Methotika : Jurnal Ilmiah Teknik Informatika
E-ISSN : 2776 – 5792
Vol. 6, No. 1, April 2026, pp. 116-127
https://ejurnal.methodist.ac.id/index.php/methotika/

Analisis Klasifikasi Lalu Lintas Serangan Menggunakan Metode CNN Pada Honeypot Server

Rostianna Br Sinuraya[1], Naikson Fandier Saragih[2], Mufria J. Purba[3]
[1,2,3]Fakultas Ilmu Komputer, Universitas Methodist Indonesia

Info Artikel
Histori Artikel:
Received, Jan 25 , 2026
Revised, Feb 15, 2026
Accepted, Feb 27, 2026

Keywords:
Convolutional Neural Network,
Deep Learning,
Batch Size,
Epoch,
Klasifikasi,
Confusion Matrix,
Pyhoneypot.

ABSTRAK

Sistem deteksi serangan terhadap server perlu terus dimaksimalkan termasuk dengan pemanfaatan kecerdasan buatan AI (Artificial Intelligence). Hal ini disebabkan oleh meningkatnya serangan siber seperti Distributed Denial of Service (DDoS) dan port scanning yang mengancam ketersediaan layanan dan keamanan data. Di sisi lain, honeypot server menghasilkan data dalam jumlah besar yang sulit dianalisis secara manual, sehingga diperlukan metode otomatis seperti Convolutional Neural Network (CNN) untuk mengklasifikasikan jenis serangan secara cepat dan akurat. Penelitian ini berfokus pada penerapan metode Convolutional Neural Network (CNN) untuk mengklasifikasikan lalu lintas serangan server berdasarkan data yang diperoleh dari honeypot server. Dataset kemudian dibagi menjadi data latih dan data uji dengan skenario pembagian yaitu 60:40. Model klasifikasi dibangun menggunakan metode *Convolutional Neural Network (CNN)* yang diimplementasikan pada *TensorFlow/Keras* melalui *Google Colaboratory*, kemudian dilatih untuk mengenali pola lalu lintas jaringan. Tahap akhir dilakukan pengujian dan evaluasi menggunakan *confusion matrix* untuk mengukur kinerja model berdasarkan nilai akurasi, presisi, recall, dan F1-score. Hasil penelitian menunjukkan bahwa model *Convolutional Neural Network (CNN)* yang dibangun mampu mengklasifikasikan lalu lintas serangan jaringan dengan tingkat kinerja yang sangat baik. Berdasarkan hasil pengujian menggunakan *confusion matrix*, model memperoleh nilai akurasi sebesar 99,98%, dengan nilai precision 89,58%, recall 90,77%, dan F1-score 90,16%. Nilai tersebut menunjukkan bahwa model memiliki kemampuan yang tinggi dalam membedakan antara lalu lintas normal (*benign*) dan berbagai jenis serangan seperti TCP Flood, UDP Flood, dan Port Scanning. Selain itu, model juga menunjukkan kestabilan performa pada berbagai skenario pembagian data latih dan data uji, sehingga dapat dikatakan efektif dan andal dalam mendukung sistem deteksi serangan jaringan secara otomatis.

Penulis Koresponden:
Naikson Fandier Saragih,
Fakultas Ilmu Komputer,
Universitas Methodist Indonesia, Medan,
Jl. Hang Tuah No.8, Medan - Sumatera Utara.
Email: saragihnaikson@gmail.com

1. PENDAHULUAN

Serangan DDoS yang terdiri dari TCP Flood dan UDP Flood merupakan salah satu serangan yang paling umum dan merugikan, serangan DDoS membanjiri server dengan trafik yang sangat tinggi, sehingga menghabiskan sumber daya server, seperti bandwidth, memori, dan CPU. Kondisi ini menyebabkan server menjadi tidak responsif atau tidak dapat diakses oleh pengguna yang sah. Pengetesan menggunakan DDoS membantu mengidentifikasi kelemahan sistem yang tidak terdeteksi oleh pengujian lain [1]. Serangan DDoS juga dapat digunakan sebagai pengalihan dari serangan lain seperti pencurian data, serta menargetkan lembaga keuangan atau pemerintahan untuk mengganggu layanan publik dan menurunkan kepercayaan masyarakat[2].

Sementara itu, Port scanning merupakan teknik yang digunakan untuk mengidentifikasi port terbuka dan layanan aktif pada sistem, yang dapat dimanfaatkan untuk menemukan potensi kerentanan keamanan. Aktivitas ini berbahaya karena dapat mengungkap informasi penting mengenai sistem target yang kemudian dapat digunakan sebagai dasar untuk melakukan eksploitasi atau serangan lanjutan. Kedua jenis serangan ini sama-sama menimbulkan ancaman serius terhadap keamanan jaringan, karena port scanning dapat menjadi pintu masuk bagi serangan lanjutan, termasuk DDoS, eksploitasi layanan, maupun pencurian data [3].

