Blend Sains
Jurnal Teknik
https://jurnal.ilmubersama.com/index.php/blendsains

Artikel Penelitian (Teknik Informatika)

Penerapan *Deep Learning* untuk Deteksi Anomali dalam Jaringan Keamanan Siber Menggunakan *Recurrent Neural Networks* (RNNs)

Muhammad Rafly Yusuf [*], *Sumarlin*

Teknik Informatika, STIKOM Uyelindo Kupang, Kupang, Indonesia

INFORMASI ARTIKEL

Diterima Redaksi: 27 April 2025
Revisi Akhir: 07 Mei 2025
Diterbitkan *Online*: 07 Mei 2025

KATA KUNCI

Anomali
Jaringan
Siber
RNNs

KORESPONDENSI (*)

Phone: +62 812-3707-6715
E-mail: srraaffllyy69@gmail.com

A B S T R A K

Keamanan siber merupakan aspek kritis dalam era digital yang terus berkembang, di mana ancaman serangan siber semakin kompleks dan dinamis. Deteksi anomali dalam jaringan menjadi salah satu metode penting untuk mengidentifikasi aktivitas mencurigakan yang dapat mengindikasikan serangan siber. Penelitian ini bertujuan untuk menerapkan teknik *deep learning*, khususnya *Recurrent Neural Networks* (RNNs), untuk mendeteksi anomali dalam jaringan keamanan siber. RNNs dipilih karena kemampuannya dalam memproses data sekuensial dan menangkap pola temporal, yang sangat relevan dalam analisis lalu lintas jaringan. Dataset yang digunakan dalam penelitian ini mencakup berbagai skenario serangan siber dan aktivitas normal untuk melatih model. Eksperimen ini diharapkan dapat menunjukkan bahwa RNNs mampu mencapai akurasi yang tinggi dalam mendeteksi anomali dengan tingkat *false positive* yang rendah. Selain itu, model ini juga menunjukkan kemampuan adaptasi yang baik terhadap pola serangan yang baru. Temuan ini diharapkan dapat mengindikasikan bahwa penerapan RNNs dapat menjadi solusi efektif untuk meningkatkan sistem deteksi anomali dalam keamanan siber, memberikan perlindungan yang lebih proaktif terhadap ancaman yang terus berkembang.

PENDAHULUAN

Keamanan siber telah menjadi isu krusial seiring meningkatnya ketergantungan pada teknologi digital. Website, sebagai pintu masuk utama transaksi online, sering menjadi target serangan siber [20]. Dengan pesatnya pertumbuhan penggunaan internet dan aplikasi berbasis web, ancaman terhadap keamanan informasi semakin meningkat. Website, sebagai pintu masuk utama untuk transaksi dan interaksi online, sering kali menjadi target serangan siber. Oleh karena itu, untuk menghadapi ancaman ini, diperlukan pemahaman tentang jenis-jenis serangan dan strategi keamanan yang efektif [12].

Penelitian ini fokus pada penerapan model RNNs dalam mendeteksi anomali jaringan pada website berbasis localhost. Lingkungan localhost digunakan sebagai tahap awal untuk pengembangan dan pengujian model sebelum diterapkan secara lebih luas. Dengan memanfaatkan model-model ini dapat diimplementasikan menggunakan TensorFlow.js, model deep learning dapat diintegrasikan langsung ke dalam sistem website yang berbasis JavaScript, sehingga memungkinkan deteksi anomali dilakukan secara real-time dan lebih efisien [18].

Seiring dengan semakin rumitnya serangan siber dan semakin banyaknya data yang dihasilkan oleh perangkat jaringan, diharapkan penerapan sistem monitoring berbasis RNNs dapat menjadi solusi yang sangat diperlukan. Sistem ini diharapkan mampu memberikan alert secara real-time ketika pola-pola abnormal terdeteksi, sehingga memungkinkan tim keamanan untuk segera mengambil tindakan pencegahan. Implementasi Deep Learning berbasis RNNs dalam jaringan keamanan siber diharapkan dapat memberikan lompatan besar dalam mendeteksi dan penanganan serangan siber yang semakin canggih.

https://doi.org/10.56211/blendsains.v3i4.800

TINJAUAN PUSTAKA

Penelitian terbaru menunjukkan efektivitas *deep learning*, khususnya *Recurrent Neural Networks* (RNNs), dalam mendeteksi anomali dan gangguan jaringan. RNNs, termasuk varian seperti LSTM dan GRU, telah menunjukkan harapan dalam menganalisis data berurutan untuk mengidentifikasi berbagai serangan seperti DoS, DDoS, dan infiltrasi jaringan [19]. Model-model ini telah mencapai akurasi yang tinggi dalam tugas klasifikasi biner dan multi-kelas di berbagai *dataset* jaringan IoT [19]. Eksperimen menggunakan dataset saat ini seperti CSE-CIC-IDS 2018 telah menunjukkan bahwa model berbasis RNNs dapat mencapai akurasi lebih dari 98% dalam klasifikasi serangan multi-kelas [19]. Penerapan RNNs pada jaringan IoT yang kompleks juga memberikan hasil yang menjanjikan, dengan akurasi mencapai 87% [21]. Namun, masih ada tantangan dalam mengimplementasikan model-model ini dalam sistem deteksi intrusi waktu nyata, dan penelitian lebih lanjut diperlukan untuk mengatasi masalah-masalah seperti kompleksitas model dan waktu inferensi [13].

