Jurnal Informatika: Jurnal pengembangan IT

Vol. 10, No. 1, 2025
ISSN : 2477-5126, DOI:10.30591/jpit.v10i1.8128

Optimasi Bobot Kelas LSTM untuk Deteksi URL Phishing pada Dataset Tidak Berimbang

Tri Ferli Handoyo , Muhammad Pajar Kharisma Putra
Program Studi Informatika, Universitas Teknokrat Indonesia,
Jl. ZA. Pagar Alam No 9-11, Kota Bandar Lampung, Kode pos 35132, Indonesia

Info Artikel

Riwayat Artikel:

Received 2024-12-18
Revised 2024-12-22
Accepted 2024-12-22

Corresponding Author:

Suttichai Premrudeeprechacharn

Email: suttichai@mail.com

Abstract - *Phishing URL detection is one of the main challenges in cybersecurity, considering the ever-increasing threats affecting internet users globally. This research aims to develop a Long Short-Term Memory (LSTM) based deep learning model to detect phishing URLs with high accuracy. The dataset used consists of 651,191 URLs, which are divided into four categories: benign, defacement, phishing, and malware. The dataset is processed through preprocessing stages, including URL cleaning and feature extraction. The LSTM model is applied with optimized hyperparameter configurations to learn patterns from the dataset. The results showed that the model was able to achieve significant accuracy during the training and validation process. Evaluation on external datasets shows that the model performs well in the benign and defacement categories, with relatively high precision and recall. However, challenges were identified in the malware and phishing categories, where recall was low due to dataset imbalance and lack of feature representation. Further analysis showed a model bias towards the majority class, as well as difficulty in detecting URLs in the minority class. This research shows the potential of using LSTM-based deep learning in phishing URL detection, but also emphasizes the importance of further optimization, such as adjusting class weights, oversampling, or using additional features. It is hoped that the resulting model can be an initial solution in improving cyber security, especially in detecting phishing threats in real-time.*

Keywords: *cybersecurity; deep learning; LSTM; threat detection; URL phishing.*

Abstrak - *Deteksi URL phishing merupakan salah satu tantangan utama dalam keamanan siber, mengingat ancaman yang terus meningkat yang mempengaruhi pengguna internet secara global. Penelitian ini bertujuan untuk mengembangkan model deep learning berbasis Long Short-Term Memory (LSTM) untuk mendeteksi URL phishing dengan akurasi yang tinggi. Dataset yang digunakan terdiri dari 651.191 URL, yang dibagi menjadi empat kategori: benign, defacement, phishing, dan malware. Dataset diproses melalui tahap preprocessing, termasuk pembersihan URL dan ekstraksi fitur. Model LSTM diterapkan dengan konfigurasi hyperparameter yang dioptimalkan untuk mempelajari pola dari dataset. Hasil penelitian menunjukkan bahwa model tersebut mampu mencapai akurasi yang signifikan selama proses pelatihan dan validasi. Evaluasi pada dataset eksternal menunjukkan bahwa model tersebut berkinerja baik pada kategori jinak dan defacement, dengan presisi dan recall yang relatif tinggi. Namun, tantangan diidentifikasi pada kategori malware dan phishing, di mana recall rendah karena ketidakseimbangan dataset dan kurangnya representasi fitur. Analisis lebih lanjut menunjukkan adanya bias model terhadap kelas mayoritas, serta kesulitan dalam mendeteksi URL pada kelas minoritas. Penelitian ini menunjukkan potensi penggunaan deep learning berbasis LSTM dalam deteksi URL phishing, tetapi juga menekankan pentingnya optimasi lebih lanjut, seperti penyesuaian bobot kelas, oversampling, atau penggunaan fitur tambahan. Model yang dihasilkan diharapkan dapat menjadi solusi awal dalam meningkatkan keamanan siber, khususnya dalam mendeteksi ancaman phishing secara real-time.*

Kata Kunci*: keamanan siber,deep learning, LSTM, deteksi ancaman, URL phishing.*

I. PENDAHULUAN

Pada era digital saat ini, internet telah menjadi bagian penting yang tidak bisa dipisahkan dari aktivitas sehari-hari. Internet adalah sebuah media informasi yang berguna untuk mencari sebuah informasi terbaru dan dapat diakses secara global[1]. Namun, ancaman keamanan siber seperti phishing semakin meningkat. Phishing adalah metode serangan siber yang meniru situs web sah untuk mencuri informasi pribadi pengguna. Meskipun berbagai metode teknis telah dikembangkan, serangan phishing tetap sulit ditangkal secara efektif S. Marchal (2014, dikutip dalam L. Tang , 2022)[2].

Dampak dari serangan *phishing* sangat menghancurkan, baik bagi individu maupun organisasi. Serangan *phishing* dapat menyebabkan kerugian finansial, pencurian identitas, dan pelanggaran data. Mereka menekankan pentingnya kesadaran dan pendidikan tentang cara mengenali dan menghindari *phishing* untuk melindungi diri dari ancaman ini[3]. Menurut laporan *Grand View Research*, pasar keamanan siber global bernilai US$203 miliar di tahun 2022, dan diproyeksikan tumbuh sebesar 12,3% selama periode 2023-2030. Adapun, nilainya diprediksi mencapai

US$500 miliar pada tahun 2030 mendatang. Sedangkan, biaya rata-rata operasional yang dikeluarkan oleh perusahaan secara global karena kasus pembobolan data mencapai US$4,35 juta pada tahun 2021 menurut laporan dari *International Business Machines Corporation* jumlah kasus serangan siber juga tercatat telah meningkat sebesar 13%. Sementara*, Federal Bureau of Investigation (FBI)* pada Maret 2023 melaporkan, kerugian akibat serangan siber sepanjang tahun 2022 telah mencapai lebih dari US$10 miliar. Jika dirupiahkan, maka nilai kerugian tersebut setara Rp147 triliun. Lebih lanjut, *FBI* mengaku telah menerima lebih dari 800 ribu aduan terkait kasus serangan siber. Adapun, akumulasi total serangan siber dalam lima tahun terakhir sudah mencapai 3,26 juta kasus dengan kerugian sebesar US$27,6 miliar atau senilai Rp406 triliun[4]. Angka- angka tersebut menunjukkan bahwa ancaman *phishing* tidak hanya menyebabkan kerugian finansial, tetapi juga menurunkan kepercayaan masyarakat terhadap transaksi online yang pada akhirnya berdampak pada bisnis secara signifikan.

Untuk mengatasi tantangan ini, berbagai pendekatan telah diusulkan. Pendekatan tradisional seperti edukasi keamanan siber kepada pengguna merupakan langkah penting untuk meningkatkan kesadaran akan tanda-tanda *phishing*. Selain itu, solusi berbasis perangkat lunak seperti filter *email*, daftar blokir domain, dan alat deteksi berbasis *heuristik* telah banyak digunakan untuk memitigasi ancaman *phishing*. Namun, solusi- solusi tersebut seringkali memiliki keterbatasan, terutama dalam menghadapi serangan *phishing* yang semakin kompleks dan dinamis. (Abuadbba, 2022) dalam penelitiannya mengkaji bagaimana situs *phishing* dapat mengelabui detektor berbasis *Machine Learning (ML)* dan menyoroti keterbatasan pendekatan tradisional dalam menghadapi serangan phishing yang semakin kompleks. Studi ini juga mengusulkan model *Anti-SubtlePhish* yang lebih tangguh dengan memasukkan fitur horizontal untuk meningkatkan akurasi deteksi[5].

Seiring dengan perkembangan teknologi, pendekatan berbasis *Artificial Intelligence (AI)* menjadi solusi yang semakin relevan. Salah satu cabangnya, *Machine Learning*, menawarkan kemampuan untuk menganalisis pola yang kompleks pada data secara otomatis[6]. *Machine learning* adalah cabang kecerdasan buatan yang memungkinkan sistem untuk belajar dari data dan pengalaman sebelumnya, dan membuat prediksi atau pengambilan keputusan tanpa adanya instruksi eksplisit. Dengan mempelajari pola dan karakteristik dalam kumpulan data, algoritma *Machine Learning* dapat menghasilkan aturan yang digunakan untuk membuat prediksi[7].

