PENERAPAN *ARTIFICIAL NEURAL NETWORK* DALAM DETEKSI SERANGAN PADA *WEB SERVER* APACHE

Arif Wicahyanto[1], Nurchim[2], Wijiyanto[3]

[123,] Program Studi Teknik Informatika, Universitas Duta Bangsa

Jl. Bhayangkara No.55, Tipes, Kec. Serengan, Kota Surakarta, Jawa Tengah 57154
[1]220103222@mhs.udb.ac.id, [2] nurchim@udb.ac.id, [3] wijiyanto@udb.ac.id

Abstract

Cyberattacks on websites are unavoidable, and government and campus websites are no exception. Cyber-attacks have a detrimental impact, ranging from the theft of sensitive data, disruption of website access, to financial losses. As cyberattack techniques become more sophisticated, rule-based and pattern-matching security systems face difficulties in detecting stealthy and adaptive attacks. Artificial Neural Network (ANN) is a machine learning method that has the ability to learn from complex attack patterns, identify invisible patterns and adapt to new attacks. This research aims to implement ANN in the form of a model to detect cyber attacks by using Apache web server access logs as a dataset data source. The research successfully built an ANN model to detect attacks on the Apache web server with an accuracy value of 0.9170.

Keywords : *cyberattack*, ANN, apache

Abstrak

Serangan siber terhadap website menjadi hal yang tidak dapat dihindarkan, tidak terkecuali website pemerintahan serta website kampus. Serangan siber memberikan dampak yang merugikan, mulai dari pencurian data sensitif, gangguan akses website, hingga kerugian finansial. Seiring dengan semakin canggihnya teknik serangan siber, sistem keamanan berbasis aturan dan pencocokan pola menghadapi kesulitan dalam mendeteksi serangan yang tersembunyi dan adaptif. *Artificial Neural Networ*k (ANN) adalah metode pembelajaran mesin yang memiliki kemampuan untuk belajar dari pola serangan yang kompleks, mengidentifikasi pola yang tidak terlihat dan beradaptasi dengan serangan baru. Penelitian ini bertujuan mengimplementasikan ANN dalam bentuk model untuk Smendeteksi serangan siber dengan menggunakan *access log web server* Apache sumber data *dataset*. Penelitian berhasil membangun model ANN untuk mendeteksi serangan pada web server Apache dengan nilai *accuracy* 0.9170.

Kata kunci : *serangan siber, ANN, apache*

1. PENDAHULUAN

Penyebaran informasi menggunakan media website menjadi strategi yang semakin dominan saat ini [1]. Media website dengan aksesibilitasnya yang luas dan kemudahan dalam pengelolaan konten menjadi platform efektif untuk menjangkau pengguna serta menyebarkan informasi secara cepat dan efisien.

Serangan siber yang terjadi terhadap website menjadi sebuah hal yang tidak bisa terhindarkan. Selama periode Januari sampai Juli 2023, Indonesia menghadapi serangan siber tertinggi di antara negara-negara di kawasan Asia Tenggara [2]. Website kampus dan pemerintahan tidak luput menjadi target serangan siber. Pelaku serangan menyasar website-website pemerintah karena pihak berwenang akan kesulitan melakukan pemblokiran website pemerintahan dan akademik [3].

Serangan siber memberikan dampak yang merugikan, mulai dari pencurian data sensitif, gangguan akses website, hingga kerugian finansial. Seiring dengan semakin canggihnya teknik serangan siber, sistem keamanan berbasis aturan dan pencocokan pola mengalami kesulitan dalam mendeteksi serangan yang tersembunyi dan adaptif [4].

Saat ini, teknologi *machine learning* menjadi solusi masalah di berbagai bidang seperti prediksi cuaca, pengenalan obyek, serta prediksi kemampuan pembelajaran [5]. *Artificial Neural Network* (ANN) yang merupakan bagian dari *machine learning* adalah sistem komputasi yang dirancang dengan meniru struktur dan fungsi otak manusia. ANN tersusun dari sejumlah besar unit pemrosesan dan saling terhubung, yang disebut neuron. Neuron bekerja bersama-sama untuk memproses informasi dan membuat keputusan [6].

Struktur ANN terdiri dari beberapa lapisan (layer), yaitu lapisan masukan, lapisan tersembunyi, dan lapisan keluaran. Proses *training* ANN melibatkan penyesuaian bobot dan bias dengan menggunakan data *training* hingga dapat memprediksi keluaran yang diinginkan.
Salah satu keunggulan utama ANN adalah kemampuannya untuk mempelajari pola yang kompleks dan beradaptasi dengan data baru [7]. Selain itu, ANN memiliki kemampuan untuk menggeneralisasi data *training* ke dalam data baru yang belum pernah ditemui sebelumnya, serta dapat menangani data yang tidak sempurna dan mengandung kebisingan.