Honeypot digunakan sebagai sistem untuk menangkap, merekam, dan memantau aktivitas serangan jaringan secara real-time. Honeypot server berfungsi sebagai umpan yang meniru layanan asli sehingga dapat menarik interaksi dari luar jaringan [4].

Deep learning adalah teknologi kecerdasan buatan yang menggunakan jaringan saraf berlapis-lapis untuk belajar dan mengenali pola dari data besar secara otomatis. Teknologi ini mampu mengekstraksi fitur penting dari data baik visual maupun tekstual, sehingga dapat melakukan klasifikasi atau prediksi dengan tingkat akurasi yang tinggi tanpa perlu pemrograman fitur secara manual [5].

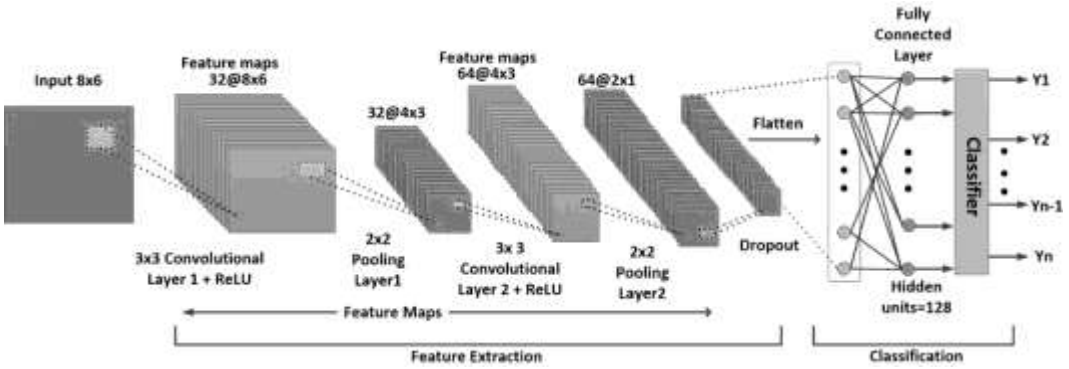
Metode Convolutional Neural Network (CNN) memiliki kemampuan untuk mengolah data yang kompleks dan mendeteksi pola-pola tersembunyi yang sulit dikenali oleh metode konvensional. CNN mampu mengubah data serangan menjadi format teks atau angka yang kemudian diproses untuk mendapatkan akurasi tinggi dalam klasifikasi serangan [6]

Klasifikasi menggunakan metode CNN digunakan untuk mendeteksi suatu aktivitas jaringan yang termasuk serangan atau bukan, mengenali tipe serangan secara otomatis berdasarkan pola lalu lintas jaringan, mempermudah respons keamanan contohnya untuk memblokir IP penyerang atau mengaktifkan sistem pertahanan, dan mengelompokkan data agar bisa dianalisis lebih lanjut [7].

2. METODE PENELITIAN

2.1 Convolutional Neural Network

CNN digunakan secara luas dalam berbagai aplikasi, terutama dalam pengenalan pola, pengolahan gambar, dan analisis video [8]. Dalam konteks klasifikasi lalu lintas serangan, CNN (Convolutional Neural Network) digunakan sebagai metode untuk mendeteksi dan mengklasifikasikan pola-pola yang mencurigakan dalam data lalu lintas jaringan. Pada gambar 1 adalah arsitektur CNN:



Gambar 1. Arsitektur CNN
Sumber Obet Siahaan dkk, 2023

Arsitektur CNN dibagi menjadi 2 bagian [9]:

1. Feature Extraction

Proses Layer ini mengubah data masukan menjadi angka yang merepresentasikan fitur penting yang dinamakan *encoding*. Terdiri dari *Convolutional Layer* untuk mengekstrak pola dan *Pooling Layer* untuk mereduksi dimensi data.

a. Convolution Layer

Layer ini berfungsi untuk mengekstraksi fitur dari data input. Dalam prosesnya, data input diambil sesuai ukuran filter, dan dilakukan perkalian yang menghasilkan peta fitur. Jumlah parameter yang harus dipelajari ditentukan oleh jumlah filter dan ukurannya.

b. Pooling Layer

Pooling layer digunakan untuk mengurangi dimensi data sambil mempertahankan informasi penting. Mengambil nilai maksimum atau rata-rata dari area tertentu, pooling menyaring fitur peta yang dihasilkan oleh convolution layer.

2. Fully Connected Layer

Layer ini umum digunakan dalam penerapan Multi-Layer Perceptron (MLP) dan bertujuan untuk melakukan transformasi pada dimensi data agar data dapat diklasifikasikan secara linear[10].

a. Loss Layer

Loss layer adalah lapisan terakhir dalam CNN dimana pada proses ini akan memperlihatkan hasil prediksi dan nilai loss pada saat proses training.