Penelitian terbaru telah mengeksplorasi berbagai teknik untuk meningkatkan kinerja sistem deteksi intrusi (IDS) dalam keamanan jaringan. Metode pra pemrosesan seperti ekstraksi fitur dan normalisasi data telah terbukti meningkatkan akurasi model *Recurrent Neural Network* (RNNs) [17]. Teknik rekayasa fitur, seperti seleksi fitur berbasis *XGBoost*, telah digunakan untuk mengurangi dimensi fitur dan fokus pada atribut yang relevan, yang mengarah pada peningkatan kinerja model [18]. Para peneliti juga telah menggabungkan RNNs dengan metode lain untuk meningkatkan akurasi deteksi. Sebagai contoh, pendekatan hibrida yang mengintegrasikan RNNs dengan teknik pengelompokan atau model berbasis graf telah diusulkan untuk menganalisis hubungan antar-simpul dalam jaringan [13]. Kombinasi ini memungkinkan deteksi anomali yang lebih komprehensif, dengan mempertimbangkan lalu lintas data dan perspektif struktur jaringan. Berbagai arsitektur RNNs, termasuk LSTM, GRU, dan BiLSTM, telah dievaluasi pada *dataset benchmark*, yang menunjukkan akurasi tinggi dalam mendeteksi gangguan dan anomali jaringan [13].

Penerapan Deteksi Anomali Jaringan dengan RNNs pada Website Berbasis Localhost

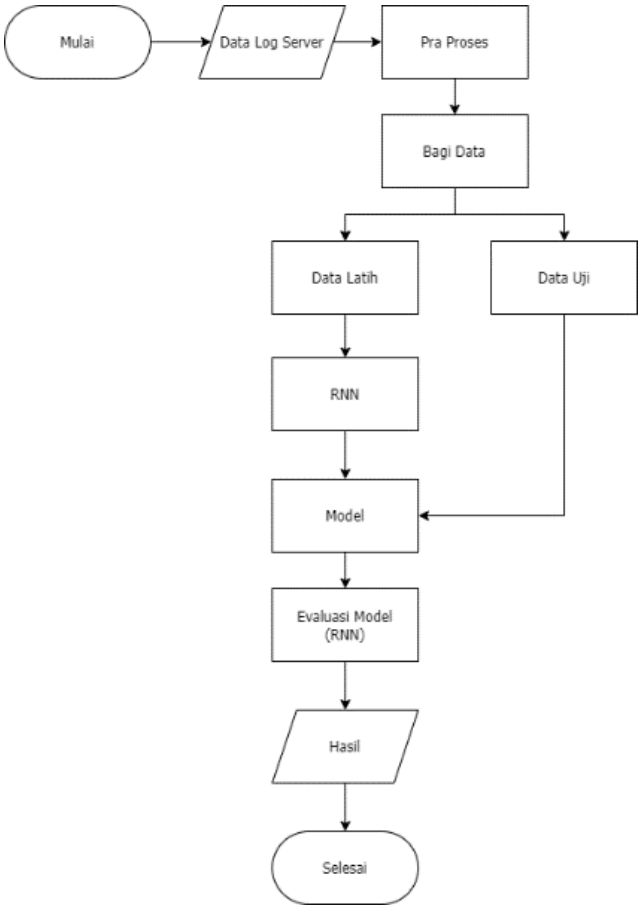
Implementasi RNNs untuk deteksi anomali pada website berbasis *localhost* berfokus pada pengawasan dan deteksi dini terhadap aktivitas mencurigakan. Dengan pengaturan *localhost*, uji coba dilakukan dalam lingkungan yang lebih terkontrol, memungkinkan analisis *real-time* terhadap pola akses pengguna. RNNs berfungsi untuk memantau data akses, mengidentifikasi anomali, dan mendeteksi potensi serangan di tingkat aplikasi website berbasis *localhost*, memberikan lapisan keamanan yang lebih dinamis dan adaptif.

Penelitian terbaru telah mengeksplorasi penggunaan Recurrent Neural Networks (RNNs) untuk deteksi anomali dalam sistem jaringan. [18] mengimplementasikan kerangka kerja Intrusion Detection System (IDS) dengan menggunakan berbagai jenis RNNs, yang mencapai akurasi tinggi pada set data benchmark. Demikian pula, [13] mengusulkan model berbasis RNNs untuk mendeteksi anomali di jaringan IoT, menunjukkan kinerja yang lebih unggul dibandingkan dengan implementasi yang ada. Sementara penelitian ini berfokus pada keamanan jaringan secara umum, [4] secara khusus membahas deteksi serangan dini di situs web menggunakan metode berbasis anomali, yang secara efektif mengidentifikasi perilaku pengunjung yang tidak biasa. Untuk lingkungan jaringan lokal, [17] menggunakan algoritma K-Nearest Neighbor untuk mengklasifikasikan serangan jaringan berdasarkan pola lalu lintas dan frekuensi koneksi. Penelitian-penelitian ini secara kolektif menyoroti potensi teknik pembelajaran mesin, khususnya RNNs, dalam meningkatkan keamanan jaringan melalui pemantauan waktu nyata dan deteksi ancaman adaptif.

METODOLOGI

Penelitian ini menggunakan data berupa referensi dari internet dan penelitian terdahulu yang akan mendukung tentang penerapan metode *Recurrent Neural Networks* dalam membangun website deteksi anomali.

