Seminar Nasional Teknologi dan Inovasi (SITASI)
Vol 1, Nomor 2, Juni 2026, hal. 336-345
P-ISSN: XXXX-XXXX | E-ISSN: XXXX- XXXX
Penerbit Institut Bisnis dan Teknologi Indonesia

Analisis dan Evaluasi Model LSTM untuk Klasifikasi Serangan Botnet pada Dataset CTU-13

I Nyoman Buda Hartawan[1*], Ni Wayan Suardiati Putri[2], Evi Dwi Krisna[3], I Kadek Garry Hartawan[4]

[1,2,3] Institut Bisnis dan Teknologi Indonesia, Denpasar, Indonesia*
*[1 *]buda.hartawan@instiki.ac.id*

INFO ARTIKEL

Article history:
Received Juni 2026
Published Juli 2026

ABSTRAK

Serangan botnet merupakan ancaman serius terhadap keamanan jaringan karena dapat mengganggu layanan, menyalahgunakan sumber daya, dan mendukung aktivitas berbahaya. Penelitian ini menganalisis penerapan Long Short-Term Memory (LSTM) untuk mendeteksi serangan botnet menggunakan dataset CTU-13, khususnya skenario 9-Neris-20110817.binetflow.parquet. Dataset yang digunakan berisi fitur aliran jaringan, seperti durasi koneksi, protokol, arah komunikasi, status koneksi, jumlah paket, total byte, dan byte dari sumber. Proses klasifikasi dilakukan secara biner untuk membedakan trafik Normal dan Attack. Penelitian ini menguji empat skenario time step, yaitu 5, 10, 20, dan 30 time steps. Oversampling diterapkan pada level sequence untuk menyeimbangkan jumlah sampel pada setiap kelas. Hasil penelitian menunjukkan bahwa penggunaan time step yang lebih panjang dapat meningkatkan performa model. Hasil terbaik diperoleh pada time step 30 dengan accuracy 0,9980, precision 0,9960, recall 1,0000, F1-score 0,9980, dan ROC-AUC 0,9997.

Kata kunci: LSTM, deteksi botnet, CTU-13, aliran jaringan, oversampling

ABSTRACT

Botnet attacks are a serious threat to network security because they can disrupt services, abuse resources, and support malicious activities. This study analyzes the application of Long Short-Term Memory (LSTM) for botnet attack detection using the CTU-13 dataset, specifically the 9-Neris-20110817.binetflow.parquet scenario. The dataset contains network flow features, including connection duration, protocol, communication direction, connection state, number of packets, total bytes, and source bytes. The classification task was designed as a binary classification problem to distinguish Normal and Attack traffic. This study tested four time step scenarios, namely 5, 10, 20, and 30 time steps. Oversampling was applied at the sequence level to balance the number of samples in each class. The results show that longer time steps improve model performance. The best result was obtained using 30 time steps, achieving 0.9980 accuracy, 0.9960 precision, 1.0000 recall, 0.9980 F1-score, and 0.9997 ROC-AUC.

Keywords: LSTM, botnet detection, CTU-13, network flow, oversampling.

1. Pendahuluan

Perkembangan teknologi informasi dan komunikasi telah membawa kemudahan dalam berbagai aktivitas digital, namun pada saat yang sama juga meningkatkan risiko terhadap keamanan jaringan. Serangan siber menjadi salah satu ancaman serius karena dapat menyebabkan gangguan layanan, pencurian data, kerugian finansial, hingga kerusakan reputasi organisasi. Berbagai bentuk serangan seperti malware, ransomware, phishing, denial-of-service, dan botnet semakin berkembang dengan pola yang kompleks serta sulit dideteksi menggunakan metode keamanan konvensional. Oleh karena itu, diperlukan sistem deteksi serangan yang mampu mengenali pola ancaman secara cepat, akurat, dan adaptif terhadap perkembangan serangan siber.

Salah satu ancaman siber yang perlu mendapat perhatian adalah serangan botnet. Botnet merupakan jaringan perangkat yang telah terinfeksi malware dan dikendalikan oleh penyerang untuk menjalankan aktivitas berbahaya, seperti pencurian data, penyebaran spam, serangan Distributed Denial of Service, maupun eksploitasi terhadap sistem lain. Karakteristik serangan botnet biasanya muncul dalam bentuk pola lalu lintas jaringan yang berulang, bertahap, dan memiliki hubungan waktu antaraktivitas. Hal ini menjadikan deteksi botnet tidak cukup hanya dilakukan berdasarkan ciri statis, tetapi juga perlu memperhatikan urutan dan pola perilaku jaringan dari waktu ke waktu.

Metode deteksi tradisional sering kali memiliki keterbatasan dalam menghadapi volume data jaringan yang besar dan pola serangan yang terus berubah. Pendekatan berbasis aturan atau signature hanya efektif untuk mendeteksi serangan yang sudah dikenal, tetapi kurang mampu mengenali pola serangan baru atau variasi serangan yang belum pernah ditemukan sebelumnya. Untuk mengatasi permasalahan tersebut, pembelajaran mesin dan deep learning mulai banyak digunakan dalam bidang keamanan siber karena mampu belajar dari data, mengenali pola tersembunyi, serta meningkatkan kemampuan klasifikasi terhadap aktivitas normal dan aktivitas serangan.