Pada penelitian sebelumnya, *Web attack detection using deep learning models* [8] peneliti menggunakan 3 metode yaitu ANN, CNN, dan RNN [9] serta menggunakan *dataset* HTTP *DATASET* CSIC 2010. Model dengan akurasi tertinggi memiliki akurasi model 94 % dengan tingkat kesalahan 6% dengan menggunakan metode RNN. Pada penelitian *Intelligent Intrusion Detection System for Apache Web Server Empowered with Machine Learning Approaches* [10], peneliti menggunakan metode Naïve Bayes untuk dalam pembuatan model. Model yang dibangun memberikan akurasi 98.57%. Pada penelitian *Web Server Attack Detection using Machine Learning* [11], penulis menggunakan metode *Decision tree*, Ada-boost, K-nearest neighbor, dan SVM. Penelitian tersebut menghasilkan model dengan akurasi nilai 98 % pada metode dengan Decision Tree dan AdaBoost. Pada penelitian *A comparative analysis of various machine learning methods for anomaly* [12], peneliti membandingkan beberapa teknik *mechine learning* untuk mendeteksi anomali yang berkaitan dengan serangan siber pada perangkat IoT. Penelitian tersebut menunjukkan teknik ANN memiliki sedikit keunggulan dibandingkan dengan teknik k-NN, *Decision Tree*, dan SVM. Pada penelitian *A survey on neural networks for (cyber-) security and (cyber-) security of neural networks* [13], peneliti membahas penggunaan ANN pada keamanan siber, khususnya pada *Intrusion Detection Systems.* Peneliti membahas peningkatan kinerja model ANN dengan metode *dimensionality reduction*, optimalisasi *hyperparameter*, dan *dataset balancing.*

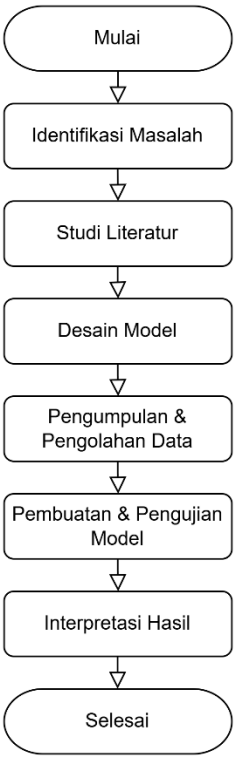
Melalui penerapan metode pengembangan eksperimental, penelitian ini bertujuan untuk menerapkan *Artificial Neural Network* (ANN) untuk mendeteksi serangan siber pada *web server* Apache. Apache merupakan *web server* yang dikenal karena stabilitas, keandalan, dan kemudahan konfigurasinya. Sampai saat ini, Apache masih menjadi salah satu *web server* yang populer digunakan di dunia [9]. Apache menerapkan arsitektur modular, di mana administrator server menambahkan atau menghapus modul sesuai kebutuhan. Modul-modul ini memberikan kemampuan tambahan, seperti dukungan untuk bahasa pemrograman, autentikasi, dan enkripsi [10].

Apache dapat menangani banyak permintaan web secara bersamaan dengan menggunakan *multithreading*, Apache dapat melayani beberapa pengguna secara bersamaan tanpa mengalami penurunan kinerja [11].

Fitur *virtual host* memungkinkan pengelola untuk menjalankan beberapa *website* pada *server* yang sama dengan konfigurasi dan pengaturan yang terpisah

Penelitian ini menggunakan *dataset* yang dihasilkan dari *access log web* [14] *server* Apache yang bersumber dari data akses website pada *virtual host*. Penelitian dengan mengimplementasikan ANN pada model yang dibangun, diharapkan dapat memberikan kontribusi pada peningkatan keamanan dan memberikan solusi alternatif dalam melawan ancaman siber yang terus berkembang.

2. METODOLOGI PENELITIAN



Gambar 1. Tahap penelitian

2.1. Identifikasi Masalah

Penelitian mengenai penerapan *Artificial Neural Network* (ANN) dalam deteksi serangan pada web server Apache dilatarbelakangi oleh beragamnya jenis serangan siber yang dialami *web server* Apache. Kompleksitas dan variasi serangan ini menuntut adanya model deteksi yang adaptif dan akurat. Dalam hal penerapan ANN, masalah performa model, pemilihan parameter yang optimal (*hyperparameter tuning*) menjadi materi yang dapat diteliti.

2.2. Studi Literatur

Studi literatur melalui buku, jurnal dan sumber internet dilakukan guna memperoleh informasi mengenai *Artificial Neural Network* (ANN), pengelolaan dan konfigurasi web server Apache.

2.3. Desain Model

Perancangan model *Artificial Neural Network* (ANN) dengan menentukan sumber data, *feature* yang digunakan dan arsitektur model.

2.4. Pengumpulan Data

Proses pengumpulan data penelitian yang bersumber dari *access log web server* Apache.

