Jurnal Penelitian Teknologi Informasi Dan Sains
Vol. 2 No. 2 Juni 2024
e-ISSN :2985-7635, p-ISSN :2985-6280, Hal 84-99
DOI: https://doi.org/10.54066/jptis.v2i2.1931

Klasifikasi Malware Menggunakan Metode Convolutional Neural Network (CNN) Berbasis Website

Septian Dwi Chandra
Universitas Muhammadiyah Jember

Hardian Oktavianto
Universitas Muhammadiyah Jember

Ari Eko Wardoyo
Universitas Muhammadiyah Jember

Alamat : Gumuk Kerang, Karangrejo, Sumbersari, Kabupaten Jember, Jawa Timur 68124
Korespondensi penulis: septiandwichandra7270@gmail.com

Abstract

This study aims to develop a web-based malware detection system using Convolutional Neural Network (CNN) utilizing the IoT23 dataset. Malware is malicious software that can exploit security vulnerabilities in computer systems, steal data, and degrade performance. The implementation of this detection system involves CNN, capable of extracting important features from both visual and textual data, applied to malware classification. The IoT23 dataset comprises 23 scenarios of IoT network traffic, including traffic from malware-infected devices. The study results show that the developed web application can detect malware attacks with accuracy, precision, recall, and F1-score of 99% on separate data scenarios. This CNN-based detection system has proven effective in identifying and classifying malware attacks, contributing to the enhancement of network and device security.

Keywords*: Malware, Convolutional Neural Network (CNN), IoT23 dataset, web-based malware detection, deep learning.*

Abstrak

Penelitian ini bertujuan untuk mengembangkan sistem deteksi *malware* berbasis web menggunakan *Convolutional Neural Network* (CNN) dengan memanfaatkan dataset IoT23. *Malware* merupakan perangkat lunak berbahaya yang dapat mengeksploitasi celah keamanan pada sistem komputer, mencuri data, dan menurunkan kinerja. Implementasi sistem deteksi ini melibatkan CNN yang mampu mengekstraksi fitur penting dari data visual maupun tekstual, diaplikasikan pada klasifikasi *malware*. Dataset IoT23 terdiri dari 23 skenario lalu lintas jaringan IoT, termasuk lalu lintas dari perangkat terinfeksi *malware*. Hasil penelitian menunjukkan bahwa aplikasi web yang dikembangkan mampu mendeteksi serangan *malware* dengan akurasi, presisi, *recall*, dan *f1-score* masing-masing sebesar 99% pada skenario data terpisah. Sistem deteksi berbasis CNN ini terbukti efektif dalam mengidentifikasi dan mengklasifikasikan serangan *malware*, berkontribusi pada peningkatan keamanan jaringan dan perangkat.

Kata kunci: *Malware*, *Convolutional Neural Network* (CNN), IoT23 dataset, deteksi *malware* berbasis web, *deep learning*.

LATAR BELAKANG

Malware adalah perangkat lunak yang diciptakan dengan tujuan tertentu, dimana penyerang mencari celah keamanan dalam sistem. Dampak buruk dari *Malware* dapat

Received April 30, 2024; Accepted Mei 27, 2024; Published Juni 30, 2024
** Septian Dwi Chandra, septiandwichandra7270@gmail.com*

dirasakan pada komputer atau pengguna, karena penyerang dapat mencuri informasi atau data pribadi seseorang. (Adenansi & Novarina, 2017) Tujuan pembuatan *Malware* oleh penyerang dapat bervariasi, termasuk merusak atau membobol suatu sistem operasi melalui skrip rahasia, dan dapat diinisiasi secara tersembunyi.

Implementasi sistem deteksi berbasis web memungkinkan pemantauan aktif terhadap aktivitas dan aliran data. Pembuatan sistem deteksi ini melibatkan penerapan deep learning, yang dapat dijelaskan sebagai model jaringan yang terdiri dari sejumlah lapisan. Menurut penelitian yang dilakukan oleh (Kusumaningrum, 2018), CNN merupakan salah satu algoritma yang berasal dari Deep learning, yang merupakan pengembangan dari Multi Layer Perceptron (MLP). Convolutional Neural Network (CNN) adalah jenis arsitektur jaringan saraf buatan yang secara khusus dirancang untuk menangani pengolahan data grid, seperti gambar atau data visual.

Penelitian ini memiliki tujuan untuk mengembangkan sebuah sistem deteksi Malware berbasis web dengan menggunakan dataset IoT23, yang telah dikembangkan oleh (Garcia dkk., 2020). Dataset ini terdiri dari 23 rekaman atau skenario yang mencakup berbagai lalu lintas jaringan IoT. Sebuah penelitian sebelumnya yang dilakukan oleh (Liang & Vankayalapati, 2021) menggunakan dataset yang sama, menerapkan metode Convolutional Neural Network (CNN) yang menghasilkan akurasi sebesar 69%, Decision Tree sebesar 73%, SVM 69%, dan Naïve Bayes sebesar 30%. Dalam konteks penelitian ini, sebuah upaya akan dilakukan untuk membuat sebuah aplikasi berbasis web yang mendeteksi serangan-serangan *Malware*.

