BAB I
PENDAHULUAN

1.1 Latar Belakang

Seiring berkembangnya zaman, teknologi informasi mengalami perkembangan pesat terutama pada bidang finansial berbasis digital yaitu sektor aset dan investasi. Salah satu inovasi yang lahir dari perkembangan teknologi pada sektor aset dan investasi ialah *cryptocurrency* atau dalam istilah bahasa indonesia ialah mata uang kripto, yakni bentuk mata uang digital sebagai alternatif dari mata uang konvensional yang tercipta dari rangkaian teknologi *blockchain* (Dani dkk., 2023). Dengan adanya teknologi *blockchain* ini, setiap transaksi dengan mata uang kripto tidak melibatkan pihak ketiga sebagai perantaranya sehingga setiap transaksi menjadi lebih transparan (Luxmana & Oktafiyani, 2022).

Salah satu mata uang kripto yang cukup populer saat ini adalah ethereum (ETH). Ethereum sendiri dibentuk oleh Vitalik Buterin dan mulai diperdagangkan pertama kali pada 7 Agustus 2015 (Sianturi dkk., 2023). Karena kepopuleran dan kebermanfaatannya yang banyak digunakan, ethereum disebut sebagai ibu dari para koin alternatif (*Mother of Altcoin*)(Hertanto dkk., 2024).

Pertumbuhan besar-besaran pasar mata uang kripto membuat harga mata uang kripto sangat fluktuatif. Ethereum memberikan potensi keuntungan yang tinggi, tetapi dengan risiko yang tinggi pula. Volatilitas nilai mata uang ethereum yang fluktuatif tersebut membuat aktivitas investasi terbilang spekulatif dan sangat berisiko (Dani dkk., 2023). Misalnya, pada periode Januari 2019 – Desember 2024, harga ETH mengalami volatilitas ekstrem, dari kisaran $104 pada 07 Februari 2019 hingga mencapai rekor $4.812 pada 08 November 2021. Volatilitas harga ethereum yang tinggi ini menciptakan tantangan sekaligus peluang bagi investor dan pelaku pasar untuk mengoptimalkan keputusan investasi mereka.

Dalam upaya memahami dan memprediksi pergerakan harga ethereum, penggunaan teknik *machine learning*, khususnya *deep learning*, telah menunjukkan hasil yang baik dalam analisis dan prediksi data keuangan, termasuk harga aset kripto. Beberapa penelitian menggunakan model seperti *Support Vector Machine* (SVM) dan *Random Forest* diterapkan untuk memprediksi pergerakan harga saham dan *cryptocurrency*. Namun, seiring perkembangan *deep learning,* metode berbasis jaringan saraf tiruan (*Artificial Neural Network*) *seperti Long Short-Term Memory* (LSTM) dan *Gated Recurrent Unit* (GRU) mulai mendapatkan perhatian lebih.

n jenis algoritma yang digunakan dalam jaringan saraf tiruan, yang menangani pemrosesan bahasa alami dan analisis urutan data. mpuan untuk mengingat informasi dari urutan data dalam jangka ıa, sehingga cocok untuk memodelkan hubungan jangka panjang aktu (Ridwan, 2023). Sementara itu, GRU mirip dengan LSTM, ada struktur *gate* yang lebih sederhana. Terdapat pengembangan yang lebih kompleks, yaitu *Bidirectional Long Short-Term Memory*

(BiLSTM) dan *Bidirectional Gated Recurrent Unit (BiGRU).* Kedua model ini bekerja dengan cara memahami dan memproses informasi dari dua arah yaitu maju (*forward layer)* dan mundur (*backward layer*) sehingga dapat menangkap lebih banyak informasi kontekstual dalam pola data waktu (Atharsyah & Romli, 2024).

Beberapa studi telah membandingkan performa berbagai model *deep learning* dalam prediksi harga mata uang kripto, namun masih terdapat kesenjangan mengenai perbandingan performa antara BiLSTM dan BiGRU dalam konteks prediksi harga ethereum secara spesifik. Misalnya, penelitian oleh Nilasari, Sudarma, & Gunantara (2023) menunjukkan bahwa BiLSTM memiliki performa yang lebih baik dalam menangkap pola harga bitcoin dibandingkan LSTM. Sementara itu, studi oleh Atharsyah & Romli (2024) membuktikan bahwa BiGRU mampu memberikan hasil yang lebih baik dalam memprediksi harga nikel dibandingkan model LSTM, GRU, dan BiLSTM.