2.5. Pembuatan Model

Proses penyiapan data latih yang dipergunakan dalam pembuatan dan pelatihan model.

2.6. Interpretasi Hasil

Tahap interpretasi hasil proses pembuatan model dan hasil pengujian model dengan menggunakan data tes.

3. HASIL DAN PEMBAHASAN

Penelitian ini mengadopsi metode pengembangan eksperimen. Peneliti membangun model dengan arsitektur ANN dengan *dataset* bersumber dari *access log web server* Apache. Penelitian menggunakan bahasa pemrograman Python dengan bantuan pustaka TensorFlow [15], Pandas [16], dan NumPy [17].

Python merupakan sebuah bahasa pemrograman jenis interpretasi tingkat tinggi yang sangat populer khususnya di bidang ilmu komputer, data sains, dan pengembangan perangkat lunak [18]. Sintaks Python yang sederhana dan mudah dibaca, serta filosofi "*readability*" [19] menjadikannya bahasa pemrograman yang ideal untuk pemrogram pemula maupun yang berpengalaman.

TensorFlow merupakan sebuah *framework open source* yang dikembangkan tim Google Brain dengan tujuan untuk membangun dan menerapkan model machine learning dan deep learning. TensorFlow memanfaatkan grafik aliran guna merepresentasikan perhitungan, keadaan bersama (shared state), dan operasi yang mengubah keadaan tersebut. TensorFlow memiliki beberapa kelebihan, antara lain fleksibilitas dan skalabilitas yang memungkinkan TensorFlow berjalan pada berbagai platform, termasuk CPU, GPU, atau bahkan pada perangkat seluler [17].

Pustaka Numpy digunakan untuk komputasi ilmiah dan numerik. Pustaka ini pertama kali dikembangkan oleh Travis Oliphant pada tahun 2005 dan telah menjadi standar untuk operasi

array di Python. NumPy memberikan dukungan untuk array multidimensi dan matriks, serta berbagai fungsi matematika tingkat tinggi untuk operasi pada data *array* [18].

Pandas adalah pustaka dalam bahasa pemrograman Python yang digunakan dalam hal manipulasi dan analisis data [19]. Pandas dibangun di atas pustaka NumPy dan memberikan fasilitas dua struktur data utama yaitu *series* dan *data frame*

Tahap-tahap yang dilakukan pada penelitian ini sebagai berikut:

3.1. Log Mining

Log Mining [20] merupakan proses pengumpulan sumber data primer yang berasal dari *access log web server* Apache. Pada setiap *virtual host* [21] di web server Apache, dilakukan konfigurasi sehingga diperoleh data *access log* dengan format yang sama dan menyatu pada sebuah file yang bernama *access_log*. Hal tersebut dicapai dengan menggunakan bantuan mod_log pada web server Apache. Adapun format yang digunakan sebagai berikut:

```
1. LogFormat "%v %h %t \"%r\" %>s %b" combined-vhost
2. CustomLog /usr/local/apache/logs/access_log combined-
   vhostLogFormat "%v %h %t
```

Gambar 2. Konfigurasi *access log* apache

Konfigurasi yang dilakukan akan menghasilkan data akses *web server* Apache dengan format: domain *virtual host*, alamat IP sumber, waktu akses, alamat URL yang diakses, *respons server*, dan ukuran data *respons*. Data *access log web server* Apache yang diperoleh kemudian dikonversi ke dalam file berformat CSV (*comma separated values*). Data *access log* yang diperoleh sebanyak 117776 baris data.

TABEL I. DATA LOG

No	Atribut	Keterangan	Tipe Data
1	*Vhost*	Domain virtual host	String
2	*Host*	Sumber IP pengakses	String
3	*Datetime*	Waktu akses	String
4	*URL*	*URL yang diakses*	String
5	*URL*	*URL yang diakses*	String
6	*Respons*	*Kode respon terhadap permintaan*	Numerik
7	*Respons Size*	*Ukuran respon*	Numerik

Berikut contoh data *access log* yang diperoleh pada proses *log mining*.

domain	ip	time	url	response	method	response_size
pulmonologi.fk.uns.ac.id	132.145.66.116	10/Aug/2024:11:23:08	/	200	GET	31925
parasitologi.fk.uns.ac.id	52.167.144.219	10/Aug/2024:11:23:09	/wp-content/plugins/w3-total-cache/pub/js/lazy...	200	GET	2356
kardiologi.fk.uns.ac.id	203.6.149.89	10/Aug/2024:11:23:12	/wp-cron.php?doing_wp_cron=1723263792.31815099...	200	POST	0
kardiologi.fk.uns.ac.id	72.14.201.144	10/Aug/2024:11:23:12	/dr-habibie-arifianto-spjpk-m-kes-fiha/	200	GET	9534
pulmonologi.fk.uns.ac.id	213.180.203.173	10/Aug/2024:11:23:34	/robots.txt	200	GET	11667

