BAB III
METODE PENELITIAN

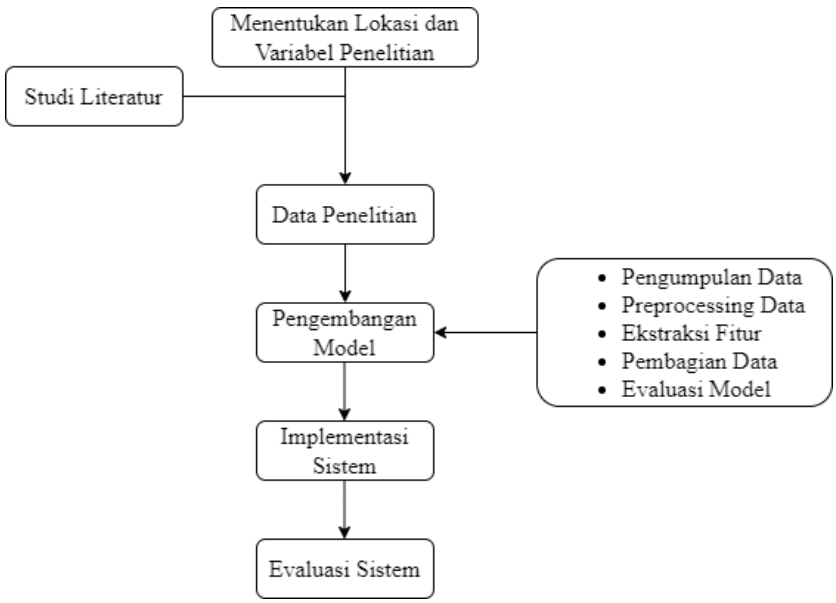
3.1 Objek Penelitian

Fokus utama dalam penelitian ini adalah penerapan algoritma *Long Short-Term Memory* pada sistem deteksi situs *phishing* berbasis aplikasi *web*. Algoritma LSTM digunakan untuk menganalisis struktur URL serta melakukan klasifikasi untuk mengkategorikan suatu situs sebagai *phishing* atau sah. Algoritma LSTM dipilih karena kemampuannya sangat baik dalam memproses data sekuensial seperti URL sehingga dapat menangkap pola-pola yang mungkin menunjukkan tanda-tanda *phishing*. Dengan demikian, penelitian ini bertujuan untuk meningkatkan akurasi model sehingga lebih efisien dalam mencegah serangan *phishing*.

3.2 Metode Penelitian

3.2.1 Jenis Penelitian dan Metode yang Digunakan

Jenis penelitian yang dilakukan pada penelitian ini adalah penerapan algoritma LSTM pada aplikasi *web* dengan tujuan untuk memberikan solusi atas permasalahan serangan situs *phishing* yang terjadi dalam kehidupan sehari-hari. Dalam penerapannya diperlukan beberapa alur untuk mencapai tujuan dari dilaksanakannya penelitian yang digambarkan pada Gambar 3.1.



Gambar 3.1 Alur Penelitian

Alur pelaksanaan penelitian pada Gambar 3.1 dimulai dari tahap perencanaan yang dijelaskan sebagai berikut:

a. Menentukan Permasalahan dan Studi Literatur

   Menentukan permasalahan yang akan dianalisis dalam penelitian dengan melakukan studi pustaka dari beberapa sumber data literatur yang relevan.

b. Merumuskan Masalah

   Merumuskan masalah yang akan diteliti dengan merumuskan judul, rumusan masalah dan tujuan dilakukannya penelitian serta membuat alur penelitian berdasarkan masalah dan tujuan yang telah ditentukan.

c. Memilih Metode dan Pendekatan Penelitian

   Dalam tahapan perencanaan, penulis juga memilih metode dan pendekatan penelitian yang digunakan. Penelitian ini dilakukan menggunakan pendekatan algoritma LSTM.