Dalam konteks menganalisis data berurutan seperti URL, algoritma Long Short-Term Memory (LSTM), yang merupakan perluasan dari Recurrent Neural Network (RNN), terbukti sangat efektif. Gurung (2024) dalam penelitianya yang mengembangkan model hibrida CNN-LSTM untuk menganalisis urutan karakter dalam URL dan membedakan antara situs web yang sah dan phishing, hasilnya menunjukkan bahwa pendekatan ini efektif dalam mendeteksi URL *phishing* dengan tingkat akurasi yang tinggi[8]. LSTM dirancang untuk mempelajari pola-pola yang kompleks dalam data yang berurutan dan mampu mengatasi masalah vanishing gradient yang sering terjadi pada RNN biasa. Dalam pendeteksian *phishing*, LSTM mampu mempelajari karakteristik URL *phishing,* seperti panjang domain, struktur karakter, dan fitur- fitur lainnya, sehingga mampu membedakan antara situs web yang sah dan situs web *phishing* dengan tingkat akurasi yang tinggi.

Penelitian ini berfokus pada pengembangan model deteksi *phishing* berbasis LSTM yang dirancang untuk mengidentifikasi URL *phishing* dengan tingkat akurasi yang tinggi. Model ini akan dievaluasi kinerjanya melalui serangkaian pengujian untuk memastikan keandalannya dalam membedakan URL *phishing* dengan URL yang aman. Dengan memanfaatkan data terbaru secara berkesinambungan dalam proses pelatihan, diharapkan model ini dapat memberikan kontribusi yang signifikan dalam meningkatkan keamanan siber dan melindungi pengguna dari ancaman *phishing*.

Dalam beberapa tahun terakhir, penggunaan algoritma *deep learning* untuk mendeteksi *phishing website* telah menjadi fokus utama penelitian. Pendekatan ini dianggap lebih efektif dibandingkan metode tradisional karena kemampuannya dalam memahami pola kompleks dalam data URL.

Penelitian oleh Saba Aslam,Hafsa Aslam,Arslan Manzoor, Hui Chen and Abdur Rasool1 (2024), menunjukkan bahwa model Long Short-Term Memory (LSTM) tetap menjadi salah satu metode yang efektif dalam mendeteksi *phishing* URL. Studi ini mengembangkan pendekatan *stacked generalization* dengan menggabungkan LSTM dan metode *ensemble* untuk meningkatkan performa deteksi *phishing*. Model yang diusulkan, *AntiPhishStack*, dilaporkan mencapai akurasi hingga 96.04% pada dataset *benchmark* URL *phishing* dan *non- phishing,* menegaskan kemampuan LSTM dalam mengenali pola sekuensial dengan presisi tinggi[9].

Selain itu, penelitian oleh Banik and Sarma (2023) mengevaluasi penggunaan *ensemble* berbasis LSTM untuk mendeteksi *phishing* URL. Dalam studi ini, model LSTM digunakan untuk menganalisis urutan karakter dalam URL, dan pendekatan *ensemble* seperti *bagging* dan *stacking* digunakan untuk meningkatkan akurasi deteksi. Hasil penelitian menunjukkan bahwa metode ini mampu mencapai performa yang lebih baik dalam mengidentifikasi URL *phishing* dibandingkan dengan teknik lain yang lebih tradisional[10].

Phishing website memiliki karakteristik atau ciri-ciri yang dapat dibedakan dengan *website* yang asli atau

legitimate website. Karakteristik tersebut dapat dilihat pada tabel 1 di bawah, Pada Tabel 1 karakteristik *phishing website* dibagi menjadi 4 fitur besar yang kemudian dibagi kembali menjadi sub-sub fitur di bawahnya[11].

TABEL 1

KARAKTERISTIK URL PHISHING WEBSITE

Kategori Fitur	Sub-Fitur
Fitur Berdasarkan Address Bar	Menggunakan IP Address pada nama domain.
	Menggunakan URL yang panjang untuk menutupi bagian yang mencurigakan pada nama domain menggunakan URL shortening services 'TinyURL'
	Redirection menggunakan '//'
	Menambahkan Prefiks aatau Surfiks yang dipisahkan dengan symbol '' pada domain
	Terdapat symbol '@' pada nama domain
	Memiliki multi sub-domain
	Menggunakan non-standar port
	URL yang terlalu panjang
	Menggunakan favicon
	Terdapat token http/https pada nama domain
Fitur berdasarkan abnormalitas	Request URL
	URL of Anchor
	Tautan pada tag, <Meta>, <Script>, dan <Link>
	Server Form Handler (SFH)
	Input informasi ke Email
	Abnormal URL
Fitur berdasarkan HTML/Javascript	Website Forwarding
	Kustomisasi Status Bar
	Menonaktifkan Right Click
	Menggunakan pop-up window
	Iframe Rediction
Fitur berdasarkan domain	Umur dari domain
	DNS Record

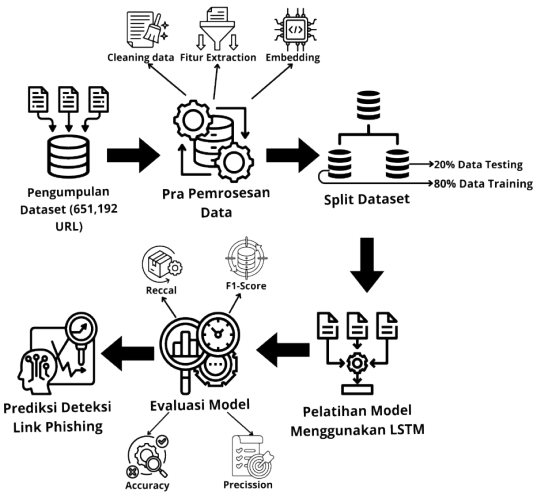
II. METODE

Penelitian ini bertujuan untuk mendeteksi link *phishing* pada situs web menggunakan model Long Short-Term Memory (LSTM). Langkah awal melibatkan penggunaan dataset URL yang terdiri dari empat kategori: *benign*, *defacement*, *phishing*, dan *malware*. Proses pra-pemrosesan data dilakukan melalui beberapa tahapan berikut:

1. Membersihkan data dari duplikasi atau URL yang tidak valid.
2. Melakukan ekstraksi fitur manual dari URL, seperti panjang URL, jumlah karakter tertentu (seperti @, -, atau .), keberadaan IP address, penggunaan *shortening service*, serta kata-kata mencurigakan.
3. Mengubah URL menjadi bentuk numerik dengan teknik seperti *embedding*.

Dataset yang telah diproses kemudian dibagi menjadi 80% data pelatihan dan 20% data pengujian. Data pelatihan digunakan untuk melatih model LSTM, sementara data pengujian dimanfaatkan untuk mengevaluasi kinerja model. Model LSTM dilatih menggunakan fungsi *loss categorical cross-entropy* dan *optimizer* Adam. Evaluasi

kinerja dilakukan dengan menggunakan beberapa metrik, termasuk *accuracy*, *precision*, *recall*, dan *F1-score*. *Confusion matrix* juga digunakan untuk memvisualisasikan hasil klasifikasi pada data pengujian. Hasil deteksi *phishing* dianalisis menggunakan berbagai metrik evaluasi untuk menentukan efektivitas model LSTM dalam mengidentifikasi *link phishing*. Gambar 1 menunjukkan diagram alur penelitian.



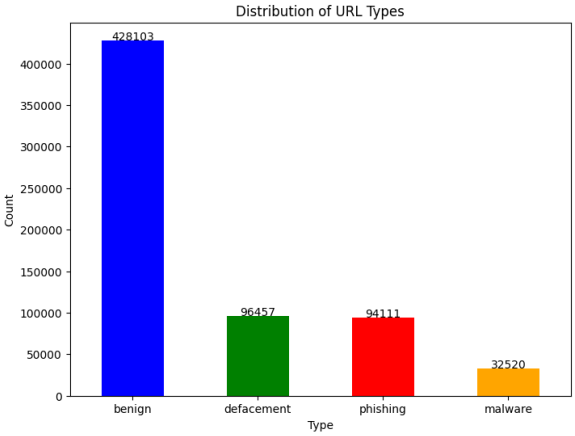
Gambar 1. Diagram Alur Penelitian

A. Dataset

Dataset yang digunakan dalam penelitian ini merupakan kompilasi yang disediakan oleh berbagai sumber tepercaya untuk memastikan representasi yang seimbang dan komprehensif dari berbagai jenis URL, yaitu URL *benign*, *phishing*, *malware*, dan *defacement*. Dataset ini mencakup data dari ISCX-URL-2016 untuk URL *benign*, *phishing*, *malware*, dan *defacement*. *Malware Domain Blacklist* untuk memperluas jumlah URL *phishing* dan *malware*. Selain itu, URL *phishing* juga diperoleh dari kumpulan data *Phishtank* dan *PhishStorm*.

