembangkan telah menunjukkan hasil yang baik, tetapi masih ada peluang untuk pengembangan penelitian di masa depan. Pengujian tambahan pada data *real-time* dengan skenario serangan yang lebih kompleks dan beragam, seperti serangan multi-vektor dan serangan terdistribusi. Selain itu, metode

pembelajaran *semi-supervised* atau *non-supervised* dapat dipelajari untuk mengidentifikasi serangan baru (serangan zero-day).

Reference

[1] S. M. S. Bukhari et al., "Secure and privacy-preserving intrusion detection in wireless sensor networks: Federated learning with SCNN-Bi-LSTM for enhanced reliability," Ad Hoc Networks, vol. 155, Mar. 2024, doi: 10.1016/j.adhoc.2024.103407.

[2] S. A. A. Bokhari and S. Myeong, "The Influence of Artificial Intelligence on E-Governance and Cybersecurity in Smart Cities: A Stakeholder's Perspective," IEEE Access, vol. 11, 2023, doi: 10.1109/ACCESS.2023.3293480.

[3] Uchenna Joseph Umoga et al., "Exploring the potential of AI-driven optimization in enhancing network performance and efficiency," Magna Scientia Advanced Research and Reviews, vol. 10, no. 1, pp. 368–378, Feb. 2024, doi: 10.30574/msarr.2024.10.1.0028.

[4] B. Riskhan et al., "An Adaptive Distributed Denial of Service Attack Prevention Technique in a Distributed Environment," Sensors, vol. 23, no. 14, Jul. 2023, doi: 10.3390/s23146574.

[5] Y. Fu, X. Duan, K. Wang, and B. Li, "Low-rate Denial of Service attack detection method based on time-frequency characteristics," Journal of Cloud Computing, vol. 11, no. 1, Dec. 2022, doi: 10.1186/s13677-022-00308-3.

[6] B. Fijatmiko and R. Sopandi, "Monitoring Dan Analisis Trafik Di Kejaksaan Negeri Jakarta Barat Peassler Router Traffic Grapher (Prtg)," Justifi, vol. 2, no. 1, pp. 23–31, 2022.

[7] R. K. Halder, M. N. Uddin, M. A. Uddin, S. Aryal, and A. Khraisat, "Enhancing K-nearest neighbor algorithm: a comprehensive review and performance analysis of modifications," J Big Data, vol. 11, no. 1, 2024, doi: 10.1186/s40537-024-00973-y.

[8] S. Wang, J. F. Balarezo, S. Kandeepan, A. Al-Hourani, K. G. Chavez, and B. Rubinstein, "Machine learning in network anomaly detection: A survey," IEEE Access, vol. 9, pp. 152379–152396, 2021, doi: 10.1109/ACCESS.2021.3126834.

[9] E. Altayef, F. Anayi, M. Packianather, Y. Benmahamed, and O. Kherif, "Detection and Classification of Lamination Faults in a 15 kVA Three-Phase Transformer Core Using SVM, KNN and DT Algorithms," IEEE Access, vol. 10, pp. 50925–50932, 2022, doi: 10.1109/ACCESS.2022.3174359.

[10] N. Çevik and S. Akleylek, "SoK of Machine Learning and Deep Learning Based Anomaly Detection Methods for Automatic Dependent Surveillance- Broadcast," IEEE Access, vol. 12, no. March, pp. 35643–35662, 2024, doi: 10.1109/ACCESS.2024.3369181.

[11] M. Owusu-Adjei, J. Ben Hayfron-Acquah, T. Frimpong, and G. Abdul-Salaam, "Imbalanced class distribution and performance evaluation metrics: A systematic review of prediction accuracy for determining model performance in healthcare systems," PLOS Digital Health, vol. 2, no. 11 November, pp. 1–19, 2023, doi: 10.1371/journal.pdig.0000290.

[12] S. Huang, Y. Lyu, Y. Peng, and M. Huang, "Analysis of Factors Influencing Rockfall Runout Distance and Prediction Model Based on an Improved KNN Algorithm," IEEE Access, vol. 7, pp. 66739–66752, 2019, doi: 10.1109/ACCESS.2019.2917868.