Gambar 3. Data hasil proses *log mining*

3.2. Data *Preprocessing*

Data dalam format CSV dibaca kemudian dilakukan proses penghapusan data yang terduplikasi dan data yang tidak lengkap. Selanjutnya dipilih 4 jenis data sebagai *feature* dalam pembuatan model yaitu *url, response, method,* dan *response_size*. Berdasar 4 jenis data tersebut, dilakukan proses pelabelan untuk menentukan permintaan ke web server yang berupa serangan atau yang bukan merupakan serangan. Permintaan yang berupa serangan diberikan label 1 sedang untuk permintaan yang bukan serangan diberikan label 0 pada kolom harm.

```
1. df = pd.read_csv('LogAkses.csv')
2. df.drop_duplicates(inplace=True, \
3. subset=['domain', 'ip','url', 'time', \
4. 'response', 'method', \
5. 'response_size'])
6. df = df.drop(columns=['domain', 'ip', 'time', 'no'])
```

Gambar 4. Proses data impor data dan pembersihan data

Proses tersebut menghasilkan 85620 baris data log akses. Dari data tersebut, 4 jenis data dipilih sebagai *feature* dalam pembuatan model yaitu url, response, method, dan response_size.

url	response	method	response_size	harm
/	200	GET	31925	0
/wp-content/plugins/w3-total-cache/pub/js/lazy...	200	GET	2356	0
/wp-cron.php?doing_wp_cron=1723263792.31815099...	200	POST	0	0
/dr-habibie-arifianto-spjpk-m-kes-fiha/	200	GET	9534	0
/robots.txt	200	GET	11667	0

Gambar 5. Data *feature*

Berdasar 4 jenis data tersebut, dilakukan proses pelabelan untuk menentukan permintaan ke *web server* yang berupa serangan atau yang bukan merupakan serangan. Permintaan yang berupa serangan diberikan label 1 sedang untuk

permintaan yang bukan serangan diberikan label 0 pada kolom harm. Pada data url yang bertipe *string*, dilakukan konversi ke numerik dengan melakukan fungsi *hash* pada bagian akhir data url.

```
1. def extract_last_part(url):
      parts = url.split("/")
      return parts[-2]+"/"+parts[-1]
2. def hash_last_part(last_part):
      hash_object = hashlib.sha256(last_part.encode())
      return int(hash_object.hexdigest(), 16)
3. df['url'] = \
4. df['url'].apply(extract_last_part).apply(hash_last_part)
```

Gambar 6. Proses konversi data url

Pada data HTTP *method* yang berupa *string* 'GET', 'POST', 'HEAD', 'PUT', dan 'OPTIONS', dilakukan proses label *encoding* menggunakan LabelEncoder [22].

```
1. from sklearn.preprocessing import LabelEncoder
2. le = LabelEncoder()
3. df['method'] = le.fit_transform(df['method'])
```

Gambar 7. Proses pelabelan data *method*

url	response	method	response_size	harm
6258676612188541924408494496769033620815304534...	200	0	31925	0
9362153947931721267206911884186103269905251517...	200	0	2356	0
2282251796761671363683145707276672505732420991...	200	3	0	0
4638412444704168652943820571781794405540833984...	200	0	9534	0
7855689986165333378320790347046269716084725696...	200	0	11667	0

Gambar 8. *Dataset* dalam bentuk numerik

Selanjutnya data dibagi ke dalam *feature* pada variabel X dan target pada variabel y, dilanjutkan pembagian data ke dalam set data *training* dan set data *test*. Sejumlah 20% data sebagai data *test* dan 80% data sebagai data *training*.

```
1. X = df.iloc[:, 0:-1].values
2. y = df.iloc[:, -1].values
3. from sklearn.model_selection import train_test_split
4. X_train, X_test, y_train, y_test = \
5. train_test_split(X, y, test_size = 0.2, random_state = 0)
```

Gambar 9. Pembagian *dataset* ke dalam data *training* dan data *test*

Data yang sudah dibagi ke dalam data *training* dan data *test* diskala untuk menghindari perbedaan nilai yang signifikan pada *dataset* yang digunakan.

```
1. from sklearn.preprocessing import StandardScaler
2. sc = StandardScaler()
3. X_train = sc.fit_transform(X_train)
4. X_test = sc.transform(X_test)
```

Gambar 10. Proses *scaling* data

Berikut hasil akhir *dataset training* dari proses data *preprocessing* yang berupa 4 jenis data *feature* dan 1 data target.

```
1  display(X_train[:1,:])
2  display(y_train[:1])
✓ 0.0s

array([[-1.5253264 ,  0.65385006, -0.30804035, -0.21675205]])

array([0], dtype=int64)
```

Gambar 11. Contoh data *feature* dan target dari *dataset*

3.3. *Pembuatan* Model

Pembuatan model dimulai dengan memuat pustaka Keras dari TensorFlow. Model ANN dibangun dengan arsitektur *sequential* [23], yang berarti lapisan-lapisan diorganisir secara berurutan. Model memiliki dua lapisan yaitu:

Lapisan pertama (*hidden layer*), merupakan lapisan tersembunyi yang menggunakan 6 neuron dengan fungsi aktivasi ReLU [24]. Lapisan pertama memiliki peran mengekstrak fitur-fitur penting dari data input. Fungsi aktivasi ReLU membantu model untuk mempelajari pola non-linear dalam data.