Salah satu metode deep learning yang sesuai untuk menganalisis data berurutan adalah Long Short-Term Memory (LSTM). LSTM merupakan pengembangan dari Recurrent Neural Network yang dirancang untuk mengatasi permasalahan ketergantungan jangka panjang pada data sekuensial. Dalam konteks deteksi serangan botnet, LSTM memiliki kemampuan untuk mempelajari pola lalu lintas jaringan berdasarkan urutan waktu, sehingga dapat mengenali hubungan antaraktivitas jaringan yang mengindikasikan adanya serangan. Kemampuan ini menjadikan LSTM relevan untuk digunakan dalam analisis data jaringan, terutama ketika pola serangan tidak muncul secara langsung pada satu data tunggal, melainkan terbentuk dari rangkaian aktivitas yang saling berkaitan.

Untuk menguji kemampuan model LSTM dalam mendeteksi serangan botnet, diperlukan dataset yang representatif. Salah satu dataset yang banyak digunakan dalam penelitian keamanan jaringan adalah CTU-13. Dataset ini berisi rekaman lalu lintas jaringan yang mencakup aktivitas normal, aktivitas botnet, dan aktivitas latar belakang. CTU-13 menyediakan data dalam bentuk packet capture dan NetFlow yang memungkinkan peneliti menganalisis pola komunikasi antarperangkat dalam jaringan. Keberadaan data botnet dan data normal dalam satu lingkungan jaringan menjadikan CTU-13 sebagai dataset yang sesuai untuk mengembangkan dan mengevaluasi model deteksi serangan botnet.

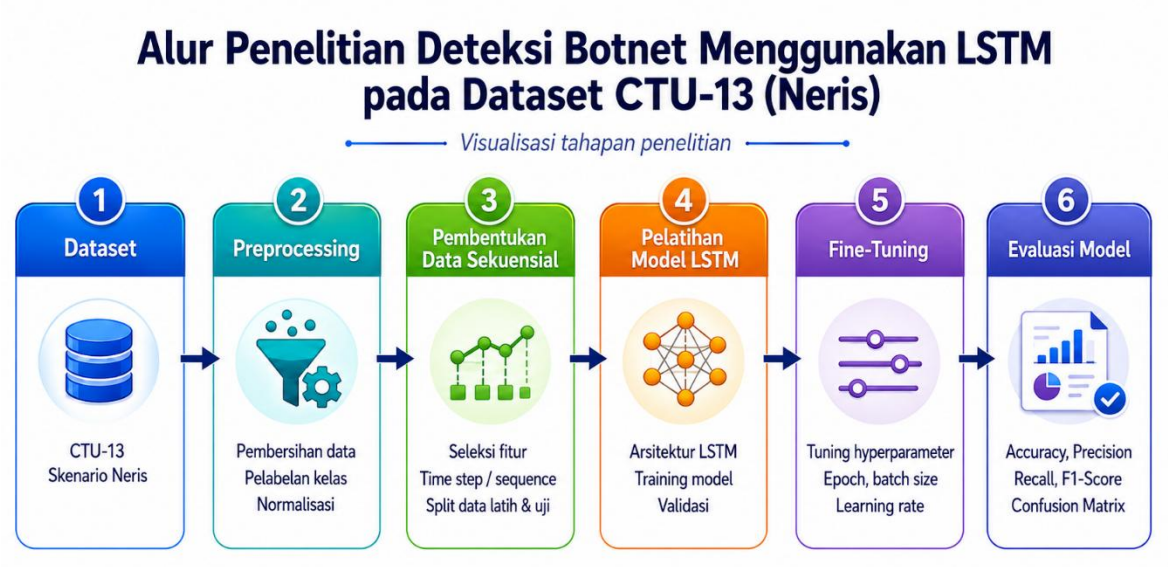
Namun, penerapan LSTM dalam deteksi botnet masih memerlukan analisis lebih lanjut, khususnya terkait bagaimana model mampu mempelajari pola sekuensial pada data CTU-13. Kinerja LSTM dapat dipengaruhi oleh beberapa faktor, seperti pemilihan fitur, proses preprocessing, pembentukan data sekuensial, pembagian data latih dan data uji, serta panjang time step yang digunakan dalam pelatihan model. Pemilihan skenario eksperimen yang tepat diperlukan agar model tidak hanya menghasilkan akurasi tinggi pada data pelatihan, tetapi juga mampu melakukan generalisasi terhadap data pengujian.

Berdasarkan permasalahan tersebut, penelitian ini bertujuan untuk menganalisis kinerja model Long Short-Term Memory dalam mendeteksi serangan botnet menggunakan dataset CTU-13. Penelitian ini diharapkan dapat memberikan gambaran mengenai efektivitas LSTM dalam mengenali pola serangan botnet berbasis data sekuensial. Selain itu, hasil penelitian ini juga diharapkan dapat menjadi kontribusi

dalam pengembangan sistem deteksi serangan siber berbasis deep learning yang lebih akurat dan adaptif terhadap pola serangan yang terus berkembang.