2.2 Confusion Matrix

Empat istilah utama dalam Confusion Matrix adalah *True Positive*, *True Negative*, *False Positive*, dan *False Negative*. Nilai *True Positive* dan *True Negative* menunjukkan saat model berhasil mengklasifikasikan data dengan benar, sedangkan *False Positive* dan *False Negative* menunjukkan ketika model melakukan kesalahan dalam klasifikasi data [11].Tabel *confusion matrix* dapat dilihat pada Tabel 1

Tabel 1. Confusion Matrix

Sample Classes		Prediksi	
		Postitif	Negative
Label Atribut	Positive	TP	FP
	Negative	FN	TN

Dalam prediksi yang dilakukan oleh model klasifikasi, *Confusion Matrix* digunakan untuk menganalisis seberapa baik model dalam mengklasifikasikan data dengan melihat nilai berikut:

1) *Accuracy* mengukur seberapa akurat model dalam mengklasifikasikan data dengan benar.

Accuracy = (TP) / (Total)

2) *Precision (Positive Predictive Value)* menunjukkan tingkat keakuratan antara data yang benar-benar positif dengan hasil prediksi positif yang diberikan oleh model.

Precision = (TP) / (TP + FP)

3) *Recall (Sensitivity / True Positive Rate)* menggambarkan kemampuan model dalam menemukan kembali data yang benar-benar positif.

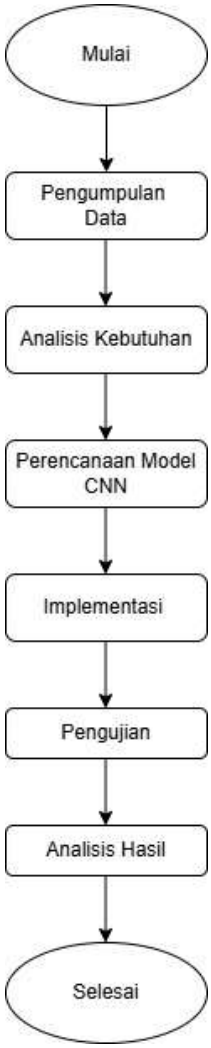
Recall = TP / (TP + FN)

4) *F1-Score* adalah rata-rata harmonis antara *Precision* dan *Recall*. Accuracy lebih tepat digunakan sebagai ukuran performa jika jumlah *False Negative* dan *False Positive* dalam dataset seimbang. Namun, jika jumlahnya tidak seimbang, *F1-Score* lebih disarankan sebagai acuan.

F-1 Score = (2 * Recall * Precision) / (Recall + Precision)

2.7 Framework Penelitian

Framework penelitian adalah struktur yang menggambarkan langkah-langkah yang harus diambil dalam suatu penelitian. Setiap tahapan disusun dengan rapi untuk memastikan penelitian berjalan sistematis dan terarah. Dapat dilihat pada gambar :



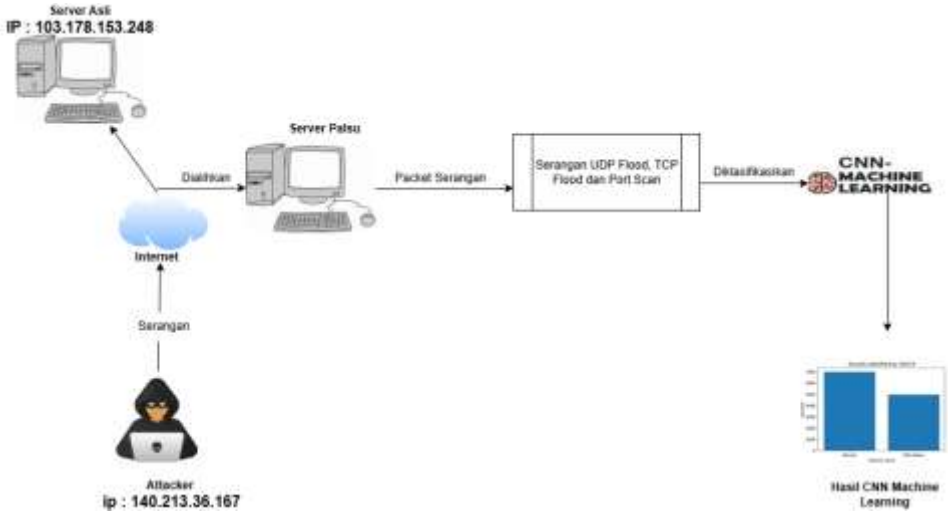
Gambar 2 Framework Penelitian

2.8 Pengumpulan Data

Proses pengumpulan data dari buku dan penelitian terdahulu sebagai referensi. Data serangan DDoS dan Port Scanning ditangkap mengunakan honeypot yang dijalankan pada virtual private server (VPS). Aktivitas serangan yang terdeteksi dicatat dalam log berisi informasi seperti ip, port, timestamp, dan jenis koneksi. Seluruh data dari hasil skenario serangan yang didapat pada honeypot server berjumlah 455340 baris data.