d. Menentukan Variabel

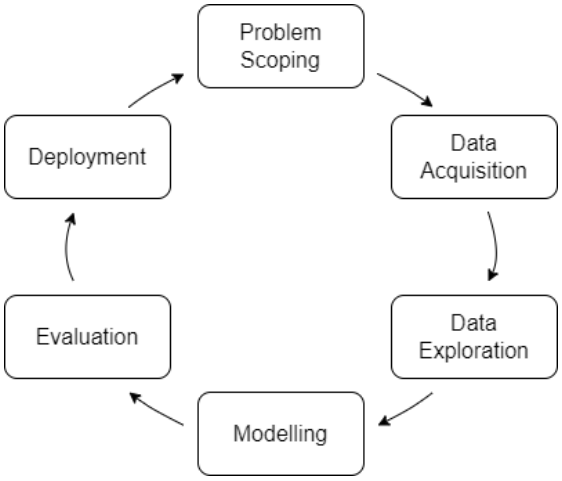
   Variabel yang ditentukan dalam penelitian ini diantaranya fitur yang diekstrak dari struktur URL menjadi bentuk numerik sebagai variabel X dan hasil dari prediksi model sebagai variabel Y.

e. Menyusun Instrumen

   Dilakukan penentuan dan penyusunan instrumen penelitian.

Berdasarkan alur penelitian diatas, terdapat dua pengembangan sistem utama yaitu pengembangan model dan implementasi pada sistem yang dilakukan setelah tahap perencanaan. Setelah seluruh tahapan telah dilakukan, maka dilanjutkan dengan tahap pembuatan laporan penelitian dalam bentuk tertulis berdasarkan kaidah-kaidah penulisan karya tulis ilmiah dan sesuai hasil penelitian berdasarkan data yang telah diolah. Untuk melaksanakan seluruh tahapan tersebut, metode penelitian yang digunakan dalam penelitian ini adalah *Framework* AI *project cycle*.

AI *project cycle* merupakan suatu pendekatan untuk membuat suatu proyek menggunakan AI secara utuh. Pendekatan metode ini disusun secara sistematis dan berurutan yang mencakup enam proses tahapan yaitu *Problem Scoping, Data Acquisition, Data Exploration, Modelling, Evaluation,* dan *Deployment* (Muhyidin & Venica, 2023). Tahapan pada proses ini direpresentasikan pada Gambar 3.2.



Gambar 3.2 Metode Penelitian AI *Project Cycle*

1. *Problem Scoping*

Pada tahap ini dilakukan identifikasi dan analisis masalah yang bertujuan untuk menyelesaikan masalah tersebut berdasarkan rumusan dan tujuan penelitian. Dengan menggunakan metode 4W, maka analisis dalam proses *problem scoping* dipetakan sebagai berikut:

a) *Who*: siapa yang terlibat? Penulis yang mengidentifikasi masalah dan mengembangkan solusi berupa aplikasi *web* berbasis algoritma LSTM dan pengguna yang mendeteksi situs *phishing* menggunakan aplikasi *web* yang dihasilkan dari penelitian ini sehingga terlindung dari serangan *phishing*.

b) *What*: apa masalah yang ditetapkan? Masalah yang ditetapkan adalah jumlah serangan *phishing* meningkat tiap tahunnya dan sangat berbahaya karena dapat mencuri informasi pribadi untuk disalahgunakan.

c) *Where*: kondisi saat masalah ditemukan? Serangan situs *phishing* dapat ditemukan dimanapun ketika mengakses internet karena dibagikan melalui berbagai platform *online*, seperti *email*, media sosial, dan situs *web*.

d) *Why* : alasan masalah perlu diselesaikan dan solusinya? Mengatasi serangan situs *phishing* perlu dilakukan untuk melindungi pengguna dari pencurian data. Solusi dalam penelitian ini yaitu membuat aplikasi *web* untuk mendeteksi situs *phishing* menggunakan algoritma LSTM.

2. *Data Acquisition*

Data yang digunakan dalam penelitian ini merupakan penggabungan dari beberapa sumber data, yaitu dataset yang dapat diakses secara publik melalui

platform UC *Irvine Machine Learning Repository* berjudul *PhiUSIIL Phishing URL (Website)* (Prasad & Chandra, 2024) dengan jumlah data sebanyak 134.850 situs sah dan 100.945 situs *phishing* dan *dataset* dari platform *Kaggle* berjudul *Phishing and Legitimate URLS* (Sudhan, 2024). Penulis juga menambahkan situs *phishing* terbaru dari *website PhishTank* yang dikumpulkan hingga tanggal 9 Mei 2024 dan menggabungkannya dengan *dataset* yang digunakan dalam penelitian oleh (Hannousse & Yahiouche, 2021).