Prosedur Penelitian



Gambar 1. Prosedur Penelitian

1. Data *log server*:
   a. Data yang dikumpulkan yaitu data empiris berupa *log* aktivitas dari *website* berbasis *localhost*. Data ini bersifat primer karena langsung dihasilkan oleh sistem yang berjumlah lebih dari 100 data.
   Contoh data dalam format *log*:
   127.0.0.1 - - [20/Nov/2024:14:23:15 +0000] "POST /login HTTP/1.1" 200 - "Mozilla/5.0 (Windows NT 10.0; Win64)"
   127.0.0.1 - - [20/Nov/2024:14:23:17 +0000] "POST /register HTTP/1.1" 200 - " Mozilla/5.0 (Windows NT 10.0; Win64)"
   127.0.0.1 - - [20/Nov/2024:14:23:19 +0000] "GET /dsasd HTTP/1.1" 404 - " Mozilla/5.0 (Windows NT 10.0; Win64)"
   127.0.0.1 - - [20/Nov/2024:14:23:20 +0000] "GET /task HTTP/1.1" 200 - "Mozilla/5.0 (Windows NT 10.0; Win64)"
   127.0.0.1 - - [20/Nov/2024:14:23:22 +0000] "GET /task-detail HTTP/1.1" 200 - " Mozilla/5.0 (Windows NT 10.0; Win64)"
   b. Data yang dikumpulkan dengan menggunakan studi literatur berupa informasi dari sumber-sumber yang sudah ada, seperti jurnal ilmiah, yang bertujuan untuk teori, konsep dan temuan sebelumnya yang relevan.
   c. Tujuan: Mengidentifikasi dan merekam aktivitas lalu lintas pada *website* berbasis *localhost*. Menyediakan *data log* yang diperlukan untuk analisis dan pelatihan model deteksi anomali. Mengumpulkan *data log* dari *website* yang berjalan di *localhost* dan menyimpan hasil monitoring ke dalam *database* untuk pra-pemrosesan data.
   d. Jenis data yang dikumpulkan: Metode HTTP (*GET, POST, PUT, PACTH, DELETE*), status kode *response* (200, 201, 204, 403, 404, 500, dan lain-lain). Alamat IP pengakses, waktu akses (*timestamp*), *user-agent* (*browser* atau aplikasi yang digunakan).
   Berikut adalah daftar kode status HTTP/HTTPS secara lengkap, mencakup semua kategori:
      1. 2xx (*Successful Responses*) Menunjukkan bahwa permintaan klien berhasil diproses oleh server.
         a. 200 OK: Permintaan berhasil dan hasilnya dikirimkan.
         b. 201 *Created*: Permintaan berhasil, dan sumber daya baru telah dibuat.
         c. 202 *Accepted*: Permintaan diterima untuk diproses, tetapi belum selesai.

 https://doi.org/10.56211/blendsains.v3i4.800

2. 4xx (*Client Error Responses*) Menunjukkan bahwa ada kesalahan pada permintaan yang dikirim oleh klien.
   a. 400 *Bad Request*: Permintaan tidak valid atau tidak dapat dipahami oleh server.
   b. 401 *Unauthorized*: Otentikasi diperlukan dan gagal atau belum diberikan.
   c. 403 *Forbidden*: Akses ke sumber daya ditolak.
   d. 404 *Not Found*: Sumber daya yang diminta tidak ditemukan di server.
   e. 405 *Method Not Allowed*: Metode HTTP yang digunakan tidak diizinkan untuk sumber daya ini.
3. 5xx (*Server Error Responses*) Menunjukkan bahwa server gagal memproses permintaan yang valid.
   a. 500 *Internal Server Error*: Kesalahan umum di sisi server.
   b. 501 *Not Implemented*: Metode yang diminta belum diimplementasikan oleh server.
   c. 502 *Bad Gateway*: Server bertindak sebagai gateway dan menerima respons yang tidak valid dari server upstream.
   d. 503 *Service Unavailable*: Server tidak dapat menangani permintaan (biasanya karena kelebihan beban atau perawatan).
   e. 504 *Gateway Timeout*: Server tidak menerima respons tepat waktu dari server upstream.

2. Penyimpanan data: menyimpan *data log* ke dalam file (misalnya *.log* atau *.json*) atau langsung ke *database* relasional/ non-relasional yaitu, *MySQL*.

Tabel 1. Contoh hasil pengumpulan data dari website *localhost*.

Timestamp	***Source IP***	***URL/ Endpoint***	***Request Type***	***Response Code***	***User Agent***
2024-11-20 14:23:15	127.0.0.1	/login	POST	200	Mozilla/5.0 (Windows NT 10.0; Win64)
2024-11-20 14:23:17	127.0.0.1	/register	POST	403	Mozilla/5.0 (Windows NT 10.0; Win64)
2024-11-20 14:23:19	127.0.0.1	/dsasd	GET	404	Mozilla/5.0 (Windows NT 10.0; Win64)
2024-11-20 14:23:20	127.0.0.1	/task	GET	200	Mozilla/5.0 (Windows NT 10.0; Win64)
2024-11-20 14:23:22	127.0.0.1	/task-detail	GET	200	Mozilla/5.0 (Windows NT 10.0; Win64)

Penjelasan:
1. *Source IP* (127.0.0.1): Semua permintaan berasal dari *localhost* (IP 127.0.0.1), karena aplikasi diakses secara lokal.
2. *URL/ Endpoint*: *Endpoint-endpoint* yang diminta dalam aplikasi.
3. *Request Type*: Metode HTTP yang digunakan (*GET, POST, PUT, PACTH, DELETE*).
4. *Response Code*: Kode status HTTP dari server (contoh: 200 = OK, 403 = *Forbidden*).
5. *User Agen*t: Atribut untuk mengidentifikasi browser atau alat yang digunakan (misalnya, `curl` mencurigakan karena sering digunakan untuk *scraping* atau *testing* otomatis).