2.9 Analisa Sistem

Analisis sistem merupakan proses memecah suatu sistem informasi secara keseluruhan menjadi bagian-bagian komponennya dengan tujuan untuk mengidentifikasi serta mengevaluasi permasalahan yang ada. Adapun tahapan atau proses yang dilakukan dapat dilihat pada gambar 3



Gambar 3. Analisa Sistem

Gambar menunjukkan bahwa seorang attacker dengan IP 140.213.36.167 melancarkan serangan melalui internet menuju server asli yang memiliki IP 103.178.153.248, namun seluruh serangan ini *tidak masuk ke server asli* karena sudah dialihkan (redirect) menuju server palsu (honeypot). Paket-paket tersebut kemudian dikirimkan ke proses analisis yang berisi jenis-jenis serangan UDP Flood, TCP Flood, dan Port Scan, lalu semua paket itu diklasifikasikan menggunakan model CNN (Machine Learning). Hasil klasifikasinya divisualisasikan dalam bentuk grafik yang menunjukkan jumlah serangan yang berhasil diidentifikasi, pada kategori Benign, Port Scan, dan DDoS Attack.

2.10 Perencanaan Model CNN

Data penelitian diperoleh dari log serangan yang direkam oleh pyhoneypot dalam format teks atau JSON pada direktori */var/log/honeypy/*. Log tersebut diekstraksi menggunakan python dan diubah menjadi dataset CSV yang kemudian diproses menjadi fitur numerik sebagai input model CNN. Dataset diberi label benign, port scan, tcp flood, dan udp flood, lalu diubah ke bentuk numerik menggunakan label encoder. Data kemudian dibagi menjadi data latih dan data uji dengan skenario 60:40. Model klasifikasi dibangun menggunakan CNN dengan lapisan convolutional, maxpooling, flatten, dense, dan dropout. Kinerja model dievaluasi menggunakan accuracy, precision, recall, f1-score, dan confusion matrix.

2.11 Implementasi

a. Import Library dan Dataset ke Google Collab

Langkah awal dalam membangun model CNN adalah melakukan import library yang diperlukan. Library ini digunakan untuk proses pengolahan data, pembagian dataset, serta pelatihan model.

```
[1]
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.metrics import confusion_matrix, classification_report, accuracy_score, ConfusionMatrixDisplay

from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv1D, MaxPooling1D, Flatten, Dense, Dropout
from tensorflow.keras.utils import to_categorical
from tensorflow.keras.callbacks import EarlyStopping
```

Gambar 4 Proses input library

Setelah itu, dataset yang tersimpan dalam format CSV pada Google Drive diinput ke dalam notebook Google Colab.

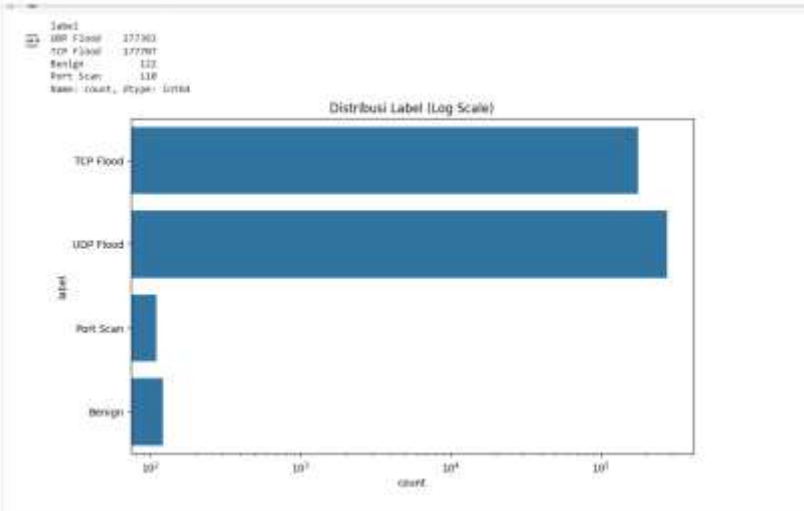
```
df
```

	timestamp	event	label
0	2025-08-11 03:20:35.111	WIB - Protocol 6 from 140.213.36.167:60206 on ...	TCP Flood
1	2025-08-11 03:20:35.118	WIB - Protocol 6 from 140.213.36.167:60206 on ...	TCP Flood
2	2025-08-11 03:20:35.124	WIB - Protocol 6 from 140.213.36.167:60206 on ...	TCP Flood
3	2025-08-11 03:20:35.125	WIB - Protocol 6 from 140.213.36.167:60206 on ...	TCP Flood
4	2025-08-11 03:20:35.125	WIB - Protocol 6 from 140.213.36.167:60206 on ...	TCP Flood
...	...	...	...
455334	2025-08-11 03:47:23.844,"WIB - Protocol 6 from...	NaN	NaN
455335	2025-08-11 03:47:25.852	WIB - Protocol 6 from 140.213.36.167:60229 on ...	Benign
455336	2025-08-11 03:47:26.084,"WIB - Protocol 6 from...	NaN	NaN
455337	2025-08-11 03:47:37.835	WIB - Protocol 6 from 140.213.36.167:61704 on ...	Benign
455338	2025-08-11 03:47:37.843,"WIB - Protocol 6 from...	NaN	NaN

455339 rows × 3 columns

Gambar 5 Input dataset

Dataset tersebut kemudian dibaca menggunakan library pandas sehingga berubah menjadi tabel terstruktur yang dapat diproses lebih lanjut oleh model.



