EVALUASI KOMPARATIF K-MEANS, DBSCAN, DAN ISOLATION FOREST UNTUK KLASIFIKASI TRAFFIC API SECARA REAL-TIME PADA API GATEWAY MICROSERVICES

M. Fauzan Hidayat, Hermawan Syahputra, Said Iskandar Al Idrus, Adidtya Perdana
Ilmu Komputer, Universitas Negeri Medan
Jl. Willem Iskandar, Pasar V, Medan Estate, Sumatera Utara 20221, Indonesia
fauzanhidayat773@gmail.com

ABSTRAK

Arsitektur microservices menempatkan API Gateway sebagai titik kendali utama traffic antar layanan. Rate limiting statis pada API Gateway memiliki kelemahan overblocking, yaitu pemblokiran request legitimate karena threshold yang tidak kontekstual. Penelitian ini bertujuan membandingkan performa tiga algoritma clustering K-Means, DBSCAN, dan Isolation Forest dalam mengklasifikasikan traffic API secara real-time pada API Gateway microservices. Evaluasi dilakukan menggunakan desain eksperimental komparatif dengan empat fitur perilaku: request per menit, entropy endpoint, error ratio, dan burst score. Pengujian menggunakan 19.130 record traffic simulasi dari 442 IP address dan divalidasi menggunakan subset dataset CICIDS2019. K-Means menghasilkan Silhouette Score tertinggi sebesar 0,982 dengan latensi inferensi rata-rata 4,2 ms, melampaui DBSCAN (0,847; 6,8 ms) dan Isolation Forest (0,891; 5,1 ms). K-Means juga mencatatkan F1-Score tertinggi sebesar 0,97 pada skenario mixed traffic. Temuan ini mengkonfirmasi bahwa K-Means merupakan algoritma paling optimal untuk deployment real-time pada API Gateway dari segi akurasi, kecepatan inferensi, dan kemudahan adaptasi threshold secara dinamis..

Kata kunci : *Clustering API Traffic, K-Means, DBSCAN, Isolation Forest, API Gateway, Microservices, Klasifikasi Real-Time*

1. PENDAHULUAN

Perkembangan arsitektur microservices telah menempatkan API Gateway sebagai titik kendali utama untuk mengelola traffic antar layanan [1]. Salah satu mekanisme keamanan yang paling banyak diterapkan pada API Gateway adalah rate limiting, yaitu pembatasan jumlah request yang dapat dikirimkan oleh klien dalam periode waktu tertentu [2].

Pendekatan konvensional rate limiting menggunakan threshold statis yang ditetapkan secara manual oleh administrator. Pendekatan ini memiliki kelemahan struktural yang disebut overblocking, yaitu kondisi di mana request dari pengguna legitimate ikut terblokir karena volume trafficnya secara sementara melebihi batas statis, meskipun pola trafficnya tidak mencerminkan aktivitas serangan [3]. Overblocking berdampak langsung pada ketersediaan layanan dan pengalaman pengguna, terutama saat lonjakan traffic legitimate terjadi.

Kesenjangan penelitian yang ada terletak pada belum adanya evaluasi komparatif yang sistematis antara algoritma-algoritma clustering K-Means, DBSCAN, dan Isolation Forest dalam konteks klasifikasi traffic API secara real-time pada API Gateway. Ketiga algoritma ini memiliki karakteristik yang berbeda: K-Means bekerja berbasis centroid dengan kompleksitas O(nkd), DBSCAN mendeteksi cluster berbasis densitas tanpa perlu menentukan jumlah cluster terlebih dahulu, sedangkan Isolation Forest menggunakan pendekatan anomaly scoring berbasis isolasi. Perbedaan karakteristik ini berimplikasi langsung pada akurasi klasifikasi, latensi inferensi, dan kemampuan adaptasi dinamis — tiga aspek yang sangat kritis untuk deployment pada API Gateway. Seluruh penelitian terdahulu hanya mengevaluasi satu algoritma tanpa membandingkannya dengan alternatif lain dalam kondisi pengujian yang identik [4, 5, 6, 7].