2. Metode Penelitian

Gambar 1 menunjukkan alur penelitian deteksi serangan botnet menggunakan metode Long Short-Term Memory (LSTM) pada dataset CTU-13 skenario Neris. Alur penelitian terdiri atas enam tahapan utama, yaitu pengumpulan dataset, preprocessing, pembentukan data sekuensial, pelatihan model LSTM, fine-tuning, dan evaluasi model..



Gambar 1 Alur Penelitian

a) Pemilihan Dataset: CTU-13

Tahap pertama adalah penggunaan dataset CTU-13 skenario Neris sebagai sumber data utama. Dataset ini berisi data lalu lintas jaringan yang mencakup aktivitas normal, background traffic, dan aktivitas botnet. Skenario Neris dipilih karena merepresentasikan pola serangan botnet yang dapat dianalisis melalui karakteristik komunikasi jaringan, sehingga sesuai untuk pengujian model deteksi berbasis deep learning.

b) Preprocessing Data

Pada tahap ini dilakukan pembersihan data untuk menghilangkan data yang tidak relevan, menangani nilai kosong atau data yang tidak konsisten, serta menyiapkan data agar dapat diproses oleh model. Selain itu, dilakukan pelabelan kelas untuk membedakan aktivitas normal dan aktivitas botnet. Proses normalisasi juga dilakukan agar setiap fitur berada pada skala nilai yang seragam, sehingga dapat membantu model LSTM belajar secara lebih stabil dan optimal.

c) Ekstraksi Fitur

Tahap ini menjadi bagian penting karena LSTM bekerja dengan data yang memiliki urutan waktu. Data yang telah melalui preprocessing kemudian disusun ke dalam bentuk sequence berdasarkan panjang time step tertentu. Selain itu, dilakukan seleksi fitur untuk memilih atribut yang relevan terhadap proses deteksi serangan botnet. Data kemudian dibagi menjadi data latih dan data uji untuk mendukung proses pelatihan dan pengujian model.

d) Pelatihan Model LSTM

Pada tahap ini, arsitektur LSTM dibangun untuk mempelajari pola sekuensial dari lalu lintas jaringan. Model dilatih menggunakan data latih agar mampu mengenali pola aktivitas normal dan pola aktivitas botnet. Proses validasi dilakukan selama pelatihan untuk memantau kemampuan model dalam melakukan generalisasi terhadap data yang tidak digunakan secara langsung pada proses training.

e) Fine Tuning

Tahap ini dilakukan untuk memperoleh konfigurasi model yang lebih optimal melalui pengaturan hyperparameter, seperti jumlah epoch, batch size, learning rate, jumlah unit LSTM, dan parameter lain yang berpengaruh terhadap performa model. Fine-tuning bertujuan untuk meningkatkan kemampuan model dalam mendeteksi serangan botnet serta mengurangi kemungkinan terjadinya overfitting maupun underfitting.

f) Evaluasi Model

Tahap terakhir adalah evaluasi model. Model yang telah dilatih dan dioptimalkan kemudian diuji menggunakan data uji. Evaluasi dilakukan dengan menggunakan beberapa metrik, yaitu accuracy, precision, recall, F1-score, dan confusion matrix. Accuracy digunakan untuk mengukur tingkat ketepatan klasifikasi secara keseluruhan. Precision menunjukkan ketepatan model dalam memprediksi data sebagai botnet. Recall menunjukkan kemampuan model dalam menemukan seluruh data botnet yang sebenarnya. F1-score digunakan untuk melihat keseimbangan antara precision dan recall, sedangkan confusion matrix digunakan untuk menganalisis hasil klasifikasi secara lebih rinci.

3. Hasil dan Pembahasan

3.1. Hasil Preprocessing Dataset

Penelitian ini menggunakan dataset CTU-13 pada skenario 9-Neris-20110817.binetflow.parquet. Dataset ini berisi data aliran jaringan atau network flow yang merepresentasikan aktivitas komunikasi jaringan. Fitur yang digunakan dalam penelitian meliputi dur, proto, dir, state, stos, dtos, tot_pkts, tot_bytes, dan src_bytes. Fitur-fitur tersebut dipilih karena merepresentasikan karakteristik dasar trafik jaringan, seperti durasi koneksi, protokol, arah komunikasi, status koneksi, jumlah paket, jumlah byte, dan jumlah byte dari sumber.

Tahap preprocessing dilakukan dengan membaca dataset secara bertahap menggunakan mekanisme batch. Pendekatan ini digunakan karena ukuran dataset cukup besar sehingga pembacaan seluruh data secara langsung dapat menyebabkan penggunaan RAM yang tinggi. Selanjutnya, nilai kosong dan nilai tidak valid pada fitur ditangani dengan mengganti nilai hilang pada fitur numerik menjadi 0, sedangkan fitur kategorikal dikonversi menjadi string agar dapat diproses lebih lanjut. Kolom kategorikal seperti proto, dir, dan state kemudian diubah menjadi bentuk numerik menggunakan one-hot encoding.