Gambar 6 Grafik dataset berdasarkan label

b. Ekstraksi Pola Teks Menggunakan Regular Expression (Regex)

Pada tahap ini proses ekstraksi fitur dilakukan dari kolom *event* yang berisi data mentah hasil tangkapan log honeypot, dan mengahapus kolom event lalu mengganti nya dengan data yang bersifat numerik agar data tersebut dapat diolah oleh model *machine learning.*

```
df.drop(columns=['event'], inplace=True)
```

```
df.info()
df.head()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 455339 entries, 0 to 455338
Data columns (total 6 columns):
 #   Column    Non-Null Count   Dtype
---  ------    --------------   -----
 0   label     455302 non-null  object
 1   protocol  455302 non-null  float64
 2   src_ip    455302 non-null  float64
 3   src_port  455302 non-null  float64
 4   dst_ip    0 non-null       float64
 5   dst_port  0 non-null       float64
dtypes: float64(5), object(1)
memory usage: 20.8+ MB
```

	label	protocol	src_ip	src_port	dst_ip	dst_port
0	TCP Flood	6.0	2.362779e+09	60206.0	NaN	NaN
1	TCP Flood	6.0	2.362779e+09	60206.0	NaN	NaN
2	TCP Flood	6.0	2.362779e+09	60206.0	NaN	NaN
3	TCP Flood	6.0	2.362779e+09	60206.0	NaN	NaN
4	TCP Flood	6.0	2.362779e+09	60206.0	NaN	NaN

Gambar 7 Pembuatan kolom baru

Setelah kolom baru terbentuk, data dengan label NaN yang merupakan data null dalam dataset perlu dihapus agaar dataset menjadi bersih dan meningkatkan performa nilai akurasi.

```
df.drop(['dst_port', 'dst_ip'], axis=1, inplace=True)
```

```
df.info()
df.head()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 455339 entries, 0 to 455338
Data columns (total 4 columns):
 #   Column    Non-Null Count   Dtype
---  ------    --------------   -----
 0   label     455302 non-null  object
 1   protocol  455302 non-null  float64
 2   src_ip    455302 non-null  float64
 3   src_port  455302 non-null  float64
dtypes: float64(3), object(1)
memory usage: 13.9+ MB
```

	label	protocol	src_ip	src_port
0	TCP Flood	6.0	2.362779e+09	60206.0
1	TCP Flood	6.0	2.362779e+09	60206.0
2	TCP Flood	6.0	2.362779e+09	60206.0
3	TCP Flood	6.0	2.362779e+09	60206.0
4	TCP Flood	6.0	2.362779e+09	60206.0

Gambar 8 Penghapusan data null

c. Pembagian Data Latih dan Data Uji

Pembagian dataset antara data latih dan data uji yaitu dengan split 60:40. Berikut adalah gambar nya:

```
from sklearn.model_selection import train_test_split

# Misal: X = fitur, y_categorical = label one-hot
X_train, X_test, y_train, y_test = train_test_split(
    X_scaled,          # hasil normalisasi fitur
    y_categorical,     # label hasil one-hot encoding
    test_size=0.3,     # 30% data untuk pengujian
    random_state=42,   # agar hasil selalu sama
    stratify=y_categorical  # menjaga proporsi kelas seimbang
)

print("Bentuk data latih:", X_train.shape)
print("Bentuk data uji:", X_test.shape)
```

```
Bentuk data latih: (318711, 3)
Bentuk data uji: (136591, 3)
```

Gambar 9 Pembagian data latih dan data uji

d. Parameter Model CNN

Model CNN yang digunakan memiliki 3 hidden layer Conv1D dan 3 layer MaxPooling1D. Setiap layer Conv1D menggunakan input shape (1,1) dengan 64 filter dan kernel size 6, sedangkan layer MaxPooling1D menggunakan pool size 4 dengan padding same. Pada bagian output digunakan softmax dengan Dense 4 untuk mengklasifikasikan empat jenis trafik, yaitu Normal, UDP Flood, TCP Food, dan Port Scan.

```python
# Bangun Arsitektur (function) pada model cnn
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv1D, MaxPooling1D, Flatten, Dense, Dropout
from tensorflow.keras.callbacks import EarlyStopping

model = Sequential()
model.add(Conv1D(filters=64, kernel_size=6, activation='relu', padding='same', input_shape=(X_train.shape[1], 1)))
model.add(MaxPooling1D(pool_size=4, padding='same'))

model.add(Conv1D(filters=64, kernel_size=6, activation='relu', padding='same'))
model.add(MaxPooling1D(pool_size=4, padding='same'))

model.add(Conv1D(filters=64, kernel_size=6, activation='relu', padding='same'))
model.add(MaxPooling1D(pool_size=4, padding='same'))

model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.3))
model.add(Dense(4, activation='softmax'))  # 4 kelas output

model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
model.summary()
```