Lapisan kedua (*output layer*), merupakan lapisan *output* memiliki 1 neuron dengan fungsi aktivasi *sigmoid*. Neuron tersebut menghasilkan *output* yang merupakan probabilitas dari kelas 'serangan', yaitu nilai antara 0 dan 1. Fungsi aktivasi *sigmoid* memungkinkan model untuk mengeluarkan probabilitas yang mewakili keyakinan model terhadap hasil prediksi.

Model ini dikompilasi dengan menggunakan *optimizer* Adam , yang merupakan algoritma yang efisien untuk mengoptimalkan bobot model selama *training*. Fungsi *loss* yang digunakan adalah *binary cross-entropy* untuk mengukur perbedaan antara probabilitas prediksi model dan nilai target aktual. Model ini juga dikonfigurasi untuk melacak metrik akurasi selama *training* dan evaluasi.

```
1. ann = tf.keras.models.Sequential()
2. ann.add(tf.keras.layers.Dense(units=6, activation='relu'))
3. ann.add(tf.keras.layers.Dense(units=1, activation='sigmoid'))
4. ann.compile(optimizer = 'adam', loss = 'binary_crossentropy', \
5. metrics = ['accuracy'])
```

Gambar 12. Pembuatan model ANN

3.4. *Training* Model

Model yang telah diinisialisasi kemudian dilatih menggunakan *dataset* yang sudah disiapkan. Proses *training* ini bertujuan untuk mengoptimalkan bobot dan bias model sehingga

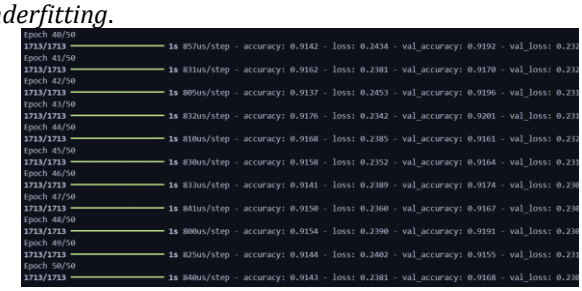
model dapat membuat prediksi yang akurat berdasarkan data input.

```
1. history = ann.fit(X_train, y_train, epochs=50,\
2. batch_size=32, validation_split=0.2)
```

Gambar 13. *Training* model ANN

Kode tersebut melatih model dengan 50 *epoch* serta memproses 32 sampel data dalam setiap *batch*, dan memisahkan 20% dari data *dataset* untuk validasi.

Variabel *history* menyimpan riwayat data *training*. Variabel tersebut digunakan untuk menganalisis kinerja model dan mengetahui apakah model mengalami *overfitting* atau *underfitting*.



Gambar 14. Proses *training* model

3.5. Pengujian Model

Model yang telah dilatih dengan data *training* kemudian diuji dengan 2 buah jenis data *web request*, yaitu 1 data yang merupakan serangan dan data yang bukan merupakan serangan.

```
1. dfcoba1    = pd.DataFrame([["//delete4.php", "404", "0",
   "2316"]])
2. dfcoba1[0] = dfcoba1[0].apply(extract_last_part)
3. dfcoba1[0] = dfcoba1[0].apply(hash_last_part)
4. display(dfcoba1)
5. print(ann.predict(sc.transform(dfcoba1)))
6. dfcoba2    = pd.DataFrame([["/sskegiatan.pdf", "200", "0",
   "13316"]])
7. dfcoba2[0] = dfcoba2[0].apply(extract_last_part)
8. dfcoba2[0] = dfcoba2[0].apply(hash_last_part)
```

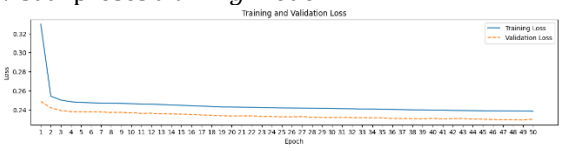
Gambar 15. Pengujian model dengan data contoh

Model yang dibuat memberikan nilai 0.8110605 untuk jenis data yang berupa serangan dan nilai 0.03470638 untuk jenis data bukan serangan.

