BAB I
PENDAHULUAN

1.1 Latar Belakang

Konsumsi listrik secara global mencerminkan kebutuhan energi listrik yang terus meningkat seiring dengan pertumbuhan populasi, ekonomi, serta kemajuan teknologi di berbagai negara. Peningkatan ini tampak nyata pada berbagai sektor kehidupan, mulai dari meningkatnya jumlah penduduk, berkembangnya industri manufaktur, hingga meningkatnya kebutuhan penerangan fasilitas umum, seperti jalan raya, lampu hias di taman kota dan lain sebagainya (Hasibuan & Siregar, 2019). Berdasarkan data terbaru PT PLN (Persero), penjualan listrik nasional pada paruh pertama tahun 2025 mencapai 155,62 TWh, tumbuh 4,36% *year-on-year* (YoY) dibandingkan periode pada tahun sebelumnya. Dengan 67,14 TWh, atau sekitar 43,14% dari total penjualan listrik nasional, sektor rumah tangga merupakan kontributor terbesar. Fakta ini menunjukkan betapa pentingnya listrik bagi kehidupan sehari-hari dan pertumbuhan ekonomi nasional (PLN, 2025).

Di tingkat regional, Kota Makassar juga menghadapi fenomena serupa. Berdasarkan data terbaru BPS Kota Makassar (2025), jumlah pelanggan listrik pada tahun 2024 mengalami peningkatan, dengan total penjualan listrik tercatat sebesar 4.260.341,29 MWh dan jumlah pelanggan sebanyak 1.122.306. Pertumbuhan jumlah pelanggan di sektor rumah tangga, bisnis, dan industri turut mendorong lonjakan kebutuhan listrik yang semakin kompleks. Namun, hingga kini pola konsumsi listrik di Makassar umumnya dianalisis secara agregat, tanpa memperhatikan variasi sektor pelanggan atau perbedaan antar wilayah. Kondisi ini berpotensi menyulitkan PLN dalam melakukan perencanaan distribusi dan pengendalian beban puncak secara optimal.

Dalam konteks prediksi konsumsi listrik, metode konvensional seperti *Autoregressive Integrated Moving Average* (ARIMA) atau regresi linear masih banyak digunakan dalam prediksi konsumsi listrik. Model ARIMA sesuai untuk prediksi jangka pendek pada data yang bersifat stasioner, namun kurang optimal ketika diterapkan pada pola jangka panjang atau dengan data yang berkarakteristik nonlinear. Sedangkan metode regresi linear mampu memberikan hasil prediksi yang cukup akurat, tetapi memiliki keterbatasan pada fluktuasi data nonlinear (Akhmad Faeda Insani et al., 2025). Sementara itu, metode yang didasarkan pada *machine learning* atau pembelajaran mesin merupakan pendekatan yang menggunakan algoritma untuk mengajarkan komputer dalam menyelesaikan suatu permasalahan berdasarkan data yang diberikan. Pendekatan ini mampu menghasilkan model adaptif dalam melakukan prediksi, klasifikasi, pengelompokan data, serta memecahkan masalah-masalah linear maupun nonlinear. Metode *Artificial Neural Network* (ANN) dan *Support Vector Machine* (SVM) merupakan pendekatan *machine learning* yang biasa digunakan untuk memprediksi data konsumsi energi listrik. Namun, meskipun kedua model tersebut dirancang untuk memproses data berurutan, keduanya masih memiliki keterbatasan dalam menangkap ketergantungan jangka panjang serta pola musiman pada *data time series* (Divina et al., 2018).

Untuk mengatasi keterbatasan metode sebelumnya, penelitian ini menggunakan *Long Short-Term Memory* (LSTM) dan *Gated Recurrent Unit* (GRU) yang mampu mengolah data *time series* dengan baik. LSTM merupakan pengembangan dari arsitektur *Recurrent Neural Network* (RNN) yang dilengkapi dengan *memory cell* untuk menyimpan informasi jangka panjang sehingga mampu mengatasi masalah *vanishing gradient* yang sering muncul pada RNN ketika memproses data sekuensial yang panjang (Pontoh et al., 2022). Sedangkan GRU merupakan proses komputasi yang lebih sederhana daripada LSTM, akan tetapi mempunyai akurasi yang setara dan cukup efektif dalam hal mengurangi permasalahan *gradient* yang hilang (Karyadi, 2022).