Penelitian terbaru yang dipublikasikan di International Journal of Creative Research Thoughts pada bulan April 2023, dengan judul "Malicious URL Detection", menyoroti pentingnya pemilihan dataset yang tepat dalam pendeteksian URL berbahaya. Jurnal tersebut menganalisis dataset yang mencakup 651.191 URL, yang terdiri dari 428.103 URL *benign*, 96.457 URL *defacement*, 94.111 URL *phishing*, dan 32.520 URL *malware*. Studi ini juga menyoroti peran penting analisis statistik dalam mengidentifikasi URL berbahaya dan memastikan validitas dataset dalam penelitian berbasis pembelajaran mesin[12]. Dataset ini dirancang dengan menggabungkan data dari berbagai sumber tepercaya, sehingga menjadi sumber data yang valid untuk penelitian di bidang keamanan siber. Dataset ini dirancang untuk mendukung penelitian dalam mendeteksi URL berbahaya. Kumpulan data ini mencakup empat kategori URL.

1. *Benign* (Aman)
   Merupakan jenis URL yang tidak memiliki potensi ancaman atau berbahaya.
2. *Defacement* (Perusakan)
   Merupakan jenis URL yang mengarah pada konten yang dimodifikasi tanpa izin biasanya untuk tujuan merusak reputasi atau penyampaian pesan tertentu.
3. *Phishing* (Penipuan)
   Merupakan jenis URL yang digunakan untuk mencoba mencuri informasi sensitif pengguna, seperti kata sandi atau informasi keuangan.
4. *Malware* (Perangkat Lunak Perusak)
   Merupakan jenis URL yang dirancang untuk mendistribusikan perangkat lunak berbahaya.



Gambar 2. Grafik Distribusi Dataset

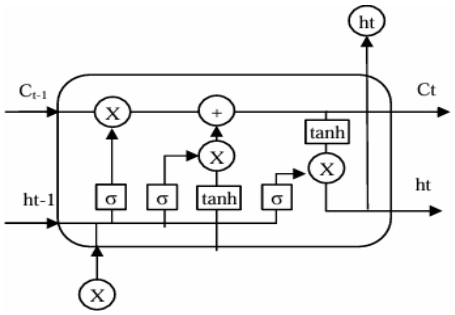
Gambar 2 menunjukkan grafik distribusi dataset, dataset ini terdiri dari total 651,191 URL yang dikumpulkan dari berbagai sumber yang relevan. Distribusi setiap kelas adalah sebagai berikut:

a. 428,103 URL *benign* (65.7%)
b. 96,457 URL *defacement* (14.8%)
c. 94,111 URL *phishing* (14.5%)
d. 32,520 URL *malware* (5.0%)

Distribusi ini menunjukkan bahwa dataset memiliki ketidakseimbangan kelas yang signifikan,dengan dominasi URL *benign* dan jumlah data *malware* yang jauh lebih sedikit. Ketidakseimbangan ini menjadi tantangan utama dalam implementasi algoritma Long Short-Term Memory (LSTM), karena model cenderung memprediksi kelas mayoritas dengan akurasi tinggi, namun kinerja pada kelas minoritas (*phishing*, *defacement*, dan *malware*) menjadi kurang optimal. Untuk mengatasi hal ini, digunakan strategi seperti *oversampling* atau *undersampling*, pembobotan untuk kelas minoritas, dan augmentasi data untuk meningkatkan jumlah data kelas minoritas. Strategi ini diharapkan dapat meningkatkan akurasi deteksi URL berbahaya oleh model LSTM di semua kelas.

B. Long Short-Term Memory (LSTM)

LSTM adalah salah satu jenis Recurrent Neural Network (RNN) yang merupakan modifikasi dari RNN, dengan menambahkan *memory cell* agar dapat menyimpan informasi dalam jangka waktu yang lama. LSTM dapat meminimalisir terjadinya *vanishing gradient*, dimana kondisi nilai gradien pada input layer lebih kecil dari *output layer*[13].



Gambar 3. Gambar Arsitektur LSTM

Secara umum, LSTM terdiri dari empat komponen utama dalam arsitekturnya: *memory cell, input gate, forget gate, dan output gate.*

Gambar 3 merupakan gambar arsitektur LSTM yang terdiri dari 4 komponen dengan fungsi setiap komponen sebagai berikut :

1. Memory Cell

Menyimpan informasi jangka panjang dan menghapus data yang tidak relevan berdasarkan instruksi forget gate.

2. *Input Gate*
   Mengontrol informasi baru yang diterima ke *memory cell* dengan memastikan hanya data relevan yang diperbarui melalui fungsi aktivasi sigmoid dan tanh.
3. *Forget Gate*
   Menghapus informasi tidak relevan dalam memory cell dengan menghasilkan nilai antara 0 (dihapus) dan 1 (dipertahankan)..
4. *Output Gate*
   Mengatur informasi yang keluar dari memory cell untuk memprediksi atau memengaruhi langkah selanjutnya.

LSTM memiliki beberapa lapisan tersembunyi, yang berfungsi ketika informasi mengalir, setiap informasi yang relevan disimpan dan yang tidak relevan dibuang pada tiap sel[14].Struktur gerbang pada LSTM terdapat pada Gambar 3.

LSTM terdiri dari rangkaian sel memori yang unik dan model LSTM menyaring informasi melalui struktur gerbang. Pada LSTM terdapat 4 gerbang atau *Gates Unit*, yaitu *forgate gates, input gate, cell gates*,dan *output gates*[15]. Persamaan dari setiap gate sebagai berikut:

1. *Input gate*:
   $it = \sigma(Wi \times Xi + Ui \times ht{-}1 + bi)$ (1)
2. *Forgate gate*:
   $ft = \sigma(Wf \times Xt + Uf \times ht{-}1 + bf)$ (2)
3. *Cell State*:
   $ct = ft \odot ct{-}1 + it \odot c't$ (3)
4. *New Candidate*:
   $c't = tanh(Wc \times Xc + Uc \times ht{-}1 + bc)$ (4)
5. *Output gate*:
   $ot = \sigma(Wo \times Xi + Uo \times ht{-}1 + bo)$ (5)
6. *Hidden State*:
   $ht = ot \odot \sigma(ct)$ (6)

Dengan mekanisme ini, LSTM dapat mempertahankan informasi penting dari data *time series* yang mungkin memiliki jeda waktu panjang antara peristiwa penting. Hal ini menjadikan LSTM sangat efektif dalam tugas-tugas seperti prediksi deret waktu, pemrosesan teks, dan pengenalan pola sekuensial.

III. HASIL DAN PEMBAHASAN

A. Data Preprocessing

Pada tahap ini data yang akan akan di proses melalui beberapa tahap persiapan dan pembersihan data sebelum digunakan untuk prediksi menggunakan model Long-Short Term Memory (LSTM). Langkah-langkah tersebut dalah sebagia berikut :

1. Pembersihan URL
   Sebelum mengekstrak fitur dari URL, fungsi clean_url digunakan untuk menghapus protokol URL seperti http:// atau https://. Proses ini penting untuk menghindari pengaruh karakter-karakter protokol yang tidak relevan dalam analisis URL.

