Jurnal Teknik Informatika (JUTIF)
Vol. 5, No. 5, October 2024, pp. 1451-1459

DOI: https://doi.org/10.52436/1.jutif.2024.5.5.2446
p-ISSN: 2723-3863
e-ISSN: 2723-3871

IMPLEMENTATION OF DEEP LEARNING FOR DETECTING PHISHING ATTACKS ON WEBSITES WITH COMBINATION OF CNN AND LSTM

Ahmad Raihan[*1], Mohammad Fadhli[2], Lindawati[*3]

[1,2,3]Telecommunication Engineering, Politeknik Negeri Sriwijaya, Indonesia
Email: [1]ahmadraihan154@gmail.com, [2]mohammad.fadhli@polsri.ac.id, [3]lindawati@polsri.ac.id

(Article received: July 04, 2024; Revision: July 18, 2024; published: October 29, 2024)

Abstract

Phishing attacks represent significant cyber threats to internet users, particularly on websites. These attacks are conducted by perpetrators seeking to acquire victims' data by impersonating legitimate websites. To address this threat, a solution is proposed using deep learning with a combined algorithm of convolutional neural network and long short-term memory. The research methodology included data collection comprising phishing and legitimate website links, pre-processing through tokenization, padding, and labeling, and splitting data into training and testing sets. The models were then trained, and grid search was employed to identify the optimal hyperparameters for each algorithm. The algorithm's performance was calculated by accuracy, precision, recall, and F1-score metrics. The outcomes indicated that using the combination algorithm achieved 95.63% accuracy, 94.60% precision, 96.81% recall, and 95.78% f1-score. This paper concludes the proposed algorithm is effective in detecting phishing attacks on websites.

Keywords: *convolutional neural network, deep learning, long short-term memory, phishing attacks, website security.*

IMPLEMENTASI *DEEP LEARNING* DALAM MENDETEKSI SERANGAN *PHISHING* PADA *WEBSITE* DENGAN KOMBINASI CNN DAN LSTM

Abstrak

Serangan *phishing* merupakan salah satu jenis serangan siber yang mengancam pengguna internet, terutama pada *website*. Serangan ini dilakukan oleh pelaku untuk mendapatkan data pribadi korban dengan cara meniru situs web asli. Untuk mengatasi ancaman ini, diusulkan sebuah solusi menggunakan *deep learning* dengan algoritma gabungan *convolutional neural network* dan *long short-term memory*. Metode Penelitian ini mencakup beberapa tahapan, dimulai dari pengumpulan data yang terdiri atas tautan situs web *phishing* dan situs web asli. Tahap berikutnya adalah *pre-processing* data yang meliputi tokenisasi, *padding*, dan pelabelan. Data kemudian dibagi menjadi data latih dan data uji. Selanjutnya, Model dilatih dengan mencari kombinasi *hyperparameter* yang optimal menggunakan metode *grid search*. Setelah itu, dilakukan pengujian dan evaluasi untuk menilai kinerja algoritma menggunakan metrik seperti *accuracy*, *precision*, *recall*, dan *f1-score*. Temuan dari penelitian ini memperlihatkan bahwa algoritma gabungan ini mencapai nilai *accuracy* 95,63%, *precision* 96,81%, *recall* 96.81% dan *f1-score* 95,78%. Penelitian ini menyimpulkan bahwa penggunaan algoritma gabungan efektif dalam mendeteksi serangan *phishing* pada *website*.

Kata kunci: *convolutional neural network, deep learning, keamanan website, long short-term memory, serangan phishing.*

1. PENDAHULUAN

Hampir setiap bisnis dan individu memanfaatkan internet untuk mendukung operasional di era modern ini. Dengan semua pengaruh, peluang, dan peningkatan penggunaan internet, banyak ancaman *online* yang juga bermunculan. *Phishing* adalah salah satu jenis ancaman yang umum saat ini [1], [2]. Serangan *phishing* dapat mencuri informasi pribadi dan pengguna melalui situs web palsu yang dibuat dengan meniru situs web asli [3]–[5]. Pelaku menggunakan situs web palsu, lalu memalsukan *uniform resource locators* (URL) untuk mencuri informasi pengguna dari halaman *login* atau pembayaran. Selain itu, pelaku juga menggunakan email, SMS, pesan suara, kode QR, dan aplikasi palsu untuk melakukan tindakan *phishing* [6], [7].

Serangan *phishing* telah menyebabkan masalah besar dan berdampak negatif pada korban, tidak hanya finansial tetapi dalam hal reputasi dan keamanan [8]. Di Indonesia, lembaga Indonesia Anti-Phishing Data Exchange (IDADX) mencatat 106.806 laporan serangan phishing sejak lima tahun terakhir dari tahun 2018. Pada tahun 2023, serangan ini tercatat mencapai 65.525 dengan laporan serangan tertinggi terjadi pada bulan Februari 2023 sebanyak 15.050 dan laporan serangan terendah terjadi pada bulan November 2023 sebanyak 1.729. Pada kuartal keempat ini, terdapat 8.161 serangan dengan menggunakan 53 nama domain. Industri yang paling terkena dampak adalah industri media sosial mencapai 64,34% [9]. Oleh karena itu, *phishing* telah menyebar melalui iklan atau pesan kepada korban di media sosial [10].

Kemajuan pesat teknologi seperti *machine learning* (ML) dan *deep learning* (DL) dalam ranah *artificial intelligence* (AI) terbukti penting mengatasi ancaman siber. Beragam kemampuan AI, mulai dari pengenalan pola hingga protokol keamanan adaptif, menjadikannya komponen yang tak tergantikan dalam sistem pertahanan siber modern. Melalui kemampuan AI untuk menganalisis data dalam jumlah besar secara *real-time* maka deteksi dan respons terhadap serangan siber bisa dilakukan secara cepat dan akurat sehingga mengurangi potensi kerugian serta meningkatkan keamanan digital secara keseluruhan [11].