Beberapa penelitian sebelumnya yang berkaitan dengan metode LSTM pernah dilakukan oleh (Fikriaziz et al., 2024) untuk memprediksi konsumsi listrik di Kabupaten Kebumen pada tahun 2023. Hasil penelitian ini menunjukkan bahwa LSTM mampu memberikan tingkat ketepatan prediksi yang baik, dibuktikan dengan nilai *Mean Absolute Percentage Error* (MAPE) sebesar 4,07% yang menandakan bahwa hasil prediksi LSTM dinilai efektif dalam memprediksi konsumsi energi listrik di Kabupaten Kebumen. Penelitian lainnya dilakukan oleh (Septiawan & Fauzi, 2025) yang menganalisis metode LSTM dan SVR dalam peramalan penjualan energi listrik. Hasil dari penelitian ini menunjukkan bahwa metode LSTM secara konsisten memberikan nilai ketepatan prediksi yang lebih baik dibandingkan SVR dengan nilai RMSE sebesar 0,29, MSE sebesar 0,08, dan MAPE sebesar 0,77% yang menandakan kemampuan LSTM dalam menangkap pola data yang kompleks dan dinamis. Selanjutnya, penelitian yang dilakukan oleh (Abumohsen et al., 2023) untuk memprediksi beban listrik menggunakan metode *deep learning* seperti LSTM, GRU, dan RNN. Dari ketiga model yang diuji, GRU menghasilkan performa terbaik dengan nilai *R-Squared* sebesar 90,23%, MSE 0,00215, dan MAE sebesar 0,03266 yang menandakan tingkat kesalahan prediksi rendah dan kemampuan model yang baik dalam mempelajari pola konsumsi listrik.

Meskipun penelitian sebelumnya menunjukkan bahwa LSTM mampu memberikan ketepatan prediksi yang baik dalam memprediksi konsumsi listrik, sedangkan GRU juga terbukti memiliki ketepatan prediksi yang kompetitif dengan arsitektur yang lebih sederhana dan efisien. Namun, penelitian-penelitian tersebut umumnya masih terbatas pada konteks luar negeri ataupun wilayah tertentu. Hanya sedikit penelitian yang secara khusus membandingkan kinerja LSTM dan GRU dalam memprediksi konsumsi listrik di Indonesia, khususnya Kota Makassar sebagai lokasi studi kasus penelitian ini. Pemilihan Kota Makassar didasarkan pada ketersediaan dan kelengkapan data konsumsi listrik, sehingga memungkinkan dilakukan analisis dan perbandingan kinerja model LSTM dan GRU secara menyeluruh.

Berdasarkan uraian yang telah dijelaskan, maka peneliti memutuskan untuk melakukan penelitian dengan membandingkan kinerja metode LSTM dan GRU dalam memprediksi konsumsi listrik di Kota Makassar. Melalui perbandingan kedua metode *deep learning* tersebut, penelitian ini diharapkan dapat memberikan kontribusi dalam meningkatkan ketepatan prediksi dan mendukung perencanaan serta pengelolaan energi listrik yang lebih efisien bagi PLN maupun pemerintah. Selain itu, penelitian ini juga mengimplementasikan hasil prediksi ke dalam *dashboard* sistem informasi berbasis *Streamlit,* yang dikembangkan untuk menyajikan visualisasi dan hasil prediksi konsumsi listrik secara interaktif dan informatif.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan sebelumnya, rumusan masalah yang akan dibahas di antaranya:

1. Bagaimana membangun dan melatih model prediksi konsumsi listrik menggunakan metode LSTM dan GRU berdasarkan data historis konsumsi listrik di Kota Makassar?
2. Metode mana yang memiliki kinerja prediksi lebih baik antara LSTM dan GRU dalam memprediksi konsumsi listrik di Kota Makassar?
3. Bagaimana merancang dan mengimplementasikan hasil prediksi serta evaluasi model ke dalam *dashboard* sistem informasi berbasis *Streamlit* yang interaktif dan informatif?