TABEL 2

DATA CLEANING URL

No	Type	URL	Clean URL
1.	**Benign**	mp3raid.co m/music/kr izz_kaliko. html	mp3raid.com/music/krizz_kaliko.htm l
2.	**Phishing**	http://paste html.com/v iew/b2i243 gkw.html	pastehtml.com/vie w/b2i243gkw.html
3.	**Defacement**	http://www.pashminao nline.com/ pure-pashminas	www.pashminaonline.com/pure-pashminas
4.	**Malware**	http://95.8. 55.230:23693/.i	95.8.55.230:23693/.i

Pada tabel 2 berisi data dari setiap kelas sebelum dan sesudah proses pembersihan URL http:// atau https://.

2. Pengecekan Keberadaan IP Address

Fungsi has_ip_address digunakan untuk memeriksa apakah sebuah URL mengandung alamat IP. URL yang mengandung alamat IP dapat menjadi indikator situs web yang lebih mencurigakan.

TABEL 3

TABEL PENGECEKAN IP URL

Has Ip Type	False	True
benign	427805.0	298.0
defacement	96457.0	0.0
malware	20719.0	11801.0
phishing	93614.0	497.0

Tabel 3 menampilkan hasil untuk fungsi pengecekan dan memberikan total dari 4 kelas apakah URL memiliki Ip atau tidak untuk diproses ke tahap selanjutnya.

3. Ekstraksi Fitur

Fungsi extract_features digunakan untuk mengekstrak berbagai fitur manual dari URL yang akan digunakan sebagai *input* model. Fitur-fitur yang diekstraksi meliputi:

a. Panjang URL
b. Jumlah karakter @, -, dan // dalam URL
c. Jumlah titik (.) dalam URL
d. Keberadaan *IP address*
e. Keberadaan URL pendek (seperti bit.ly)
f. Panjang domain dan *TLD (Top-Level Domain)*
g. Keberadaan kata-kata mencurigakan dalam URL seperti "login", "verify", "secure", dll.
h. Memuat Model dan *One-Hot Encoding*

TABEL 4

TABLE EKSTRAKSI FITUR

Type	Url	Features
Benign	mp3raid.com/music/krizz_kaliko.html	[35, 0, 0, 0, 2, 0, 0, 7, 3, 0]
Defacement	http://www.pashminaonline.com/purepashminas	[37, 0, 1, 0, 2, 0, 0, 14, 3, 0]
Malware	trtsport.cz	[11, 0, 0, 0, 1, 0, 0, 8, 2, 0]
Phishing	br-icloud.com.br	[16, 0, 1, 0, 2, 0, 0, 9, 6, 0]

Dari tabel 4 diatas adalah output dari proses ektraksi fitur yang dimana semua kategori yang masuk kedalam Fitur ekstraksi akan dihitung.

4. Memuat Model dan *One-Hot Encoding*

Langkah ini melibatkan pemrosesan data awal, seperti membaca dataset, membersihkan URL, mengekstraksi fitur, dan mengonversi label kategori menjadi numerik menggunakan One-Hot Encoding. Proses ini mengubah setiap label kategori menjadi vektor biner, di mana setiap kategori diwakili oleh kolom terpisah. Misalnya, untuk kelas benign, vektornya adalah [1, 0, 0, 0], untuk kelas defacement [0, 1, 0, 0], dan seterusnya.Tabel 5 menunjukkan bagaimana masing-masing kategori dikonversi:

TABEL 5

TABEL HASIL ENCODING

Label	benign	defacement	malware	phishing
benign	1	0	0	0
defacement	0	1	0	0
malware	0	0	1	0
phishing	0	0	0	1

5. *Split* Dataset

Pada proses ini dataset dakan dibagi menjadi 2 yaitu data training dan data *testing* dengan rasio 80% : 20%. Proses ini memastikan data pelatihan mencakup semua kategori dengan proporsi yang seimbang.

TABEL 6

TABEL PEMBAGIAN DATASET

Distribusi Kelas pada x_train		
	Class	Count
0	0	342482
1	1	77165
2	3	75289
3	2	26016
Distribusi Kelas pada x_test		
	Class	Count
0	0	85621
1	1	19292
2	2	18822
3	3	6504

6. Mengubah Format Input untuk LSTM

LSTM membutuhkan input dalam format 3D. Oleh karena itu, data pelatihan dan pengujian diubah bentuknya menjadi [samples, time steps, features].

Karena setiap URL direpresentasikan sebagai satu *vector* fitur, panjang timestep (*T*) diset ke 1. Transformasi dilakukan dengan:

$$X_{train\_lstm} = X_{train}.\text{reshape}(N_{train},1,F) \quad (7)$$

$$X_{test\_lstm} = X_{test}.\text{reshape}(N_{test},1,F) \quad (8)$$

Yang dimana :

1. N_{train} : adalah jumlah sampel data pada data latih
2. N_{test} : adalah jumlah sampel data pada data uji.
3. F : adalah jumlah fitur.

Maka :

$$N_{train} = 80\% \times 651{,}191 = 520{,}952 \quad (9)$$

$$N_{test} = 20\% \times 651{,}191 = 130{,}239 \quad (10)$$

Perhitungan dimensi :

$$\text{Dimensi Xtrain\_lstm} = (520{,}952,1,10) \quad (11)$$

$$\text{Dimensi Xtest\_lstm} = (130{,}239,1,10) \quad (12)$$

Dengan F= 10 karena memiliki 10 fitur. Karena setiap URL direpresentasikan sebagai satu vektor fitur, *timestep* (*T*) diset ke 1.

Setelah Transformasi :

- Data latih memiliki dimensi: (520,952, 1, 10)
- Data uji memiliki dimensi: (130,239, 1, 10)

Dimensi ini memastikan bahwa data kompatibel untuk digunakan dalam model LSTM, yang membutuhkan *input* dalam format [samples, timesteps, features].

7. Costom Loss

Fungsi *custom*_loss digunakan untuk memberikan bobot yang lebih besar pada kelas tertentu selama pelatihan Rumus *Costume Loss Function*.

Custom loss function yang diterapkan adalah:

$$\text{Weighted Loss} = \frac{1}{N}\sum_{i=1}^{N} CrossEntropy(y_{true}^{(i)}, y_{pred}^{(i)}) \text{ x} w_{y_{true}^{(i)}} \quad (13)$$

Dimana :

1. N : Jumlah total sampel dalam batch.
2. $y_{true}^{(i)}$) : Label sebenarnya untuk sampel ke-i.
3. $y_{pred}^{(i)}$) : Probabilitas prediksi untuk sampel ke-i.
4. $w_{y_{true}^{(i)}}$: Bobot yang diberikan ke kelas dari label sebenarnya.

Langkah Perhitungan :

$$CrossEntropy\ (y_{true}^{(i)}, y_{pred,c}^{(i)}) = -\sum_{C=1}^{C} y_{true,c}^{(i)} \cdot \log(y_{pred,c}^{(i)}) \quad (14)$$

1. C : Jumlah kelas.
2. y_(true,c)^((i))) : Label benar (one-hot) untuk kelas c pada sampel i
3. y_(pred,c)^((i))) : Probabilitas prediksi model untuk kelas c pada sampel i

Bobot Kelas:

weights = tf.constant([1.0, 1.0, 2.0, 1.0]) (15)

Bobot ini memberi perhatian ekstra pada kelas ketiga dengan w =2.0, karena kelas tersebut adalah minoritas. Untuk setiap sampel, bobot $w_{y_{true}^{(i)}}$ ditentukan berdasarkan label sebenarnya $y_{true}^{(i)}$).

Menggabungkan Bobot :

Setelah menghitung Cross-Entropy untuk setiap sampel, hasilnya dikalikan dengan bobot $w_{y_{true}^{(i)}}$:

$$\text{Weighted Loss per sample} = \text{CrossEntropy}(y_{true,}^{(i)}\ y_{pred}^{(i)})\text{x}\ w_{y_{true}^{(i)}} \quad (16)$$

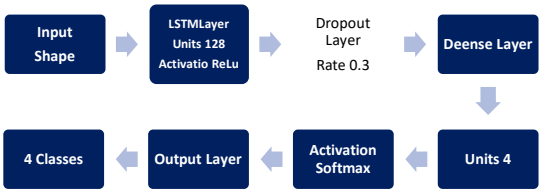
Rata-rata Loss:

$$\text{Weighted Loss}\ \frac{1}{N}\ \sum_{i=1}^{N} Weight\ Loss\ per\ sample \quad (17)$$

Keuntungan menggunakan *custom loss* ini adalah kelas minoritas (yang sering sulit diprediksi) diberi perhatian lebih besar dan juga mengatasi ketidakseimbangan kelas dengan cara yang fleksibel.