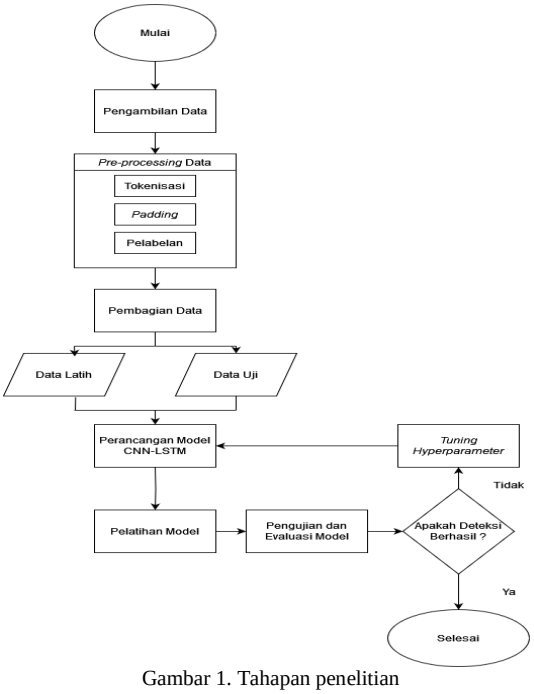
ML memiliki kemampuan yang andal saat memproses data berbasis teks, seperti URL dalam mendeteksi serangan *phishing* dari situs web. Sebagai contoh, penelitian yang dilakukan oleh Shouq Alnemari menggunakan beberapa algoritma ML yang berbeda, yaitu *decision tree* (DT), *random forest* (RF), dan *support vector machine* (SVM). Hasilnya menunjukkan bahwa model dengan algoritma RF mencapai akurasi tertinggi, sekitar 96,86% sebelum normalisasi dan 97,3% setelah normalisasi. Selain itu, model dengan algoritma lainnya juga mencapai akurasi tinggi, dengan nilai di atas 90% [12].

Penelitian sebelumnya sudah melakukan analisis untuk mendeteksi serangan *phishing* pada *website*. Akan tetapi, pendekatan ML yang digunakan dalam penelitian ini memiliki keterbatasan, seperti ketergantungan pada ekstraksi fitur manual yang mencakup 31 karakteristik yang diidentifikasi, yaitu panjang URL, pengiriman email, layanan pemendekan URL, URL abnormal, keberadaan simbol '@', dan pengalihan [12]. Fitur-fitur ini dipilih secara manual dan sangat bergantung pada pengetahuan *website* mendalam. Penelitian lain juga menyoroti kelemahan yang sama dari ekstraksi fitur manual, yaitu memerlukan keahlian manusia dalam memilih dan merancang fitur-fitur yang relevan dari data mentah dan berkinerja kurang baik saat mengekstrak data dari *big data* [13]–[15]. Hal ini menyebabkan peneliti menggunakan DL untuk mengatasi masalah tersebut. DL memungkinkan ekstraksi fitur otomatis sehingga mengurangi kebutuhan akan campur tangan manusia [16]–[18]. Berdasarkan penelitian di atas, diambil inisiatif untuk menggunakan DL dalam mendeteksi serangan *phishing* menggunakan algoritma gabungan *convolutional neural network* (CNN) dan *long short-term memory* (LSTM).

Adapun alasan penggunaan algoritma LSTM dikarenakan algoritma ini merupakan jenis dari *recurrent neural network* (RNN) yang dapat memahami urutan informasi dari data teks dan mengingatnya dalam jangka panjang [19], [20]. Sementara itu, alasan penggunaan algoritma CNN dikarenakan algoritma ini memiliki kemampuan dalam memahami fitur penting dari data teks [21]. Sebagai contoh, sebuah penelitian dari Lal Khan menggunakan algoritma CNN-LSTM untuk menganalisis sentimen teks dalam bahasa Inggris dan Roman Urdu yang dibagikan di media sosial menggunakan empat *dataset* yang terbagi menjadi tiga *dataset* Roman Urdu (UCL, RUSA-19, dan RUSA) dan satu dataset bahasa Inggris (IMDB). Temuan dari penelitian ini memperlihatkan bahwa algoritma CNN-LSTM yang dikembangkan pada tiga *dataset* Roman Urdu, yaitu *dataset* RUSA mencapai akurasi 84.1%, 74.8 % pada dataset RUSA-19, dan 74.0% pada dataset UCL. Sementara itu, pada dataset bahasa Inggris IMDB, model ini mencapai akurasi sebesar 90.4% [22].

2. METODE PENELITIAN

Penelitian ini terbagi menjadi 6 tahap, yaitu pengumpulan data, *pre-processing* data, pembagian data, perancangan arsitektur model DL, pelatihan model, dan pengujian serta evaluasi model. Gambar 1 menyajikan tahapan penelitian dari awal hingga akhir.



Gambar 1. Tahapan penelitian

2.1. Pengambilan Data

Tahap pengambilan data merupakan langkah awal yang krusial karena menyediakan kumpulan data yang diperlukan untuk melatih dan menguji model DL dalam mendeteksi serangan *phishing*. Penelitian ini memanfaatkan data berupa URL situs web yang mencakup URL asli dan URL *phishing*. Data dikumpulkan dari situs GitHub melalui https://github.com/larranaga/phishing-url-detection. Data ini terdiri dari 194.798 situs yang terbagi menjadi dua kelas, yaitu 97.389 situs asli dan 97.409 situs *phishing*.

2.2. *Pre-Processing Data*

Setelah menyelesaikan tahapan pengumpulan data, selanjutnya dilakukan tahap *preprocessing*. Tahap *preprocessing* melibatkan pengubahan data mentah menjadi data yang siap digunakan oleh model DL dalam mendeteksi situs *phishing*. Pada penelitian ini, tahapan dilakukan terbagi menjadi 3 bagian yaitu tokenisasi, *padding*, dan pelabelan. Adapun proses ini dilakukan sebagai berikut:

1. Tokenisasi

   Proses tokenisasi bertujuan untuk mengubah setiap karakter dalam URL menjadi token integer sehingga mempermudah model DL untuk memproses data dalam bentuk numerik.