1.3 Batasan Masalah

Untuk menjaga fokus penelitian dan menghindari perluasan masalah, maka ditetapkan beberapa batasan sebagai berikut:

1. Penelitian ini hanya menggunakan algoritma *Recurrent Neural Network* (RNN), yaitu LSTM dan GRU sebagai model utama untuk prediksi.
2. Data yang digunakan dalam penelitian ini merupakan data historis konsumsi listrik pelanggan PLN di wilayah Kota Makassar, yang mencakup dua Unit Pelaksana Pelayanan Pelanggan (UP3), yaitu UP3 Makassar Selatan dan UP3 Makassar Utara, dengan rentang waktu Januari 2019 hingga Agustus 2024.
3. Penelitian hanya berfokus pada pemodelan dan evaluasi prediksi konsumsi listrik tanpa membahas aspek eksternal seperti kondisi ekonomi, cuaca, atau kebijakan energi.
4. Metode pengukuran hasil prediksi menggunakan *Mean Absolute Percentage Error* (MAPE).
5. *Dashboard* sistem informasi dibangun menggunakan *framework Streamlit*, dan hanya digunakan untuk menampilkan hasil prediksi dan evaluasi model dalam bentuk visualisasi.

1.4 Tujuan Penelitian

Berdasarkan rumusan masalah yang telah diuraikan, tujuan penelitian sebagai berikut:

1. Membangun dan melatih model prediksi konsumsi listrik menggunakan metode LSTM dan GRU berdasarkan data historis konsumsi listrik di Kota Makassar.
2. Melakukan evaluasi kinerja model LSTM dan GRU menggunakan metrik MAPE untuk menentukan model yang paling optimal dalam menghasilkan prediksi.
3. Mengembangkan *dashboard* sistem informasi berbasis *Streamlit* untuk menyajikan hasil prediksi dan evaluasi model secara interaktif dan informatif.

1.5 Manfaat Penelitian

Penelitian ini diharapkan memiliki manfaat sebagai berikut:

1. Memberikan kontribusi ilmiah terkait implementasi dan evaluasi *model deep learning time series* dalam bidang prediksi konsumsi energi.
2. Memberikan pemahaman yang lebih mendalam mengenai pola konsumsi listrik di Kota Makassar sehingga dapat menjadi dasar dalam upaya efisiensi energi serta perencanaan dan pengelolaan sumber daya listrik di berbagai sektor.
3. Menghasilkan *dashboard* sistem informasi prediktif berbasis *Streamlit* yang dapat digunakan sebagai media visualisasi interaktif untuk membantu peneliti, akademisi, maupun instansi terkait dalam menganalisis dan memanfaatkan hasil prediksi konsumsi listrik.

1.6 Landasan Teori

1.6.1 Konsumsi Energi Listrik

Energi listrik merupakan salah satu sumber utama dalam kehidupan manusia seiring dengan perkembangan ilmu pengetahuan dan teknologi. Energi ini banyak digunakan dalam kehidupan sehari-hari seperti penerangan dan peralatan rumah tangga, hingga keperluan industri, perkantoran, transportasi, dan layanan publik (Melinda & Longgom Nasution, 2023). Selain itu, ketersediaan dan efisiensi penggunaan energi listrik juga menjadi indikator kemajuan suatu daerah, karena semakin tinggi tingkat konsumsi listrik biasanya mencerminkan peningkatan kegiatan ekonomi, pembangunan infrastruktur, dan kualitas hidup masyarakat.