1. Pra-proses:
   a. Normalisasi dan *encoding* untuk mengkonversikan atribut non-numerik seperti alamat IP dan URL ke bentuk numerik.

Tabel 2. Contoh data hasil normalisasi dan *encoding*.

Timestamp	***Source IP***	***URL/ Endpoint***	***Request Type***	***Response Code***	***User Agent***	***Label***
1	1	1	2	200	1	0
2	1	2	2	403	1	1
3	1	3	1	404	1	1
4	1	4	1	200	1	0
5	1	5	1	200	1	0

Penjelasan:
1. *Source IP*: `127.0.0.1` = 1.
2. *URL/ Endpoint*:

a. `/login` = 1
b. `/register` = 2
c. `/dsasd` = 3
d. dll.

3. *Request Type*:
   a. *GET* = 1
   b. *POST* = 2
   c. *PUT* = 3
   d. *PACTH* = 4
   e. *DELETE* = 5
4. User Agent: `Mozilla/5.0 (Windows NT 10.0; Win64)` = 1.
5. Label: 0 = Normal, 1 = Anomali.

b. Membagi *data log* menjadi *sequences* berdasarkan *timestamp* atau kejadian tertentu.

Tabel 3. Contoh pembagian data berdasarkan *sequences* satu.

Timestamp	***Source IP***	***URL/ Endpoint***	***Request Type***	***Response Code***	***User Agent***	***Label***
1	1	1	2	200	1	0
2	1	2	2	403	1	1
3	1	3	1	404	1	1

Tabel 4. Contoh pembagian data berdasarkan *sequences* dua.

Timestamp	***Source IP***	***URL/ Endpoint***	***Request Type***	***Response Code***	***User Agent***	***Label***
4	1	4	1	200	1	0
5	1	5	1	200	1	0

c. Melabeli data mana yang tergolong anomali, jika terdapat data historis yang sudah diberi label, digunakan sebagai pembelajaran (*supervised learning*).

Tabel 5. *Sequences* satu.

Feature Vector	**Label**
[1, 1, 2, 200, 1]	0
[1, 2, 2, 403, 1]	1
[1, 3, 1, 404, 1]	1

Tabel 6. *Sequences* dua.

Feature Vector	**Label**
[1, 4, 1, 200, 1]	0
[1, 5, 1, 200, 1]	0

2. Bagi Data:
   Memisahkan data menjadi data latih, dan data uji.
   a. Data latih.
      Data latih diambil sebanyak 70% dari hasil pra-proses, ini digunakan untuk pelatihan model.
   b. Data uji.
      Data uji diambil sebanyak 30% dari hasil pra-proses, ini digunakan untuk pengujian model.
3. RNNs (*Recurrent Neural Networks*):
   RNNs dilatih menggunakan data latih untuk mengenali pola dalam data yang jarang terjadi dalam *data log* dengan bantuan *TensorFlow.js*.
4. Model:
   Setelah pelatihan selesai, model diuji menggunakan data uji dan data latih untuk mengukur kinerjanya.
5. Evaluasi Model:
   Pengujian model dengan data uji dan data latih untuk menilai akurasi dalam mendeteksi anomali dengan metrik evaluasi seperti *accuracy*, *precision*, *recall*, dan *F1 score*.

 https://doi.org/10.56211/blendsains.v3i4.800

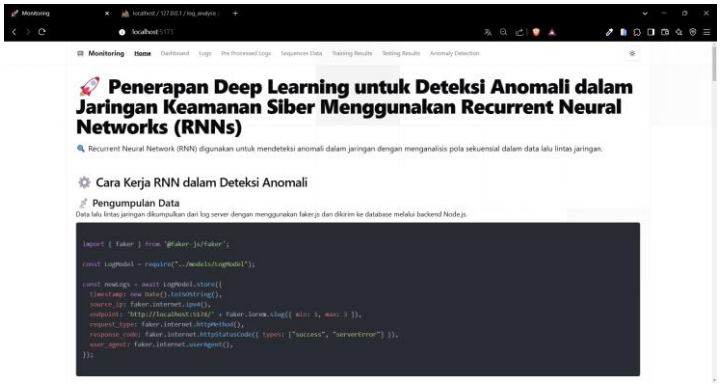
6. Hasil: Hasil evaluasi dianalisis untuk memahami kinerja model dan apakah model dapat digunakan untuk deteksi anomali.

HASIL DAN PEMBAHASAN

Implementasi sistem berbasis website *localhost* ini menggunakan bahasa pemrograman Javascript, dan Python serta MySQL sebagai *database*. Javascript digunakan sebagai tampilan untuk pengguna dan komunikasi dengan server. Python digunakan untuk pra proses, bagi data sequence, training, testing, dan deteksi anomali. MySQL berguna untuk menyimpan data dan pengembangan lebih lanjut.