KAJIAN TEORITIS

1. ***Website***

   website merupakan suatu koleksi halaman web yang mengandung teks, gambar, suara, dan berbagai jenis informasi lainnya. Website berfungsi sebagai layanan internet yang menghubungkan dokumen lokal dan jarak jauh, dan tautan pada website memungkinkan pengguna untuk berpindah antar halaman dengan format hypertext.

2. ***Flask***

   Flask merupakan sebuah module dari python berupa web framework. Flask digunakan untuk mengembangkan sebuah aplikasi web (Python, 2021). Framework ini dirancang untuk menjadi ringan, mudah digunakan, dan fleksibel, memungkinkan pengembang untuk membangun aplikasi web dengan cepat.

3. Malware

Malware menurut (Akhtar & Feng, 2022) merupakan sebuah program atau sekumpulan instruksi yang diciptakan untuk mencelakai komputer, pengguna, bisnis, atau sistem komputer.

4. Machine learning

Machine learning menurut (Suprayogi & Marwan, 2022), merujuk pada seperangkat teknik yang digunakan untuk mengelola dan memprediksi data besar melalui penerapan algoritma pembelajaran.

5. Penelitan Terdahulu

Penelitian ini menggunakan beberapa penelitian terdahulu sebagai referensi. Tujuan dilampirkannya tabel daftar penelitian di bawah ini adalah sebagai tolak ukur berjalannya penelitian ini.

Penulis dan Tahun	**Judul**	**Tujuan**	**Model**	**Elemen Pembeda**
(Nurfauzi, 2022)	Deteksi Serangan *Malware* Pada *Cloud Server* Menggunakan *Metode Anomaly Based*	Deteksi serangan *Malware* pada cloud server	*Anomaly Based*	Pemilihan fitur kurang efektif serta dataset tidak seimbang
(Hananta Firdaus dkk., 2022)	Klasifikasi Penyakit Katarak Pada Mata Menggunakan Metode *Convolutional Neural Network (CNN)* Berbasis Web	Mengelompokkan citra mata katarak menggunakan *deep learning* berbasis web	*CNN*	Tipe data masukan adalah citra atau gambar
(Adiputra & Setiawan, 2023)	Klasifikasi *Malicious URL* Menggunakan Algoritma *Improved Random Forest* dan *Random Fores*t Berbasis Web	Membuat sistem deteksi *Malware* berdasarkan *URL* berbasis web	*Improved Random Forest, Random Fores*t	Data berbentuk *URL* jenis *Malware* dan *URL Benign* (jinak)

Penelitian terdahulu di atas dianggap relevan sebagai referensi karena masing-masing studi memberikan wawasan berharga mengenai deteksi *malware*, penggunaan metode CNN dalam klasifikasi, dan implementasi sistem berbasis web, yang semuanya merupakan aspek penting dalam penelitian "Klasifikasi *Malware* Menggunakan Metode Convolutional Neural Network (CNN) Berbasis Website"

METODE PENELITIAN

1) **Praproses**

Sebelum melakukan pemodelan, diperlukan beberapa tahap praproses pada data. Seluruh langkah praproses dalam penelitian ini dilakukan melalui *Jupyter Notebook* dengan memanfaatkan berbagai pustaka yang sesuai, seperti *pandas* dan *numpy*. Berikut adalah langkah langkah praposes pada data yang pertama reduksi dimensi, kedua one-hot encoding, ketiga label encoding, selanjutnya normalisasi. Berikut contoh perhitungan normalisasi MinMax :

$$x_{norm} = \frac{x - \min f}{\max - \min f}$$

$$x_{norm} = \frac{34.603264 - 0}{78840.329305 - 0} = 4.389031\text{e} - 04$$

Tahap normalisasi diterapkan kepada setiap data di setiap kolom kecuali kolom label sehingga menghasilkan data.

*2) **CNN Deep learning***

Proses pelatihan menggunakan metode CNN dilakukan dengan beberapa tahapan yaitu pembagian data latih dan data uji, pembentukan lapisan, perhitungan fungsi aktivasi, dan evaluasi model.

Tabel 3. 1 Sampel Data

		x1	x2	x3	x4	x5	label	Keterangan
data latih	**1**	0,000054218	0,000000894	0	1	0,71	0	*Benign*