Label data dikonversi menjadi dua kelas, yaitu Normal dan Attack. Kelas Attack diberikan pada data yang terindikasi sebagai trafik botnet, sedangkan kelas Normal diberikan pada trafik normal, background, benign, atau non-botnet. Dengan demikian, tugas klasifikasi dalam penelitian ini adalah klasifikasi biner untuk membedakan trafik botnet dan trafik non-botnet.

Setelah proses encoding, fitur dinormalisasi menggunakan StandardScaler. Normalisasi ini penting karena setiap fitur memiliki rentang nilai yang berbeda. Sebagai contoh, fitur tot_bytes dan src_bytes dapat memiliki nilai yang jauh lebih besar dibandingkan fitur lain. Jika tidak dilakukan normalisasi, model dapat lebih banyak dipengaruhi oleh fitur yang memiliki skala besar.

3.2. Pembentukan Data Sekuensial untuk LSTM

Karena model yang digunakan adalah Long Short-Term Memory (LSTM)**,** data perlu dibentuk menjadi format sekuensial. LSTM dirancang untuk mempelajari pola dari data yang memiliki urutan. Oleh karena itu, data network flow tidak hanya diperlakukan sebagai baris data tunggal, tetapi dibentuk menjadi beberapa urutan flow berdasarkan nilai time step**.**

Dalam penelitian ini digunakan empat skenario time step, yaitu:

Tabel 1 Skenario Pembentukan Data Sekuensial

Skenario	Time Step
S1	5
S2	10

S3	20
S4	30

Pada skenario S1, satu input LSTM terdiri dari 5 flow jaringan berurutan. Pada skenario S2 terdiri dari 10 flow, S3 terdiri dari 20 flow, dan S4 terdiri dari 30 flow. Label sequence ditentukan berdasarkan label pada flow terakhir dari setiap sequence. Pendekatan ini bertujuan agar model dapat mempelajari pola temporal dari aktivitas jaringan, bukan hanya karakteristik satu flow secara individual.

3.3. Hasil Eksperimen dengan Oversampling

Pada eksperimen dengan oversampling, data dibuat seimbang pada level sequence. Artinya, jumlah sequence untuk kelas Normal dan Attack dibuat setara. Teknik ini dilakukan karena distribusi data awal tidak seimbang, sehingga model berpotensi lebih banyak belajar dari kelas mayoritas.

Berdasarkan hasil eksperimen, diperoleh tabel perbandingan performa sebagai berikut:

Tabel 2 Hasil eksperimen dengan oversampling

Skenario	Time Step	Accuracy	Precision	Recall	F1-Score	ROC-AUC
S1	5	0.8273	0.7560	0.9667	0.8484	0.9371
S2	10	0.9127	0.8513	1.0000	0.9197	0.9759
S3	20	0.9953	0.9908	1.0000	0.9954	0.9984
S4	30	0.9980	0.9960	1.0000	0.9980	0.9997

Hasil tersebut menunjukkan bahwa peningkatan panjang time step memberikan pengaruh positif terhadap performa model LSTM. Skenario S1 dengan time step 5 menghasilkan accuracy sebesar 0,8273 dan F1-score sebesar 0,8484**.** Nilai ini menunjukkan bahwa model sudah mampu mengenali pola trafik botnet, tetapi konteks urutan yang digunakan masih relatif pendek.

Pada skenario S2 dengan time step 10, performa model meningkat secara signifikan. Accuracy meningkat menjadi 0,9127, precision menjadi 0,8513, recall mencapai 1,0000**,** dan F1-score meningkat menjadi 0,9197. Hal ini menunjukkan bahwa penambahan panjang sequence dari 5 menjadi 10 flow memberikan konteks temporal yang lebih baik bagi model.

Peningkatan performa yang lebih besar terlihat pada skenario S3 dengan time step 20. Model memperoleh accuracy sebesar 0,9953, precision sebesar 0,9908, recall sebesar 1,0000, F1-score sebesar 0,9954, dan ROC-AUC sebesar 0,9984. Nilai ini menunjukkan bahwa model hampir sempurna dalam membedakan trafik Normal dan Attack pada data uji yang telah diseimbangkan.