Halaman Beranda

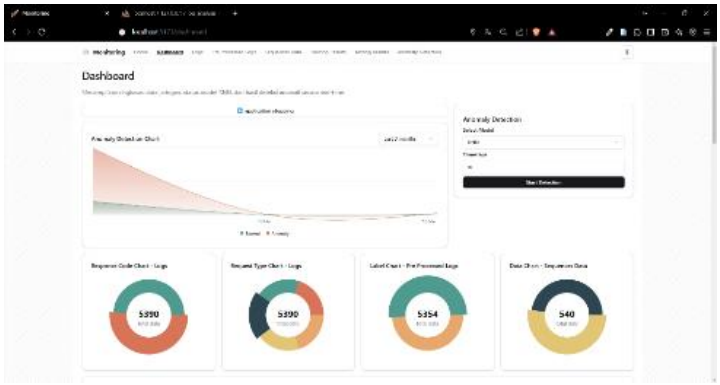
Halaman ini menampilkan informasi tentang cara kerja sistem dan teknologi yang digunakan serta *source code*.



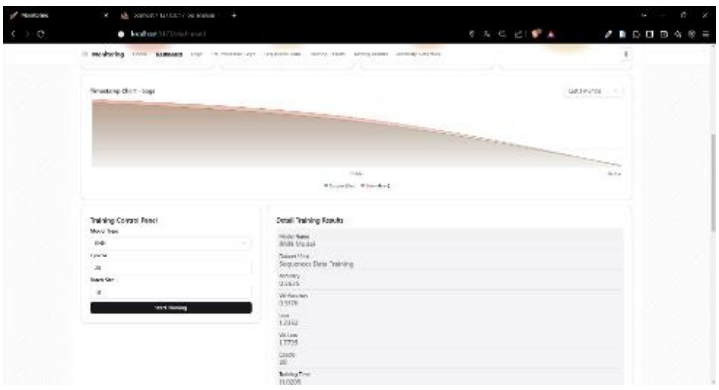
Gambar 2. Halaman Beranda

Halaman Dashboard

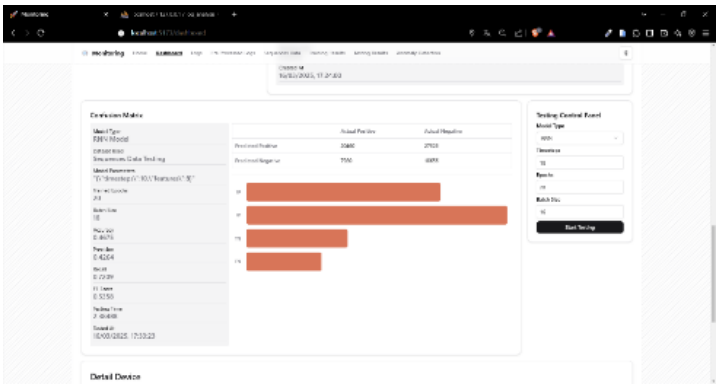
Pada halaman dashboard, publik dapat melihat bagaimana proses pengumpulan data dilakukan. Menampilkan beberapa grafik untuk melihat pembagian data secara serta control panel pelatihan, pengujian serta deteksi anomali secara real-time. Grafik lainnya juga menampilkan jumlah data yang dikumpulkan sepanjang hari dengan rentang waktu tertentu. Pada bagian akhir dari dashboard, terdapat informasi dari pengguna yang mengakses website ini, seperti nama peramban, lokasi dan juga ip address.



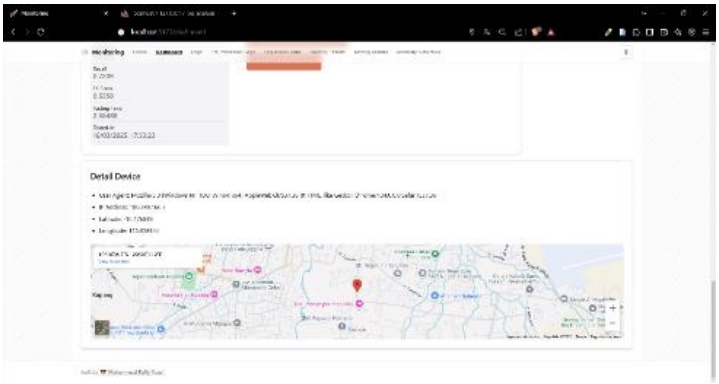
Gambar 3. Halaman *dashboard*



Gambar 4. Tampilan grafik *timestamps* dan *Training Control Panel*



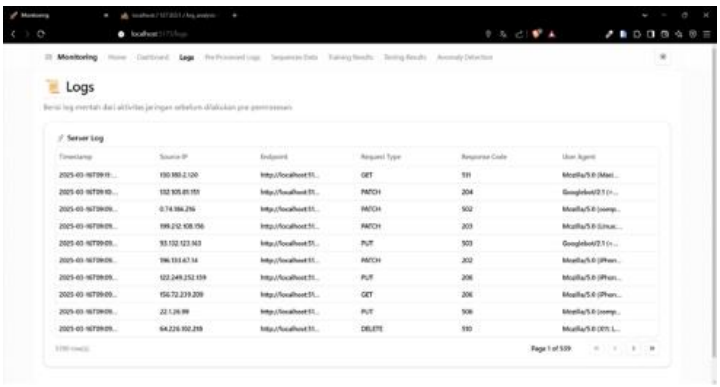
Gambar 5. Tampilan *Confusion Matrix* dan *Testing Control Panel*



Gambar 6. *Detail Device*

Halaman Logs

Pada halaman ini, menampilkan tabel dari data logs yang dikumpulkan dari log server, serta kolom-kolom pada tabel logs.