[13] S. Pitafi, T. Anwar, I. D. M. Widia, B. Yimwadsana, and S. Pitafi, "Revolutionizing Perimeter Intrusion Detection: A Machine Learning-Driven Approach with Curated Dataset Generation for Enhanced Security," IEEE Access, vol. 11, no. October, pp. 106954–106966, 2023, doi: 10.1109/ACCESS.2023.3318600.

[14] W. Xing and Y. Bei, "Medical Health Big Data Classification Based on KNN Classification Algorithm," IEEE Access, vol. 8, pp. 28808–28819, 2020, doi: 10.1109/ACCESS.2019.2955754.

[15] S. Zhang, "Challenges in KNN Classification," IEEE Trans Knowl Data Eng, vol. 34, no. 10, pp. 4663–4675, 2022, doi: 10.1109/TKDE.2021.3049250.

[16] J. Gu, Q. Zou, C. Deng, and X. Wang, "A Novel Robust Online Extreme Learning Machine for the Non-Gaussian Noise," Chinese Journal of Electronics, vol. 32, no. 1, pp. 130–139, 2023, doi: 10.23919/cje.2021.00.122.

[17] A. A. Bakar, O. H. Yi, and N. Chepa, "Journal of digital system development," vol. 2, no. 2, pp. 79–92, 2024.

[18] Y. Liu, M. Ji, S. S. Lin, M. Zhao, and Z. Lyv, "Combining Readability Formulas and Machine Learning for Reader-oriented Evaluation of Online Health Resources," IEEE Access, vol. 9, pp. 67610–67619, 2021, doi: 10.1109/ACCESS.2021.3077073.

[19] M. F. Kucuk and I. Uysal, "Anomaly Detection in Self-Organizing Networks: Conventional Versus Contemporary Machine Learning," IEEE Access, vol. 10, pp. 61744–61752, 2022, doi: 10.1109/ACCESS.2022.3182014.

[20] S. Puttinaovarat and P. Horkaew, "Flood Forecasting System Based on Integrated Big and Crowdsource Data by Using Machine Learning Techniques," IEEE Access, vol. 8, pp. 5885–5905, 2020, doi: 10.1109/ACCESS.2019.2963819.

[21] G. Ramesh, J. Logeshwaran, and A. P. Kumar, "The Smart Network Management Automation Algorithm for Administration of Reliable 5G Communication Networks," Wirel Commun Mob Comput, vol. 2023, 2023, doi: 10.1155/2023/7626803.

[22] I. Ullah and Q. H. Mahmoud, "A Framework for Anomaly Detection in IoT Networks Using Conditional Generative Adversarial Networks," IEEE Access, vol. 9, pp. 165907–165931, 2021, doi: 10.1109/ACCESS.2021.3132127.

[23] S. Zainab, D. H. Nugroho, M. R. T. Siregar, H. S. WD, A. Multi, and M. N. Huda, "Network Metadata (NETTA) : Sistem Monitoring Jaringan Dan Metadata UPT BMKG Dengan Notifikasi Berbasis Telegram," Sainstech: Jurnal Penelitian Dan Pengkajian Sains Dan Teknologi, vol. 33, no. 1, pp. 52–61, 2023, doi: 10.37277/stch.v33i1.1653.

[24] S. Kamamura, Y. Takei, M. Nishiguchi, Y. Hayashi, and T. Fujiwara, "Network Anomaly Detection Through IP Traffic Analysis With Variable Granularity," IEEE Access, vol. 11, no. October, pp. 129818–129828, 2023, doi: 10.1109/ACCESS.2023.3334212.

[25] M. Shajari, H. Geng, K. Hu, and A. Leon-Garcia, "Tensor-Based Online Network Anomaly Detection and Diagnosis," IEEE Access, vol. 10, no. August, pp. 85792–85817, 2022, doi: 10.1109/ACCESS.2022.3197651.