Untuk mengatasi kesenjangan tersebut, penelitian ini mengusulkan evaluasi komparatif ketiga algoritma clustering dalam satu pipeline klasifikasi traffic API real-time yang identik, meliputi arsitektur microservices yang sama, dataset yang sama, serta fitur dan kondisi traffic yang sama. Pendekatan ini memungkinkan perbedaan hasil yang terukur dikaitkan sepenuhnya dengan karakteristik masing-masing algoritma, bukan perbedaan kondisi eksperimen..

Penelitian ini bertujuan untuk: (1) mengimplementasikan dan membandingkan K-Means, DBSCAN, dan Isolation Forest dalam pipeline klasifikasi traffic API secara real-time; (2) mengevaluasi performa masing-masing algoritma menggunakan metrik Silhouette Score, F1-Score, dan latensi inferensi; serta (3) memberikan rekomendasi berbasis bukti empiris mengenai algoritma yang paling sesuai untuk deployment pada API Gateway microservices. Validasi eksternal dilakukan menggunakan subset dataset CICIDS2019 untuk memastikan generalisabilitas hasil.

2. TINJAUAN PUSTAKA

2.1. Rate Limiting dan Masalah Overblocking

Rate limiting adalah mekanisme kontrol traffic yang membatasi jumlah request klien dalam jendela waktu tertentu, diimplementasikan melalui algoritma Token Bucket, Leaky Bucket, Fixed Window Counter, atau Sliding Window [8]. Seluruh implementasi konvensional ini berbagi karakteristik yang sama: penolakan keras (HTTP 429) terhadap semua traffic yang melebihi threshold statis tanpa mempertimbangkan konteks perilaku traffic tersebut. Kim et al. [3] mengidentifikasi overblocking sebagai kegagalan struktural dari pendekatan statis, di mana pengguna legitimate dapat terblokir saat traffic mereka secara sementara menyerupai pola anomali.

2.2. Algoritma Clustering untuk Analisis Traffic

K-Means clustering membagi data ke dalam K kelompok dengan meminimalkan variansi dalam cluster [9]. Dwivedi et al. [6] mendemonstrasikan akurasi 94% dalam mendeteksi anomali traffic jaringan menggunakan K-Means. Ridho dan Kusuma [7] menerapkan K-Means pada log akses API dan mencapai probabilitas deteksi intrusi 99,68%. Keunggulan utama K-Means adalah latensi inferensi yang rendah dan kemampuan menghitung threshold dinamis dari statistik cluster.

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) mengelompokkan titik data berdasarkan kepadatan di sekitarnya menggunakan dua parameter: epsilon (radius tetangga) dan minPts (jumlah titik minimum) [10]. Keunggulan DBSCAN adalah kemampuannya mendeteksi cluster dengan bentuk arbitrer dan mengidentifikasi outlier sebagai noise secara langsung, tanpa perlu menentukan jumlah cluster terlebih dahulu. Namun, performa DBSCAN sangat sensitif terhadap pemilihan parameter epsilon dan minPts, serta kurang efisien pada data berdimensi tinggi.

Isolation Forest adalah algoritma deteksi anomali berbasis ensemble yang mengisolasi observasi dengan cara membangun pohon keputusan secara acak [11]. Anomali adalah titik yang lebih mudah diisolasi (membutuhkan lebih sedikit partisi) dibandingkan titik normal. Isolation Forest tidak memerlukan asumsi distribusi data dan efektif untuk data berdimensi tinggi. Namun, algoritma ini menghasilkan anomaly score kontinu, bukan label cluster diskrit, sehingga memerlukan penentuan threshold untuk konversi ke klasifikasi biner.