Gambar 16. Hasil pengujian model dengan data contoh

3.6. Pengamatan Proses *Training* Model

Pada proses *training*, data *training* pada tiap epoch disimpan di sebuah variabel "*history*". Data pada variabel tersebut digunakan untuk membuat grafik, untuk melakukan pengamatan secara visual proses *training* model.



Gambar 17. *Grafik riwayat training berupa training loss dan validation loss*

Grafik di atas menggambarkan *training loss* dan *validation loss* selama proses *training* model. Garis tanpa putus menunjukkan *training loss*, sedangkan garis putus-putus menunjukkan *validation loss*. Dari hasil pengamatan visual, *training loss* dan *validation loss* menurun secara stabil selama awal *epoch*. Hal tersebut menunjukkan model mengalami proses belajar yang efektif.

Training loss menunjukkan penurunan yang cepat pada awal proses *training*, hal tersebut menunjukkan model mempelajari cara menyesuaikan diri dengan data *training*. Setelah sekitar 10 *epoch*, *training loss* menjadi datar, hal ini menunjukkan model telah mencapai titik di mana tidak terjadi peningkatan kinerja yang signifikan pada data *training*.

Validation loss menunjukkan penurunan yang lebih bertahap dan tetap relatif stabil sepanjang proses *training*. Hal tersebut menunjukkan bahwa model melakukan generalisasi dengan baik pada data yang tidak terlihat dan tidak mengalami *overfitting* terhadap data *training*.

Berdasarkan grafik tersebut dapat disimpulkan bahwa model telah dilatih secara efektif dan telah mencapai keseimbangan pada proses belajar dari data *training* dan melakukan generalisasi pada data baru.

Gambar 18. Grafik riwayat *training* berupa *training accuracy* dan *validation accuracy*

Grafik di atas menampilkan *training accuracy* dan *validation accuracy* selama proses *training* model. Garis tanpa putus menunjukkan *training accuracy*, sedangkan garis putus-putus menunjukkan *validation accuracy*.

Dari hasil pengamatan visual, terjadi peningkatan akurasi yang signifikan pada tahap awal *training*. *Training accuracy* meningkat tajam hingga sekitar epoch ke-5, lalu mulai datar di sekitar 0,915. Hal ini menunjukkan bahwa model telah mencapai titik di mana model tidak banyak belajar lagi dari data *training*.

Validation *accuracy* juga meningkat pada awal proses *training*, tetapi dengan fluktuasi yang lebih besar dibandingkan dengan *training accuracy*. Hal tersebut menunjukkan bahwa pada tahap *epoch* tersebut model mengalami *overfitting* terhadap data *training*.

Training accuracy tidak meningkat secara signifikan setelah epoch ke-5, *validation accuracy* terus meningkat lalu dan menjadi mendatar di sekitar nilai 0,915. Hal ini menunjukkan bahwa model telah mencapai kinerja yang cukup baik pada data yang tidak terlihat.

Secara keseluruhan, grafik tersebut menunjukkan bahwa model telah dilatih secara efektif dan mencapai tingkat akurasi yang baik pada data *training* maupun validasi. Nilai *validation accuracy* yang tinggi menunjukkan bahwa model mampu menggeneralisasi dengan baik pada data baru.

3.7. Pengujian Data *Test*

Untuk mengevaluasi performa model *Artificial Neural Network* (ANN) yang telah dilatih, dilakukanlah proses prediksi menggunakan data tes yang dipisahkan secara acak yang berjumlah 17.124 atau sebesar 20% dari keseluruhan *dataset*. Data *test* memiliki fungsi sebagai data yang belum pernah ditemui oleh model selama fase *training*, sehingga memungkinkan menilai kemampuan generalisasi model dalam menghadapi data baru yang tidak dikenalnya.

Proses prediksi dilakukan dengan menggunakan data *test* sehingga akan diperoleh nilai yang dapat digunakan untuk melakukan evaluasi model.

```
1. y_pred = ann.predict(X_test)
2. y_pred = (y_pred > 0.5)
3. print(np.concatenate((y_pred.reshape(len(y_pred),1),
   y_test.reshape(len(y_test),1)),1))
```

Gambar 19. Prediksi dengan data *test*

```
536/536 ━━━━━━━━━━━━━━━━━━━━ 0s 363us/step
[[0 0]
 [0 0]
 [0 0]
 ...
 [0 0]
 [0 0]
 [1 1]]
```

Gambar 20. Hasil prediksi dengan data *test*

Hasil proses prediksi menggunakan model yang telah dibangun dapat digunakan untuk membuat *confiusion matix* [25].

```
1. from sklearn.metrics import confusion_matrix,
   accuracy_score
2. cm = confusion_matrix(y_test, y_pred)
3. print(cm)
```

Gambar 21. Pembuatan *confiusion matrix*

TABEL II. *CONFUSION MATRIX*

	Predicted Positive	*Predicted Negative*
Actual Positive	12100 *(True Positif/TP)*	889 *(False Negative/FN)*
Actual Negative	533 *(False Negatif/FN)*	3602 *(True Negative/TN)*

Tabel *confusion matrix* menghasilkan nilai *accuracy, precision, recall dan F1-Score*. Nilai-nilai tersebut digunakan untuk mengevaluasi kinerja model yang dibangun.