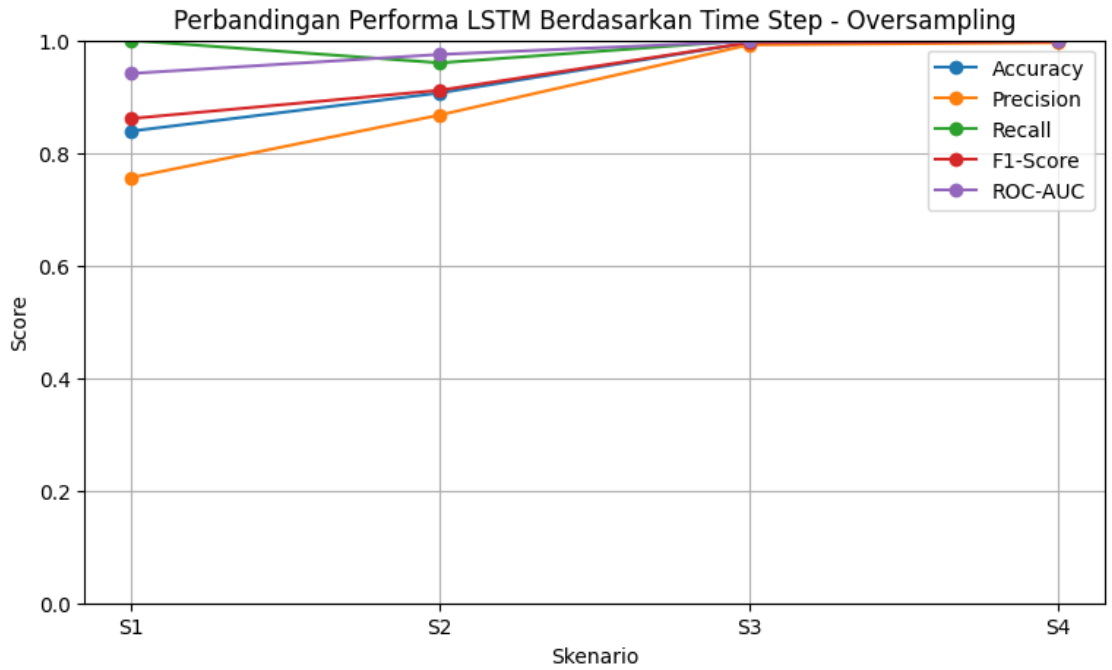
Skenario terbaik diperoleh pada S4 dengan time step 30. Pada skenario ini, model mencapai accuracy sebesar 0,9980**,** precision sebesar 0,9960, recall sebesar 1,0000, F1-score sebesar 0,9980, dan ROC-AUC sebesar 0,9997. Hasil ini menunjukkan bahwa semakin panjang sequence yang diberikan kepada LSTM, semakin baik kemampuan model dalam mengenali pola trafik botnet.

3.4. Pembahasan Pengaruh Time Step terhadap Performa LSTM

Hasil eksperimen menunjukkan bahwa panjang time step memiliki pengaruh yang signifikan terhadap performa model LSTM. Skenario dengan time step pendek, yaitu S1, menghasilkan performa yang lebih rendah dibandingkan skenario dengan time step yang lebih panjang. Hal ini dapat dijelaskan karena botnet tidak selalu terlihat dari satu flow atau sejumlah kecil flow, tetapi sering kali muncul sebagai pola aktivitas jaringan yang berulang dan berurutan.

Pada time step 5, model hanya menerima konteks lima flow berurutan. Jumlah ini kemungkinan belum cukup untuk merepresentasikan pola komunikasi botnet secara utuh. Akibatnya, meskipun recall sudah tinggi, precision masih lebih rendah dibandingkan skenario lain. Precision yang lebih rendah menunjukkan bahwa masih terdapat data Normal yang salah diklasifikasikan sebagai Attack.

Ketika time step dinaikkan menjadi 10, 20, dan 30, model memperoleh konteks urutan trafik yang lebih panjang. Hal ini membantu LSTM dalam mengenali hubungan antar-flow, seperti pola durasi koneksi, jumlah paket, arah komunikasi, dan volume byte yang muncul secara berulang. Semakin panjang sequence, semakin besar peluang model menangkap pola temporal yang menjadi ciri aktivitas botnet.



Gambar 2 Perbandingan Performa LSTM berdasarkan Time Step

Skenario S3 dan S4 menunjukkan performa yang sangat tinggi. Hal ini mengindikasikan bahwa sequence dengan panjang 20 hingga 30 flow sudah cukup kuat untuk merepresentasikan pola serangan botnet pada skenario Neris. Dengan demikian, LSTM lebih efektif ketika diberi input berupa urutan flow yang cukup panjang dibandingkan hanya menggunakan sequence pendek.

3.5. Perbandingan F1-Score Berdasarkan Time Step

Grafik tersebut menunjukkan perbandingan nilai F1-Score model LSTM pada empat skenario time step, yaitu S1, S2, S3, dan S4.