2. *Padding*

   Setelah proses tokenisasi, setiap urutan token diperpanjang (*padding*) menjadi 75 karakter agar semua URL memiliki panjang yang konsisten.

3. Pelabelan

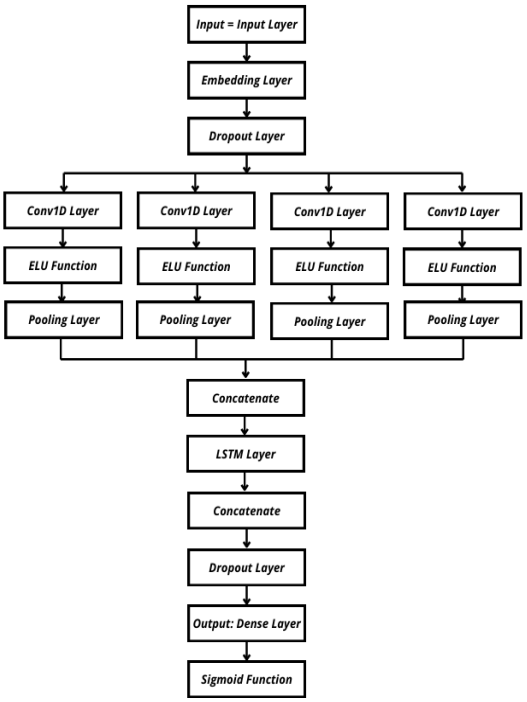
   Pada tahap terakhir, setiap URL diberi label, yaitu 1 untuk situs *phishing* dan 0 untuk situs asli. Proses ini memastikan bahwa model DL dapat membedakan antara dua jenis URL berdasarkan label yang diberikan.

2.3. Pembagian Data

Tahap pembagian data melibatkan pemisahan menjadi data latih dan data uji yang bertujuan untuk mempermudah pelatihan dan pengujian model DL. Data latih digunakan oleh model untuk mempelajari pola dan karakteristik data, sementara data uji digunakan untuk mengukur performa model setelah proses pelatihan. Sebanyak 80% dari total data dialokasikan sebagai data latih, sedangkan 20% sisanya digunakan sebagai data uji. Data uji ini tidak pernah diakses oleh model selama pelatihan dan berfungsi untuk menguji seberapa baik model dapat menangani data yang belum pernah dilihat sebelumnya.

2.4. Perancangan Model CNN-LSTM

Model yang digunakan dalam penelitian ini adalah kombinasi algoritma dari CNN dan LSTM yang dirancang untuk mendeteksi serangan *phishing*. CNN digunakan untuk mengekstraksi fitur-fitur penting dari data. Sementara itu, LSTM berfungsi untuk mengingat informasi jangka panjang dan memahami urutan data. Kombinasi ini membuat model CNN-LSTM cocok digunakan dalam deteksi *phishing* yang memerlukan analisis mendalam terhadap struktur dan urutan karakter dalam URL. Untuk memahami lebih dalam mengenai implementasi model ini, Gambar 2 di dibawah menunjukkan cara kerja lapisan pada model CNN-LSTM yang dirancang untuk penelitian ini.



Gambar 2 Lapisan CNN-LSTM

Pada tahap awal, terdapat lapisan *input* yang menerima data berupa URL. Selanjutnya, terdapat lapisan *embedding* yang bertugas untuk mengonversi data teks menjadi representasi vektor sehingga dapat diproses oleh model. Setelah itu, terdapat lapisan *dropout* yang berfungsi untuk mencegah *overfitting* selama pelatihan model berlangsung.

Proses selanjutnya dari CNN dimulai dengan lapisan *convolution* yang berguna untuk mengekstraksi pola-pola penting dari data teks. Pada struktur ini, terdapat tiga lapisan *convolution* yang masing-masing diikuti oleh fungsi aktivasi *exponential linear unit* (Elu) yang membantu model untuk mempelajari pola-pola yang kompleks dalam data. Setiap lapisan *convolution* diikuti oleh lapisan *pooling* yang berfungsi untuk mengurangi dimensi data sehingga mengurangi kompleksitas komputasi.

Setelah melalui lapisan *pooling*, hasil dari ketiga jalur lapisan *convolution* digabungkan (*concatenate*) menjadi satu representasi data yang lebih komprehensif. Data hasil gabungan ini kemudian diteruskan ke lapisan LSTM yang berfungsi untuk menangkap dan memproses informasi urutan dari data teks, memungkinkan model untuk memahami konteks temporal dari informasi yang diterima.

Output dari lapisan LSTM ini kembali digabungkan (*concatenate*) dan diteruskan ke lapisan *dropout* untuk mencegah *overfitting*. Hasil akhir diperoleh melalui lapisan *dense* dengan fungsi aktivasi *sigmoid* yang menghasilkan prediksi berdasarkan input yang telah diberikan. Dengan struktur ini, model CNN-LSTM dapat memproses data teks secara efektif dan diharapkan dapat menghasilkan prediksi yang akurat dalam mendeteksi situs *phishing* atau situs asli.

2.5. Pelatihan Model

Tahap pelatihan model dilakukan dengan menggunakan data latih untuk membangun dan mengoptimalkan model DL. Pada tahap ini, model dilatih untuk mempelajari data yang berisi URL situs web agar dapat membuat prediksi dengan akurat. Proses pelatihan melibatkan penyesuaian *hyperparameter* untuk meminimalkan kesalahan prediksi dan meningkatkan performa model. Untuk menemukan nilai *hyperparameter* sesuai yang menghasilkan performa model terbaik maka digunakan metode *grid search*. Keunggulan *grid search* terletak pada kemampuannya memberikan informasi yang detail mengenai pengaruh pengaturan *hyperparameter* terhadap performa model [23]..