Berdasarkan uraian latar belakang dan penelitian sebelumnya, peneliti tertarik untuk melakukan penelitian yang berjudul **"Perbandingan Performa Algoritma *Bidirectional Long Short-Term Memory* (BiLSTM) dan *Bidirectional Gated Recurrent Unit* (BiGRU) dalam Prediksi Harga Ethereum Tahun 2019–2024"**. Penelitian ini diharapkan dapat berkontribusi dalam pengembangan model prediksi harga aset kripto yang lebih akurat, serta memberikan wawasan bagi investor dalam pengambilan keputusan investasi yang lebih baik.

1.2 Rumusan Masalah

Rumusan masalah dalam penelitian ini berfokus pada perbandingan kinerja algoritma *Bidirectional Long Short-Term Memory* (BiLSTM) dan *Bidirectional Gated Recurrent Unit* (BiGRU) dalam melakukan prediksi harga ethereum menggunakan metrik evaluasi *Root Mean Squared Error (*RMSE*)* dan *Mean Absolute Percentage Error (*MAPE*).*

1.3 Tujuan Penelitian

Penelitian ini bertujuan untuk mengidentifikasi algoritma terbaik di antara BiLSTM dan BiGRU yang lebih unggul dan memberikan hasil prediksi harga ethereum yang paling akurat sehingga bermanfaat memberikan wawasan dan rekomendasi yang berguna bagi investor, trader, dan pihak terkait lainnya dalam mengambil keputusan terkait investasi dan perdagangan ethereum.

1.4 Landasan Teori

1.4.1 Mata Uang Kripto (*Cryptocurrency*)

Cryptocurrency atau mata uang kripto merupakan sistem mata uang virtual yang berfungsi serupa dengan mata uang konvensional, memungkinkan pengguna untuk secara digital tanpa perantara pihak ketiga. Keamanan dan amin melalui teknologi *blockchain* dengan mekanisme konsensus ingga tidak bergantung pada otoritas keuangan pusat seperti bank aknya mata uang konvesional, mata uang kripto tidak memiliki diatur oleh rangkaian kode atau yang bisa disebut *blockchain* (Dani *ocurrency* juga diangap sebagai salah satu jenis investasi paling popular dan menjanjikan (Sianturi dkk., 2023).

1.4.2 Ethereum

Ethereum adalah platform *blockchain* terdesentralisasi berbasis *smart contract* untuk melakukan layanan pertukaran *online*. Ethereum merupakan salah satu aset digital menggunakan teknologi *peer-to-peer* dimana transaksi dapat dilakukan tanpa kartu kredit atau melalui bank sentral (Pradana, 2021). Mata uang ini mulai diperdagangkan pertama sekali pada 7 Agustus 2015 dengan harga US$ 2,83. Meskipun banyak orang yang melakukan investasi pada ethereum, *cryptocurrency* ini masih memiliki fluktuasi harga yang tinggi sehingga menyebabkan risiko yang cukup besar.

1.4.3 *Machine Learning*

Machine Learning adalah bagian dari kecerdasan buatan (*artificial intelligence*). *Machine Learning* mengacu pada konsep melatih mesin sedemikian rupa sehingga dapat belajar dari pengalaman masa lalu (data yang diberikan) dan menghasilkan sebuah keputusan. Konsep *Machine Learning* dapat diimplementasikan di berbagai bidang seperti internet, pendidikan, kesehatan, hiburan, robotika, perbankan, industri dan sebagainya menggunakan berbagai algoritma. *Machine learning* secara umum diklasifikasikan menjadi pembelajaran yang diawasi (*supervised learning*), dan pembelajaran tanpa pengawasan (*unsupervised learning*) (Shitole & Priyadarshini, 2021).