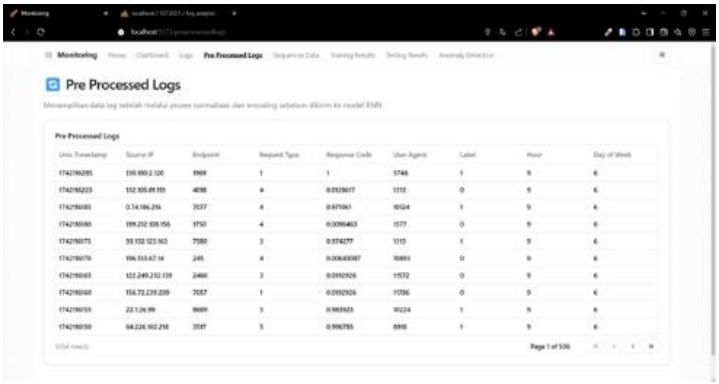


Gambar 7. Halaman *logs*

 https://doi.org/10.56211/blendsains.v3i4.800

Halaman Preprocessed Logs

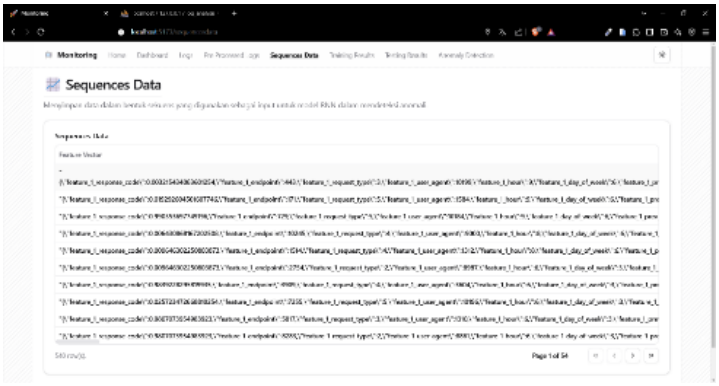
Pada halaman ini, menampilkan tabel dari data *preprocossed logs* yang telah dilakukan normalisasi dan *encoding* dari data *logs*. Data ini dihasilkan oleh yang telah dibuat dan dilatih sebelumnya untuk menghasilkan data yang dapat digunakan dalam pelatihan dan pengujian model.



Gambar 8. Halaman *preprocossed logs*

Halaman Sequences Data

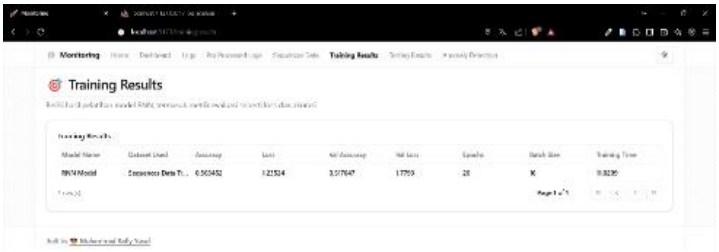
Pada halaman ini, menampilkan tabel dari data sequences data yang telah dibagi berdasarkan waktu dari data preprocessed logs, data kemudian diberi label 0 atau 1 sesuai dengan seberapa banyak label 0 atau 1 dari satu sequences. Kemudian data diberi nama secara acak, yaitu training atau testing.



Gambar 9. Halaman *sequences data*

Halaman Training Results

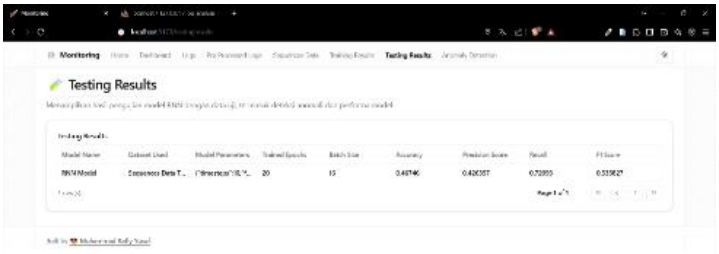
Pada halaman ini, menampilkan tabel dari data training results yang mengambil data sebanyak 70% data dari sequences data dengan nama data training. Data kemudian dilakukan pelatihan berdasarkan model yang dipilih serta beberapa inputan lainnya guna meningkatkan akurasi model.



Gambar 10. Halaman *training results*

Halaman Testing Results

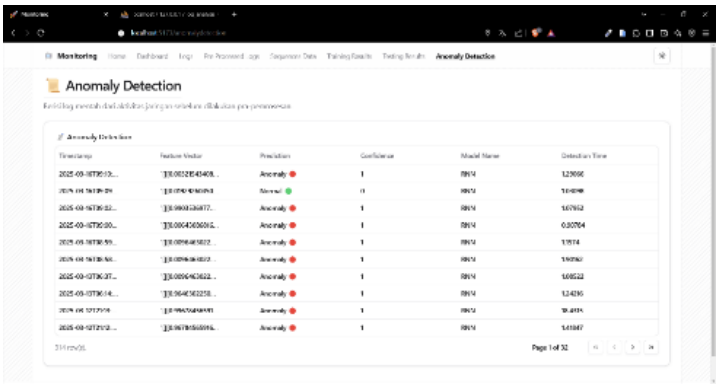
Pada halaman ini, menampilkan tabel dari data *testing results* yang mengambil data sebanyak 30% data dari *sequences data* dengan nama data *testing*. Data kemudian dilakukan pengujian berdasarkan model yang dipilih serta beberapa inputan lainnya guna melihat hasil dan mengevaluasi dari model yang dipilih.