8. Definisi dan Kompilasi Model

Model LSTM dibangun dan dikompilasi dengan menggunakan *optimizer* adam dan fungsi *loss* kustom yang telah didefinisikan sebelumnya.



Gambar 4. Arsitektur Model Penerapan

Model LSTM didefinisikan dengan arsitektur sederhana yang terdiri dari beberapa lapisan berikut:

TABEL 7

TABEL HYPERPARAMETER MODEL LSTM

Lapisan	Hypermeter	Nilai/Deskripsi
Input Layer	Input shape	[samples, timesteps, features]
LSTM Layer	Units	128
	Activation	Relu
Dropout Layer	Dropout rate	30%
Dense Layer	Units	4

	Activation	Softmax
Output Layer	Classes	4 (benign, phishing, malware, defacement)

Model dikompilasi menggunakan *optimizer* `Adam`, yang dikenal efisien untuk pelatihan jaringan dalam berbagai kasus. Fungsi *loss* yang digunakan adalah fungsi kustom berbobot *(`custom_loss`)*, yang dirancang untuk memberikan perhatian lebih pada kelas minoritas, mengatasi ketidakseimbangan kelas dalam dataset. Selain itu, metrik evaluasi yang digunakan adalah akurasi, yang memberikan gambaran tentang performa model pada data latih dan validasi selama pelatihan.

9. Pengertian *Early Stopping*

Early stopping digunakan untuk menghentikan pelatihan jika model tidak menunjukkan perbaikan pada *loss* validasi setelah beberapa *epoch*.

10. Pelatihan Model

Model dilatih dengan data pelatihan dan validasi. *Callback early_stopping* digunakan untuk menghentikan pelatihan lebih awal jika perlu. Proses pelatihan model dilakukan menggunakan fungsi model.fit(), di mana data latih X_train_lstm dan y_train diformat dalam tensor 3D. Pelatihan berlangsung hingga 30 *epoch* dengan *batch size* 64, menggunakan data validasi (X_test_lstm, y_test) untuk evaluasi performa di setiap *epoch*. *Callback early_stopping* diterapkan untuk menghentikan pelatihan jika *loss* validasi tidak membaik selama 5 *epoch* berturut-turut, serta memastikan bobot terbaik dipulihkan.

Hasil pelatihan disimpan dalam variabel *history*, yang mencatat metrik seperti *loss* dan akurasi untuk data latih dan validasi. Optimasi dilakukan menggunakan algoritma Adam dengan fungsi *loss* kustom berbobot untuk mengatasi ketidakseimbangan kelas, mencegah *overfitting*, dan meningkatkan performa model.

11. Evaluasi Model dan Hasil Pengujian

Hasil evaluasi model ditampilkan dalam bentuk tabel yang mencakup metrik *precision*, *recall*, *f1-score*, dan support untuk setiap kelas *(benign, defacement, malware, dan phishing)*. Berikut penjelasan dari setiap metrik:

TABEL 8

EVALUASI MODEL dan HASIL PENGUJIAN I

	Precission	Recall	F1-Score	Support
Benign	0.81	0.95	0.88	85601
Defacement	0.75	0.64	0.69	19272
Malware	0.94	0.59	0.72	6494
Phishing	0.73	0.38	0.50	18802
Accuracy			0.80	130169
	0.81	0.64	0.70	130169
	0.80	0.80	0.79	130169

1. *Precision*: Mengukur proporsi prediksi positif yang benar-benar positif. Formula:

$$\text{Precisson} = \frac{True\ Positive\ (TP)}{True\ Positve\ (TP)+False\ Positive\ (FP)} \qquad (18)$$

2. *Reccal*: Mengukur kemampuan model dalam menangkap data positif dengan benar . Formula

$$\text{Precisson} = \frac{True\ Positive\ (TP)}{True\ Positve\ (TP)+False\ Negatif\ (FN)} \qquad (19)$$

3. *F1-Score*: Merupakan rata-rata harmonis antara *precision* dan *recall*, yang memberikan keseimbangan keduanya. Formula:

$$\text{F1 -Score} = \frac{Precission - Reccal}{Precission+Recccal} \qquad (20)$$

4. Support : Jumlah sampel sebenarnya pada masing-masing kelas.

   a. Kelas ***benign***:

1. Precision 0.81 menunjukkan bahwa 81% prediksi *benign* benar.
2. Recall 0.95 menunjukkan bahwa model berhasil mengenali 95% sampel benign dengan benar.
3. F1-Score 0.88 menunjukkan keseimbangan yang tinggi antara precision dan recall untuk kelas ini.

b. Kelas ***defacement***:
1. Precision 0.75 menunjukkan akurasi prediksi kelas ini cukup baik, tetapi tidak sebaik kelas *benign*.
2. Recall 0.64 menunjukkan model hanya mampu mengenali 64% sampel kelas *defacement*.
3. F1-Score 0.69 menunjukkan kinerja keseluruhan pada kelas ini relatif lebih rendah dibanding kelas *benign*.

c. Kelas ***malware***:
1. Precision 0.94 menunjukkan model sangat akurat dalam memprediksi kelas ini.
2. Namun, recall yang rendah (0.59) menunjukkan model gagal mengenali sebagian besar sampel kelas *malware*.
3. F1-Score 0.72 mencerminkan ketidakseimbangan antara precision dan recall pada kelas ini.

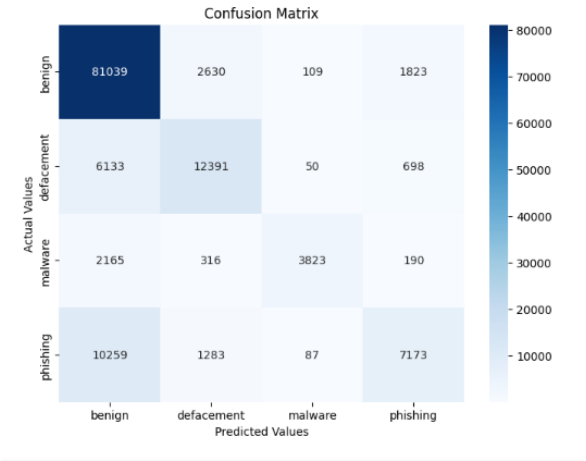
d. Kelas ***phishing***:
1. Precision 0.73 cukup baik, tetapi recall hanya 0.38 menunjukkan bahwa model kesulitan mendeteksi kelas *phishing* dengan baik.
2. Recall 0.38 menunjukkan model hanya mampu mengenali 38% sampel kelas *defacement*
3. F1-Score 0.50 menunjukkan kinerja keseluruhan pada kelas ini masih perlu ditingkatkan.

Dari data hasil pelatihan model diatas dapat disimpulkan bahwa:

1. Akurasi Total: Model memiliki akurasi keseluruhan sebesar 80%, yang mencerminkan persentase prediksi yang benar terhadap total sampel.
2. Rata-Rata Makro (Macro Avg):
   a. Memberikan rata-rata tanpa mempertimbangkan support (jumlah sampel per kelas).
   b. Menunjukkan skor 0.81 untuk precision, 0.64 untuk recall, dan 0.70 untuk f1-score.
3. Rata-Rata Tertimbang (*Weighted Avg*):
   a. Mempertimbangkan support saat menghitung rata-rata.