2.6. Pengujian dan Evaluasi Model

Tahap pengujian dan evaluasi dilakukan menggunakan data uji untuk mengevaluasi performa model yang telah dirancang. Pada proses evaluasi ini, metrik digunakan sebagai variabel indikator untuk menilai performa model. Sebelum memahami metrik yang digunakan dalam mengevaluasi model, penting untuk terlebih dahulu memahami *confusion matrix*. *Confusion matrix* merupakan tabel yang berisi evaluasi performa model klasifikasi dengan membandingkan nilai aktual dan prediksi. *Confusion matrix* terbagi beberapa elemen, yaitu *true positive* (TP), *true negative* (TN), *false positive* (FP), dan *false negative* (FN). Jumlah hasil klasifikasi yang tepat sebagai kasus positif dan negatif ditunjukkan oleh elemen TP dan TN, sementara itu total prediksi yang diklasifikasikan salah sebagai kasus positif dan negatif ditunjukkan oleh elemen FP dan FN [24]. Penggunaan *confusion matrix* dapat digunakan untuk menghitung masing-masing nilai metrik evaluasi, yaitu *accuracy*, *precision*, *recall*, dan *f1-score* [25]. Berikut adalah penjelasan singkat mengenai masing-masing metrik tersebut.

Accuracy adalah metrik yang menunjukkan seberapa sering klasifikasi dilakukan dengan benar. Metrik ini berupa rasio jumlah prediksi yang akurat terhadap jumlah total prediksi. Perhitungan *accuracy* ditampilkan di Persamaan (1).

$$accuracy = \frac{\mathrm{TP+TN}}{\mathrm{TP+FP+TN+FN}} \quad (1)$$

Precision adalah metrik yang menetukan berapa banyak sampel yang diklasifikasikan sebagai positif yang benar-benar merupakan sampel kelas positif. Perhitungan *precision* ditampilkan di Persamaan (2).

$$precision = \frac{TP}{TP+FP} \quad (2)$$

Recall merupakan metrik yang menentukan berapa banyak sampel yang termasuk dalam kelas positif yang diklasifikasikan sebagai positif oleh model klasifikasi. Perhitungan *recall* diperlihatkan di Persamaan (3).

$$recall = \frac{TP}{TP+FN} \quad (3)$$

F1-score merupakan metrik yang berupa rata-rata harmonik antara *precision* dan *recall*. Perhitungan *f1-score* diperlihatkan di Persamaan (4).

$$f1-score = \frac{precision \times recall}{precision + recall} \quad (4)$$

3. HASIL DAN PEMBAHASAN

3.1. Dataset

Adapun *dataset* yang diterapkan dalam penelitian ini yaitu berisikan data berbasis teks. Data tersebut dibuat dan dikumpulkan dengan mengambil sejumlah URL situs web yang mencakup URL asli dan URL *phishing* lalu mengaturnya ke dalam format tabel menggunakan aplikasi Spreadsheet. Seluruh data yang telah diambil mencakup 194.798 URL. Berikut Tabel 1 yang menunjukkan *dataset* dari kumpulan URL yang telah dikumpulkan.

Tabel 1. Kumpuldan *dataset*

No	URL
1.	https://123people.com/s/marc+pageau
2.	https://byu.edu/about-byu/accreditation
3.	http://spiritualsupportcenter.com/wp-admin/aje/googletr
4.	185.25.60.138/upd/8

3.2. *Pre-processing*

Tahapan dari *pre-processing* meliputi tokenisasi untuk mengubah URL menjadi urutan token karakter, diikuti dengan proses *padding* untuk menyesuaikan panjang urutan token. Setelah itu, dilakukan pelabelan untuk mengklasifikasikan sebagai situs *phishing* atau situs asli. Setelah proses *pre-processing*, *dataset* siap digunakan untuk pelatihan model. Tabel 2 menampilkan *output* dari proses ini.

Tabel 2. *Output* tahapan *pre-processing*

URL	*Tokenized* URL	*Labels*
https://123people.com/s/marc+pageau	[11, 22, 33, 44, 55, 66, 77, 88, 99 100, 35, 20, ...]	0
https://byu.edu/about-byu/accreditation	[29, 58, 87, 16, 45, 74, 3, 32, 61, 90, 44, 31, ...]	0
http://spiritualsupportcenter.com/w	[47, 94, 41, 88, 35, 82, 29, 76, 23, 88, 95	1

p-admin/aje/googlet r	23, 21, ...]	
185.25.60.138/up d/8	[45, 86, 30, 12, 14, 15, 29, 30, 11, 23, 95, 21, 25, ...]	1

3.3. Pembagian Data

Setelah selesai melakukan *pre-processing*, *dataset* lalu dibagi menjadi data latih dan data uji. Agar pengujian dapat mendeteksi dengan baik dan mencapai tingkat keakuratan yang tinggi dibutukan data latih yang banyak dalam proses pelatihan sehingga model dapat mengenali dan mempelajari data URL situs web. Kedua dataset dibagi menjadi 20% data untuk pengujian dan 80 % data untuk pelatihan. Pembagian *dataset* disajikan pada Gambar 3.

Gambar 3 menampilkan data latih dengan dua kategori label yaitu *benign* (URL asli) dan *malicious* (URL *phishing*). Total data latih yang digunakan adalah 155.838, meliputi 77.905 URL asli dan 77.933 URL *phishing*, mewakili 80% dari keseluruhan *dataset*. Kemudian data uji juga ditampikan dengan dua kategori label, yang terdiri dari 19.494 URL asli dan 19.466 URL *phishing*. Secara keseluruhan, data testing terdiri dari 38960 atau 20% *dataset*.