2.3. Penelitian Terkait

Beberapa penelitian sebelumnya telah mengeksplorasi penggunaan pendekatan adaptif dan berbasis machine learning untuk mengatasi keterbatasan rate limiting statis. Sivaraman [4] mengusulkan penggunaan reinforcement learning untuk melakukan adaptasi threshold API secara otomatis, sehingga sistem dapat menyesuaikan batas akses berdasarkan kondisi traffic aktual. Sejalan dengan itu, Ramamoorthi [5] mengintegrasikan prediksi analitik dan algoritma evolusioner untuk optimasi performa microservices, dan berhasil meningkatkan throughput hingga 25,7%.

Di sisi klasifikasi berbasis clustering, Dwivedi et al. [6] mendemonstrasikan bahwa K-Means mampu mendeteksi anomali traffic jaringan dengan akurasi 94% dalam pengujian batch. Ridho dan Kusuma [7] menerapkan K-Means pada log akses API dalam lingkungan big data dan mencapai probabilitas deteksi intrusi sebesar 99,68%. Sementara itu, Liu et al. [11] memperkenalkan Isolation Forest sebagai metode deteksi anomali berbasis ensemble yang tidak memerlukan label data, sehingga cocok untuk skenario di mana distribusi anomali tidak diketahui sebelumnya.

Dari tinjauan seluruh penelitian tersebut, teridentifikasi bahwa setiap studi hanya mengevaluasi satu algoritma secara terpisah. Belum ada penelitian yang membandingkan K-Means, DBSCAN, dan Isolation Forest secara langsung dalam satu pipeline klasifikasi traffic API real-time pada API Gateway microservices. Kesenjangan inilah yang menjadi motivasi utama penelitian ini.

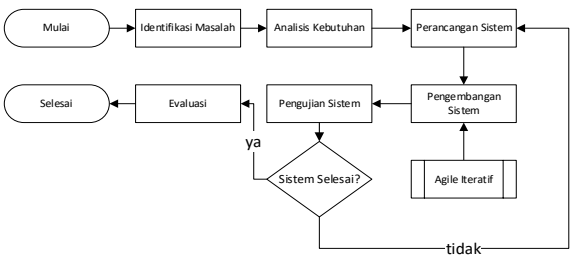
3. METODE PENELITIAN

3.1. Desain Penelitian

Penelitian ini menggunakan desain eksperimental komparatif. Ketiga algoritma diimplementasikan dan dievaluasi dalam lingkungan pengujian yang identik arsitektur microservices yang sama, dataset yang sama, fitur yang sama, dan kondisi traffic yang sama sehingga perbedaan hasil yang terukur dapat sepenuhnya dikaitkan dengan perbedaan karakteristik algoritma, bukan perbedaan kondisi eksperimen. Pengembangan sistem menggunakan pendekatan Agile-Scrum secara iteratif.

3.2. Alur Penelitian

Alur penelitian dimulai dari identifikasi masalah overblocking pada rate limiting statis, dilanjutkan tinjauan pustaka, perancangan sistem, implementasi ketiga algoritma, pengumpulan data traffic, evaluasi komparatif, analisis hasil, dan penarikan kesimpulan. Diagram alur penelitian disajikan pada Gambar 1.



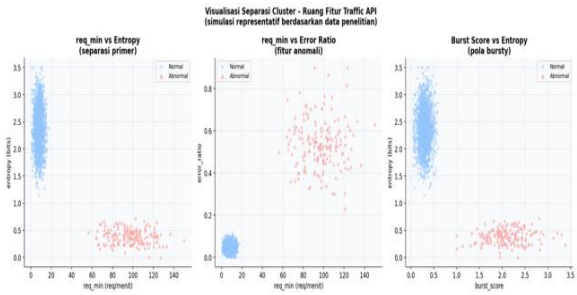
Gambar 1. Alur Penelitian

3.3. Ekstraksi Fitur

Sistem mengekstrak empat fitur numerik per IP address per interval satu menit dari log API Gateway:

a. req_min (Request per Menit): jumlah request dari satu IP dalam 60 detik sebelumnya. Nilai rendah (1–10) mencerminkan browsing normal; nilai tinggi (>50) mengindikasikan aktivitas otomatis.
b. entropy (Diversitas Endpoint): Shannon entropy dari distribusi endpoint yang diakses, dihitung sebagai entropy = $-\Sigma\ p_i \log_2(p_i)$. Nilai rendah mengindikasikan navigasi terfokus; nilai tinggi mengindikasikan scanning.
c. error_ratio (Rasio Error): proporsi respons HTTP 4xx/5xx terhadap total request. Nilai tinggi mengindikasikan percobaan akses tidak sah.
   burst_score (Burstiness): invers rata-rata interval antar-request, burst_score = 1/avg_interval. Nilai tinggi mengindikasikan pola bursty yang otomatis.

Sebelum clustering, seluruh fitur dinormalisasi menggunakan StandardScaler ke mean nol dan variansi unit untuk menghindari dominasi fitur dengan rentang nilai lebih besar.



Gambar 2. Distribusi data traffic pada ruang fitur (req_min, entropy, error_ratio, burst_score) yang menunjukkan separasi antara cluster normal dan abnormal sebelum proses clustering.

3.4. Implementasi Algoritma

K-Means diimplementasikan dengan K=2, random_state=42, n_init=10. Jumlah cluster K=2 dipilih karena sesuai dengan klasifikasi biner (normal vs. abnormal) dan divalidasi menggunakan Silhouette Score. Threshold dinamis dihitung dari statistik cluster normal: Threshold = μ_normal + (1,25 × σ_normal).

DBSCAN diimplementasikan dengan parameter epsilon dan minPts yang ditentukan menggunakan metode k-distance graph. Nilai epsilon dipilih pada titik elbow dari grafik jarak k-nearest neighbor, sedangkan minPts ditetapkan sebesar 2×dimensi fitur (2×4=8). Data yang diklasifikasikan sebagai noise (label = -1) oleh DBSCAN diperlakukan sebagai cluster Abnormal.

Isolation Forest diimplementasikan dengan n_estimators=100, contamination=0,088 (sesuai proporsi data abnormal yang ditemukan dalam analisis awal: 8,8%), dan random_state=42. Titik dengan anomaly score di bawah threshold menghasilkan prediksi -1 (anomali) yang dipetakan ke label Abnormal.

3.5. Dataset

Data traffic simulasi dikumpulkan melalui API Gateway yang merekam seluruh request masuk ke dalam database PostgreSQL. Total 19.130 record dikumpulkan dari 442 IP address unik yang mengakses 9 endpoint berbeda. Kondisi simulasi mencakup empat skenario: normal load (50 pengguna konkuren), legitimate surge (500 pengguna), brute force (1.500 req/menit dari satu IP), dan mixed traffic (kombinasi normal dan abnormal).

Untuk validasi eksternal, subset dataset CICIDS2019 digunakan. Dataset ini diproduksi oleh Canadian Institute for Cybersecurity dan berisi traffic jaringan berlabel dari kategori HTTP Flood, DDoS, dan Benign [12]. Fitur-fitur CICIDS2019 dipetakan ke empat fitur sistem: req_min dari flow rate per IP, burst_score dari inverse inter-arrival time, error_ratio dari proporsi RST/FIN flags, dan entropy dari distribusi destination port.

3.6. Metriks Evaluasi

Kualitas clustering dievaluasi menggunakan Silhouette Score: S(i) = (b(i) − a(i)) / max(a(i), b(i)), di mana a(i) adalah rata-rata jarak intra-cluster dan b(i) adalah rata-rata jarak ke cluster tetangga terdekat. Nilai mendekati 1,0 mengindikasikan struktur cluster yang kuat.