Model: "sequential"

Layer (type)	Output Shape	Param #
conv1d (Conv1D)	(None, 3, 64)	448
max_pooling1d (MaxPooling1D)	(None, 1, 64)	0
conv1d_1 (Conv1D)	(None, 1, 64)	24,640
max_pooling1d_1 (MaxPooling1D)	(None, 1, 64)	0
conv1d_2 (Conv1D)	(None, 1, 64)	24,640
max_pooling1d_2 (MaxPooling1D)	(None, 1, 64)	0
flatten (Flatten)	(None, 64)	0
dense (Dense)	(None, 128)	8,320
dropout (Dropout)	(None, 128)	0
dense_1 (Dense)	(None, 4)	516

Total params: 58,564 (228.77 KB)
Trainable params: 58,564 (228.77 KB)
Non-trainable params: 0 (0.00 B)

Gambar 10 Parameter Model CNN

3. HASIL DAN PEMBAHASAN

3.1 Hasil Akurasi

Nilai akurasi yang didapat dari proses pelatihan dan pengujian model CNN adalah 99,989%.

```
Epoch 7/20
2135/2135 ━━━━━━━━━━━━━━━━━━━━ 16s 7ms/step - accuracy: 0.9999 - loss: 5.9880e-04 - val_accuracy: 0.9999 - val_loss: 7.8332e-04
Epoch 8/20
2135/2135 ━━━━━━━━━━━━━━━━━━━━ 14s 7ms/step - accuracy: 0.9998 - loss: 5.1314e-04 - val_accuracy: 0.9998 - val_loss: 8.0922e-04
Epoch 9/20
2135/2135 ━━━━━━━━━━━━━━━━━━━━ 15s 7ms/step - accuracy: 0.9999 - loss: 5.0931e-04 - val_accuracy: 0.9999 - val_loss: 6.9321e-04
Epoch 10/20
2135/2135 ━━━━━━━━━━━━━━━━━━━━ 15s 7ms/step - accuracy: 0.9999 - loss: 0.0036 - val_accuracy: 0.9999 - val_loss: 8.8712e-04
Epoch 11/20
2135/2135 ━━━━━━━━━━━━━━━━━━━━ 15s 7ms/step - accuracy: 0.9999 - loss: 3.5429e-04 - val_accuracy: 0.9999 - val_loss: 7.3747e-04
Epoch 12/20
2135/2135 ━━━━━━━━━━━━━━━━━━━━ 15s 7ms/step - accuracy: 0.9999 - loss: 3.7219e-04 - val_accuracy: 0.9998 - val_loss: 8.7458e-04
Epoch 13/20
2135/2135 ━━━━━━━━━━━━━━━━━━━━ 15s 7ms/step - accuracy: 0.9999 - loss: 4.3177e-04 - val_accuracy: 0.9998 - val_loss: 0.0010
Epoch 14/20
2135/2135 ━━━━━━━━━━━━━━━━━━━━ 15s 7ms/step - accuracy: 0.9998 - loss: 6.8153e-04 - val_accuracy: 0.9999 - val_loss: 8.8599e-04
Epoch 15/20
2135/2135 ━━━━━━━━━━━━━━━━━━━━ 15s 7ms/step - accuracy: 0.9999 - loss: 3.2898e-04 - val_accuracy: 0.9999 - val_loss: 7.5189e-04
Epoch 16/20
2135/2135 ━━━━━━━━━━━━━━━━━━━━ 15s 7ms/step - accuracy: 0.9999 - loss: 3.6485e-04 - val_accuracy: 0.9999 - val_loss: 8.2105e-04
Epoch 17/20
2135/2135 ━━━━━━━━━━━━━━━━━━━━ 15s 7ms/step - accuracy: 0.9999 - loss: 3.3847e-04 - val_accuracy: 0.9999 - val_loss: 8.5995e-04
Epoch 18/20
2135/2135 ━━━━━━━━━━━━━━━━━━━━ 15s 7ms/step - accuracy: 0.9999 - loss: 6.8877e-04 - val_accuracy: 0.9999 - val_loss: 6.7793e-04
Epoch 19/20
2135/2135 ━━━━━━━━━━━━━━━━━━━━ 15s 7ms/step - accuracy: 0.9999 - loss: 5.9712e-04 - val_accuracy: 0.9999 - val_loss: 6.4588e-04
Epoch 20/20
2135/2135 ━━━━━━━━━━━━━━━━━━━━ 15s 7ms/step - accuracy: 0.9999 - loss: 5.8347e-04 - val_accuracy: 0.9999 - val_loss: 6.1157e-04
```

```
# periksa kinerja model CNN
scores = model.evaluate(X_test, y_test)
print("Akurasi: %.3f%%" % (scores[1] * 100))
```