	2	0,000054218	0,000000894	0,00003903117	1	0,71	0	*Benign*
	3	0,000054218	0,000000526	0,00000000018	0	0,43	0	*Benign*
	4	0,000054218	0,000000526	0,00000000381	0	0,43	0	*Benign*
	5	0,001301236	0,000023139	1	0	0,71	0	*Benign*
	6	0,000054218	0,000000894	0,00000288796	1	0,71	0	*Benign*
	7	0,000054218	0,000000999	0,00000567703	0,5	0,57	0	*Benign*
	8	0,000054218	0,000000526	0,00000290500	0	0,43	1	*Malware*
	9	0,000054218	0,000000526	0,00001719820	0	0,43	1	*Malware*
	10	0,000054218	0,000000999	0,00001350788	0,5	0,57	0	*Benign*
	11	0,000054218	0,000000526	0,00001719827	0	0,43	1	*Malware*
	12	0,000054218	0,000000999	0,00000210686	0,5	0,57	0	*Benign*
	13	0,000054218	0,000000526	0,00001719848	0	0,43	1	*Malware*
	14	0,000054218	0,000000894	0,00000027440	1	0,71	0	*Benign*
	15	0,000054218	0,000000736	0,00000042479	1	0,71	0	*Benign*
	16	0,000054218	0,000000999	0,00000338579	0,5	0,57	0	*Benign*
	17	0,000054218	0,000000526	0,00000000018	0	0,43	1	*Malware*
	18	0,000054218	0,000000526	0,00001719854	0	0,43	1	*Malware*
	19	0,000054218	0,000000999	0,00000396086	0,5	0,57	0	*Benign*
	20	0,000054218	0,000000526	0,00001719838	0	0,43	1	*Malware*
Data uji	**21**	0,000054218	0,000000999	0,00000432127	0,5	0,57	0	*Benign*
	22	0,000054218	0,000000736	0,00000270302	1	0,71	0	*Benign*

	23	0,000054218	0,000000999	0,00000498179	0,5	0,57	0	*Benign*
	24	0,000054218	0,000000999	0,00000098696	0,5	0,57	0	*Benign*
	25	0,000054218	0,000000947	0,00000189655	1	0,71	0	*Benign*

Terdapat 2 *Hidden Layer* yang berperan dalam pembelajaran pola dari data *input*. Setiap *Hidden Layer* tersebut memiliki jumlah *neuron* yang berbeda dan akan menghasilkan parameter (biasanya disebut sebagai bobot dan bias). Parameter inilah yang nantinya akan dipelajari oleh model pada saat pelatihan untuk membuat prediksi. Rumus perhitungan parameter :

$$param = (besar\ input * besar\ output) + besar\ output$$

Perhitungan parameter pada *Hidden Layer* pertama adalah seperti berikut:

$$param = (5 * 1000) + 1000 = 5000 + 1000 = 6000$$

Pada lapisan ini, data masukan akan diubah dengan menerapkan fungsi aktifasi ReLU pada setiap neuron. Langkah pertama pada lapisan ini yaitu menghitung bobot yang diberikan secara acak. Berikut adalah daftar nilai w yang telah ditentukan menggunakan fungsi *rand()* di *Microsoft Excel*:

Tabel 3. 2 Daftar bobot

w	nilai
w1	0,154451
w2	0,559455
w3	0,538638
w4	0,799724
w5	0,519061

Sehingga nilai V1 yang didapat dari persamaan 3.1 adalah :

$$V_1 = (0{,}000054218 * 0{,}154) + (0{,}000000894 * 0{,}559455) + (0 * 0) + (1 * 0{,}799724) + (0 * 0{,}71) = 1{,}17049 \quad (3.2)$$

Pada klasifikasi *binary*, kelas yang lebih mendekati 0 akan mereprentasikan satu kelas (*Benign*) dan sebaliknya (*Malware*).

$$\begin{cases} 0\ jika\ label \leq thresold \\ 1\ jika\ label \geq thresold \end{cases}$$

Performa model merupakan sebuah tolak ukur untuk menguji kelayakan sebuah model klasifikasi. Pada penelitian ini, tolak ukur tersebut adalah akurasi, presisi, dan *recall*. Akurasi mengukur sejauh mana model klasifikasi memprediksi dengan benar seluruh kelas. Presisi mengukur sejauh mana prediksi positif dibuat oleh model klasifikasi benar. *Recall* mengukur sejauh mana model klasifikasi mampu mendeteksi *instance* yang sebenarnya positif. Adapun rumus dari Akurasi, presisi, dan *recall* adalah sebagai berikut:

$$Akurasi = \frac{TP + TN}{TP + TN + FP + FN}$$
$$Presisi = \frac{TP}{TP + FP} \quad (3.5)$$
$$Recall = \frac{TP}{TP + FN}$$

Keterangan :

TP = *True Positive* (Benar mengklasifikasikan kelas positif)

TN = *True Negative* (Benar mengklasifikasikan kelas negatif)

FP = *False Positive* (Salah mengklasifikasikan kelas positif)

FN = *False Negative* (Salah mengklasifikasikan kelas negatif)

HASIL DAN PEMBAHASAN

1. Hasil Pelatihan

Proses pelatihan dilakukan dengan memperhatikan nilai *loss*, *val_los*, *accuracy*, dan *val_accuracy* dan juga *early stopping*. Parameter-parameter ini berfungsi sebagai tolak ukur

kestabilan proses pelatihan. Proses pelatihan dilakukan sebanyak dua kali dengan cara memecah dataset menjadi dua bagian untuk menghemat waktu.