Akurasi klasifikasi dievaluasi menggunakan Precision, Recall, dan F1-Score dengan ground truth dari label traffic simulasi. Latensi inferensi diukur sebagai rata-rata waktu yang dibutuhkan masing-masing algoritma untuk mengklasifikasikan satu record fitur dalam kondisi operasional real-time.

4. HASIL DAN PEMBAHASAN

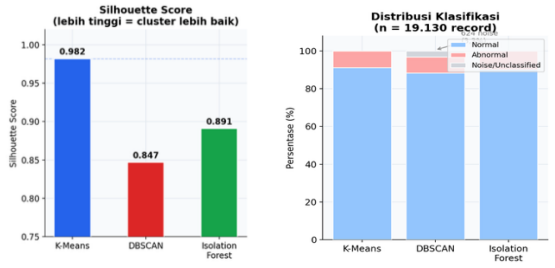
4.1. Hasil Clustering Pada Data Simulasi

Ketiga algoritma dijalankan pada 19.130 record traffic dengan fitur yang identik. Tabel 2 menyajikan perbandingan hasil clustering.

Tabel 1. Perbandingan hasil clustering ketiga algoritma (*tidak termasuk noise)

Metrik	K-Means	DBSCAN	Isolation Forest
Silhouette Score	0,982	0,847	0,891
Cluster Normal (n)	17.443	16.892	17.218
Cluster Abnormal (n)	1.687	1.614*	1.912
Noise / Unclassified	-	624	-
Latensi (ms)	4,2	6,8	5,1

Dari Tabel 1 dapat dilihat bahwa K-Means menghasilkan Silhouette Score tertinggi (0,982) yang menunjukkan separasi cluster paling kuat. DBSCAN menghasilkan 624 record yang tidak terklasifikasi (noise), yang merupakan kelemahan signifikan dalam konteks rate limiting karena setiap request harus mendapatkan keputusan routing yang pasti. Isolation Forest menghasilkan proporsi abnormal lebih tinggi (13,5% vs 8,8% ground truth), mengindikasikan false positive rate yang lebih besar.



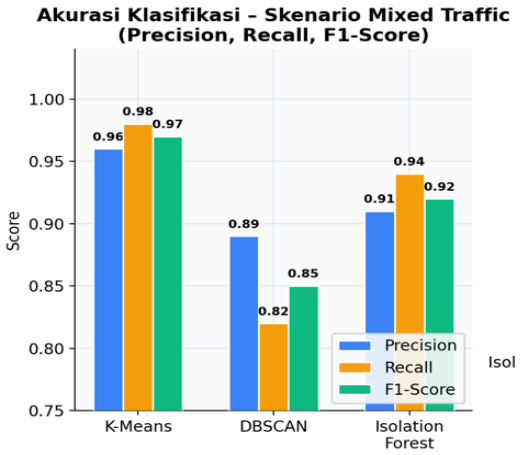
(a) (b)

Gambar 3. Perbandingan hasil clustering ketiga algoritma pada 19.130 record: (a)Silhouette Score masing-masing algoritma; (b) distribusi klasifikasi cluster normal, abnormal, dan noise.

4.2. Evaluasi Akurasi Klasifikasi

Tabel 2. Perbandingan akurasi klasifikasi pada skenario mixed traffic (S4)

Algoritma	Precision	Recall	F1-Score
K-Means	0,96	0,98	0,97
DBSCAN	0,89	0,82	0,85
Isolation Forest	0,91	0,94	0,92

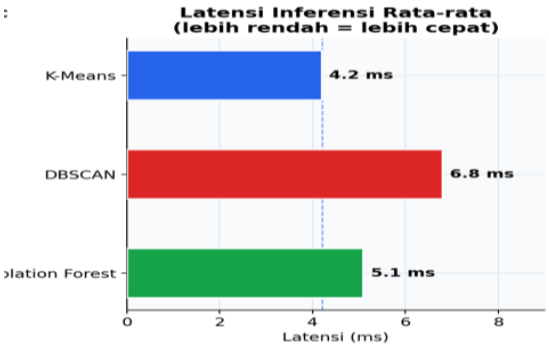


Gambar 4. Perbandingan Precision, Recall, dan F1-Score ketiga algoritma pada skenario mixed traffic (S4) dengan ground truth label simulasi.