TABEL III. NILAI EVALUASI MODEL

No	Matrik	Nilai
1	*Accuracy*	0.9170
2	*Precision*	0.8020
3	*Recall*	0.8711
4	F1-*Score*	0.8351

Nilai *accuracy* yang diperoleh adalah 0.9170. Semakin tinggi nilai akurasi yang tinggi menunjukkan model memiliki kinerja yang baik secara umum.

Nilai *precision* yang diperoleh adalah 0.8020 yang bermakna dari semua *request* yang diprediksi sebagai berbahaya, 80.2% benar-benar berbahaya. Angka tersebut relatif rendah dibandingkan nilai akurasi. Hal tersebut menunjukkan model cenderung menghasilkan *false positives* (menandai *request* yang aman sebagai berbahaya).

Nilai *recall* yang dihasilkan adalah 0.8711 yang bermakna dari semua *request* yang sebenarnya berbahaya, model berhasil mendeteksi 87.11%. Angka ini lebih tinggi dari presisi, yang menunjukkan model lebih baik dalam mendeteksi *request* berbahaya daripada menghindari klasifikasi *false positives*. Dari nilai *recall*, masih terdapat beberapa *request* berbahaya yang terlewat (*false negatives*).

Nilai F1-*Score* yang dihasilkan adalah 0.8351 yang menunjukkan keseimbangan antara *precision* dan *recall*. Karena nilai presisi lebih rendah dari *recall*, maka nilai F1-score lebih mendekati nilai presisi.

4. KESIMPULAN DAN SARAN

Penelitian yang dilakukan berhasil membuat model *Artificial Neural Network* (ANN) untuk mendeteksi adanya serangan pada *web server Apache*. Model dibangun menggunakan *dataset* yang dihasilkan dari data *access log web server* Apache.

Penelitian yang dilakukan berhasil membuat model *Artificial Neural Network* (ANN) yang dibangun dengan menggunakan *dataset* dari *access log web server* Apache. Model yang dibangun menghasilkan nilai *accuracy* 0.9170, nilai *precision* 0.8020, *recall* 0.8711 dan F1-*Score* 0.8351.

Model memiliki kinerja yang baik dalam hal akurasi dan *recall*, nilai presisi yang rendah yang berakibat model cenderung menandai *request* yang aman sebagai berbahaya, masih memerlukan pengembangan pada penelitian selanjutnya.

5. UCAPAN TERIMA KASIH

Ucapan terima kasih kami sampaikan kepada Bapak Nurchim, M.Kom dan Bapak Wijiyanto, M.Kom yang telah membantu dan membimbing dalam penelitian ini serta seluruh civitas Fakultas Ilmu Komputer, Universitas Duta Bangsa Surakarta.

Daftar Pustaka:

[1] C. Siahaan, J. A. Tampubolon, and N. B. Sinambela, "Diseminasi Informasi Melalui Media Online Sebagai Transformasi Media Konvensional," *J. Signal*, vol. 9, no. 2, p. 322, 2021, doi: 10.33603/signal.v9i2.6288.

[2] J. Chakravarti, "Indonesia Hardest Hit by Cyberattacks in the Region," *Bank Info Security*, 2023. https://www.bankinfosecurity.asia/indonesia-hardest-hit-by-cyberattacks-in-region-a-22720 (accessed Oct. 26, 2024).

[3] Kompas, "Judi Online Sasar Website Kampus, Pakar Ungkap Ada Celah Keamanan," 2023. https://www.kompas.com/tren/read/2023/09/11/193000165/judi-online-sasar-website-kampus-pakar-ungkap-ada-celah-keamanan?page=all (accessed Oct. 24, 2024).

[4] P. P. Putra and D. Toresa, *Keamanan informasi dan jaringan komputer*. 2021.

[5] Wijiyanto, A. Marjuni, A. Z. Fanani, and R. S. Basuki, "Enhancing Machine Learning Algorithms Using Recursive Feature Elimination For Student Performance Prediction," in *2024 International Seminar on Application for Technology of Information and Communication (iSemantic)*, 2024, pp. 409–414. doi: 10.1109/iSemantic63362.2024.10762675.

[6] R. Qamar and B. Zardari, "Artificial Neural Networks: An Overview," *Mesopotamian J. Comput. Sci.*, vol. 2023, pp. 130–139, 2023, doi: 10.58496/MJCSC/2023/015.

[7] M. G. M. Abdolrasol *et al.*, "Artificial neural networks based optimization techniques: A review," *Electron.*, vol. 10, no. 21, 2021, doi: 10.3390/electronics10212689.