Tabel 1. 3 Hasil Pelatihan Tiap Epoch pada Bagian dataset pertama

Epoch	***accuracy***	***loss***	***val_accuracy***	***val_loss***
1	0.9673	0.1638	0.9762	0.1132
2	0.9765	0.1119	0.9762	0.1118
3	0.9761	0.1127	0.9762	0.1116
4	0.9767	0.1106	0.9762	0.1117
5	0.9765	0.1111	0.9762	0.1116
6	0.9762	0.1120	0.9762	0.1115
7	0.9766	0.1107	0.9762	0.1110
8	0.9761	0.1121	0.9762	0.1117
9	0.9761	0.1118	0.9764	0.1114
10	0.9768	0.1091	0.9762	0.1105

Tabel 4. 3 Hasil Pelatihan Tiap Epoch pada Bagian Dataset Kedua

Epoch	***accuracy***	***loss***	***val_accuracy***	***val_loss***
1	0.9542	0.1788	0.9990	0.0100
2	0.9988	0.0091	0.9990	0.0082
3	0.9990	0.0064	0.9990	0.0110
4	0.9989	0.0064	0.9990	0.0135
5	0.9988	0.0064	0.9990	0.0162
6	0.9989	0.0061	0.9990	0.0200
7	0.9989	0.0058	0.9990	0.0222
8	0.9988	0.0061	0.9990	0.0250
9	0.9988	0.0060	0.9990	0.0299
10	0.9988	0.0078	0.9990	0.0275
11	0.9989	0.0055	0.9990	0.0334

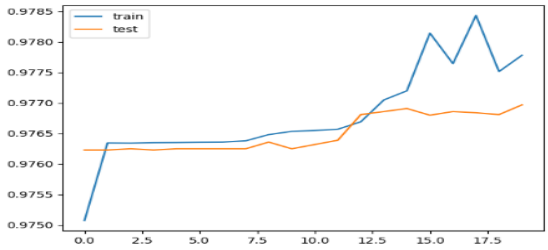
Proses pelatihan berhenti pada Epoch ke-11 karena parameter early stopping terpicu pada Epoch ke-6. *Val_loss* mengukur kinerja model pada data baru atau data yang tidak terlihat.

Sedangkan *accuracy* mengukur keakuratan model dalam memprediksi benar terhadap target aktual. Sama halnya dengan *val_loss, val_accuracy* menggunakan data yang tidak terlihat. Idealnya, model yang baik memiliki tingkat *loss* yang rendah dan *akurasi* yang tinggi.

Gambar 1.1 Visualisasi loss dan val_loss per epoch pada dataset bagian pertama

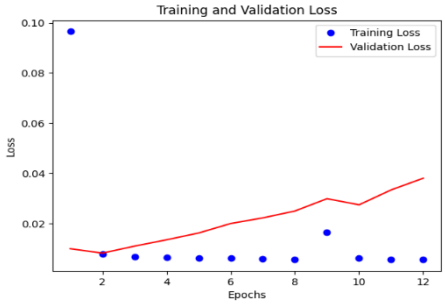
menunjukkan perbandingan antara "Training Loss" (biru) dan "Validation Loss" (merah) selama pelatihan.

Namun, *validation loss* tetap stabil hingga mendekati *epoch* ke-18, lalu meningkat tajam, menunjukkan overfitting di mana model terlalu menyesuaikan diri dengan data pelatihan dan kurang mampu menggeneralisasi pada data baru hanya di *epoch* ke-18.

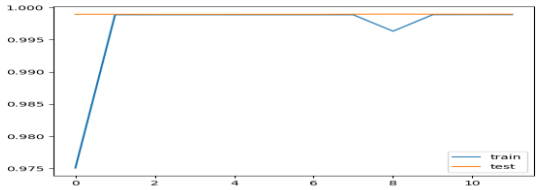


Gambar 1.2 Visualisasi accuracy dan val_accuracy per epoch pada dataset bagian pertama

menunjukkan perbandingan akurasi antara data pelatihan (train) dan data uji (test) selama pelatihan model CNN. Peningkatan akurasi data pelatihan yang lebih tajam dibandingkan dengan data uji mengindikasikan bahwa model mungkin mulai mengalami *overfitting*, di mana model lebih cocok dengan data pelatihan daripada data baru.

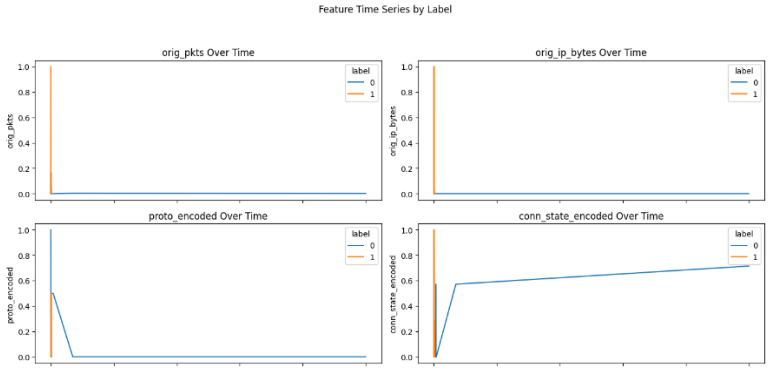


Gambar 1.4 menampilkan perbandingan "Training Loss" dan "Validation Loss" selama pelatihan model CNN pada dataset bagian kedua. *Validation loss* yang terus meningkat sementara *training loss* tetap rendah dan stabil menguatkan indikasi ini, menunjukkan bahwa model lebih cocok dengan data pelatihan daripada data validasi.