Menunjukkan skor 0.80 untuk precision dan recall, serta 0.79 untuk f1-score, yang lebih mencerminkan kinerja model secara keseluruhan karena memperhitungkan ketidakseimbangan jumlah sampel pada masing-masing kelas.

a. Confusiion Matrix untuk Analisis Model



Gambar 5. Confusiion Matrix Data Internal

Confusion Matrix di atas menggambarkan distribusi hasil prediksi model untuk masing-masing kelas terhadap nilai aktualnya. Setiap baris menunjukkan jumlah sampel dari kelas aktual, sementara setiap kolom menunjukkan jumlah sampel yang diprediksi sebagai kelas tertentu.

a. Kelas *Benign*
1. Prediksi benar (*true positive*)
   Terdapat 81,039 sampel berhasil diklasifikasikan dengan benar sebagai *benign*.
2. Prediksi salah
   Didapatkan 2,630 sampel salah diklasifikasikan sebagai defacement, 109 sebagai *malware*, dan 1,823 sebagai *phishing*.

b. Kelas *Defacement*
   1. Prediksi benar
      Didapatkan 12,391 sampel diklasifikasikan dengan benar sebagai *defacement*.
   2. Prediksi salah
      Didapatkan 6,133 salah diklasifikasikan sebagai *benign*, 50 sebagai *malware*, dan 698 sebagai *phishing*.

c. Kelas *Malware*
   1. Prediksi benar
      Didapatkan 3,823 sampel diklasifikasikan dengan benar sebagai *malware*.
   2. Prediksi salah
      Didapatkan 2,165 salah diklasifikasikan sebagai *benign*, 316 sebagai defacement, dan 190 sebagai *phishing*.

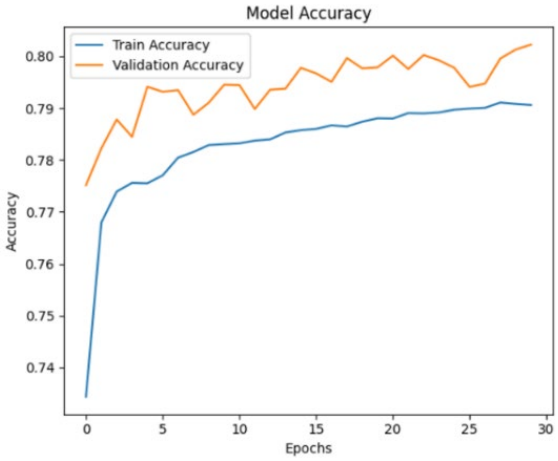
d. Kelas *Phishing*
   1. Prediksi benar
      Didapatkan 7,173 sampel diklasifikasikan dengan benar sebagai *phishing*.
   2. Prediksi salah
      Didapatkan 10,259 salah diklasifikasikan sebagai benign, 1,283 sebagai *defacement*, dan 87 sebagai malware.

Model menunjukkan performa yang sangat baik untuk kelas *benign* dengan akurasi prediksi yang tinggi, namun mengalami kesulitan dalam mengklasifikasikan kelas *phishing* dan *malware*, yang terlihat dari jumlah kesalahan prediksi yang cukup besar pada kelas-kelas tersebut. Ketidakseimbangan data antar kelas kemungkinan memengaruhi performa model, sehingga strategi seperti *oversampling* atau penyesuaian bobot kelas dapat diterapkan untuk meningkatkan akurasi pada kelas minoritas.

B. Evaluasi Model Berdasarkan Grafik

Grafik yang dihasilkan dari proses pelatihan dan evaluasi model menunjukkan tren kinerja yang penting untuk dianalisis lebih lanjut.

a. Grafik Model Accuracy

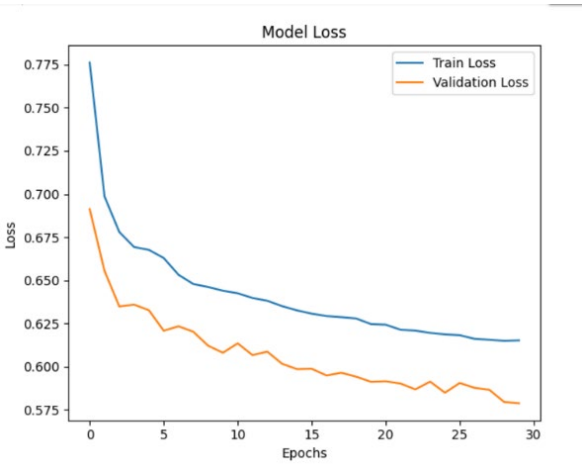


Gambar 6. Grafik Model Accuracy

Grafik pada Gambar 6 menunjukkan perkembangan akurasi model selama proses pelatihan untuk data pelatihan (*Train Accuracy*) dan validasi (*Validation Accuracy*).

1. Sumbu X: Representasi jumlah epoch (iterasi pelatihan).
2. Sumbu Y: Akurasi model.
3. Pada grafik ini, terlihat bahwa akurasi untuk data pelatihan terus meningkat seiring bertambahnya epoch, menunjukkan bahwa model belajar dari data pelatihan dengan baik.
4. Akurasi validasi juga meningkat, meskipun ada beberapa *fluktuasi* kecil. Ini menunjukkan bahwa model memiliki kemampuan generalisasi yang cukup baik terhadap data yang belum pernah dilihat.

b. Grafik Model Loss

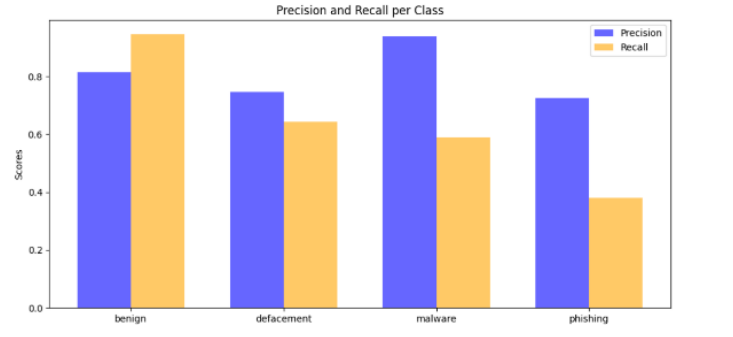


Gambar 7. Grafik Model Loss

Grafik pada Gambar 7 menunjukkan nilai loss (kesalahan) selama proses pelatihan untuk data pelatihan (*Train Loss*) dan validasi (*Validation Loss*).

1. Sumbu X: Jumlah epoch.
2. Sumbu Y: Nilai loss.
3. Nilai loss data pelatihan menurun secara konsisten seiring bertambahnya epoch, menunjukkan bahwa model mempelajari pola dari data pelatihan dengan baik.
4. Nilai loss untuk data validasi juga menurun, meskipun dengan pola yang sedikit fluktuatif. Penurunan ini mengindikasikan bahwa model berhasil meminimalkan kesalahan prediksi untuk data yang belum pernah dilihat, tanpa mengalami overfitting.

c. Grafik *Precision* dan *Recall* per Class



Gambar 8. Grafik Precision dan Recall per Class

Grafik pada Gambar 8 memperlihatkan nilai precision dan recall untuk masing-masing kelas (*benign*, *defacement*, *malware*, dan *phishing*).

1. Sumbu X: Label kelas.
2. Sumbu Y: Nilai skor untuk *precision* dan *recall*.
3. Interpretasi per kelas.
   a. *Benign*
      Memiliki skor *precision* dan *recall* yang tinggi, menunjukkan bahwa model sangat baik dalam mengenali URL benign dan minim kesalahan.
   b. *Defacement*
      Precision cukup baik, tetapi *recall* lebih rendah, menunjukkan model tidak menangkap semua sampel *defacement* dengan baik.
   c. *Malware*
      Precision sangat tinggi, tetapi *recall* lebih rendah, menunjukkan model mampu mengenali sebagian besar prediksi *malware* dengan benar tetapi kehilangan beberapa sampel.

d. Phishing
Memiliki skor *precision* yang moderat, tetapi recall rendah. Hal ini menunjukkan bahwa model kesulitan menangkap URL *phishing* secara menyeluruh.

C. Pengujian Menggunakan Sampel Dataset Eksternal

Pada pengujian ini, model yang telah dilatih sebelumnya akan disimpan bersama dengan *label encoder* yang digunakan untuk memetakan label kelas ke dalam bentuk numerik. Pengujian dilakukan menggunakan 100 URL untuk kelas *benign*, *phishing*, *defacement*, dan 50 URL untuk *malware*, yang tidak termasuk dalam data pelatihan. Proses ini mengukur kemampuan generalisasi model dengan data eksternal melalui langkah *preprocessing* yang sama, seperti pembersihan URL dan ekstraksi fitur manual.