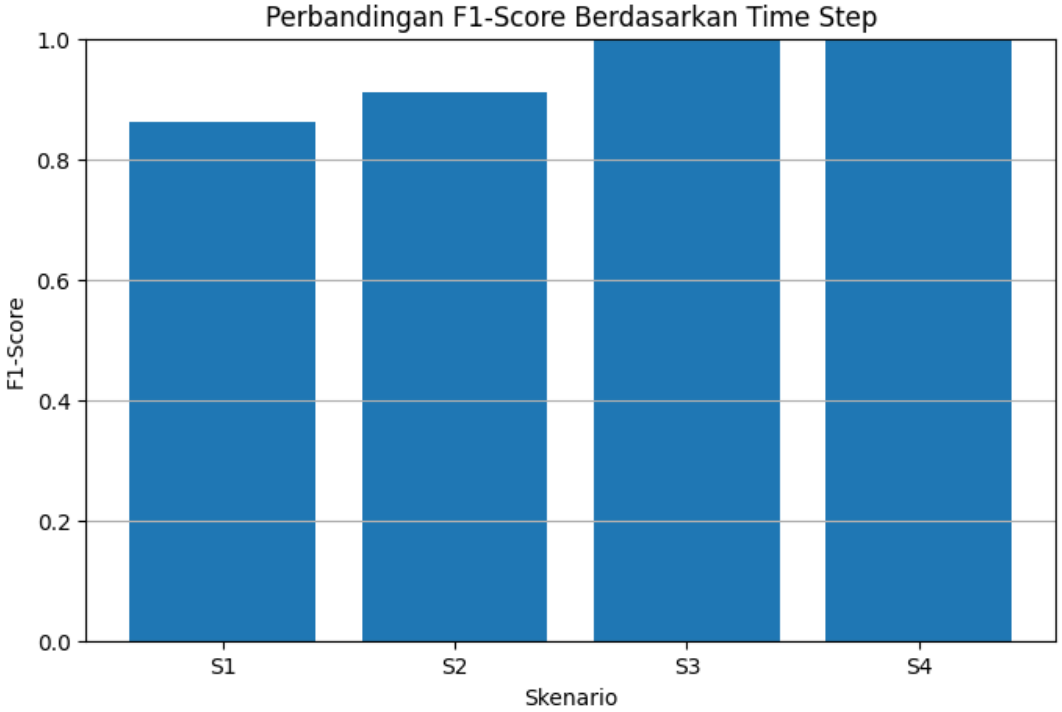
Secara umum, grafik memperlihatkan bahwa semakin panjang time step yang digunakan, nilai F1-Score semakin meningkat. Hal ini menunjukkan bahwa model LSTM semakin baik dalam menyeimbangkan kemampuan klasifikasi antara trafik Normal dan Attack.

Pada skenario S1, nilai F1-Score berada sekitar 0,85. Nilai ini menunjukkan bahwa penggunaan 5 flow berurutan sebagai input sudah mampu menghasilkan performa yang cukup baik, tetapi masih terdapat kesalahan klasifikasi. Dengan kata lain, konteks urutan 5 flow belum sepenuhnya cukup untuk menangkap pola trafik botnet secara optimal.

Pada skenario S2, nilai F1-Score meningkat menjadi sekitar 0,92. Peningkatan ini menunjukkan bahwa penggunaan 10 time steps memberikan informasi temporal yang lebih baik dibandingkan 5 time steps. Urutan flow yang lebih panjang membantu model mengenali pola serangan dengan lebih baik.

Pada skenario S3, nilai F1-Score meningkat sangat tinggi dan mendekati 1,00. Hal ini menunjukkan bahwa time step 20 sudah mampu memberikan konteks urutan trafik yang kuat bagi LSTM dalam membedakan trafik Normal dan Attack.

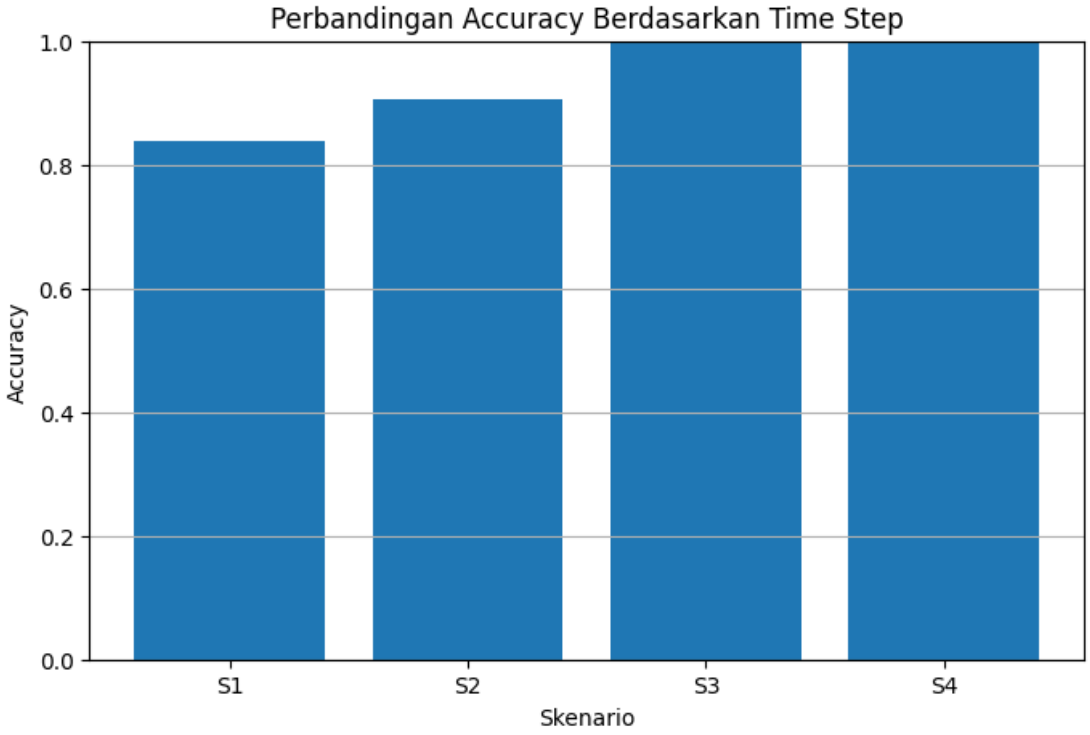
Skenario S4 juga menunjukkan F1-Score yang sangat tinggi dan mendekati 1,00. Hasil ini menunjukkan bahwa penggunaan 30 time steps memberikan performa terbaik atau hampir sama dengan S3. Dengan sequence yang lebih panjang, model memiliki lebih banyak informasi untuk mempelajari pola aktivitas botnet.