Gambar 1.5 Visualisasi accuracy dan val_accuracy per epoch pada Dataset Bagian Kedua

Grafik 1.5 menunjukkan perbandingan akurasi antara data pelatihan (*train*) dan data uji (*test*) selama pelatihan model CNN pada dataset bagian kedua. Kedua kurva akurasi meningkat stabil dengan jumlah *epoch*, dengan akurasi data pelatihan sedikit menurun pada



epoch ke-8 lalu kembali naik sampai akhir pelatihan.

Gambar 1.6 Tren Pola serangan

Gambar diatas menjabarkan bagaimana tiap fitur yang digunakan berkorelasi dengan label seiring waktu pelatihan berjalan.

2. Evaluasi Model

Evaluasi model merujuk pada evaluasi kinerja model selama proses pelatihan. Evaluasi ini dilakukan dengan cara melihat metrik akurasi, presisi, recall, f1 *score*, dan *confusion matrix* dengan pembagian data uji 80% dan data latih 20%.

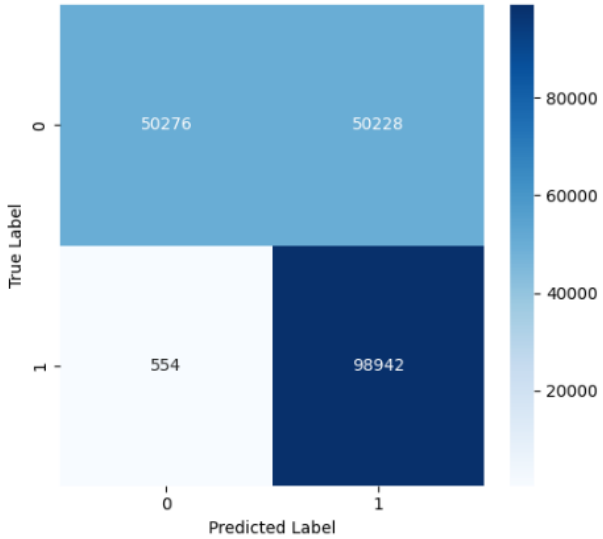
Tabel 1.3 Hasil Pelatihan Model CNN Pada Dataset Bagian Pertama

Akurasi	Presisi	Recall	F1 *Score*
97%	99%	60%	74%

Tabel 1.4 Hasil Pelatihan Model CNN Pada Dataset Bagian Kedua

Akurasi	Presisi	Recall	F1 *Score*
99%	99%	99%	99%

Selain metrik pada tabel di atas, tolak ukur evaluasi model lainnya adalah *Confusion matrix*. *Confusion matrix* bertujuan untuk mengetahui distribusi hasil prediksi pada setiap kelas yang nantinya akan dihitung untuk menemukan akurasi, presisi, *recall* dan f1 *score.*

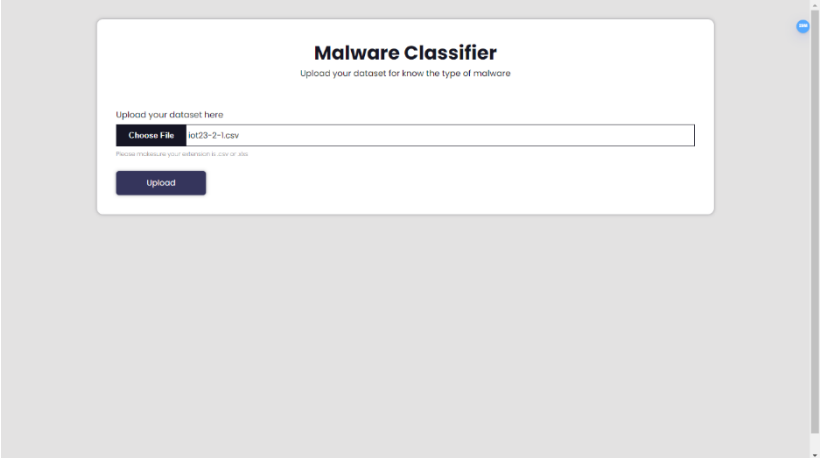


Gambar 1.7 Confusion Matrix

Terlihat nilai *True Positive* memiliki jumlah terbesar daripada nilai yang lain. Hal ini menandakan kehandalan model memprediksi data aktual positif menjadi benar positif, dengan kata lain sebanyak 98942 data adalah berlabel 1 (*malware).*