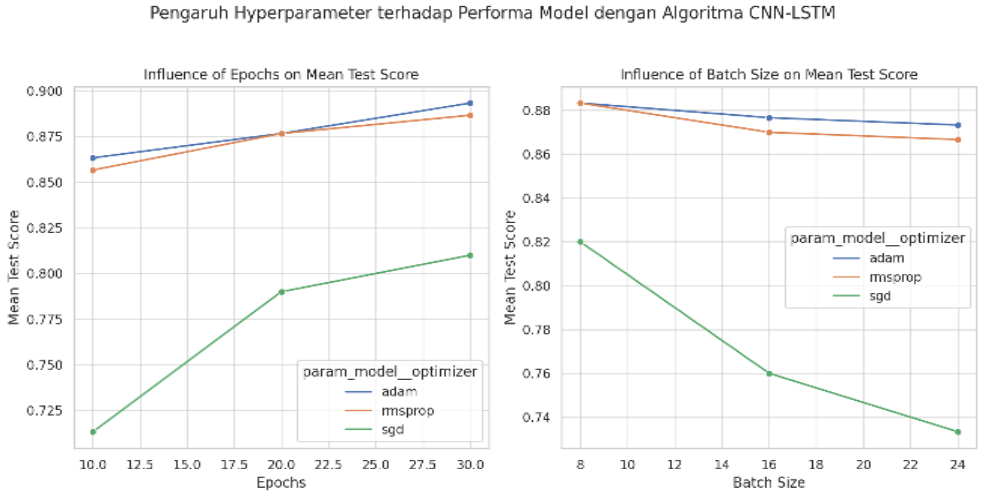
```
Training Data Distribution:
Label                    Training Data Count
Benign (URL asli)          77,905
Malicious (URL phishing)   77,933
Total 155,838 (80% of total)


Testing Data Distribution:
Label                    Testing Data Count
Benign (URL asli)          19,494
Malicious (URL phishing)   19,466
Total 38,960 (20% of total)
```

Gambar 3. Pembagian Data

3.4. Pelatihan Model

Selanjutnya pada saat pelatihan model, dilakukan analisis terhadap hasil yang diperoleh selama tahapan pelatihan yang dapat dilihat melalui diagram visualisasi yang telah dibuat dengan menggunakan *library* Python. *Library* ini menyediakan fitur visualisasi yang memungkinkan pengguna untuk memvisualisasikan berbagai informasi terkait model yang dilatih, termasuk analisis hubungan antara *hyperparameter* terhadap performa model.

Dalam eksperimen ini, penulis menentukan kombinasi *hyperparameter* yang optimal. Optimizer yang diuji meliputi Adam, RMSprop, dan SGD. Jumlah *epoch* yang diuji adalah 10, 20, dan 30, sedangkan *batch size* yang diuji adalah 8, 16, dan 32. Nilai *learning rate* disesuaikan dengan *optimizer* yang digunakan. Secara khusus, Gambar 4 menunjukkan hasil pelatihan dalam bentuk grafik yang menggambarkan pengaruh berbagai *hyperparameter* terhadap performa model dengan algoritma CNN-LSTM.



Gambar 4. Pengaruh *Hyperparameter* terhadap Performa Model dengan Algoritma CNN-LSTM

Berdasarkan grafik yang disajikan pada Gambar 4, terlihat pengaruh dari tiga *hyperparameter*, yaitu jumlah *epochs*, *batch size*, dan *optimizer* terhadap performa model yang menggunakan algoritma CNN-LSTM. *Mean test score* yang disajikan dalam grafik merupakan nilai rata-rata dari beberapa metrik evaluasi. Kedua grafik memberikan pandangan yang jelas mengenai bagaimana perubahan ketiga *hyperparameter* mempengaruhi *mean test score* dari model tersebut.

Grafik pertama menunjukkan pengaruh jumlah *epochs* terhadap *mean test score*. Secara umum, seiring dengan penambahan jumlah *epochs*, terdapat tren peningkatan *mean test score* yang menunjukkan model semakin baik dalam membuat prediksi yang akurat. Selain itu, penggunaan *optimizer* Adam denhan konsisten memberikan hasil yang lebih baik

daripada RMSprop dan SGD di berbagai tingkat *epochs*. Hal ini menunjukkan bahwa Adam lebih efektif dalam mengoptimalkan model CNN-LSTM untuk berbagai kondisi pelatihan, dibandingkan dengan *optimizer* lain yang lebih sensitif terhadap perubahan jumlah *epochs*.

Grafik kedua menampilkan pengaruh *batch size* terhadap *mean test score*. Secara keseluruhan, seiring dengan pengurangan *batch size*, terdapat peningkatan *mean test score*. Penurunan ini menunjukkan model CNN-LSTM mengoptimalkan manfaat dari *batch size* yang lebih kecil dengan fokus pada pemrosesan urutan informasi dalam data. Di samping itu, *optimizer* Adam kembali menunjukkan performa yang lebih baik dibandingkan dengan RMSprop dan SGD di semua batch size yang diuji. Keunggulan ini menandakan bahwa Adam lebih efektif dalam mengelola perubahan sehingga membuatnya lebih cocok untuk pelatihan dengan algoritma CNN-LSTM.

Berdasarkan hasil analisis di atas, kombinasi *hyperparameter* yang cocok pada CNN-LSTM adalah menggunakan *optimizer* Adam dengan jumlah *epochs* sebanyak 30 dan *batch size* sebesar 8. Kombinasi ini menawarkan performa yang stabil dan efisien yang memanfaatkan penggunaan Adam dalam memperbarui bobot data yang lebih sering dan akurat pada *batch size* yang lebih kecil. Kombinasi inilah yang akan digunakan untuk pengujian model selanjutnya. Hasil dari kombinasi *hyperparamter* disajikan dalam Tabel 3.

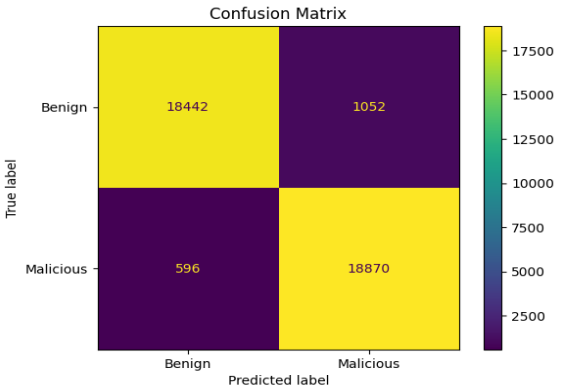
Tabel 3. Kombinasi *Hyperparameter*

No.	*Hyperparameter*	Nilai
1.	*Optimizer*	Adam
2.	*Epochs*	30
3.	*Batch size*	8

3.5. Pengujian Model dengan Data Uji

Uji coba dalam penelitian ini bertujuan untuk mengukur performa algoritma gabungan CNN dan LSTM menggunakan data pengujian yang telah melalui tahapan *pre-processing*. Selain itu, uji coba ini dimaksudkan untuk menunjukkan bahwa prediksi yang dihasilkan oleh kedua algoritma memiliki tingkat akurasi yang tinggi. Hasil pengujian disusun menggunakan *confusion matrix* yang disajikan pada *software* Jupyter notebook dengan menggunakan bahasa pemrograman Python ditampilkan dalam Gambar 5.