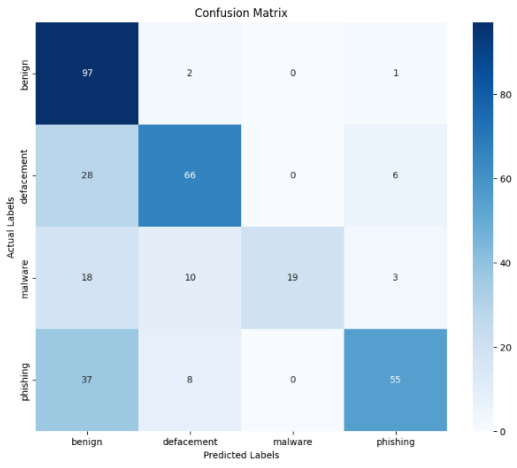
Hasil pengujian memberikan gambaran kemampuan model dalam mengklasifikasikan URL dunia nyata, mengidentifikasi kelemahan pada kelas tertentu (seperti *phishing* atau *malware*), serta peluang untuk optimasi performa lebih lanjut.

TABEL 9
HASIL EVALUASI MODEL PADA DATASET EKSTERNAL

	Precission	Recall	F1-Score	Support
Benign	0.54	0.97	0.69	100
Defacement	0.77	0.66	0.71	100
Malware	1.0	0.38	0.55	50
Phishing	85	0.55	0.67	100
Accuracy			0.68	350
	0.79	0.64	0.65	350
	0.76	0.68	0.67	350

Hasil evaluasi model pada dataset eksternal menunjukkan akurasi keseluruhan sebesar 68%. Model memiliki performa baik pada kelas mayoritas seperti *benign* (*precision*: 0.54, recall: 0.97) dan *defacement* (*precision*: 0.77, *recall*: 0.66). Namun, performa menurun pada kelas minoritas seperti *malware* (*recall*: 0.38) dan *phishing* (*recall*: 0.55). Ketidakseimbangan antara precision dan recall menghasilkan *F1-score* yang bervariasi (tertinggi pada defacement: 0.71). Hasil ini menunjukkan kebutuhan optimasi lebih lanjut, terutama pada kelas *malware* dan *phishing*.

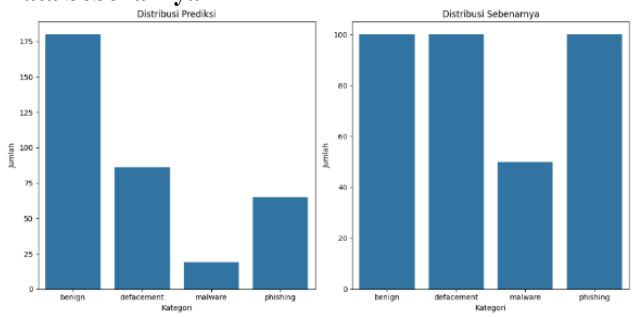
a. Confusiion Matrix Data External



Gambar 9. Confusiion Matrix Data External

Confusion matrix pada Gambar 9 menunjukkan model memiliki performa baik pada kelas *benign* (97 prediksi benar) dan *defacement* (66 prediksi benar). Namun, model kesulitan pada kelas *malware* (19 prediksi benar) dan *phishing* (55 prediksi benar), dengan banyak kesalahan klasifikasi ke kelas mayoritas seperti *benign*. Perbaikan diperlukan untuk meningkatkan deteksi kelas minoritas.

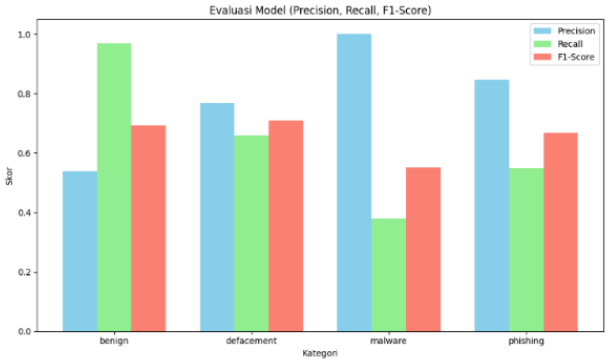
b. Grafik Prediksi dan Data Sebenarnya



Gambar 10. Grafik Prediksi dan Data Sebenarnya

Grafik pada Gambar 10 menunjukkan distribusi prediksi model dibandingkan distribusi sebenarnya pada dataset eksternal. Model cenderung mendominasi kelas *benign* dengan lebih dari 175 prediksi, sementara kelas *defacement* memiliki sekitar 100 prediksi, mendekati distribusi sebenarnya. Kelas *phishing* hanya memiliki sekitar 75 prediksi, dan kelas *malware* kurang dari 50, menunjukkan kesulitan model dalam mengenali pola kelas ini. Distribusi sebenarnya relatif merata untuk kelas *benign*, *defacement*, dan *phishing*, namun kelas *malware* lebih sedikit. Model menunjukkan bias terhadap kelas mayoritas(*benign*) dan kesulitan mendeteksi kelas minoritas seperti *malware*, sehingga optimasi diperlukan untuk meningkatkan performa model.

c. Grafik Evaluasi Precision, Recall, dan F1-score



Gambar 11. Grafik evaluasi precision, recall, dan F1-score

Grafik pada Gambar 11 menunjukkan evaluasi performa model berdasarkan metrik *precision*, *recall*, dan *F1-score* untuk masing-masing kategori (*benign*, *defacement*, *malware*, dan *phishing*). Hasil evaluasi ini mengungkapkan kekuatan dan kelemahan model dalam mendeteksi berbagai jenis URL, serta memberikan wawasan tentang kesalahan prediksi yang terjadi. Penjelasan dan analisis rinci untuk setiap kategori dijabarkan sebagai berikut.

1. **Kelas *Benign***
   Model menunjukkan precision rendah (~0.54) namun recall tinggi (~0.97), artinya banyak prediksi positif salah, namun hampir semua URL *benign* terdeteksi. F1-Score (~0.69) mencerminkan keseimbangan moderat antara *precision* dan *recall*.
   Kesalahan prediksi: http://www.designeremdoces.com/components/com_contact/ggdrives/ (*benign*, namun *phishing*). Faktor kesalahannya adalah struktur URL mirip URL *benign*, model kurang efektif membedakan keduanya. Pengembangan fitur analisis URL atau metadata dapat memperbaiki deteksi.
2. **Kelas *Defacement***
   Precision cukup baik, namun recall rendah, menunjukkan model kesulitan membedakan URL defacement dari benign. F1-Score mencerminkan ketidakseimbangan antara identifikasi *defacement* dan kesalahan prediksi.
   Kesalahan prediksi: http://larcadelcarnevale.com/catalogo/palloncini (*defacement*, diprediksi *benign*). Faktor kesalahannya adalah penggunaan domain sah yang dimodifikasi. Model perlu lebih sensitif terhadap perubahan kecil. Penggunaan fitur analisis metadata atau deteksi perubahan konten dapat meningkatkan akurasi.

3. **Kelas *Malware***

   Precision sangat tinggi (1.0), namun recall rendah (~0.38), menunjukkan banyak URL malware tidak terdeteksi dan diklasifikasikan sebagai benign. F1-Score (~0.55) mencerminkan ketidakseimbangan deteksi malware dan kesalahan prediksi.

   Kesalahan prediksi: http://www.824555.com/app/member/SportOption.php?uid=guest&langx=gb (*malware*, diprediksi *benign*). Faktor kesalahannya adalah URL mirip sah, model kurang sensitif terhadap ciri *malware*. Penggunaan analisis perilaku situs atau metadata dapat meningkatkan sensitivitas model.

4. **Kelas *Phishing***

   Pada kelas *phishing* Model sangat kesulitan mendeteksi URL phishing, dengan *precision* dan *recall* sangat rendah.

   Kesalahan prediksi: http://www.marketingbyinternet.com/mo/e56508df639f6ce7d55c81ee3fcd5ba8/ (*phishing*, diprediksi *benign*). Faktor kesalahannya adalah model tidak dapat menangkap pola khas URL *phishing*. Pengembangan lebih lanjut dengan fitur analisis halaman atau model lebih canggih dibutuhkan.