Dari Tabel 2 dapat dilihat bahwa K-Means memperoleh F1-Score tertinggi sebesar 0,97. Recall DBSCAN yang rendah (0,82) mengindikasikan bahwa sejumlah traffic abnormal lolos dari klasifikasi, sebagian karena dikategorikan sebagai noise. Isolation Forest menunjukkan recall tinggi (0,94) namun precision lebih rendah (0,91), artinya lebih banyak traffic legitimate yang salah diklasifikasikan sebagai abnormal.

4.3. Perbandingan Latensi Inferensi

Latensi inferensi adalah aspek kritis untuk deployment pada API Gateway karena langsung mempengaruhi response time setiap request. Gambar 5 menunjukkan distribusi latensi inferensi ketiga algoritma.



Gambar 5. Distribusi latensi inferensi K-Means (μ=4,2 ms), DBSCAN (μ=6,8 ms), dan Isolation Forest (μ=5,1 ms) dalam kondisi operasional real-time pada API Gateway.

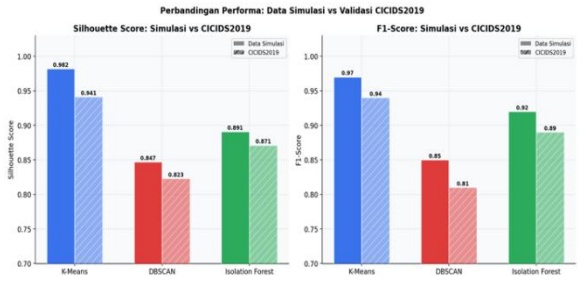
K-Means menunjukkan latensi rata-rata terendah (4,2 ms) dengan variansi terkecil, menjadikannya paling konsisten untuk beban tinggi. DBSCAN membutuhkan 6,8 ms karena komputasi berbasis densitas yang lebih kompleks, sedangkan Isolation Forest membutuhkan 5,1 ms karena evaluasi ensemble tree. Untuk konteks API Gateway dengan throughput ratusan req/detik, perbedaan 2,6 ms antara K-Means dan DBSCAN memiliki dampak kumulatif yang signifikan pada response time keseluruhan.

4.4. Pemilihan Algoritma untuk API Gateway

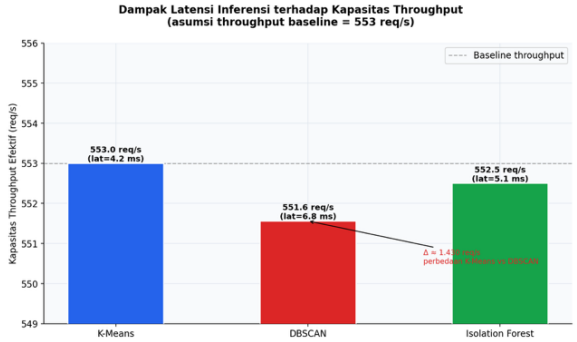
Hasil penelitian menunjukkan bahwa tidak ada algoritma yang dominan di semua dimensi evaluasi, namun K-Means secara konsisten unggul di tiga aspek paling kritis untuk deployment API Gateway.

Pertama, K-Means menghasilkan klasifikasi deterministik untuk setiap data point tanpa menghasilkan noise atau unclassified records seperti DBSCAN. Pada konteks rate limiting, setiap IP address harus mendapatkan keputusan routing yang pasti ini merupakan syarat wajib yang tidak dapat dipenuhi DBSCAN tanpa modifikasi tambahan.