Gambar 3 Perbandingan F1-Score Berdasarkan Time Step

3.6. Perbandingan Accuracy berdasarkan Time Step

Grafik tersebut menunjukkan perbandingan nilai accuracy model LSTM pada empat skenario time step, yaitu S1, S2, S3, dan S4. Accuracy digunakan untuk mengukur proporsi prediksi yang benar dari seluruh data uji. Semakin tinggi nilai accuracy, semakin besar kemampuan model dalam mengklasifikasikan trafik jaringan secara tepat.



Gambar 4 Perbandingan Accuracy Berdasarkan Time Step

Berdasarkan grafik, terlihat bahwa nilai accuracy meningkat seiring bertambahnya panjang time step. Pada skenario S1 dengan 5 time steps, model memperoleh accuracy sekitar 0,83. Hasil ini menunjukkan bahwa penggunaan 5 flow berurutan sudah mampu memberikan informasi awal bagi model LSTM untuk membedakan trafik Normal dan Attack, tetapi performanya belum optimal.

Pada skenario S2 dengan 10 time steps, accuracy meningkat menjadi sekitar 0,91. Peningkatan ini menunjukkan bahwa penambahan panjang urutan flow memberikan konteks temporal yang lebih baik bagi model. Dengan jumlah flow yang lebih panjang, LSTM dapat mempelajari pola hubungan antar-flow secara lebih jelas.

Skenario S3 dengan 20 time steps menunjukkan peningkatan accuracy yang sangat tinggi, yaitu mendekati 1,00. Hal ini mengindikasikan bahwa urutan 20 flow sudah cukup kuat untuk merepresentasikan pola trafik botnet pada dataset yang digunakan. Model mampu mengenali karakteristik trafik dengan tingkat kesalahan yang sangat kecil.

Pada skenario S4 dengan 30 time steps, accuracy juga mencapai nilai yang sangat tinggi dan mendekati 1,00. Hasil ini menunjukkan bahwa penggunaan sequence yang lebih panjang memberikan informasi temporal yang lebih lengkap kepada model LSTM. Dengan demikian, model dapat membedakan trafik Normal dan Attack secara lebih akurat.

Secara keseluruhan, grafik ini menunjukkan bahwa panjang time step berpengaruh terhadap performa model LSTM. Semakin panjang sequence yang digunakan, semakin tinggi accuracy yang diperoleh. Hal ini mendukung bahwa deteksi serangan botnet tidak hanya bergantung pada satu flow jaringan, tetapi juga pada pola urutan beberapa flow secara berkelanjutan. Oleh karena itu, penggunaan time step yang lebih panjang, khususnya S3 dan S4, lebih efektif dalam meningkatkan kemampuan model LSTM untuk mendeteksi serangan botnet pada dataset CTU-13.

4. Kesimpulan

Berdasarkan hasil penelitian, model Long Short-Term Memory (LSTM) mampu digunakan untuk mendeteksi serangan botnet pada dataset CTU-13 skenario 9-Neris-20110817.binetflow.parquet dengan memanfaatkan data flow jaringan yang dibentuk ke dalam beberapa skenario time step. Hasil eksperimen menunjukkan bahwa panjang time step berpengaruh terhadap performa model, di mana semakin panjang urutan flow yang digunakan, semakin baik kemampuan LSTM dalam mengenali pola trafik botnet. Skenario S1 dengan 5 time steps menghasilkan accuracy sebesar 0,8273 dan F1-Score sebesar 0,8484, sedangkan skenario S4 dengan 30 time steps memberikan performa terbaik dengan accuracy sebesar 0,9980, precision sebesar 0,9960, recall sebesar 1,0000, F1-Score sebesar 0,9980, dan ROC-AUC sebesar 0,9997. Penerapan oversampling pada level sequence juga membantu menyeimbangkan data Normal dan Attack sehingga model tidak bias terhadap kelas mayoritas. Dengan demikian, kombinasi LSTM, pembentukan sequence berdasarkan time step, dan oversampling efektif digunakan untuk meningkatkan kinerja deteksi serangan botnet berbasis data flow pada dataset CTU-13.