Konsumsi energi listrik adalah jumlah listrik yang digunakan konsumen dalam kilowatt jam (kWh), yang mencerminkan kebutuhan energi pada rumah tangga, industri, bisnis, dan lainnya (Akhmad Faeda Insani et al., 2025). Perhitungan konsumsi energi listrik dalam kWh adalah sebagai berikut.

$$\text{Energi (kWh)} = \text{Daya (kW)} \times \text{Waktu (jam)} \quad (1)$$

Keterangan:
Daya (kW) = besarnya daya listrik yang digunakan
Waktu (jam) = lamanya waktu listrik digunakan

1.6.2 Data *Time Series*

Data *time series* merupakan serangkaian pengamatan yang terurut berdasarkan waktu dengan jarak yang sama. Data ini sering ditemui dalam kehidupan sehari-hari karena data tersebut dikumpulkan melalui waktu interval, yaitu harian, bulanan, maupun tahunan (Mahfud Al et al., 2020). Selama data tersebut disimpan berdasarkan urutan waktu, data tersebut termasuk dalam kategori data *time series.* Dengan demikian, dapat dikatakan bahwa data *time series* memiliki karakteristik yang berbeda dibandingkan dengan jenis data lainnya, sehingga metode analisis yang digunakan untuk mengolah dan memprediksi data ini juga memerlukan pendekatan yang berbeda.

Analisis data *time series* adalah proses memahami data dengan cara melakukan analisis terhadap suatu data berdasarkan urutan waktu dari sebuah pengamatan yang memiliki nilai interval konsisten. Melalui tahap analisis tersebut, pola perubahan pada data akan teridentifikasi. Pola analisis *time series* dapat dikategorikan menjadi tiga, yaitu pola tren yang menunjukkan kecenderungan naik atau turun dalam jangka panjang, pola siklus (*cycle*) yang menggambarkan fluktuasi yang berulang dalam periode tertentu, serta pola musiman yang muncul secara teratur pada periode waktu yang spesifik seperti bulanan atau tahunan (Nur Cahyo & Susanti, 2023).

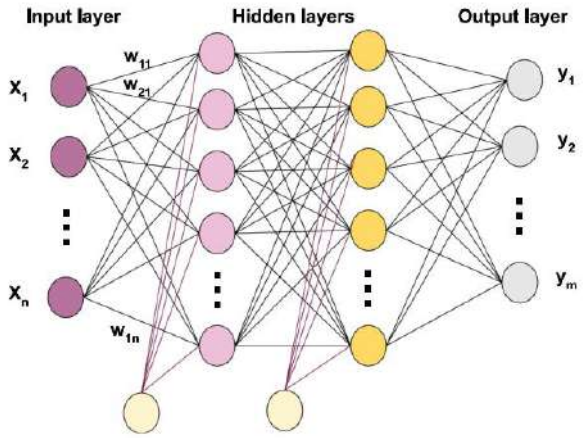
1.6.3 *Deep Learning*

Machine learning adalah salah satu cabang dari *artificial intelligence* yang saat ini banyak digunakan dalam sebuah penelitian karena kemampuannya yang kompleks dan juga fleksibel. Teknologi ini berkembang pesat serta menjadi inovasi penting di berbagai sektor, seperti kesehatan, keuangan, keamanan, pengelolaan data, serta analisis tren dan juga prediksi (Pandey et al., 2019). Secara prinsip, *machine learning* memungkinkan sistem untuk belajar serta mengambil keputusan sendiri tanpa instruksi pemrograman langsung. Hal ini menjadikan mesin tidak hanya mampu berperilaku mengambil keputusan, namun juga dapat beradaptasi dengan perubahan yang terjadi.

Deep Learning merupakan salah satu bidang dari *machine learning* yang memanfaatkan jaringan saraf tiruan (*neural network*) untuk implementasi dataset yang besar. Dalam algoritma *deep learning*, data akan melalui beberapa lapisan, di mana setiap lapisan secara bertahap mengekstraksi fitur serta mengirimkan informasi ke lapisan berikutnya. Lapisan awal berfungsi untuk mengekstraksi karakteristik tingkat rendah, yang kemudian digabungkan oleh lapisan-lapisan selanjutnya untuk membentuk representasi komprehensif (Perumal et al., 2024).