Pada Gambar 5, algoritma gabungan CNN-LSTM memprediksi 596 situs *phishing* sebagai situs asli (FN) dan 1.052 situs asli sebagai situs *phishing* (FP). Dari total 38.960 data uji, terdapat 18.442 prediksi yang tepat untuk situs asli (TN) dan 18.870 prediksi yang tepat untuk situs *phishing* (TP). Hal ini menunjukkan bahwa CNN-LSTM lebih sensitif dalam mendeteksi situs *phishing* dengan hasil prediksi benar yang lebih banyak (TP lebih tinggi), tetapi juga diiringi dengan peningkatan FP dan sedikit penurunan prediksi benar terhadap situs asli (TN lebih rendah).



Gambar 5. *Confusion matrix* algoritma CNN-LSTM

Berdasarkan hasil *confusion matrix*, pengaruh FN dan FP pada algoritma CNN-LSTM dapat dilihat dengan jelas. Nilai FN yang lebih rendah menunjukkan bahwa CNN-LSTM lebih jarang salah mengklasifikasikan situs asli sebagai situs *phishing*. Sebaliknya, nilai FP yang sedikit lebih tinggi memperlihatkan bahwa CNN-LSTM lebih sering memberikan peringatan palsu dengan mengklasifikasikan situs asli sebagai situs *phishing*. Selain itu, pengaruh nilai TP dan TN juga penting untuk diperhatikan. Nilai TP yang tinggi menunjukkan deteksi yang baik terhadap ancaman sebenarnya, sedangkan nilai TN yang tinggi menunjukkan bahwa algoritma ini dapat memastikan situs asli benar-benar aman. Perbandingan performa algoritma berdasarkan metrik disajikan dalam Tabel 4.

Tabel 4. Perbadingan performa metrik pada model CNN-LSTM

No.	Metrik	Nilai
1.	*Accuracy*	95.63%
2.	*Precision*	94.60%
3.	*Recall*	96.81%
4.	*F1-score*	95.78%

Hasil metrik dari tabel berikut menunjukkan bahwa algoritma gabungan CNN-LSTM memiliki tingkat *accuracy* sebesar 95,63%, *precision* 94,60%, *recall* 96,81%, dan *f1-score* 95,78%. Tingkat *accuracy* yang tinggi menunjukkan bahwa CNN-LSTM andal dalam memprediksi situs asli dan situs *phishing* secara keseluruhan. *Precision* yang tinggi menunjukkan bahwa algoritma ini memberikan kesalahan yang lebih sedikit dalam mengklasifikasikan situs *asli* sebagai situs *phishing* sehingga menghasilkan lebih sedikit peringatan palsu. Sementara itu, *recall* yang tinggi menunjukkan bahwa CNN-LSTM dapat mendeteksi situs *phishing* dengan baik. F1-score yang tinggi memperlihatkan keseimbangan antara *precision* dan *recall*. Jadi, algoritma CNN-LSTM unggul secara keseluruhan dalam keseimbangan antara mendeteksi situs *phishing* dan mengurangi kesalahan deteksi situs asli sebagai situs *phishing*.

3.6. Pengujian Model dengan Data Baru

Pengujian dalam penelitian ini bertujuan untuk memeriksa apakah model CNN-LSTM yang telah dikembangkan dapat secara efektif membedakan antara situs *phishing* dan situs asli dengan menggunakan data baru yang tidak termasuk dari data yang disediakan dari Github. Kedua model tersebut diuji dengan memasukkan 20 URL yang terbagi atas 10 situs *phishing* dan 10 situs asli. Detail dari hasilnya ditunjukan pada Tabel 5.

Tabel 5. Pengujian dengan data baru pada model CNN-LSTM

No	URL	Aktual	Prediksi CNN-LSTM
1.	https://www.facebook.com/?locale=id_ID	Asli	Asli
2.	http://beasiswa.ui.ac.id/web/	Asli	Asli
3.	https://drive.google.com/drive/u/1/home	Asli	*Phishing*
4.	http://gosugamers.net/lol/streams	Asli	Asli
5.	http://nmgnews.com.cn/system/2016/11/26/012198481.shtml	Asli	Asli
6.	https://www.path.org/where-wework/africa//mozambique/	Asli	Asli
7.	http://2144.cn/www.2144.cn/html/11/73873	Asli	Asli
8.	https://wordpress.com/	Asli	Asli
9.	https://twitter.com/rahmat	Asli	Asli
10.	https://play.google.com/store/apps/details?id=com.lazada.android&hl=en_US&pli=1	Asli	Asli
11.	http://www.faceboo0k.Com	*Phishing*	*Asli*
12.	221.229.173.133:8084/sony	*Phishing*	*Phishing*
13.	http://rooferexpert.com/css/8933617-dosar nr-1817842015/394c-4735-82399c8f64a5248/botosani_firme/adc7b380b3f1002f7a0972358f6c0398/?email=abuse@yahoo.com.com.com	*Phishing*	*Phishing*
14.	http://kelownatownhomes.com/8g74crec	*Phishing*	*Phishing*
15.	209.193.67.173/sdoc22.pdf	*Phishing*	*Phishing*
16.	http://visionntime.com/wapadmin /pdf/Dropbox.html	*Phishing*	*Phishing*
17.	http://laautotransport.com/app.login/westpac%20normal/index.htm	*Phishing*	*Phishing*
18.	http://kvalitne.cz/656465/d5678h9.exe	*Phishing*	*Phishing*
19.	91.239.25.6:6892	*Phishing*	*Phishing*
20.	grouper.in/buy/yhfile/Yahooooooooo oooooooooooooooo.html	*Phishing*	*Phishing*

Berdasarkan hasil dari Tabel 5, terlihat model CNN - LSTM memiliki performa yang baik dalam mendeteksi URL situs *web*, karena keduanya berhasil mengklasifikasikan hampir semua URL dengan benar. Akan tetapi, model CNN-LSTM membuat satu kesalahan dengan mengklasifikasikan URL asli sebagai *phishing* (TP) dan satu kesalahan dengan mengklasifikasi URL *phishing* sebagai URL asli (TN). Ini menunjukkan bahwa model cukup efektif dalam mendeteksi serangan *phishing* pada *website*.