5. Ucapan Terima Kasih

Dengan ini, kami mengucapkan terima kasih yang sebesar-besarnya kepada Institut Bisnis dan Teknologi Indonesia, melalui Direktoran Riset dan Pengabdian Masyarakat (DRPM) yang telah memberikan dukungan dan fasilitasi yang luar biasa dalam penelitian ini. Bantuan yang diberikan, baik berupa sumber daya, fasilitas, maupun bimbingan, sangat berharga dan menjadi faktor penting dalam kelancaran dan kesuksesan penelitian ini.

Daftar Pustaka

Abu Talib, M., Nasir, Q., Bou Nassif, A., Mokhamed, T., Ahmed, N., & Mahfood, B. (2022). APT beaconing detection: A systematic review. *Computers and Security*, *122*. https://doi.org/10.1016/j.cose.2022.102875

Alabdulatif, A., Thilakarathne, N. N., & Aashiq, M. (2024). Machine Learning Enabled Novel Real-Time IoT Targeted DoS/DDoS Cyber Attack Detection System. *Computers, Materials and Continua*, *80*(3), 3655–3683. https://doi.org/10.32604/cmc.2024.054610

Bartwal, U., Mukhopadhyay, S., Negi, R., & Shukla, S. (2022). Security Orchestration, Automation, and Response Engine for Deployment of Behavioural Honeypots. *5th IEEE Conference on Dependable and Secure Computing, DSC 2022 and SECSOC 2022 Workshop, PASS4IoT 2022 Workshop SICSA International Paper/Poster Competition in Cybersecurity*, 1–8. https://doi.org/10.1109/DSC54232.2022.9888808

Chen, M., Zhu, K., Lu, B., Li, D., Yuan, Q., & Zhu, Y. (2025). AECR: Automatic attack technique intelligence extraction based on fine-tuned large language model. *Computers and Security*, *150*. https://doi.org/10.1016/j.cose.2024.104213

Chen, Y., Cui, M., Wang, D., Cao, Y., Yang, P., Jiang, B., Lu, Z., & Liu, B. (2024). A survey of large language models for cyber threat detection. *Computers and Security*, *145*. https://doi.org/10.1016/j.cose.2024.104016

Firdaus, R., Hadiana, A. I., & Kasyidi, F. (2023). Model Deteksi Botnet Menggunakan Algoritma Decision Tree Dengan Untuk Mengidentifikasi Serangan Click Fraud. *Journal of Informatics and Communication Technology (JICT)*, *4*(2), 10–20. https://doi.org/10.52661/j_ict.v4i2.122

Irshad, E., & Basit Siddiqui, A. (2023). Cyber threat attribution using unstructured reports in cyber threat intelligence. *Egyptian Informatics Journal*, *24*(1), 43–59. https://doi.org/10.1016/j.eij.2022.11.001

Mishra, C., Sarma, H., & M., S. (2025). PageLLM: Incremental approach for updating a Security Knowledge Graph by using Page ranking and Large language model. *Information Processing and Management*, *62*(3). https://doi.org/10.1016/j.ipm.2024.104045

Ng, C. K., Rajasegarar, S., Pan, L., & ... (2020). VoterChoice: A ransomware detection honeypot with multiple voting framework. *Concurrency and …*. https://doi.org/10.1002/cpe.5726

Pérez-Gomariz, M., Cerdán-Cartagena, F., & García, J. (2025). LM-Hunter: An NLP-powered graph method for detecting adversary lateral movements in APT cyber-attacks at scale. *Computer Networks*, *262*. https://doi.org/10.1016/j.comnet.2025.111181

Putra, M. A. R., Hostiadi, D. P., & Ahmad, T. (2022). Botnet dataset with simultaneous attack activity. *Data in Brief*, *45*, 108628. https://doi.org/10.1016/j.dib.2022.108628

Putri, V. U., Cahyono, E. B., & Azhar, Y. (2020). Deteksi Botnet Pada Passive DNS Dengan Menggunakan Metode K Nearest Neighbor. *Jurnal Repositor*, *2*(12), 1631. https://doi.org/10.22219/repositor.v2i12.450

Rahali, A., & Akhloufi, M. A. (2023). MalBERTv2: Code Aware BERT-Based Model for Malware Identification. *Big Data and Cognitive Computing*, *7*(2). https://doi.org/10.3390/bdcc7020060

Udiyono, A., Lim, C., & Lukas. (2020). Botnet Detection Using DNS and HTTP Traffic Analysis. *ACM International Conference Proceeding Series*. https://doi.org/10.1145/3429789.3429818

Vellela, S. S., D, R., Purimetla, N. M. R., Thalakola, S. R., Vuyyuru, L. R., & Vatambeti, R. (2025). Cyber threat detection in industry 4.0: Leveraging GloVe and self-attention mechanisms in BiLSTM for enhanced intrusion detection. *Computers and Electrical Engineering*, *124*. https://doi.org/10.1016/j.compeleceng.2025.110368

Zhen, Z., & Gao, J. (2023). Chinese Cyber Threat Intelligence Named Entity Recognition via RoBERTa-wwm-RDCNN-CRF. *Computers, Materials and Continua*, *77*(1), 299–321. https://doi.org/10.32604/cmc.2023.042090