1.4.4 *Deep Learning*

Deep Learning merupakan salah satu implementasi dari *Machine Learning*. *Deep Learning* mulai populer digunakan sejak tahun 2006, menggunakan mekanisme *deep architecture of learning. Deep Learning* memanfaatkan jaringan saraf tiruan atau biasa disebut *Artificial Neural Network (ANN)*, disebut sebagai *deep* karena *hidden layer* (neuron) dengan jumlah yang sangat banyak (Hariz dkk., 2022). Arsitektur tersebut dapat digunakan untuk melakukan pemrosesan dengan beberapa tahap yang hasilnya dapat digunakan untuk *feature learning* (Diponegoro dkk., 2021). Deep learning memiliki karakteristik utama yang membedakannya dari metode pembelajaran mesin tradisional. Pertama, deep learning menggunakan representasi hierarkis data, yang berarti informasi diproses secara bertahap melalui beberapa lapisan abstraksi. Kedua, deep learning membutuhkan data dalam jumlah besar untuk menghasilkan model yang akurat, terutama untuk menghindari overfitting. Ketiga, pendekatan ini bergantung pada algoritma optimasi berbasis gradien, seperti backpropagation, untuk menyesuaikan bobot jaringan agar mendekati solusi optimal. Algoritma ini memungkinkan jaringan saraf tiruan mempelajari pola-pola kompleks dengan efisiensi tinggi (Luthfi & Syah, 2025).

1.4.5 Prediksi

Prediksi adalah usaha menduga atau memperkirakan sesuatu yang akan terjadi di waktu mendatang dengan memanfaatkan berbagai informasi yang relevan pada waktu-waktu) melalui suatu metode ilmiah. Tujuan dari prediksi adalah asi apa yang akan terjadi di masa datang dengan probabilitas lah satu metode prediksi kuantitatif adalah menggunakan analisis *ries)* (Wanto & Windarto, 2017). Prediksi tidak harus memberikan tentang apa yang akan terjadi, tetapi harus berusaha mencari t mungkin dengan apa yang akan terjadi (Mulyana dkk., 2022).

1.4.6 Runtun Waktu (*Time Series*)

Runtun waktu (*Time Series*) adalah serangkaian pengamatan terhadap suatu variabel yang diambil dari waktu ke waktu dan dicatat secara berurutan menurut urutan kejadiannya dengan interval waktu yang tetap. Analisis deret waktu pada dasarnya digunakan untuk melakukan analisis data yang mempertimbangkan pengaruh waktu (G. A. Putri dkk., 2017).

1.4.7 *Data Preprocessing*

Kondisi data yang kurang baik adalah salah satu kendala untuk menggunakan data secara efektif, untuk itu tahap *preprocessing* ini dibutuhkan untuk meningkatkan kualitas data. Tahap *preprocessing* dilakukan untuk menghasilkan data berkualitas atau data yang bebas dari duplikasi data, data tidak lengkap, dan data tidak konsisten, serta pengaturan bentuk data agar sesuai dengan kebutuhan model (Perwitasari dkk., 2023).

1.4.7.1 *Data Splitting*

Data Splitting berfungsi untuk membagi dataset menjadi 3 bagian, yaitu *data train, data validation,* dan *data test* dengan proporsi tertentu. Nantinya *data train* akan digunakan sebagai data untuk melatih model membuat prediksi pada *machine learning*, *data validation* untuk menguji kinerja model selama proses training, dan *data test* digunakan untuk mengevaluasi hasil *fit* model yang dibuat (Putri dkk., 2022)

1.4.7.2 *Data Normalization*

Normalisasi adalah proses penskalaan nilai pada data agar nilai data berada pada rentang tertentu. Normalisasi yang digunakan dalam penelitian ini adalah *Min-max Normalization*. Proses normalisasi ini menggunakan cara transformasi linier terhadap data aslinya (Sugiartawan dkk., 2018). Normalisasi dapat dilakukan dengan persamaan (1).

$$x_i' = \frac{x_i - x_{min}}{x_{max} - x_{min}} \qquad i = 1, 2, 3, \ldots, t \tag{1}$$

Dimana:

x_i' = Hasil normalisasi

x_i = Data ke-i

x_{min} = Data dengan nilai terkecil

x = Data dengan nilai terbesar

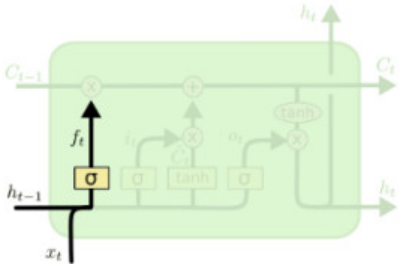
rm Memory (LSTM)

lemory (LSTM) adalah arsitektur jaringan saraf tiruan yang k mengatasi masalah *vanishing gradient* dimana model li begitu lambat atau bahkan berhenti (Atharsyah & Romli, 2024). emiliki memori jangka panjang, LSTM digunakan untuk menangani

masalah ini yang dimana LSTM memiliki memori jangka panjang yang berguna dalam memprediksi data. Sebuah LSTM memiliki tiga gerbang yaitu:

1. *Forget Gate*

   Forget gate adalah gerbang pertama yang dilalui oleh nilai input yang menggunakan fungsi aktivasi sigmoid.