1.6.4 *Multilayer Perceptron*

Multilayer Perceptron (MLP) adalah salah satu pemodelan dalam teknologi jaringan saraf tiruan dengan karakteristik memiliki nilai bobot yang lebih baik daripada pemodelan yang lain. MLP dapat digunakan tanpa pengetahuan sebelumnya dan algoritma dari MLP juga dapat diimplementasikan dengan mudah serta mampu menyelesaikan masalah linear dan nonlinear (Prasetya Wibawa et al., 2020). MLP merupakan pengembangan dari *perceptron* tunggal sehingga memiliki beberapa lapisan maupun *hidden layer*, yang terletak diantara ruang *input* dan *output layer.*



Gambar 1. Visualisasi *Multilayer Perceptron* (MLP) (Chan et al., 2023)

Gambar 1 menampilkan visualisasi MLP yang terdiri atas tiga unit yang dapat dibedakan, di antaranya lapisan masukan (*input layer*), lapisan tersembunyi (*hidden layer*), dan lapisan luaran (*output layer*) (Resha & Syamsu, 2022).

1. Lapisan masukan (*input layer*) terdiri dari *neuron* yang menerima data masukan dari variabel *input.* Semua *neuron* pada lapisan ini dapat terhubung ke *neuron* pada *hidden layer* atau langsung ke *output layer* jika jaringan tidak menggunakan lapisan tersembunyi.
2. Lapisan tersembunyi *(hidden layer)* terdiri dari *neuron* yang menerima data dari *input layer.*
3. Lapisan luaran (*output layer*) terdiri dari *neuron* yang menerima data dari *hidden layer* atau langsung dari *input layer.*

1.6.4.1 Iterasi *Training* Data/*Epoch*

Epoch merupakan *hyperparameter* dalam *deep learning* yang menentukan berapa kali seluruh dataset dilalui atau diproses oleh algoritma pembelajaran. Satu *epoch* berarti setiap sampel dalam dataset telah digunakan sekali untuk memperbarui parameter internal atau *weight* (bobot) pada jaringan saraf. Dalam satu *epoch*, data pelatihan biasanya dibagi menjadi beberapa *batch*, di mana setiap *batch* berisi sebagian kecil dari dataset untuk membantu mempercepat proses pelatihan, menghemat penggunaan memori, serta menjaga stabilitas pembaruan bobot (Dewi & Riwurohi, 2024).

Seiring bertambahnya jumlah *epoch*, model akan semakin sering menyesuaikan bobotnya berdasarkan kesalahan (*error*) yang ditemukan, sehingga tingkat akurasi model dapat meningkat. Namun, penggunaan *epoch* yang terlalu banyak juga dapat menyebabkan *overfitting*, yaitu suatu kondisi di mana model terlalu pas dengan data pelatihan dan kehilangan kemampuannya untuk digeneralisasi ke data baru. Oleh karena itu, jumlah *epoch* yang ideal harus dipilih sesuai dengan fitur dan kompleksitas dataset.

1.6.4.2 *Batch Size*

Batch size merupakan jumlah sampel dari dataset yang dimasukkan ke dalam model pada setiap iterasi. Banyaknya dataset yang digunakan pada proses pelatihan dapat memengaruhi pemilihan dari *batch size.* Semakin besar *batch size* yang digunakan, semakin banyak data yang diproses sekaligus dalam satu iterasi, sehingga dapat mempercepat proses konvergensi dan membuat pembaruan bobot menjadi lebih stabil. Namun, ukuran *batch* yang terlalu besar seringkali mengarah pada hasil akhir yang kurang optimal karena model cenderung kehilangan kemampuan untuk beradaptasi terhadap variasi data. Jika *batch size* terlalu kecil, proses pelatihan bisa menjadi lebih lambat dan menghasilkan pembaruan bobot yang fluktuatif atau *noise*, karena model hanya belajar dari sebagian kecil data yang tidak selalu merepresentasikan distribusi keseluruhan dataset dengan baik (Akmal Hariz et al., 2022).