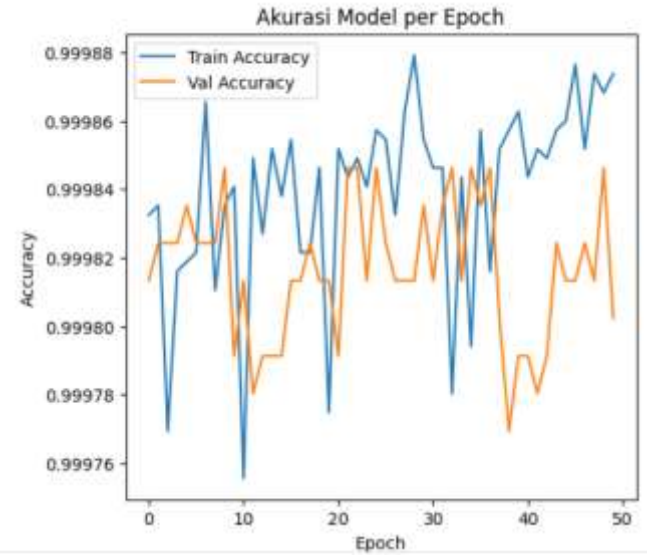
```
5692/5692 ━━━━━━━━━━━━━━━━━━━━ 11s 2ms/step - accuracy: 0.9999 - loss: 4.2746e-04
Akurasi: 99.989%
```

Gambar 11 Hasil Akurasi

Dapat dilihat pada gambar 11 hasil nilai akurasi yang didapat dari proses pelatihan dan pengujian model CNN adalah 99,989%.

3.2 Grafik Nilai Akurasi dan Validasi Akurasi

Setelah proses pelatihan dan pengujian menggunakan model CNN, dilakukan evaluasi untuk memperoleh nilai akurasi dan validasi akurasi. Grafik hasil pelatihan dan validasi akurasi dapat dilihat pada gambar berikut :

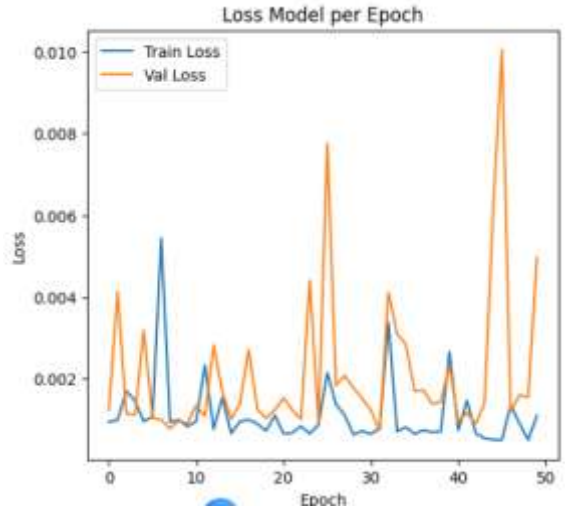


Gambar 12 Grafik Akurasi dan Validasi Akurasi

Nilai train accuracy dan val accuracy berada pada kisaran 99,98% dan menunjukkan pola stabil tanpa perbedaan signifikan. Hal ini menunjukkan bahwa model mampu melakukan generalisasi dengan baik dan tidak mengalami oerfitting.

3.3 Grafik nilai Loss dan Validasi Loss

Selanjutnya hasil pengujian dengan nilai loss dan validasi loss. Adapun grafiknya dapat dilihat sebagai berikut :



Gambar 13 Grafik nilai loss dan validasi loss

Nilai train loss dan validation loss berada dibawah 0,01 dan menunjukkan pola yang stabil. Perbedaan keduanya tidak signifikan, sehinga model CNN tidak mengalami overfitting maupun underfitting dan mampu melakukan generalisasi dengan baik.

3.4 Confusion Matrix

Hasil pengujian model CNN menunjukkan bahwa model mampu mengklasifikasikan jenis serangan jaringan dengan baik. Confusion matrix memperlihatkan bahwa kesalahan prediksi yang terjadi sangat minim.

Gambar 14 Confusion Matrix

Confusion matrix ini menunjukkan bahwa model CNN bekerja dengan sangat baik dalam mengenali pola serangan jaringan. Dua kelas terbesar, yaitu TCP Flood dan UDP Flood, hampir seluruhnya terklasifikasi benar dengan sangat sedikit kesalahan, yang menandakan bahwa model mampu mempelajari pola kedua serangan tersebut secara kuat dan konsisten. Pada kelas benign dan port scan memang terdapat beberapa kesalahan prediksi yakni 5,3,5,5 (dapat dilihat pada gambar 13).

Angka yang menunjukkan 53312 dan 83204 menunjukkan nilai data yang tinggi dan jelas sudah terklasifikasi dengan baik. Secara umum, hasil ini menggambarkan bahwa model memiliki kemampuan klasifikasi yang sangat tinggi dan dapat diandalkan untuk mendeteksi trafik normal maupun berbagai jenis serangan secara otomatis.