Gambar 1. Arsitektur Forget Gate (Sumber: Nomidl, 2022, *https://www.nomidl.com/deep-learning/what-is-lstm-and-how-does-it-work/*)

Nilai *forget gate* dapat diperoleh menggunakan persamaan (2).

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f) \quad (2)$$

Dengan fungsi aktivasi sigmoid:

$$\sigma = \frac{1}{1+e^{-x}} \quad (3)$$

Dimana:

f_t = *forget gate*

W_f = *weight input* untuk *forget gate*

h_{t-1} = *output hidden state* t-1

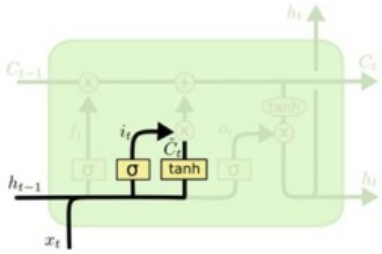
x_t = input saat t

b_f = bias untuk *forget gate*

e^{-x} = Nilai eksponsial negatid dari nilai data input

2. *Input Gate*

Pada *input gate* terjadi proses untuk menentukan apakah input baru akan diperbolehkan masuk atau tidak.



Gambar 2. Arsitektur Input Gate (Sumber: Nomidl, 2022, https://www.nomidl.com/deep-learning/what-is-lstm-and-how-does-it-work/)

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i) \quad (4)$$

i_t = *input gate*

W_i = *weight input* untuk *input gate*

h_{t-1} = *output hidden state* t-1

x_t = *input* saat t

b_i = bias *input gate*

Pada *input gate* juga terdapat proses ke dua, yaitu menghitung nilai kandidat *cell state*. Nilai kandidat *cell state* diperoleh menggunakan fungsi aktivasi tanh seperti terlihat pada persamaan (5) (Cahyani dkk., 2023).

$$\widetilde{C_t} = \tanh\ (W_c[h_{t-1}, x_t] + b_c) \quad (5)$$

Dengan fungsi aktivasi tangen hiperbolik:

$$tanh = \frac{e^x - e^{-x}}{e^x + e^{-x}} \quad (6)$$

Dimana:

$\widetilde{C_t}$ = *cell state candidate*

W_c = *weight input cell state*

put hidden state t-1

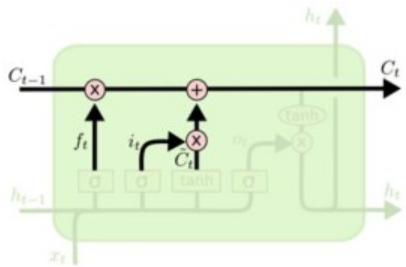
ut saat t

s *cell state*

i eksponsial positif dari nilai data input

e^{-x} = nilai eksponsial negatif dari nilai data input

Cell state digunakan sebagai memori dalam LSTM dan menampung data dari LSTM sebelumnya lalu membawa data untuk LSTM selanjutnya. *Cell State* ini menjadi bagian penting dalam LSTM karena dapat mengingat data dari proses sebelumnya (Erlangga dkk., 2023).



Gambar 3 Arsitektur Cell State (Sumber: Nomidl, 2022, https://www.nomidl.com/deep-learning/what-is-lstm-and-how-does-it-work/)

$$C_t = f_t \odot C_{t-1} + i_t \odot \widetilde{C}_t \quad (7)$$

Dimana:

C_t = cell state t

f_t = forget gate

C_{t-1} = cell state t-1

i_t = input gate

$\widetilde{C}_t$ = cell state candidate

Produk Hadamard ($\odot$) adalah operasi biner antara dua vektor (atau matriks) yang memiliki dimensi identik. Operasi ini menghasilkan sebuah vektor baru dengan dimensi yang sama, di mana setiap elemennya merupakan hasil perkalian dari elemen-elemen yang saling bersesuaian pada kedua vektor awal. Misalkan kita memiliki dua buah vektor f_t dan C_{t-1} yang masing-masing memiliki n elemen sebagai berikut:

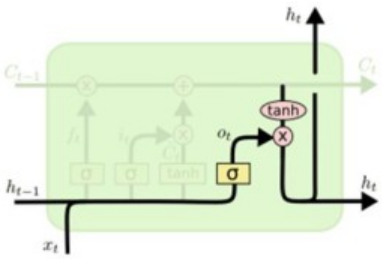
$$f_t = \begin{bmatrix} f_{t_1} \\ f_{t_2} \\ \vdots \\ f_{t_n} \end{bmatrix} \quad \text{dan} \quad C_{t-1} = \begin{bmatrix} C_{t-1_1} \\ C_{t-1_2} \\ \vdots \\ C_{t-1_n} \end{bmatrix}$$

‹ Hadamard dari f_t dan C_{t-1} akan menghasilkan vektor baru:

$$f_t \odot C_{t-1} = \begin{bmatrix} f_{t_1} & \times & C_{t-1_1} \\ f_{t_2} & \times & C_{t-1_2} \\ \vdots & \vdots & \vdots \\ f_{t_n} & \times & C_{t-1_2} \end{bmatrix}$$

3. *Output Gate*

Pada *g*ate ini nilai output digunakan untuk menghasilkan nilai *hidden state* baru bersama dengan nilai *cell state (Cahyani dkk., 2023)*. Output gate diperoleh menggunakan persamaan (8).



Gambar 4 Arsitektur Output Gate (Sumber: Nomidl, 2022, https://www.nomidl.com/deep-learning/what-is-lstm-and-how-does-it-work/)

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \quad (8)$$

Dimana:

o_t = *output gate*

W_o = *weight output gate*

h_{t-1} = *output hidden state* dari t-1

x_t = *input* saat t

b_o = bias *output gate*

Nilai *hidden state* terbaru diperoleh dengan mengalikan nilai output gate dan cell state yang telah melalui proses fungsi tanh, seperti terlihat pada persamaan (9).

$$h_t = o_t \odot \tanh\ (C_t) \quad (9)$$

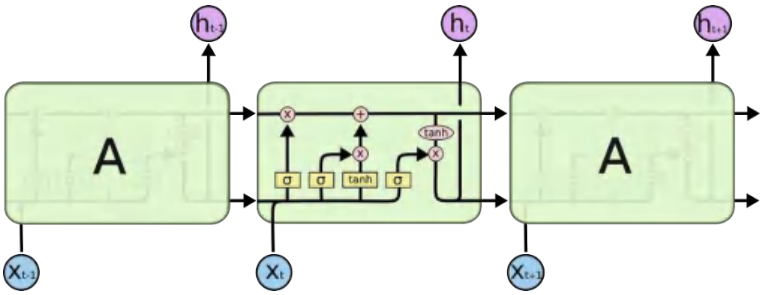
Dimana:

put perhitungan ke t

put gate

l state t

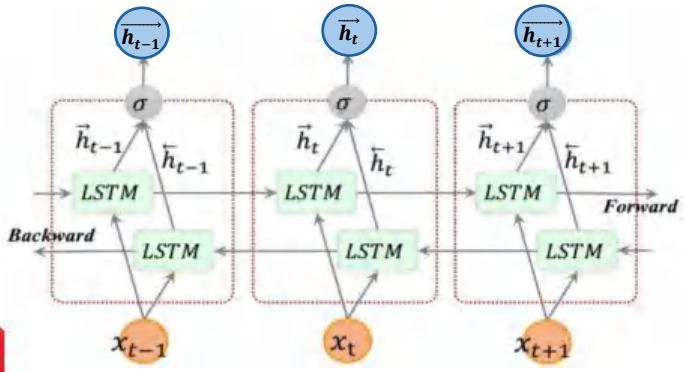
Ketiga *gate* ini bekerja karena fungsi aktivasi yang menentukan perilaku apa yang akan dilakukan kepada data-data yang ada pada setiap layernya. Pada LSTM akan digunakan fungsi aktivasi sigmoid yang mengeluarkan nilai 0 - 1, karena membutuhkan nilai positif dari setiap gerbang untuk mengetahui apakah fitur tersebut harus disimpan atau dihapus (Moghar & Hamiche, 2020). Jika nilai mendekati 0 mengartikan bahwa gerbang memblokir fitur untuk lewat dan jika nilai mendekati 1 maka gerbang mengizinkan fitur untuk lewat. Arsitektur Lengkap LSTM dapat dilihat pada Gambar 5:



Gambar 5 Arsitektur lengkap LSTM (Sumber: Medium, 2019, *https://medium.com/coinmonks/character-to-character-rnn-with-pytorchs-lstmcell-cd923a6d0e72*)

1.4.8.1 *Bidirectional Long Short-Term Memory* (BiLSTM)

Pada *Bidirectional Long Short Term Memory* (BiLSTM) terdapat dua jaringan LSTM dimana jaringan LSTM pertama berfungsi dalam memproses urutan masukan data ke arah depan (*forward*) dan jaringan LSTM kedua berfungsi dalam memproses urutan data dari arah sebaliknya (*backward*). Kemudian *output* dari jaringan LSTM *forward* dan *backward* digabungkan pada setiap urutan waktu. Dengan adanya dua layer yang berlawanan arah tersebut, model dapat mempelajari informasi masa lalu dan informasi masa mendatang untuk setiap *sequence input* (Karyadi & Santoso, 2022). Nilai BiLSTM dapat diperoleh dengan persamaan (10).



Gambar 6 Arsitektur BiLSTM (Puteri, 2023)

$$h_t\ BiLSTM = [\overrightarrow{h_t}, \overleftarrow{h_t}] \quad (10)$$

Dimana:

$\overrightarrow{h_t}$ = *Forward*

$\overleftarrow{h_t}$ = *Backward*

$h_t\ BiLSTM$ = *output* hidden state BiLSTM (penggabungan)

x_t = *input* saat t

h_{t-1} = *output* hidden state ke t-1

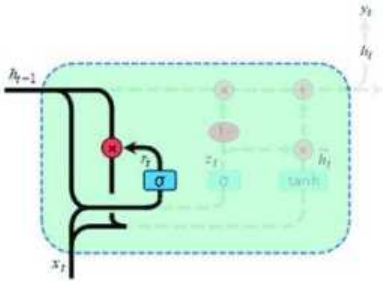
h_{t+1} = *output* hidden state ke t+1

1.4.9 *Gated Recurrent Unit (*GRU*)*

Gated Recurrent Unit pertama kali diperkenalkan oleh Kyunghyun Cho pada tahun 2014. *Gated Recurrent Unit* ini merupakan turunan dari arsitektur *Recurrent Neural Network* (RNN) di mana GRU serupa dengan LSTM karena GRU juga menggunakan sistem *gate*, namun asrsitektur GRU lebih sederhana daripada LSTM (Chung dkk., 2014). Perbedaan utama GRU dengan LSTM adalah bahwa satu unit gerbang secara bersamaan mengontrol faktor lupa dan keputusan untuk memperbarui unit dalam *hidden state*. Dengan kata lain GRU menggabungkan proses untuk menentukan faktor unit yang ingin dibuang dan unit mana yang ingin diperbarui ke dalam satu *gate* yang disebut *update gate* (Ripto & Heryanto, t.t.).

Arsitektur GRU tidak menggunakan *cell state*, tetapi memanfaatkan *hidden state* untuk menyimpan informasi (Prayogi dkk., 2024). Di dalam struktur GRU, terdapat komponen yang disebut *gate* berfungsi untuk mengatur alur informasi model GRU (Ripto & Heryanto). GRU mempunyai dua *gate*, yaitu:

1. *Reset Gate*
   Reset gate pada GRU menggabungkan input baru dengan informasi masa lalu.



Gambar 7 Arsitektur Reserch Gate GRU (Lin dkk., 2022)

$$r_t = \sigma(W_r x_t + U_r h_{t-1} + b_r) \quad (11)$$

Dimana:

r_t = reset gate
W_r = weight input reset gate
x_t = data input saat t
U_r = weight hidden state sebelumnya untuk reset gate
h_{t-1} = hidden state dari t-1
b_r = nilai bias pada reset gate

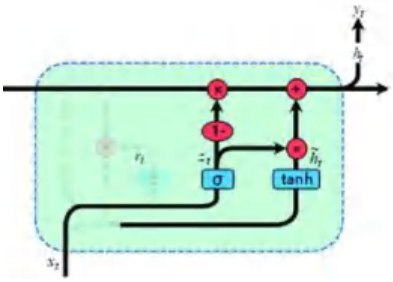
Setelah *reset gate*, komputasi dilanjutkan ke *hidden state* dengan rumus sebagai berikut:

$$\widetilde{h_t} = \tanh\left(W_h x_t + r_t \times U_h h_{t-1} + b_h\right) \quad (12)$$

Dimana:

$\widetilde{h_t}$ = kandidat hidden state
r_t = reset gate
W_h = weight input
x_t = data input saat t
U_h = weight hidden state sebelumnya
h_{t-1} = hidden state dari t-1
b_h = nilai bias

2. *Update Gate*
Update gate menentukan berapa banyak informasi masa lalu yang harus tetap disimpan.