Penggabungan data dari berbagai sumber yang berbeda ini menghasilkan satu set data yang lebih selaras dan konsisten untuk digunakan pada pelatihan dan pengujian model. Total kumpulan data dalam penelitian ini berjumlah 784.762 URL dengan pembagian data diantaranya 392.381 URL *phishing* berlabel 1 dan 392.381 URL sah berlabel 0. Atribut dari *dataset* terdiri dari:

a) URL: Atribut yang berisi alamat situs *web* yang akan dianalisis oleh algoritma LSTM.
b) Status: Label atau kelas yang menunjukkan URL sebagai *phishing* atau sah.

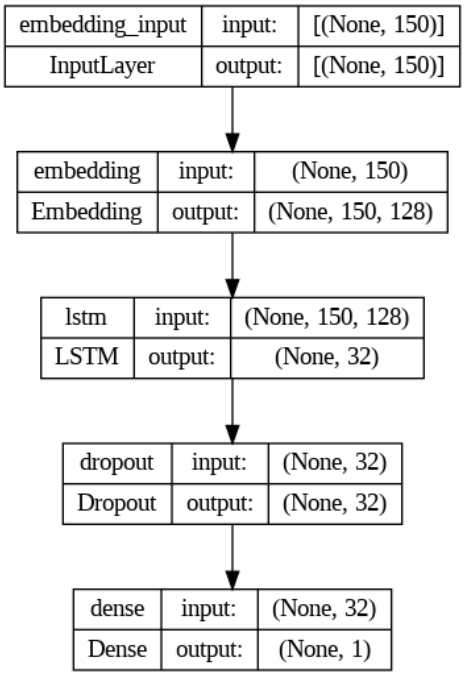
Kumpulan data ini dirancang untuk digunakan sebagai data penelitian dalam sistem deteksi situs *phishing* berbasis aplikasi *web* dengan representasi domain yang beragam. Pembagian data dalam *dataset* ini seimbang dimana 50% URL merupakan URL *phishing* dan 50% URL sah sehingga mengindikasikan bahwa model yang dihasilkan memiliki potensi untuk menghasilkan performa yang baik dalam mendeteksi situs *phishing*.

3. *Data Exploration*

Untuk mempelajari isi *dataset* serta memahami karakteristik dan pola data yang ada, maka dilakukan tahap *pre-processing* dan *feature extraction* sebagai pemrosesan *input* data. Proses ini melibatkan penggunaan *library pandas* yang merupakan *library python* untuk analisis data. Tahap awal *pre-processing* merupakan analisis jumlah data dan isinya untuk memastikan tidak ada data yang kosong dan analisis karakteristik data agar data yang digunakan dalam pelatihan model konsisten dan representatif. Proses selanjutnya yaitu ekstraksi fitur pada struktur URL kemudian dilakukan pembagian data menjadi data latih dan data uji, dimana data latih digunakan untuk pemrosesan menggunakan algoritma LSTM dan data uji digunakan untuk pengujian model.

4. *Modelling*

Untuk menemukan pola-pola dalam data untuk membuat keputusan prediksi dalam berdasarkan model yang dihasilkan maka diterapkan algoritma LSTM pada kumpulan data pelatihan dan melakukan *training* model. Algoritma LSTM merupakan bentuk *sequential* model dimana lapisannya bekerja secara berurutan, sehingga pada proses ini digunakan fungsi *sequential* agar lapisan-lapisan ditambahkan secara berurutan satu per satu. Proses pelatihan model diimplementasikan di *keras*, yang merupakan API *neural* tingkat tinggi yang bekerja dengan *framework* pembelajaran mesin *open source* yang disebut *TensorFlow*. Sebelum masuk ke pelatihan model ditentukan beberapa *hyperparameter* yang merupakan parameter jaringan saraf yang ditetapkan sebelum pelatihan dimulai yaitu: ukuran *batch* untuk mengatur banyak sampel yang diproses saat pelatihan untuk mengurangi penggunaan memori dibandingkan memproses seluruh *dataset* sekaligus, *epochs* untuk mengatur proses pengulangan seluruh kumpulan data pelatihan, *loss* untuk mengevaluasi efektivitas model dan *optimizer* untuk memperbarui parameter model berdasarkan gradien yang dihitung selama *backpropagation* (Aldakheel dkk., 2023). Arsitektur lapisan model LSTM untuk pelatihan data dalam penelitian ini digambarkan pada Gambar 3.3.