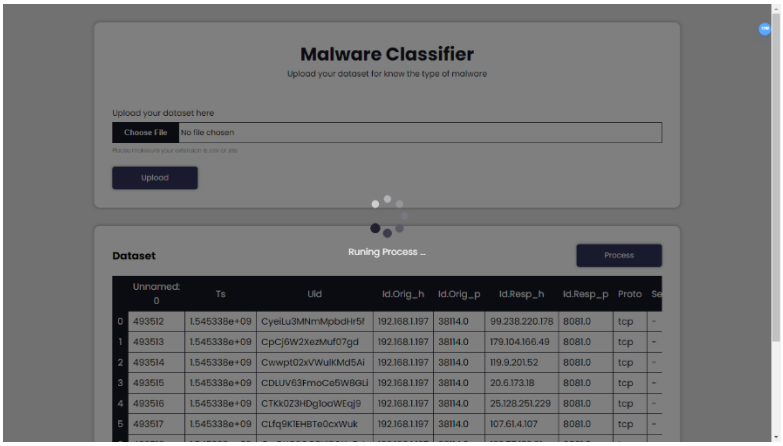
3. Tampilan Website

Agar proses klasifikasi terlihat lebih rapi dan terstrukur, dibuat sebuah aplikasi berbasis web menggunakan bahasa pemrograman *python* dengan *framework flash* untuk pelatihan modelnya. Berikut tampilan dari menu awal yaitu upload dataset

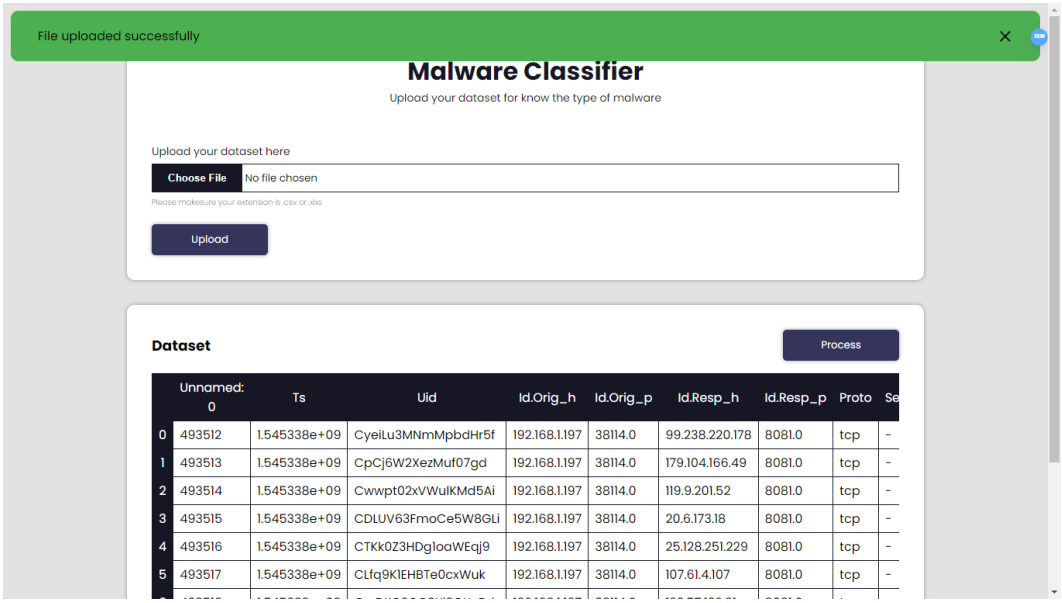


Gambar 1.8 Tampilan Upload Dataset

Tekan tombol *upload* untuk mengunggah dataset dari *folder* lokal. Akan muncul *pop-up* yang menandakan dataset berhasil diunggah setelah proses *upload* selesai.