[8] J. I. Christy Eunaicy and S. Suguna, "Web attack detection using deep learning models," *Mater. Today Proc.*, vol. 62, pp. 4806–4813, 2022, doi: 10.1016/j.matpr.2022.03.348.

[9] N. Giarsyani, "Komparasi Algoritma Machine Learning dan Deep Learning untuk Named Entity Recognition : Studi Kasus Data Kebencanaan," *Indones. J. Appl. Informatics*, vol. 4, no. 2, p. 138, 2020, doi: 10.20961/ijai.v4i2.41317.

[10] M. U. Ullah, A. Hassan, M. Asif, M. S. Farooq, and M. Saleem, "Intelligent Intrusion Detection System for Apache Web Server Empowered with Machine Learning Approaches," *Int. J. Comput. Innov. Sci.*, vol. 1, no. 1, pp. 21–27, 2022, [Online]. Available: https://ijcis.com/index.php/IJCIS/article/view/13

[11] S. Saleem, M. Sheeraz, M. Hanif, and U. Farooq, "Web Server Attack Detection using Machine Learning," *1st Annu. Int. Conf. Cyber Warf. Secur. ICCWS 2020 - Proc.*, pp. 1–7, 2020, doi: 10.1109/ICCWS48432.2020.9292393.

[12] M. M. Inuwa and R. Das, "A comparative analysis of various machine learning methods for anomaly detection in cyber attacks on IoT networks," *Internet of Things (Netherlands)*, vol. 26, no. January, p. 101162, 2024, doi: 10.1016/j.iot.2024.101162.

[13] M. Pawlicki, R. Kozik, and M. Choraś, "A survey on neural networks for (cyber-) security and (cyber-) security of neural networks," *Neurocomputing*, vol. 500, pp. 1075–1087, 2022, doi: 10.1016/j.neucom.2022.06.002.

[14] T. Valentine, "Installing and Using the Apache Web Server," 2023, pp. 145–160. doi: 10.1007/978-1-4842-9792-6_9.

[15] F. J. John Joseph, S. Nonsiri, and A. Monsakul, "Keras and TensorFlow: A Hands-On Experience," 2021, pp. 85–111. doi: 10.1007/978-3-030-66519-7_4.

[16] S. K. S. Joy, F. Ahmed, A. H. Mahamud, and N. C. Mandal, *An Empirical Studies on How the Developers Discussed about Pandas Topics*. 2022. doi: 10.48550/arXiv.2210.03519.

[17] R. Hazrat, "The numpy Library," 2024, pp. 183–209. doi: 10.1007/978-3-031-49780-3_7.

[18] M. R. S. Alfarizi, M. Z. Al-farish, M. Taufiqurrahman, G. Ardiansah, and M. Elgar, "Penggunaan Python Sebagai Bahasa Pemrograman untuk Machine Learning dan Deep Learning," *Karya Ilm. Mhs. Bertauhid (KARIMAH TAUHID)*, vol. 2, no. 1, pp. 1–6, 2023.

[19] M. Segedinac, G. Savić, I. Zeljković, J. Slivka, and Z. Konjović, "Assessing code readability in Python programming courses using eye-tracking," *Comput. Appl. Eng. Educ.*, vol. 32, no. 1, p. e22685, 2024, doi: https://doi.org/10.1002/cae.22685.

[20] M. Patankar, G. Dangare, and P. Mathurkar, "A STUDY OF WEB USAGE MINING: FINDING PATTERNS IN WEB LOG DATA," p. 2022, Aug. 2024.

[21] T. Valentine, "Installing and Using the Apache Web Server," 2023, pp. 145–160. doi: 10.1007/978-1-4842-9792-6_9.

[22] M. S. Hashim and A. adil Yassin, "Breast Cancer Prediction Using Soft Voting Classifier Based on Machine Learning Models," *IAENG Int. J. Comput. Sci.*, vol. 50, pp. 705–714, May 2023.

[23] H. Zhao, Y. Feng, H. Koide, and K. Sakurai, "An ANN Based Sequential Detection Method for Balancing Performance Indicators of IDS," in *2019 Seventh International Symposium on Computing and Networking (CANDAR)*, 2019, pp. 239–244. doi: 10.1109/CANDAR.2019.00039.

[24] S. Zhang and J. Lu, "Deep Network Approximation : Beyond ReLU to Diverse Activation Functions," vol. 25, pp. 1–39, 2024.

[25] Maruli Tua Silaen, "Klasifikasi Karakteristik Kepribadian Siswa Berdasarkan the Big Five Personality Dengan Menggunakan Metode K- Nearest Neighbor (Knn)," *J. Inform. dan Rekayasa Elektron.*, vol. 6, no. 1, pp. 121–129, 2023, doi: 10.36595/jire.v6i1.860.