1.6.4.3 *Optimizer* dan *Learning Rate*

Optimizer digunakan untuk menyesuaikan parameter jaringan saraf dalam meminimalkan *cost function*, dalam penelitian ini menggunakan *Adaptive Moment Estimation* (Adam). Adam adalah algoritma pengoptimalan yang dapat digunakan untuk memperbarui *weight network* secara iteratif berdasarkan data *training*. Adam adalah algoritma yang populer digunakan karena mencapai hasil yang baik dan cepat dibanding dengan *optimizer* lainnya (Wardana, 2020). *Learning rate* untuk mengontrol kecepatan pelatihan dan memengaruhi ketelitian jaringan untuk menghitung nilai koreksi pada proses pembelajaran. Semakin besar *learning rate* mengakibatkan pembelajaran akan menjadi kurang optimal karena terlalu cepat dan proses training menjadi tidak stabil, sedangkan jika semakin kecil *learning rate* maka *training* akan membutuhkan waktu yang lama (Rochmawati et al., 2021).

1.6.4.4 Fungsi Aktivasi

Fungsi aktivasi merupakan operasi matematik yang dikenakan pada *layer.* Fungsi aktivasi berfungsi untuk menghitung nilai *output* berdasarkan nilai *input* dan bobot pada *neuron*. Cara kerja fungsi aktivasi dalam *neural network* dimulai ketika data *input* diberikan ke jaringan melalui *input layer*. Pada setiap neuron di lapisan berikutnya, nilai *input* dikalikan dengan bobot (*weight*) yang sesuai, kemudian hasilnya dijumlahkan bersama dengan bias. Nilai total tersebut kemudian dilewatkan ke fungsi aktivasi, yang berperan untuk menentukan apakah neuron tersebut akan aktif atau tidak. Fungsi aktivasi juga membantu jaringan saraf dalam mempelajari hubungan nonlinear pada data, sehingga model dapat menghasilkan prediksi yang lebih akurat dan kompleks (Akil, 2023).

Beberapa fungsi aktivasi yang digunakan dalam penelitian ini, di antaranya (Hikmah et al., 2023):

1. *Rectified Linear Unit* (ReLU)

Fungsi aktivasi ReLU tidak seperti fungsi aktivasi *sigmoid* dan *tanh* yang mampu mengatasi *gradient vanishing.* Tetapi fungsi aktivasi ini lebih efisien digunakan pada *deep learning* tanpa bergantung pada prapemrosesan. Adapun persamaan dari fungsi aktivasi ReLU sebagai berikut.

$$f(x) = \begin{cases} 0, x < 0 \\ x, x \geq 0 \end{cases} \quad (2)$$

Keterangan:
x = data inputan

2. *Tanh*
Fungsi aktivasi *tanh* merupakan transformasi dari fungsi aktivasi *sigmoid* dengan nilai unit interval -1 dan +1. Fungsi aktivasi *tanh* dapat diuraikan pada persamaan berikut.

$$f(x) = tanh(x) = \frac{2}{(1 + e^{-2x})} - 1 \quad (3)$$

Keterangan:
x = data inputan
$e \approx 2.718281828459045\ldots$

3. *Sigmoid*
Fungsi aktivasi *sigmoid* memiliki rentang antara 0 dan 1. Fungsi ini dapat menjadi model probabilitas. Fungsi aktivasi *sigmoid* dapat diuraikan pada persamaan berikut.

$$f(x) = \sigma(x) = \frac{1}{1 + e^{x}} \quad (4)$$

Keterangan:
x = data inputan
e $\approx 2.718281828459045\ldots$
$\sigma(x)$ = fungsi aktivasi *sigmoid*

1.6.4.5 *Evaluasi Kinerja*

Dalam konteks penelitian, evaluasi kinerja merupakan proses untuk menilai seberapa baik model yang dikembangkan dalam memprediksi hasil yang diinginkan. Pada penelitian ini, penilaian kinerja model dilakukan menggunakan *loss function* selama proses *training* serta metrik evaluasi, yaitu *Mean Absolute Percentage Error* (MAPE).