Gambar 11. Halaman *testing results*

Halaman Anomaly Detection

Pada halaman ini, menampilkan tabel dari data *anomaly detection* yang telah diberi label 0 sama dengan normal, 1 sama dengan anomali, dan juga menampilkan nama model yang dipakai dalam mendeteksi anomali.

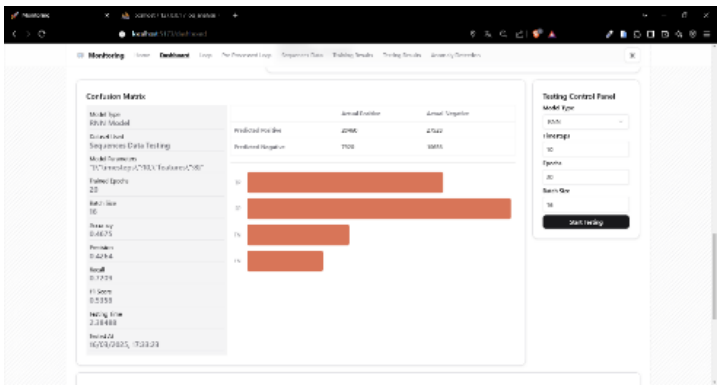


Gambar 12. Halaman *anomaly detection*

Pengujian sistem adalah proses untuk mengevaluasi hasil dari kemampuan model dalam mendeteksi anomali pada jaringan. Beberapa model lainnya seperti LTSM dan GRU juga digunakan dalam pengujian ini. Pengujian ini menampilkan *confusion matrix* untuk melihat seberapa akurat model dalam mendeteksi anomali pada jaringan.
Output yang dihasilkan dari pengujian menggunakan model RNN, sebagai berikut:

"model_name": "RNN Model",
"dataset_used": "Sequences Data Testing",
"model_parameters": "{\"timesteps\":10,\"features\":8}",
"trained_epochs": 20,
"batch_size": 16,
"accuracy": 0.46746,
"precision_score": 0.426357,
"recall": 0.72093,
"f1_score": 0.535827,
"tp": 20460,
"fp": 27528,
"tn": 10656,
"fn": 7920,
"testing_time": 2.38488,
"tested_at": "2025-03-16T09:33:23.000Z"

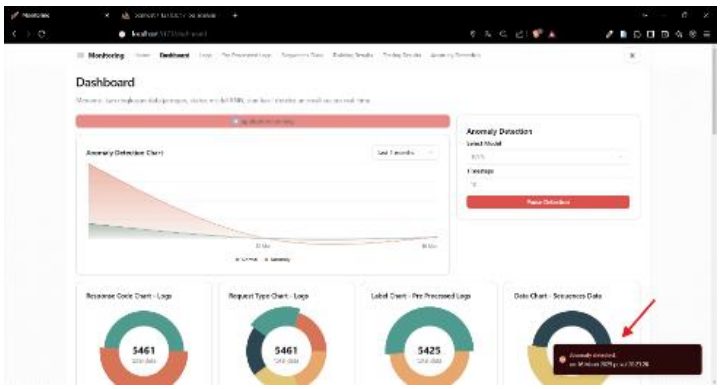
Output dalam bentuk *confusion matrix*:



Gambar 13. Tampilan *confusion matrix*

Dari tampilan ini kita bisa melihat bagaimana model melakukan pengujian, hasil akurasi menunjukkan model telah mempelajari data dan mengklasifikasikan anomali dan normal dengan baik.

Pada tampilan berikutnya terdapat notifikasi yang ditampilkan jika terdapat anomali pada jaringan *server*.



Gambar 14. Tampilan notifikasi jika terdapat anomali

KESIMPULAN DAN SARAN

Berdasarkan hasil penelitian ini, dapat disimpulkan bahwa penerapan *Recurrent Neural Networks* (RNN) untuk deteksi anomali pada lalu lintas jaringan terbukti efektif dan memberikan performa yang superior. Model RNN yang dilatih berhasil mengekstraksi pola temporal dari data sekuensial, menghasilkan akurasi klasifikasi yang tinggi dan meminimalkan tingkat *false positive* dalam mengidentifikasi anomali siber. Validasi implementasi operasional melalui integrasi ke dalam platform berbasis web yang beroperasi secara *real-time* lebih lanjut menegaskan kapabilitas model dalam beradaptasi dengan dinamika lalu lintas jaringan aktual dan memberikan dasar untuk pemantauan proaktif. Secara akademik, studi ini menunjukkan bahwa RNN merupakan arsitektur jaringan saraf yang kuat dan relevan untuk mengatasi kompleksitas deteksi anomali siber di lingkungan jaringan yang terus berkembang, serta menyajikan prototipe sistem yang memiliki potensi signifikan untuk meningkatkan postur keamanan siber kontemporer.

DAFTAR PUSTAKA

[1] A. R. N. Nisa, A. D. Wijayanto, A. P. J. Priana, dan A. Setiawan, “Analisis Log Server untuk mendeteksi Serang DDoS pada Keamanan Jaringan di Website,” *Journal of Internet and Software Engineering*, vol. 1, no. 3, pp. 1–17, 2024.

[2] J. E. D. A. Albuquerque Filho *et al.*, “Review of Neural Networks for Anomaly Detection,” *IEEE Access*, vol. 10, pp. 112345–112347, 2022. doi: 10.1109/ACCESS.2022.3216007.