Gambar 3.3 Arsitektur Model Penelitian

a) *Input Layer: Input Shape* (*None*, 150); *Output Shape* (*None*, 150)

Pada lapisan pertama *input* URL diterima dalam bentuk urutan kata sebagai indeks integer dengan panjang maksimal 150 karakter. *None* menunjukkan bahwa *batch size* tidak ditentukan, sehingga model dapat menerima berbagai ukuran *batch.*

b) *Embedding Layer*: *Input Shape* (*None*, 150); *Output Shape* (*None*, 150, 128)

Lapisan *embedding* digunakan untuk mengubah *input* URL menjadi vektor 128-dimensi menggunakan layer *embedding* dan ukuran kamus token ditambah satu untuk menangani token yang tidak ada didalam daftar kamus menggunakan *len(vocabulary) + 1*. Proses ini dilakukan agar model dapat memahami dan memproses karakter-karakter dalam URL dengan mempelajari pola representasi dari urutan karakter atau kata dalam URL tanpa perlu ekstraksi fitur manual. Lapisan ini berfungsi untuk menjaga keseimbangan antara kemampuan representasi model dan kompleksitas model dan menghasilkan urutan vektor untuk diproses lapisan LSTM sebagai urutan 150 langkah, karena panjang *input* URL yang digunakan pada lapisan *embedding* berjumlah 150 karakter.

c) LSTM *Layer*: *Input Shape*: (*None*, 150, 128); *Output Shape*: (*None*, 32)

Selanjutnya urutan dipadatkan menjadi vektor berdimensi 32 karena jumlah *neuron* yang digunakan dalam lapisan LSTM adalah 32 atau satu lapisan LSTM. Proses analisis informasi dalam lapisan LSTM diproses secara berurutan untuk mempelajari pola-pola informasi dalam data melalui tiga gerbang LSTM yaitu:

1) *Forget Gate*

   Output dari *forget gate* menghasilkan nilai antara 0 jika melupakan informasi yang tidak relevan dan nilai 1 jika mempertahankan informasi untuk setiap elemen dalam sel memori.

2) Input Gate

   Input gate menggunakan fungsi *tanh* untuk menghasilkan vektor yang ditambahkan ke sel memori. *Input gate* juga menggunakan fungsi aktivasi *sigmoid* sehingga menghasilkan nilai antara 0 untuk

mengabaikan informasi dan 1 untuk menyimpan informasi setiap elemen yang akan ditambahkan ke sel memori.

3) *Update* Sel Memori

Sel memori diperbarui dengan mengurangi informasi yang akan dilupakan berdasarkan *output* dari *forget gate* dan menambahkan informasi baru berdasarkan *output* dari *input gate*.

4) *Output Gate*

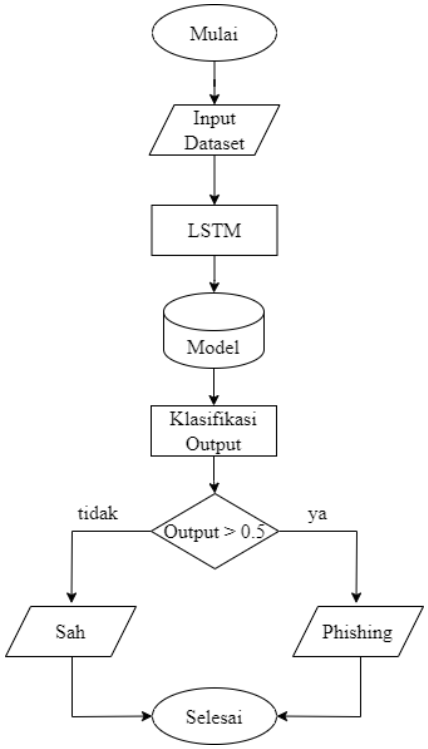
Output dari sel LSTM dihasilkan dengan mengalikan *output* dari sel memori dengan *output* dari *output gate*.