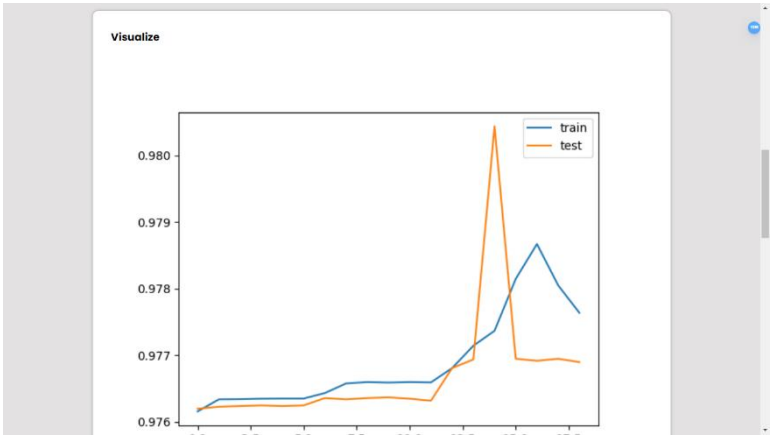
bar 1.9 Proses Upload



	Unnamed: 0	Ts	Uid	Id.Orig_h	Id.Orig_p	Id.Resp_h	Id.Resp_p	Proto	Se
0	493512	1.545338e+09	CyeiLu3MNmMpbdHr5f	192.168.1.197	38114.0	99.238.220.178	8081.0	tcp	-
1	493513	1.545338e+09	CpCj6W2XezMuf07gd	192.168.1.197	38114.0	179.104.166.49	8081.0	tcp	-
2	493514	1.545338e+09	Cwwpt02xVWulKMd5Ai	192.168.1.197	38114.0	119.9.201.52	8081.0	tcp	-
3	493515	1.545338e+09	CDLUV63FmoCe5W8GLi	192.168.1.197	38114.0	20.6.173.18	8081.0	tcp	-
4	493516	1.545338e+09	CTkk0Z3HDgloaWEqj9	192.168.1.197	38114.0	25.128.251.229	8081.0	tcp	-
5	493517	1.545338e+09	CLfq9K1EHBTe0cxWuk	192.168.1.197	38114.0	107.61.4.107	8081.0	tcp	-

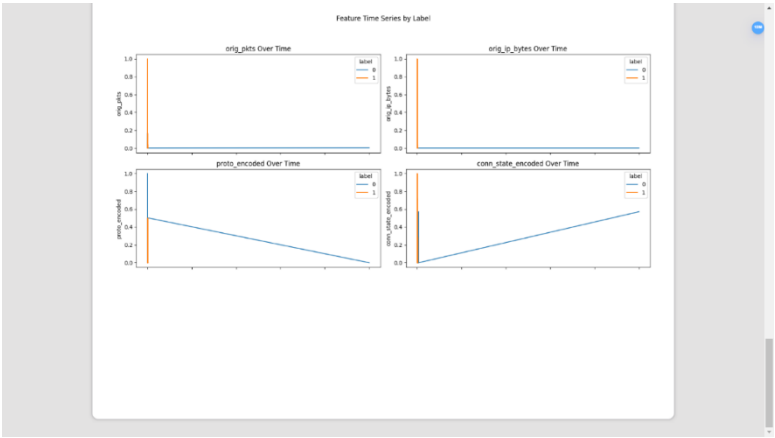
Gambar 1. 10 Upload Sukses

Pengguna bisa melihat dan menganalisa dataset yang telah diupload untuk kebutuhan analisa pada menu Dataset.



Gambar 1. 11 Visualisasi Time Akurasi

Tahap praproses dan arsitektur layer dilakukan secara *background* atau di belakang layar karena dirasa kurang perlu untuk ditampilkan. Setelah pengguna menekan tombol *process*, proses pelatihan akan berjalan dan akan menghasilkan metrik akurasi, presisi, *recall*, dan *f1-score.* Pengguna juga bisa melihat evaluasi kinerja model seperti visualisasi akurasi dan visualisasi serangan seiring waktu atau *time series analysis*.



Gambar 1. 12 Visualisasi Time series

Sebagai perbandingan, berikut hasil pelatihan pada pembagian data latih 50% dan data uji 50%.

Tabel 1.5 Hasil Pelatihan Model CNN Pada Dataset Bagian Pertama

Akurasi	Presisi	Recall	F1 *Score*
0,58%	0,55%	100%	0,1%

Tabel 1.6 Hasil Pelatihan Model CNN Pada Dataset Bagian Kedua

Akurasi	Presisi	Recall	F1 *Score*
0,67%	1,2%	0,71%	0,13%

KESIMPULAN DAN SARAN

Berdasarkan hasil penelitian yang sudah dijabarkan, bisa disimpulkan bahwa:

1) Klasifikasi malware menggunakan metode CNN berbasis web berhasil dibuat dan dijalankan
2) Model CNN yang dijalankan pada web tersebut menghasilkan akurasi tertinggi sebesar 99%, presisi tertinggi sebesar 99%, recall tertinggi sebesar 99% dan f1-score tertinggi sebesar 99%.

Adapun saran yang dapat diperhatikan untuk peneliti selanjutnya tentang topik terkait adalah:

1. Gunakan metode praproses yang lebih efektif menangani masalah kekosongan data tanpa harus menghapusnya seperti mean/median filling. Dalam kasus penelitian ini, tahap praproses melibatkan banyak penghapusan atribut yang mungkin saja memiliki informasi tambahan.
2. Perhatikan arsitektur lapisan atau layer karena sangat berpengaruh terhadap metrik yang dihasilkan oleh model. Atur layer sesuai karakteristik dataset yang digunakan.
3. Di sisi pengembangan website, bisa ditambahkan menu yang lebih interaktif seperti menu praproses terpisah atau pengaturan parameter model yang terintegrasi.

DAFTAR REFERENSI

Adenansi, R., & Novarina, L. A. (2017). *MALWARE DYNAMIC*.