[3] C. D. Berliana, T. A. Saputra, dan I. Gunawan, “Analisis Serangan dan Keamanan pada Denial of Service (DOS): Sebuah Review Sistematik,” *Jurnal Ilmiah Informatika dan Komputer*, vol. 1, no. 2, pp. 33–38, 2022. doi: 10.51901/jiifkom.v1i2.229.

[4] Fariadi dan M. R. R. Islami, “Deteksi Dini Serangan Pada Website Menggunakan Metode Anomali Based,” *Jurnal Informatika dan Komputer*, vol. 5, no. 3, pp. 224–229, 2022. doi: 10.33387/jiko.

[5] Y. Fu, "Computer Network Intrusion Anomaly Detection with Recurrent Neural Network," *Mobile Information Systems*, vol. 2022, Art. no. 6576023.

[6] S. Gautam *et al.*, "A Composite Approach of Intrusion Detection Systems: Hybrid RNN and Correlation-Based Feature Optimization," *Electronics*, vol. 11, no. 3529, p. 1, 2022. doi: 10.3390/electronics11213529.

[7] H. Hindy *et al.*, "Machine learning based IoT intrusion detection system: An MQTT case study (MQTT-IoT-IDS2020 Dataset)," in *Advances in Intelligent Systems and Computing*, 2020.

[8] Isnawaty, L. I. Uzlah, dan R. A. Saputra, "Deteksi Serangan Siber Pada Jaringan Komputer Menggunakan Metode Random Forest," *Jurnal Mahasiswa Teknik Informatika*, vol. 8, no. 3, p. 2787, 2024. doi: 10.36040/jati.v8i3.8891.

[9] H. Kang *et al.*, "IoT network intrusion dataset," *IEEE Dataport*, Tech. Rep., 2020. doi: 10.21227/q70p-q449.

[10] P. Krishnegowda *et al.*, "Enhancing Intrusion Detection Systems with Recurrent Neural Networks," *International Journal of Innovative Research in Engineering*, vol. 2024, pp. 209–216. doi: 10.59256/ijire.20240503026.

[11] S. J. Kumaresan *et al.*, "Investigating the Effectiveness of Recurrent Neural Networks for Network Anomaly Detection," in *2024 International Conference on Intelligent and Innovative Technologies in Computing, Electrical and Electronics (IITCEE)*, pp. 1–10. doi: 10.1109/IITCEE59897.2024.10467790.

[12] T. G. Laksana dan S. Mulyani, "Pengetahuan Dasar Identifikasi Dini Deteksi Serangan Kejahatan Siber Untuk Mencegah Pembobolan Data Perusahaan," *Jurnal Ilmiah Multidisiplin*, vol. 3, no. 1, pp. 109–122, 2024. doi: 10.56127/jukim.v3i01.1143

[13] Q. H. Mahmoud dan I. Ullah, "Design and Development of RNN-based Anomaly Detection Model for IoT Networks," *Journal of Big Data*, vol. 10, no. 1, pp. 1–16, 2022. doi: 10.1186/s40537-021-00448-4

[14] N. Mamuriyah, S. E. Prasetyo, dan A. O. Sijabat, "Rancangan Sistem Keamanan Jaringan dari serangan DDoS Menggunakan Metode Pengujian Penetrasi," *Jurnal Teknologi Dan Sistem Informasi Bisnis*, vol. 6, no. 1, pp. 162–167, 2024.

[15] S. Pandey, "Advanced Abnormality Recognition in IoT Networks Using LSTM-RNNs for Dynamic Security Enhancement," in *IEEE Xplore*, 2023, p. 35. doi: 10.1109/ICCSAI59793.2023.10420960.

[16] A. Parmisano, S. Garcia, dan M. J. Erquiaga, "IoT-23: A labeled dataset with malicious and benign IoT network traffic," Stratos Lab, Czech Republic, Tech. Rep., 2020. doi: 10.5281/zenodo.4743746.

[17] D. S. Prasetyo, K. Auliasari, dan M. R. P. Syalabi, "Klasifikasi Serangan Jaringan Menggunakan Metode K-Nearest Neighbour pada Data Riwayat Jaringan," in *Seminar Nasional Sinergitas Era Digital 5.0*, ITN Malang, 9 Desember 2023.

[18] F. Sjafrina, N. A. Pratama, dan P. D. Arnesia, "Aplikasi Artificial Intelligence Untuk Mendeteksi Objek Berbasis Web Menggunakan Library Tensorflow Js, React Js Dan Coco Dataset," *Jurnal Sistem Informasi*, vol. 9, no. 1, pp. 62–69, 2022. doi: 10.30656/jsii.v9i1.4243.

[19] I. Ullah dan Q. H. Mahmoud, "A scheme for generating a dataset for anomalous activity detection in IoT networks," *Advances in Artificial Intelligence*, vol. 12109, pp. 508–520, 2020. doi: 10.1007/978-3-030-47358-7_52.

[20] A. Wijaya dan T. Sutabri, "Mendesain Cyber Security Untuk Keamanan Website Menggunakan Web Aplication Firewall Pada Kantor BKPSDM Ogan Ilir," *Blantika: Multidisciplinary Journal*, vol. 2, no. 4, p. 1, 2024. doi: 10.57096/blantika.v2i4.121.

[21] E. A. Winanto, Kurniabudi, dan Sharipuddin, "Deteksi Serangan pada Jaringan Kompleks IoT menggunakan Recurrent Neural Network," *JURIKOM (Jurnal Riset Komputer)*, vol. 9, no. 6, pp. 1996–2002, 2022. doi: 10.30865/jurikom.v9i6.5298.