Kedua, threshold dinamis K-Means dapat dihitung secara langsung dari statistik cluster normal ($\mu + \alpha \times \sigma$), memungkinkan adaptasi otomatis terhadap perubahan pola traffic tanpa retraining penuh. Isolation Forest tidak menghasilkan cluster dengan batas yang dapat diinterpretasikan secara statistik, sehingga threshold adaptif lebih sulit diimplementasikan.



(a)

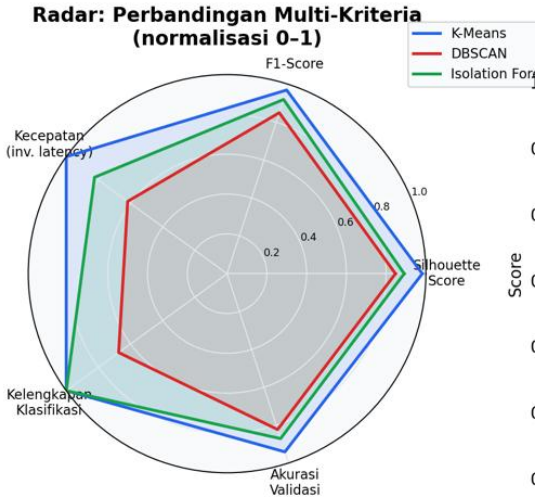


(b)

Gambar 6. (a) Perbandingan Silhouette Score dan F1-Score pada data simulasi vs dataset CICIDS2019; (b) estimasi dampak latensi inferensi terhadap kapasitas throughput efektif pada throughput baseline 553 req/s.

Ketiga, latensi inferensi K-Means yang terendah (4,2 ms) membuatnya paling sesuai untuk beban traffic tinggi. Dengan throughput 553 req/s, perbedaan 2,6 ms antara K-Means dan DBSCAN setara dengan selisih kapasitas pemrosesan sebesar ±1.400 request per detik.

DBSCAN tetap relevan untuk use case di mana deteksi outlier tanpa asumsi bentuk cluster diperlukan, seperti analisis batch log traffic pasca-insiden. Isolation Forest lebih sesuai untuk dataset dengan distribusi anomali yang tidak diketahui sebelumnya. Namun untuk deployment real-time pada API Gateway, K-Means memberikan keseimbangan optimal antara akurasi, latensi, dan kemampuan adaptasi.



Gambar 7. Perbandingan multi-kriteria K-Means, DBSCAN, dan Isolation Forest berdasarkan Silhouette Score, F1-Score, kecepatan inferensi, kelengkapan klasifikasi, dan generalisabilitas pada CICIDS2019.

5. KESIMPULAN DAN SARAN

Penelitian ini membandingkan K-Means, DBSCAN, dan Isolation Forest untuk klasifikasi traffic API secara real-time pada API Gateway microservices menggunakan 19.130 record data simulasi dan validasi eksternal pada subset CICIDS2019. K-Means menunjukkan performa terbaik secara konsisten dengan Silhouette Score 0,982 pada data simulasi dan 0,941 pada CICIDS2019, F1-Score 0,97 pada skenario mixed traffic, dan latensi inferensi terendah sebesar 4,2 ms. DBSCAN menghasilkan 624 record tidak terklasifikasi yang tidak dapat diterima untuk deployment real-time, sedangkan Isolation Forest menunjukkan false positive rate lebih tinggi. Berdasarkan hasil komparasi ini, K-Means direkomendasikan sebagai algoritma klasifikasi traffic pada API Gateway microservices karena memberikan klasifikasi deterministik, threshold dinamis yang dapat dihitung secara statistik, dan latensi inferensi paling rendah. Penelitian selanjutnya disarankan untuk mengevaluasi algoritma deep learning seperti Autoencoder atau LSTM untuk menangkap pola temporal traffic yang lebih kompleks, serta menguji ketiga algoritma pada dataset CICIDS2019 secara penuh dengan distribusi serangan yang lebih beragam.