4. DISKUSI

Pada penelitian yang dilakukan sebelumnya [13], algoritma *random forest* mencapai akurasi tertinggi sekitar 96,86% sebelum normalisasi dan 97,3% setelah normalisasi. Meskipun akurasi ini sedikit lebih tinggi, metode *machine learning* tradisional memiliki keterbatasan dalam ekstraksi fitur manual. Metode ini mengharuskan identifikasi manual dari 31 karakteristik seperti panjang URL, pemendekan URL, URL abnormal, keberadaan simbol '@', dan pengalihan. Kelemahan utama dari pendekatan ini adalah memerlukan pemahaman yang mendalam mengenai *website*, sehingga proses ekstraksi fitur menjadi sangat subjektif dan rentan terhadap kesalahan manusia.

Sebaliknya, metode *deep learning* memungkinkan ekstraksi fitur secara otomatis dari data mentah sehingga meminimalkan campur tangan manusia. Hal ini mengurangi bias yang mungkin terjadi pada ekstraksi fitur manual dan meningkatkan konsistensi serta keakuratan hasil. Kombinasi CNN dan LSTM dalam penelitian ini berhasil menangkap pola dan urutan informasi dalam data teks URL, sehingga memberikan deteksi yang lebih baik terhadap serangan phishing.

Penelitian ini mengindikasikan bahwa ekstraksi fitur otomatis dapat memudahkan pembaruan model *deep learning* dengan cukup menambahkan data baru tanpa perlu mengetahui fitur spesifik dari data tersebut. Hal ini berbeda pada penelitian sebelumnya yang menggunakan *machine learning* tradisional dikarenakan pembaruan model memerlukan penambahan data serta fitur-fiturnya agar model dapat mendeteksi data baru lainnya. Dengan demikian, metode *deep learning* memberikan fleksibilitas dan efisiensi yang lebih baik dalam pemrosesan dan pembaruan data.

Selain itu, pemilihan *hyperparameter* dapat mempengaruhi performa model DL yang menggunakan algoritma gabungan CNN dan LSTM. Dengan menambah jumlah *epochs*, model CNN-LSTM menunjukkan peningkatan performa. Sementara itu, penggunaan *batch size* yang lebih kecil juga meningkatkan efektivitas model CNN-LSTM karena dapat mengoptimalkan pemrosesan informasi. Di sisi lain, pemilihan *optimizer* Adam terbukti lebih efektif dalam meningkatkan kinerja model CNN-LSTM. Hasil ini mengonfirmasi bahwa penyesuaian *hyperparameter* yang tepat memiliki pengaruh signifikan terhadap efektivitas model dalam mendeteksi serangan *phishing* pada *website*.

5. KESIMPULAN

Penelitian ini telah berhasil mengimplementasikan algoritma CNN-LSTM untuk mendeteksi serangan *phishing* pada *website* menggunakan URL. Melalui eksperimen, *hyperparameter* dievaluasi untuk mencari nilai optimal, dan hasil evaluasi model menunjukkan *accuracy* sebesar 95,63%, *precision* sebesar 94,62%, *recall* sebesar 96,81%, dan *f1-score* sebesar 95,78%. Kedepannya, penulis akan mempertimbangkan pengembangan dengan menggunakan tiga kelas yang terdiri dari situs asli, *suspicious*, dan *phishing* sehingga dapat memberikan deteksi yang lebih spesifik dan meningkatkan perlindungan bagi pengguna internet.

DAFTAR PUSTAKA

[1] G. Mohamed, J. Visumathi, M. Mahdal, J. Anand, and M. Elangovan, "An Effective and Secure Mechanism for Phishing Attacks Using a Machine Learning Approach," *Processes*, vol. 10, no. 7, pp. 1356, Jul. 2022, doi: 10.3390/pr10071356.

[2] M. W. Shaukat, R. Amin, M. M. A. Muslam, A. H. Alshehri, and J. Xie, "A Hybrid Approach for Alluring Ads Phishing Attack Detection Using Machine Learning," *Sensors*, vol. 23, no. 19, pp. 8070, Sep. 2023, doi: 10.3390/s23198070.

[3] S. Kapan and E. Sora Gunal, "Improved Phishing Attack Detection with Machine Learning: A Comprehensive Evaluation of Classifiers and Features," *Appl. Sci.*, vol. 13, no. 24, pp. 13269, Dec. 2023, doi: 10.3390/app132413269.

[4] M. Sánchez-Paniagua, E. Fidalgo, E. Alegre, and R. Alaiz-Rodríguez, "Phishing websites detection using a novel multipurpose dataset and web technologies features," *Expert Syst. Appl.*, vol. 207, p. 118010, Jun. 2022, doi: https://doi.org/10.1016/j.eswa.2022.118010.

[5] E. D. Frauenstein, S. Flowerday, S. Mishi, and M. Warkentin, "Unraveling the behavioral influence of social media on phishing susceptibility: A Personality-Habit-Information Processing model," *Inf. Manag.*, vol. 60, no. 7, p. 103858, Sep. 2023, doi: https://doi.org/10.1016/j.im.2023.103858.