d) *Dropout Layer*: *Input Shape*: (*None*, 32); *Output Shape*: (*None*, 32)

Selama proses pelatihan menggunakan LSTM, lapisan *dropout* digunakan untuk mencegah *overfitting* yang secara acak mengabaikan sebagian *neuron* selama pelatihan untuk mencegah ketergantungan antar *neuron*. Dengan menggunakan *Dropout(0.5)*, tingkat *drop* yang diberikan dalam penelitian ini sebesar 0.5, yang berarti setiap *neuron* di *layer dropout* memiliki probabilitas 50% untuk dinonaktifkan secara acak selama pelatihan. Penggunaan satu lapisan LSTM dengan *dropout rate* sebesar 0.5 didasarkan pada penelitian sebelumnya karena efisien digunakan untuk mencapai tingkat akurasi yang tinggi (Do dkk., 2021).

e) *Dense Layer*: *Input Shape*: (*None*, 32); *Output Shape*: (*None*, 1)

Lapisan terakhir ialah *layer fully connected* atau lapisan *dense* yang digunakan untuk menghasilkan satu unit *output* dengan aktivasi *sigmoid* untuk mengubah *output* menjadi nilai antara 0 dan 1 dan memprediksi probabilitas biner tersebut. Model menghasilkan klasifikasi berdasarkan analisis mendalam terhadap struktur URL selama proses pelatihan sehingga mengenali pola-pola dalam URL *phishing* atau sah. Dari model yang telah dihasilkan, jika probabilitas pola *phishing* lebih besar dari 0.5, maka keluarannya akan diprediksi 1 *(phishing)* dan jika lebih kecil dari atau sama dengan 0.5, maka keluaran yang diprediksi adalah 0 (sah). Diagram alir proses klasifikasi model digambarkan pada Gambar 3.4.



Gambar 3.4 Diagram Alir Proses Klasifikasi

5. *Evaluation*

Pengukuran performa sistem diukur menggunakan metrik evaluasi akurasi karena memberikan informasi tentang seberapa baik model dapat mengklasifikasikan data dengan benar, sehingga pengukuran ini relevan digunakan dalam deteksi situs *phishing.* Proses pelatihan data menghasilkan model untuk melakukan evaluasi dari hasil pengujian data dengan mengklasifikasikan semua URL dalam data uji sebagai *phishing* atau sah yang dibandingkan dengan label aslinya untuk mengukur performa model.

6. *Deployment*

Deployment dalam penelitian ini merupakan integrasi model LSTM dengan aplikasi *web* deteksi situs *phishing* untuk dapat melakukan prediksi pada aplikasi yang dijalankan secara lokal. Aplikasi *web* dibangun sebagai langkah pencegahan serangan situs *phishing* menggunakan *flask* yang merupakan kerangka *web* dengan bahasa pemrograman *python*. Tahap *deployment* dibagi menjadi dua proses utama yaitu:

a) Analisis Kebutuhan Sistem

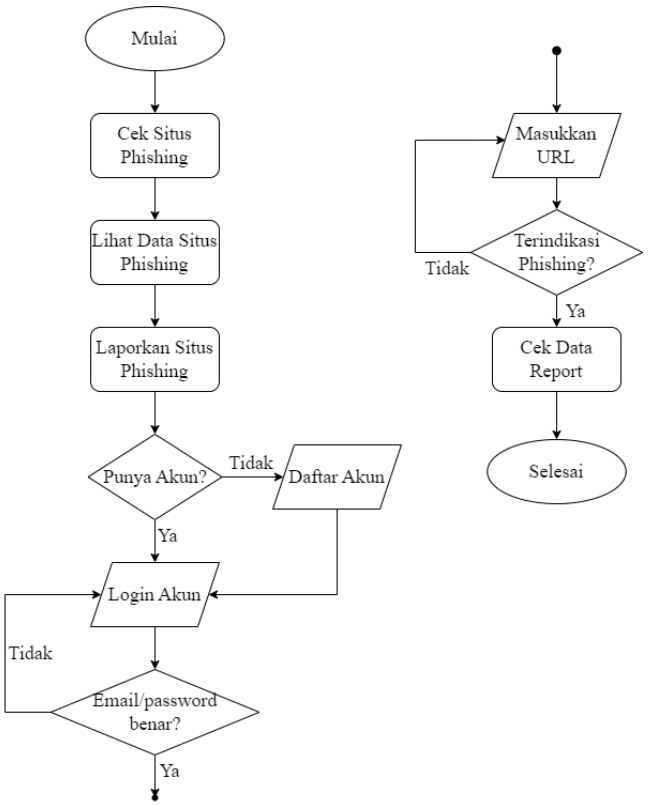
Pembuatan aplikasi *web* sebagai sistem deteksi situs *phishing* melibatkan pengembangan *frontend* dan *backend* diantaranya:

1) *Frontend:*
   - Menggunakan perpaduan bahasa markup HTML, CSS, dan *JavaScript* untuk membangun struktur, tampilan, dan interaktivitas halaman *web* yang terdiri dari laman masuk dan daftar, beranda, data *phishing*, tentang, halaman pengguna, data laporan, dan edit profil.
   - Menggunakan *framework bootstrap* untuk mempermudah pembuatan tata letak responsif dan *styling* yang konsisten di seluruh halaman *web* yang digunakan pada komponen tabel dan tombol.
   - Menggunakan *library javaScript* yaitu *jQuery* untuk membuat efek interaktif DOM digunakan untuk membuat fitur pencarian atau fitur hapus yang ditandai dalam HTML sebagai tag *scripting* dan menyembunyikan atau menampilkan elemen, seperti klik tombol.
2) *Backend:*
   - Menggunakan *flask* yang merupakan *microframework* bahasa *python* untuk mengelola rute dan logika dari aplikasi, mengelola CRUD terhadap *database* dan menampilkan halaman HTML menggunakan fungsi *render_template*.
   - Menggunakan MySQL sebagai *database* untuk menyimpan URL *phishing* yang dilaporkan dan menyimpan data pengguna menggunakan ekstensi *Flask-MySQLdb*.
   - Menggunakan *keras* dan *TensorFlow* untuk memuat *model.json* dan *model_weights.h5* yang merupakan arsitektur dan bobot model keras untuk melakukan prediksi URL juga memuat *vocabulary.json* yang berisi kamus token karakter URL.
   - Menggunakan *library python* yaitu *werkzeug* untuk *hashing password*.

b) Perancangan Sistem

Tahap ini mencakup perancangan sistem sebelum melakukan pengkodean untuk mengimplementasikan semua komponen sistem, mulai dari pengembangan *frontend* dan *backend* hingga integrasi model yang telah dibangun dengan aplikasi

web. Langkah-langkah utama yang ada dalam aplikasi *web* digambarkan pada diagram alir kerja sistem pada Gambar 3.5.



Gambar 3.5 Diagram Alir Kerja Sistem

Berdasarkan Gambar 3.5, sistem deteksi situs *phishing* berbasis aplikasi *web* menyediakan formulir cek situs untuk mengklasifikasikan URL menggunakan model LSTM. Jika hasil cek situs adalah *phishing* maka situs tersebut dapat dilaporkan agar tersimpan di sistem untuk ditampilkan di bagian informasi data *phishing*. Sebelum melaporkan situs, pengguna harus masuk akun untuk menjaga kevalidan URL yang dilaporkan. Sehingga data *phishing* yang ditampilkan pada aplikasi *web* terdiri dari dua sumber, yaitu situs *phishing* dalam data pelatihan yang digunakan dalam penelitian dan situs *phishing* laporan pengguna.

3.2.2 Spesifikasi Perangkat

Untuk melaksanakan seluruh tahapan proses penelitian yang telah ditentukan, maka pemrosesan data yang telah didapat membutuhkan instrumen perangkat. Perangkat yang digunakan dalam penelitian ini terdiri dari perangkat keras dan perangkat lunak. Perangkat disesuaikan dengan fungsionalitas untuk

mendukung proses pengembangan sistem juga lingkungan uji coba. Adapun spesifikasi perangkat keras ditunjukkan pada Tabel 3.1.

Tabel 3.1

Spesifikasi Perangkat Keras

Perangkat	**Spesifikasi**
Laptop	ASUS X412FA
Prosesor	Intel Pentium
RAM	4 GB
Penyimpanan	500 GB

Selain perangkat keras, diperlukan perangkat lunak yang spesifikasinya ditunjukkan pada Tabel 3.2.