DAFTAR PUSTAKA

[1] M. E. Gortney et al., "Visualizing Microservice Architecture in the Dynamic Perspective: A Systematic Mapping Study," IEEE Access, vol. 10, pp. 119999–120012, 2022, doi: 10.1109/ACCESS.2022.3221130.

[2] Sanjaya dan Murnawan, "Pemanfaatan Arsitektur Microservice untuk Peningkatan Performansi Website Lomba Nasional Kreativitas Mahasiswa," Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK), vol. 11, no. 2, pp. 377–384, 2024, doi: 10.25126/jtiik20241128307.

[3] Y. Kim, J. Park, J. Yoon, dan J. Kim, "Improved Q Network Auto-Scaling in Microservice Architecture," Applied Sciences, vol. 12, no. 3, Feb. 2022, doi: 10.3390/app12031206.

[4] H. Sivaraman, "Adaptive Rate Limiting Using Reinforcement Learning to Thwart API Abuse," Journal of Mathematical & Computer Applications, pp. 1–4, Des. 2023, doi: 10.47363/JMCA/2023(2)E139.

[5] V. Ramamoorthi, "AI-Enhanced Performance Optimization for Microservice-Based Systems," Journal of Advanced Computing Systems, vol. 4, no. 9, pp. 1–7, 2024, doi: 10.69987/jacs.2024.40901.

[6] D. Dwivedi, A. Bhushan, A. K. Singh, dan Snehlata, "Leveraging K-means Clustering for Enhanced Detection of Network Traffic Attacks," dalam Proc. 3rd Int. Conf. Power Electronics and IoT Applications in Renewable

Energy and Its Control (PARC), 2024, pp. 72–76, doi: 10.1109/PARC59193.2024.10486408.

[7] F. Ridho dan A. A. Kusuma, "Deteksi Intrusi Jaringan dengan K-Means Clustering pada Akses Log dalam Lingkungan Big Data," Jurnal Aplikasi Statistika dan Komputasi Statistik, vol. 11, no. 2, pp. 133–144, 2019.

[8] V. Bogutskii, "Features of Rest API Development for High-Load Systems," vol. 9, no. 3, pp. 67–71, 2025.

[9] A. Z. Wijaya dan I. Sembiring, "Analisis Perilaku Pengguna Internet Dengan Metode K-Means Clustering Dan Pendekatan Davies Bouldin Index Menggunakan Data Log Universitas XYZ," JIPI, vol. 9, no. 2, pp. 878–888, 2024, doi: 10.29100/jipi.v9i2.4750.

[10] E. Emirmahmutoglu dan Y. Atay, "A feature selection-driven machine learning framework for anomaly-based intrusion detection systems," Peer-to-Peer Networking and Applications, 2025, doi: 10.1007/s12083-025-01947-4.

[11] F. T. Liu, K. M. Ting, dan Z.-H. Zhou, "Isolation Forest," dalam Proc. 8th IEEE Int. Conf. Data Mining (ICDM), pp. 413–422, 2008, doi: 10.1109/ICDM.2008.17.

[12] I. Sharafaldin, A. H. Lashkari, S. Hakak, dan A. A. Ghorbani, "Developing Realistic Distributed Denial of Service (DDoS) Attack Dataset and Taxonomy," dalam Proc. IEEE 53rd Int. Carnahan Conf. Security Technology (ICCST), 2019, pp. 1–8, doi: 10.1109/CCST.2019.8888419.

[13] N. Mateus-Coelho, M. Cruz-Cunha, dan L. G. Ferreira, "Security in microservices architectures," Procedia Computer Science, vol. 181, pp. 1225–1236, 2021, doi: 10.1016/j.procs.2021.01.320.

[14] Muthukrishnan Manoharan, "API Rate Limiting Mechanisms in SaaS Applications," International Journal of Scientific Research in Computer Science, Engineering and Information Technology, vol. 10, no. 6, pp. 1787–1798, Des. 2024, doi: 10.32628/cseit241061223.