3.5 Hasil Confusion Matrix

Hasil dari confusion adalah nilai akurasi, precision, recall, dan f1-score.

Laporan Klasifikasi CNN:

	precision	recall	f1-score	support
Benign	0.74	0.78	0.76	37
Port Scan	0.85	0.85	0.85	33
TCP Flood	1.00	1.00	1.00	53312
UDP Flood	1.00	1.00	1.00	83209
accuracy			1.00	136591
macro avg	0.90	0.91	0.90	136591
weighted avg	1.00	1.00	1.00	136591

Gambar 15 Hasil Confusion

Laporan klasifikasi menunjukkan bahwa model CNN mampu mendeteksi trafik jaringan dengan akurat. Kelas TCP Flood dan UDP Flood memiliki nilai precision, recall, dan f1-score sebesar 1.00, sedangkan kelas benign dan port scan juga menunjukkan hasil yang baik. Secara keseluruhan, model mampu membedakan trafik normal dan serangan dengan tingkat ketetapan yang sangat tinggi.

4. KESIMPULAN

Berdasarkan hasil yang diperoleh dari proses klasifikasi DdoS dan Port Scanning menggunakan metode CNN yaitu :

1. Berdasarkan hasil penelitian, metode Convolutional Neural Network (CNN) dapat digunakan untuk mengklasifikasikan lalu lintas serangan jaringan pada honeypot server melalui tahapan pengumpulan data, preprocessing, pelabelan, serta proses pelatihan model untuk mengenali pola serangan secara otomatis.
2. Setelah menggunakan berbagai kombinasi pelatihan dan pengujian, didapat hasilnya maksimal pada pembagian data 60:40. Data diproses menggunakan metode CNN dengan mengatur jumlah epoch 20 dan batch size 128 didapatkan Hasil kinerja metode CNN pada perhitungan *Confusion Matrix* seluruh data kelas Normal, Serangan DdoS dan Port Scanning mendapatkan hasil pengujian dengan nilai akurasi 99.98%, presesi 89.58%, recall 90.77%, dan f1-score 90.16%. Dari hasil ini, dapat disimpulkan penggunaan model CNN yang dirancang dikategorikan baik.

REFERENSI

[1] Alfidzar, H., & Parga Zen, B. (n.d.-a). *Journal of Informatics, Information System, Software Engineering and Applications Implementasi HoneyPy Dengan Malicious Traffic Detection System (Maltrail) Guna Mendeteksi Serangan DOS Pada Server*. *4*(2), 32–045. https://doi.org/10.20895/INISTA.V4I2

[2] Solichin, A., & Nugroho, L. (2024). Deteksi Dini Gangguan Jaringan Distributed Denial Of Service (DDOS) Menggunakan Metode Shannon Entropy Pada Software Defined Network (SDN). *Jurnal Teknologi Informasi Dan Ilmu Komputer*, *11*(3), 461–474.

[3] Roslan, F. H. (2023). A Comparative Performance of Port Scanning Techniques. *Journal of Soft Computing and Data Mining*, *4*(2), 43–51.

[4] Njoera, Y. A. D., Hartawan, I. N. B., Ariana, A. A. G. B., & Krisna, E. D. (2024). The Analysis of Honeypot Performance Using Grafana Loki and ELK Stack Visualization. *Jurnal Info Sains: Informatika dan Sains*, *14*(03), 297-309.

[5] Septian Dwi Chandra, Hardian Oktavianto, & Ari Eko Wardoyo. (2024). Klasifikasi Malware Menggunakan Metode Convolutional Neural Network (CNN) Berbasis Website. *Jurnal Penelitian Teknologi Informasi Dan Sains*, *2*(2), 84–99.
[6] Yoshanda, M. I., & Alamsyah, A. (2023). Penerapan Model Hibrida CNN-GRU-BiLSTM-PCA Untuk Meningkatkan Akurasi Deteksi Serangan Jaringan Pada Intrusion Detection System. Indonesian Journal of Mathematics and Natural Sciences, 46(2), 61-67.
[7] Ali, S. Y., Farooq, U., Anum, L., Mian, N. A., Asim, M., & Alyas, T. (2024). Securing cloud environments: a Convolutional Neural Network (CNN) approach to intrusion detection system. Journal of Computing & Biomedical Informatics, 6(02), 295-308.
[8] Herdianto, H. (2022). Klasifikasi objek menggunakan metode convolutional neural network (CNN). SNASTIKOM, 1(01), 330-336.
[9] Kurniawan, A. A. (2020). TA: Intrusion Detection System Menggunakan Deep Learning untuk Deteksi Serangan DoS (Doctoral dissertation, Universitas Dinamika).
[10] AYENI, J. A. (2022). Convolutional Neural Network (CNN): The architecture and applications. *Applied Journal of Physical Science*, *4*(4), 42–50.
[11] Sathyanarayanan, S., & Tantri, B. R. (2024). Confusion matrix-based performance evaluation metrics. African Journal of Biomedical Research, 27(4S), 4023-4031.