[6] L. Tang and Q. H. Mahmoud, "A Survey of Machine Learning-Based Solutions for Phishing Website Detection," *Mach. Learn. Knowl. Extr.*, vol. 3, no. 3, pp. 672–694, Aug. 2021, doi: 10.3390/make3030034.

[7] P. Saravanan and S. Subramanian, "A Framework for Detecting Phishing Websites using GA based Feature Selection and ARTMAP based Website Classification," *Procedia Comput. Sci.*, vol. 171, pp. 1083–1092, Jun. 2020, doi: https://doi.org/10.1016/j.procs.2020.04.116.

[8] A. Taha, "Intelligent Ensemble Learning Approach for Phishing Website Detection Based on Weighted Soft Voting," *Mathematics*, vol. 9, no. 21, pp. 2799 Nov. 2021, doi: 10.3390/math9212799.

[9] Indonesia Anti-Phishing Data Exchange, "Phishing Activity Report - 4th Quarter 2023," 2023. https://idadx.id/ (accessed Feb. 07, 2024).

[10] E. O. O. A. S. S. Carolyn Oreoluwa Tinubu Olorunjube James Falana and S. A. Rufai, "PHISHGEM: a mobile game-based learning for phishing awareness," *J. Cyber Secur. Technol.*, vol. 7, no. 3, pp. 134–153, Feb. 2023, doi: 10.1080/23742917.2023.2167276.

[11] A. Basit, M. Zafar, X. Liu, A. R. Javed, Z. Jalil, and K. Kifayat, "A comprehensive survey of AI-enabled phishing attacks detection techniques," *Telecommun. Syst.*, vol. 76, no. 1, pp. 139–154, Jan. 2021, doi: 10.1007/s11235-020-00733-2.

[12] S. Alnemari and M. Alshammari, "Detecting Phishing Domains Using Machine Learning," *Appl. Sci.*, vol. 13, no. 8, p. 4649, Apr. 2023, doi: 10.3390/app13084649.

[13] Y. N. Kunang, S. Nurmaini, D. Stiawan, and B. Y. Suprapto, "Attack classification of an intrusion detection system using deep learning and hyperparameter optimization," *J. Inf. Secur. Appl.*, vol. 58, p. 102804, May. 2021, doi: https://doi.org/10.1016/j.jisa.2021.102804.

[14] W. Bakasa and S. Viriri, "Vgg16 feature extractor with extreme gradient boost classifier for pancreas cancer prediction," *J. Imaging*, vol. 9, no. 7, p. 138, Jul. 2023, doi: https://doi.org/10.3390/jimaging9070138.

[15] M. Hakim, A. A. B. Omran, A. N. Ahmed, M. Al-Waily, and A. Abdellatif, "A systematic review of rolling bearing fault diagnoses based on deep learning and transfer learning:

Taxonomy, overview, application, open challenges, weaknesses and recommendations," *Ain Shams Eng. J.*, vol. 14, no. 4, p. 101945, Apr. 2023, doi: https://doi.org/10.1016/j.asej.2022.101945.

[16] M. M. Taye, "Understanding of Machine Learning with Deep Learning: Architectures, Workflow, Applications and Future Directions," *Computers*, vol. 12, no. 5, p. 91, Apr. 2023, doi: 10.3390/computers12050091.

[17] M. Abbasi, A. Shahraki, and A. Taherkordi, "Deep Learning for Network Traffic Monitoring and Analysis (NTMA): A Survey," *Comput. Commun.*, vol. 170, pp. 19–41, Mar. 2021, doi: https://doi.org/10.1016/j.comcom.2021.01.021.

[18] N. Q. Do, A. Selamat, O. Krejcar, E. Herrera-Viedma, and H. Fujita, "Deep Learning for Phishing Detection: Taxonomy, Current Challenges and Future Directions," *IEEE Access*, vol. 10, pp. 36429–36463, Apr. 2022, doi: 10.1109/ACCESS.2022.3151903.

[19] R. R. Rajalaxmi, L. V. Narasimha Prasad, B. Janakiramaiah, C. S. Pavankumar, N. Neelima, and V. E. Sathishkumar, "Optimizing Hyperparameters and Performance Analysis of LSTM Model in Detecting Fake News on Social media," *ACM Trans. Asian Low-Resource Lang. Inf. Process.*, pp. 1083-1092, Mar. 2022, doi: 10.1145/3511897.

[20] A. Al Bataineh, V. Reyes, T. Olukanni, M. Khalaf, A. Vibho, and R. Pedyuk, "Advanced Misinformation Detection: A Bi-LSTM Model Optimized by Genetic Algorithms," *Electronics*, vol. 12, no. 15, p. 3250, Jul. 2023, doi: 10.3390/electronics12153250.

[21] S. Soni, S. S. Chouhan, and S. S. Rathore, "TextConvoNet: a convolutional neural network based architecture for text classification," *Appl. Intell.*, vol. 53, no. 11, pp. 14249–14268, Jun. 2023, doi: 10.1007/s10489-022-04221-9.

[22] L. Khan, A. Amjad, K. M. Afaq, and H.-T. Chang, "Deep Sentiment Analysis Using CNN-LSTM Architecture of English and Roman Urdu Text Shared in Social Media," *Appl. Sci.*, vol. 12, no. 5, p. 2694, Mar. 2022, doi: 10.3390/app12052694.

[23] X. Jiang and C. Xu, "Deep Learning and Machine Learning with Grid Search to Predict Later Occurrence of Breast Cancer Metastasis Using Clinical Data," *J. Clin. Med.*, vol. 11, no. 19, p. 5572, Sep. 2022, doi: 10.3390/jcm11195772.

[24] Y. S. Taspinar, M. Koklu, and M. Altin, "Classification of flame extinction based on acoustic oscillations using artificial intelligence methods," *Case Stud. Therm. Eng.*, vol. 28, p. 101561, Dec. 2021, doi: 10.1016/j.csite.2021.101561.

[25] J. Kozak, B. Probierz, K. Kania, and P. Juszczuk, "Preference-Driven Classification Measure," *Entropy*, vol. 24, no. 4, p. 531, Apr. 2022, doi: 10.3390/e24040531.