Adiputra, O., & Setiawan, E. (2023). Klasifikasi Malicious URL Menggunakan Algoritma Improved Random Forest dan Random Forest Berbasis Web. *Jurnal Sains dan Informatika*, *9*(1), 8–14. https://doi.org/10.22216/jsi.v9i1.1378

Akhtar, M. S., & Feng, T. (2022). *Malware* Analysis and Detection Using Machine Learning Algorithms. *Symmetry*, *14*(11). https://doi.org/10.3390/sym14112304

Azzahra Nasution, D., Khotimah, H. H., & Chamidah, N. (2019). *PERBANDINGAN NORMALISASI DATA UNTUK KLASIFIKASI WINE MENGGUNAKAN ALGORITMA K-NN* (Vol. 4, Nomor 1).

Damanik, R. A., Seta, H. B., & Theresiawati. (2023). *ANALISIS TROJAN DAN SPYWARE MENGGUNAKAN METODE HYBRID ANALYSIS*.

Garcia, S., Parmisano, A., & Erquiaga, M. J. (2020). *IoT-23: A labeled dataset with malicious and benign IoT network traffic*.

Hamdi, F. S., & Maita, I. (2022). Pelatihan Pembuatan Website Memanfaatkan Wix Untuk Blog Pribadi Pada Siswa SMAN 2 Gunung Talang. *CONSEN: Indonesian Journal of Community Services and Engagement*, *2*(2), 64–69. https://doi.org/10.57152/consen.v2i2.471

Hananta Firdaus, D., Imran, B., Darmawan Bakti, L., & Suryadi, E. (2022). KLASIFIKASI PENYAKIT KATARAK PADA MATA MENGGUNAKAN METODE CONVOLUTIONAL NEURAL NETWORK (CNN) BERBASIS WEB. Dalam *Jurnal Kecerdasan Buatan dan Teknologi Informasi (JKBTI)* (Vol. 1, Nomor 3).

Hartono, B. (2023). Ransomware: Memahami Ancaman Keamanan Digital. *Bincang Sains dan Teknologi*, *2*(02), 55–62. https://doi.org/10.56741/bst.v2i02.353

Kusumaningrum, T. F. (2018). *IMPLEMENTASI CONVOLUTION NEURAL NETWORK (CNN) UNTUK KLASIFIKASI JAMUR KONSUMSI DI INDONESIA MENGGUNAKAN KERAS*.

Lee, H., & Song, J. (2019). Introduction to convolutional neural network using Keras; An understanding from a statistician. *Communications for Statistical Applications and Methods*, *26*(6), 591–610. https://doi.org/10.29220/CSAM.2019.26.6.591

Liang, Y., & Vankayalapati, N. (2021). *Machine Learning and Deep Learning Methods for Better Anomaly Detection in IoT-23 Dataset Cybersecurity*.

Libovický, J. (2017). *Deep Learning for Natural Language processing Introduction to Natural Language Processing*.

Mahdi, F. A., Lukito, C. A., Parwita, D., Nofri, A., Madjid, V. A., & Prasvita, D. S. (2021). Pengaruh Principal Component Analysis terhadap Akurasi Model Machine Learning dengan Algoritma Artificial Neural Network untuk Prediksi Kebangkrutan Perusahaan. Dalam *Seminar Nasional Mahasiswa Ilmu Komputer dan Aplikasinya (SENAMIKA) Jakarta-Indonesia*.

Nurfauzi, N. A. (2022). *DETEKSI SERANGAN MALWARE PADA CLOUD SERVER MENGGUNAKAN METODE ANOMALY BASED*.

Putra, J. W. G. (2020). *Pengenalan Pembelajaran Mesin dan Deep Learning*. 150–151.

Python. (2021). *Python FLask*.

Rangga, M., Nasution, A., & Hayaty, M. (2019). Perbandingan Akurasi dan Waktu Proses Algoritma K-NN dan SVM dalam Analisis Sentimen Twitter. *JURNAL INFORMATIKA*, *6*(2), 212–218. http://ejournal.bsi.ac.id/ejurnal/index.php/ji

Rizki, M., Basuki, S., & Azhar, Y. (2020). Implementasi Deep Learning Menggunakan Arsitektur Long Short Term Memory Untuk Prediksi Curah Hujan Kota Malang. *REPOSITOR*, *2*(3), 331–338.

Sahu, A. K., Sharma, S., Tanveer, M., & Raja, R. (2021). Internet of Things attack detection using hybrid Deep Learning Model. *Computer Communications*, *176*, 146–154. https://doi.org/10.1016/j.comcom.2021.05.024

Silviana, Kurniawan, R., Nazir, A., Budianita, E., Syarifa, F., & Gusti, K. S. (2022). *PENGKLASTERAN RISIKO COVID-19 DI RIAU MENGGUNAKAN TEKNIK ONE HOT ENCODING DAN ALGORITMA K-MEANS CLUSTERING*.

Stoian, N.-A. (2020). *Machine Learning for Anomaly Detection in IoT networks: Malware analysis on the IoT-23 Data set*.

Suprayogi, C., & Marwan, M. A. (2022). Classification of Network Traffic Data Mirai *Malware* Attacks on Internet of Things Devices Using the K-Nearest Neighbor Method. *International Research Journal of Advanced Engineering and Science*, *7*(4), 39–43.