Tabel 3.2

Spesifikasi Perangkat Lunak

Nama	**Keterangan**
Sistem Operasi	*Windows* 10
Dokumen Editor	*Microsoft Word, Microsoft Excel*
Code Editor	*Google Colab* (Pelatihan dan pengembangan model) *Visual Studio Code* (Pengembangan *web*, *scripting*, dan penulisan kode *Python*)
Web Browser	*Google Chrome* (Pengujian dan *debugging* aplikasi *web*)
Bahasa Pemrograman	*Python* (Pengembangan aplikasi *web* dan pelatihan model LSTM) *JavaScript* (Membuat elemen interaktif dan responsif pada halaman *web*) HTML dan CSS (Struktur dan konten halaman *web* juga desain dan tata letak halaman *web*)

3.2.3 Teknik Pengumpulan Data

Teknik pengumpulan data yang digunakan dalam penelitian ini adalah penggabungan data atau integrasi data. Penggabungan data melibatkan pengumpulan data dari berbagai sumber yang berbeda dan digabungkan menjadi

satu set data agar format dan struktur data menjadi lebih konsisten. Teknik ini membantu untuk melengkapi informasi data *phishing* dan sah agar model dapat menganalisis pola data yang lebih konsisten.

3.2.4 Pengujian Validitas dan Reliabilitas

a. Pengujian Validitas

Pengujian validitas yang dilakukan melibatkan pengujian bahwa model dapat mengklasifikasikan atau memprediksi data pengujian dengan benar dan memberikan hasil yang valid. Pengujian ini dilakukan untuk mengevaluasi apakah prediksi yang dihasilkan oleh algoritma LSTM sesuai dengan data sebenarnya

b. Pengujian Reliabilitas

Pengujian reliabilitas dilakukan untuk memastikan bahwa model memberikan hasil yang konsisten dan stabil ketika diuji pada data baru yang bukan merupakan bagian dari data pengujian atau data pelatihan.

3.2.5 Rancangan Analisis Data

a. Rancangan Analisis Data Deskriptif

Data yang digunakan dalam penelitian ini terdiri dari dua atribut, yaitu atribut URL dan status. Atribut status memiliki dua kelas yaitu 1 untuk URL *phishing* dan 0 untuk URL resmi atau sah. Dalam penelitian ini, URL merupakan bentuk teks string dan dianalisis berdasarkan karakter-karakter tertentu seperti huruf, angka, dan simbol yang membentuk alamat situs *web*. Berdasarkan studi literatur, pendekatan yang dilakukan untuk pengolahan URL adalah ekstraksi fitur-fitur dari URL tersebut. Dalam penelitian ini model LSTM digunakan untuk mengekstrak fitur semantik dan konteks dari struktur URL dengan mengidentifikasi representasi urutan karakter URL secara langsung pada jumlah data URL yang banyak dan pola-pola URL yang kompleks, dimana pola tersebut tidak bisa terlihat jika menggunakan ekstraksi fitur secara manual. Fitur semantik mengacu pada aspek-aspek seperti struktur URL, pola umum dalam nama domain atau direktori, serta penggunaan parameter yang mencurigakan. Pendekatan ini bersesuaian dengan pendekatan yang dilakukan oleh (Bahnsen dkk., 2017). Model LSTM memiliki kemampuan untuk mengenali dan mempelajari pola sekuensial atau pola dengan urutan terstruktur dan berkesinambungan. Pola sekuensial ini sangat penting untuk meningkatkan kinerja model dengan mengenali pola fitur yang

diekstraksi. Dengan menggunakan model LSTM, penelitian ini bertujuan untuk mendeteksi URL *phishing* secara lebih efektif dengan memanfaatkan pola sekuensial yang ada dalam struktur URL. LSTM mampu mengidentifikasi karakteristik URL *phishing* dan membedakannya dari URL yang sah berdasarkan pola-pola yang telah dipelajari. Fitur yang telah dikenali kemudian diklasifikasikan menggunakan fungsi aktivasi *sigmoid* menjadi *output* hasil prediksi.