Hasil evaluasi di atas menunjukkan bahwa performa model bervariasi pada setiap kategori. Model menunjukkan performa yang baik pada kelas mayoritas (*benign* dan *defacement*), tetapi memiliki keterbatasan pada kelas minoritas (*malware* dan *phishing*), terutama dalam hal *recall*. Keterbatasan ini dapat disebabkan oleh beberapa faktor, antara lain:

1. Ketidakseimbangan Dataset.
   Kelas minoritas seperti *malware* dan *phishing* memiliki jumlah sampel yang jauh lebih sedikit dibandingkan kelas mayoritas, sehingga model kurang terlatih untuk mengenali pola dari kelas ini.
2. Fitur yang Kurang *Discriminative*.
   Beberapa fitur yang digunakan model, seperti panjang URL, keberadaan karakter khusus, atau struktur domain, tidak cukup untuk membedakan URL dari kelas minoritas.
3. *Overfitting* pada Kelas Mayoritas.
   Model cenderung lebih fokus pada pola dari kelas mayoritas, sehingga performa pada kelas minoritas menurun.

Secara keseluruhan, hasil evaluasi menunjukkan bahwa model mampu mendeteksi URL dengan tingkat keberhasilan yang bervariasi di setiap kategori. Kinerja terbaik ditunjukkan pada kelas *benign* dan *defacement*, yang mencerminkan kemampuan model dalam mengenali pola dari URL yang umum ditemukan. Sebaliknya, tantangan signifikan terlihat pada kelas *malware* dan *phishing*, di mana model mengalami kesulitan dalam mengidentifikasi URL yang tergolong dalam kategori ini. Hasil ini menunjukkan kompleksitas deteksi ancaman keamanan siber, khususnya pada URL dengan karakteristik yang sulit dibedakan.

IV. SIMPULAN

Penelitian ini berhasil mengimplementasikan metode Long Short-Term Memory (LSTM) untuk mendeteksi URL phishing menggunakan dataset dengan empat kategori yaitu kategori *benign*, *defacement*, *phishing*, dan *malware*. Model mencapai akurasi 68% pada dataset eksternal, dengan performa terbaik pada kategori benign (recall 0,97). Namun, kinerja menurun pada kategori *phishing* (*recall* 0,55) dan *malware* (recall 0,38) akibat ketidakseimbangan data dan pola yang menyerupai *benign*. Optimasi lebih lanjut diperlukan, seperti *oversampling*, penyesuaian bobot kelas, dan penambahan fitur seperti metadata. Penelitian ini berkontribusi dalam pengembangan sistem deteksi ancaman siber berbasis deep learning dan membuka peluang untuk peningkatan akurasi pada dataset yang lebih beragam dan realistis, sehingga dapat menghasilkan solusi keamanan siber yang lebih andal di masa depan.

UCAPAN TERIMAKASIH

Penulis mengucapkan syukur kepada Tuhan Yang Maha Esa atas terselesaikannya junal ini. Terima kasih kepada Universitas Teknokrat Indonesia, dosen pembimbing, keluarga, dan pihak-pihak yang telah memberikan dukungan, arahan, serta motivasi selama proses penelitian ini. Semoga penelitian ini bermanfaat dan dapat menjadi kontribusi bagi perkembangan ilmu pengetahuan.

DAFTAR PUSTAKA

[1] V. A. Windarni, A. F. Nugraha, S. T. A. Ramadhani, D. A. Istiqomah, F. M. Puri, and A. Setiawan, "Deteksi Website Phishing Menggunakan Teknik Filter Pada Model Machine Learning," *Inf. Syst. J.*, vol. 6, no. 01, pp. 39–43, 2023, doi: 10.24076/infosjournal.2023v6i01.1268.

[2] L. Tang and Q. H. Mahmoud, "A Deep Learning-Based Framework for Phishing Website Detection," *IEEE Access*, vol. 10, pp. 1509–1521, 2022, doi: 10.1109/ACCESS.2021.3137636.

[3] M. A. B. Dewanto, M. Fathurrahman, D. R. Firdaus, and A. Setiawan, "Penipuan Penambah Followers Instagram: Analisis Serangan Phising dan Dampaknya pada Keamanan Data," *J. Internet Softw. Eng.*, vol. 1, no. 4, p. 11, 2024, doi: 10.47134/pjise.v1i4.2672.

[4] Bjcoid2, "Serangan Phishing di Indonesia Terus Meningkat," bankjombang.co.id. Accessed: Dec. 17, 2024. [Online]. Available: https://bankjombang.co.id/serangan-phishing-di-indonesia-terus-meningkat-berikut-data-lengkapnya/

[5] A. Abuadbba *et al.*, "Towards Web Phishing Detection Limitations and Mitigation," 2022, [Online]. Available: http://arxiv.org/abs/2204.00985

[6] A. S. Sitio and F. A. Sianturi, "Penerapan Algoritma Machine Learning dalam Analisis Pola Perilaku Penggunaan Internet," *DIKE J. Ilmu Multidisiplin*, vol. 2, no. 2, pp. 46–51, 2024, doi: 10.69688/dike.v2i2.102.

[7] A. Fathurohman, "Machine Learning Untuk Pendidikan: Mengapa Dan Bagaimana," *J. Inform. dan Teknol. Komput.*, vol. 1, no. 3, pp. 57–62, 2021, [Online]. Available: https://journal.amikveteran.ac.id/index.php/jitek/article/view/306

[8] H. Gurung, R. Nepal, and S. Nepal, "Phishing URL Detection Using CNN-LSTM and Random Forest Classifier," *Int J Med Net*, vol. 2, no. 5, pp. 1–6, 2023, [Online]. Available: https://www.

[9] S. Aslam, H. Aslam, A. Manzoor, H. Chen, and A. Rasool, "AntiPhishStack: LSTM-Based Stacked Generalization Model for Optimized Phishing URL Detection," *Symmetry (Basel).*, vol. 16, no. 2, 2024, doi: 10.3390/sym16020248.

[10] B. Banik and A. Sarma, "Phishing Url Detection Using Lstm Based Ensemble Learning Approaches," *Int. J. Comput. Networks Commun.*, vol. 15, no. 1, pp. 17–33, 2023, doi: 10.5121/ijcnc.2023.15102.

[11] S. Shabudin, N. S. Sani, K. A. Z. Ariffin, and M. Aliff, "Feature selection for phishing website classification," *Int. J. Adv. Comput. Sci. Appl.*, vol. 11, no. 4, pp. 587–595, 2020, doi: 10.14569/IJACSA.2020.0110477.

[12] R. Indu, M. Bhavya, V. Pardhasaradhi, Y. S. Ram, and Y. Suresh, "Malicious url detection 1," vol. 11, no. 4, pp. 612–618, 2023, [Online]. Available: https://ijcrt.org/papers/IJCRT2304563.pdf

[13] N. P. S. Wati and C. Pramartha, "Penerapan Long Short Term Memory dalam Mengklasifikasi Jenis Ujaran Kebencian pada Tweet Bahasa Indonesia," *J. Nas. Teknol. Inf. dan Apl.*, vol. 1, no. 1, pp. 755–762, 2022.

[14] A. Hasiholan, I. Cholissodin, and N. Yudistira, "Analisis Sentimen Tweet Covid-19 Varian Omicron pada Platform Media Sosial Twitter menggunakan Metode LSTM berbasis Multi Fungsi Aktivasi dan GLOVE," *J. Pengemb. Teknol. Inf. dan Ilmu Komput.*, vol. 6, no. 10, pp. 4653–4661, 2022, [Online]. Available: https://j-ptiik.ub.ac.id/index.php/j-ptiik/article/view/11648

[15] T. Bastian Sianturi, I. Cholissodin, and N. Yudistira, "Penerapan Algoritma Long Short-Term Memory (LSTM) berbasis Multi Fungsi Aktivasi Terbobot dalam Prediksi Harga Ethereum," *J. Pengemb. Teknol. Inf. dan Ilmu Komput.*, vol. 7, no. 3, pp. 1101–1107, 2023, [Online]. Available: http://j-ptiik.ub.ac.id