Gambar 8 Arsitektur Update Gate GRU (Lin dkk., 2022)

$$z_t = \sigma(W_z x_t + U_z h_{t-1} + b_z) \quad (13)$$

Dimana:

z_t = update gate
x_t = data input saat t
W_z = weight input
h_{t-1} = *hidden state* dari t-1
U_z = *weight hidden state sebelumnya*
b_z = nilai bias pada *update gate*

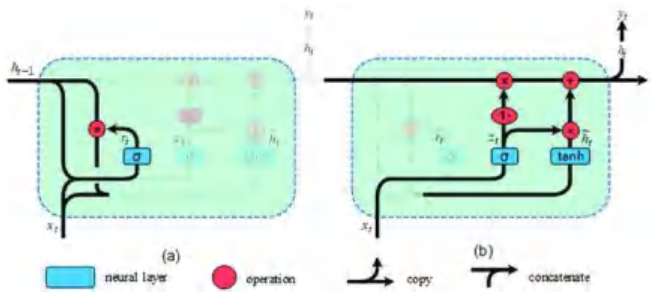
Setelah itu, *hidden state final* dihitung dengan persamaan (14).

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t \quad (14)$$

Dimana:

h_t = output
$\tilde{h}_t$ = kandidat hidden state
z_t = output pada update gate
h_{t-1} = hidden state dari t-1

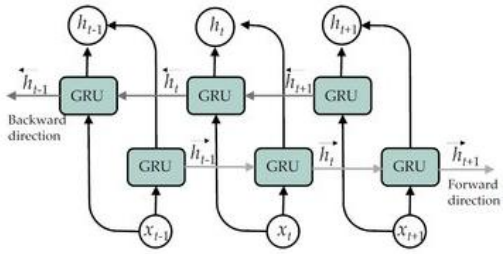
Sama seperti LSTM, pada GRU juga digunakan fungsi aktivasi sigmoid untuk menentukan apakah fitur tersebut harus disimpan atau dihapus (Ripto & Heryanto, t.t.). Arsitektur lengkap GRU dapat dilihat pada Gambar 9:



Gambar 9. Arsitektur Lengkap GRU

1.4.9.1 *Bidirectional Gated Recurrent Unit* (BiGRU)

*Bidirectional Gated Recurrent Unit (*BiGRU) adalah desain RNN khusus yang dirancang untuk menangani data berurutan dengan menangkap pola kompleks dalam urutan maju terdiri dari dua lapisan GRU dengan model GRU sebagai fondasi mengamati urutan input dalam lintasan maju, dan lapisan lainnya a mundur (Atharsyah & Romli, 2024). Arsitektur BiGRU dapat dilihat



Gambar 10 Arsitektur Lengkap BiGRU (Mekruksavanich & Jitpattanakul, 2023)

$$h_t\ BiGRU = [\overrightarrow{h_t}, \overleftarrow{h_t}] \quad (15)$$

Dimana:

$\overrightarrow{h_t}$ = *Forward*

$\overleftarrow{h_t}$ = *Backward*

$h_t\ BiGRU$ = *output* hidden state BiGRU (penggabungan)

x_t = *input* saat t

h_{t-1} = *output* hidden state ke t-1

h_{t+1} = *output* hidden state ke t+1

1.4.10 *Hyperparameter Tuning*

Hyperparameter adalah properti dari algoritma pembelajaran yang biasanya memiliki nilai numerik yang mempengaruhi cara kerja algoritma. Nilai-nilai itu tidak dipelajari oleh algoritma itu sendiri dari data, tetapi harus ditetapkan sebelum menjalankan algoritma. Penyetelan *hyperparameter* berfokus untuk memastikan bahwa model tidak mengalami *underfitting* maupun *overfitting* terhadap dataset pelatihan, sambil mempelajari struktur data secepat mungkin (Hariz dkk., 2022). Dalam penelitian kali ini *hyperparameter* yang digunakan adalah *Hyperband.*

1.4.10.1 *Hyperband*

Hyperband adalah algoritma optimasi *hyperparameter* yang menggunakan strategi alokasi sumber daya adaptif untuk meningkatkan efisiensi pencarian model terbaik
ing dan *deep learning*. Metode ini merupakan pengembangan dari
e Halving, yang secara bertahap mengeliminasi konfigurasi
g kurang optimal dan memberikan lebih banyak sumber daya
ang memiliki performa lebih baik (Wang dkk., 2018).