b. Rancangan Pengujian Hipotesis

Rancangan pengujian hipotesis dalam penelitian ini dibagi menjadi tiga rancangan untuk membuktikan atau menyangkal hipotesis nol (H0) dan mendukung hipotesis alternatif (H1). Tiga rancangan pengujian hipotesis yaitu:

1. Pengujian untuk Mengetahui Proses Klasifikasi Algoritma LSTM dalam Mendeteksi Serangan Situs *Phishing*

   Rancangan pengujian pertama dilakukan untuk mengetahui cara pemrosesan algoritma LSTM dalam melakukan klasifikasi situs *phishing* dan sah melalui pengujian model dengan tahapan:

   a) Hitung Metrik Evaluasi Akurasi

      Evaluasi model dilakukan dengan menghitung nilai akurasi berdasarkan keakuratan prediksi yang dilakukan sistem (Wahyudi dkk., 2022).

   b) Pengujian Model

      Pengujian model dilakukan untuk mengevaluasi apakah prediksi yang dihasilkan oleh algoritma LSTM sesuai dengan data kelas yang sebenarnya pada data pengujian. Pengujian ini menghasilkan analisis mengenai cara kerja model dalam mendapat hasil prediksi. Untuk memahami kemampuan model dalam membedakan antara *phishing* dan sah, maka dihitung juga *precision* dan *recall* terhadap setiap kelas yang ada. Dengan membandingkan hasil prediksi model dengan label sebenarnya, jika didapat hasil uji yang menunjukkan bahwa LSTM secara signifikan dapat mendeteksi URL *phishing* atau sah dengan benar, maka H1 dapat diterima.

2. Pengujian untuk Pengembangan Aplikasi *Web* sebagai Sistem Deteksi Situs *Phishing*

   Pengujian selanjutnya merupakan pengujian sistem secara keseluruhan yang dilakukan untuk mengetahui keberhasilan fungsionalitas sistem dan

memastikan bahwa semua fitur yang ada pada aplikasi *web* berjalan dengan baik. Rancangan pengujian yang dilakukan ialah pemeriksaan bahwa setiap fitur dari aplikasi *web* berfungsi dengan optimal sebagai sistem deteksi situs *phishing*, serta mengidentifikasi dan memperbaiki kesalahan baik dari sumber eksternal maupun internal aplikasi.

Rancangan pengujian sistem menggunakan metode pengujian *black box* yang merupakan pendekatan dimana sejumlah *input* diberikan pada program untuk menguji fungsionalitasnya (Perbawa & Nurohim, 2020). Teknik yang digunakan merupakan teknik analisis *boundary value* untuk mengidentifikasi kesalahan pada dua aspek, yaitu kesalahan yang berasal dari sumber eksternal dan kesalahan yang bersumber dari internal aplikasi *web*. Jika aplikasi *web* berhasil menjalankan sebagian besar fitur dengan baik sesuai hasil yang diharapkan, maka H1 dapat diterima.

3. Pengujian untuk Penerapan dan Kinerja Model LSTM pada Aplikasi *Web*.

   Pengujian ketiga merupakan pengukuran kinerja model LSTM pada aplikasi *web* untuk memastikan bahwa model LSTM yang diintegrasikan ke dalam aplikasi berfungsi dengan baik dalam mendeteksi *phishing* pada data baru yang belum pernah diproses sebelumnya. Pada tahap ini, dilakukan juga evaluasi keberhasilan penerapan algoritma LSTM untuk memastikan bahwa saat melakukan klasifikasi situs pada aplikasi *web*, model LSTM dapat berfungsi dengan baik dan menghasilkan klasifikasi dengan benar.

   Rancangan pengujian yang dilakukan melibatkan data baru yang tidak termasuk data pelatihan maupun data pengujian. Adapun rancangan pengujiannya sebagai berikut:

   a) Mengintegrasikan model LSTM kedalam *backend server* aplikasi *web* yang menerima permintaan dari pengguna untuk memproses situs *web* baru secara langsung.
   b) Sistem akan melakukan *preprocessing* URL lalu hasilnya diproses oleh model LSTM untuk dilakukan prediksi dan mengembalikan hasil deteksi secara langsung kepada pengguna melalui antarmuka *web*. Jika aplikasi *web* menghasilkan prediksi yang akurat dalam mendeteksi situs *phishing* baru, maka H1 diterima.