1.4.11 *Data Denormalization*

Denormalisasi dilakukan untuk mengembalikan data menjadi data aktual dari data yang telah di normalisasi sebelumnya. Nilai prediksi yang didapat dari perhitungan dilakukan proses denormalisasi terlebih dahulu sebelum menghitung akurasi hasil prediksi agar sesuai dengan ukuran data asli (Yunizar dkk., 2023). Perhitungan proses denormalisasi dapat dilihat pada persamaan (16).

$$x_i = x_i' \times (x_{max} - x_{min}) + x_{min} \quad (16)$$

Dimana:

x_i = Data ke-i

x_i' = Hasil normalisasi

x_{min} = Data dengan nilai terkecil

x_{max} = Data dengan nilai terbesar

1.4.12 Evaluasi

Metrik evaluasi digunakan untuk mengukur kinerja dari prediksi model. Pada model time-series, evaluasi dapat digunakan dengan menghitung kesalahan yang dibuat oleh model. Ada beberapa metrik yang digunakan dalam evaluasi model penelitian kali ini, diantaranya adalah *Root Mean Squared Error (RMSE), Mean Square Error (MSE), dan Mean Percentage Absolute Error (MAPE)* (Meriani & Rahmatulloh, 2024).

1.4.12.1 *Root Mean Squarred Error* (RMSE)

RMSE menghitung selisih nilai yang diprediksi dan nilai yang sebenarnya terjadi oleh suatu model, dimana semakin kecil nilai RMSE maka hasil prediksi dianggap semakin akurat (Karnisih dkk., 2025). Nilai evaluasi RMSE dapat diperoleh dengan persamaan (17).

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}{n}} \quad (17)$$

Dimana:

$\hat{y}_i$ = nilai prediksi pada data ke-i

y_i = nilai aktual pada data ke-i

1.4.12.2 *Mean Absolute Percentage Error* (MAPE)

Mean Absolute Percentage Error (MAPE) merupakan nilai absolut dari persentase *error* data terhadap nilai rata-rata. Pada MAPE diukur seberapa besar selisih nilai model peramalan dibandingkan dengan nilai yang sebenarnya (Yunizar dkk., 2023). Nilai evaluasi MAPE dapat diperoleh dengan persamaan (18).

$$MAPE = \frac{\sum_{i=1}^{n} \frac{(\hat{y}_i - y_i)}{y_i}}{n} \times 100 \quad (18)$$

Dimana:

$\hat{y}_i$ = nilai prediksi pada data ke-i
y_i = nilai aktual pada data ke-i
n = jumlah data


PDF
Optimized using
trial version
www.balesio.com

BAB II
METODOLOGI PENELITIAN

2.1 Pendekatan dan Tahapan Penelitian

Pada penelitian ini, digunakan pendekatan kuantitatif dengan metode yang digunakan yaitu metode *Bidirectional Long Short-Term Memory (BiLSTM)* dan model *Bidirectional Gated Recurrent Unit (BiGRU).* Dalam proses pengolahan data, penulis menggunakan bantuan platform *Google Colab*. Adapun tahapan pengolahan data dalam penelitian ini adalah sebagai berikut:

1. Melakukan pengumpulan data dengan metode *scrapping.*
2. Melakukan data understanding untuk memperoleh informasi dari data.
3. Memeriksa missing value.
4. Melakukan pembagian data train dan data test.
5. Melakukan normalisasi data.
6. Menentukan timespteps pada data.
7. Menentukan parameter untuk *hyperparameter tuning*.
8. Membangun model BiLSTM.
9. Melatih model BiLSTM pada data train.
10. Melakukan prediksi data test dengan model BiLSTM.
11. Mengevaluasi model BiLSTM dengan metrik RMSE dan MAPE.
12. Membangun model BiGRU.
13. Melatih model BiGRU pada data train.
14. Melakukan prediksi data test dengan model BiGRU.
15. Mengevaluasi model BiGRU dengan metrik RMSE dan MAPE.
16. Membandingkan hasil evaluasi model BiLSTM dan model BiGRU.

2.2 Alur Kerja Penelitian