Loss function merupakan fungsi yang digunakan untuk menggambarkan besarnya kerugian (*loss*) yang muncul dari setiap kemungkinan hasil yang dihasilkan oleh model. *Loss function* dihitung berdasarkan data *training* dan data *validation* serta interpretasinya didasarkan pada seberapa baik kinerja model dalam dua set ini. *Loss function* yang baik adalah fungsi yang menghasilkan *error* yang diharapkan paling rendah. Dalam penelitian ini digunakan *Mean Squared Error* (MSE) sebagai *loss function*, karena cocok untuk permasalahan regresi dan sensitif terhadap kesalahan berukuran besar (Wulandari et al., 2020).

Mean Absolute Percentage Error (MAPE) adalah ukuran akurasi relatif yang digunakan untuk menentukan persentase deviasi dari hasil prediksi. Dengan kata lain, MAPE merupakan rata-rata kesalahan mutlak selama periode tertentu dan kemudian dikembalikan 100% agar memberikan hasil secara persentase (Lattifia et al., 2022). Persamaan MAPE dapat dilihat pada persamaan 5.

$$MAPE = \frac{1}{n} \sum_{i=1}^{n} \left| \frac{y_i - \hat{y}_i}{y_i} \right| . 100\% \quad (5)$$

Keterangan:
y_i = nilai yang sebenarnya pada daya ke-i
$\hat{y}_i$ = nalai prediksi pada data ke-i
n = jumlah data

Nilai MAPE dapat diartikan ke dalam 4 kategori (Amansyah et al., 2024), yang ditampilkan dalam Tabel 1.

Tabel 1. *Range* nilai MAPE

***Range* MAPE**	**Arti Nilai**
MAPE < 10%	Kemampuan prediksi sangat baik
10% ≤ MAPE < 20 %	Kemampuan prediksi baik
20% ≤ MAPE < 50 %	Kemampuan prediksi cukup
MAPE ≥ 50%	Kemampuan prediksi buruk

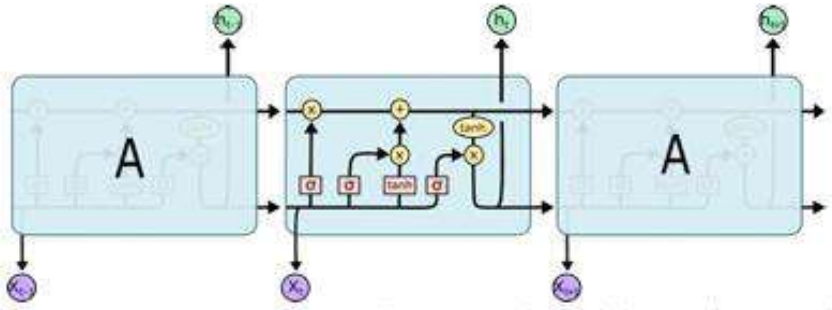
1.6.5 Recurrent Neural Network (RNN)

Recurrent Neural Network (RNN) merupakan jaringan saraf tiruan yang prosesnya dipanggil secara berulang untuk data masukan yang sifatnya *sequential*. RNN adalah bagian dari *deep learning* karena pemrosesan data dilakukan dengan banyak *layer.* RNN juga telah mengalami kemajuan pesat dan telah merevolusi bidang-bidang seperti pemrosesan bahasa alami, pengenalan suara, pemrosesan data finansial seri waktu, dan sebagainya (Firmansyah et al., 2020). Dalam pengembangan lebih lanjut, RNN juga banyak digunakan untuk menyelesaikan tugas yang terkait dengan data *time series.*

Pada proses *training*, RNN tidak membuang data atau informasi proses sebelumnya. RNN menyimpan data memori agar mendapatkan pola data yang lebih baik, yang kemudian hasilnya akan digunakan untuk mendapatkan informasi prediksi yang akurat. Metode yang digunakan dalam RNN untuk menyimpan informasi atau data masa lalu yaitu dengan cara melakukan *looping,* sehingga secara otomatis informasi masa lalu tetap tersimpan (Karyadi & Santoso, 2022).

1.6.6 *Long Short-Term Memory* (LSTM)

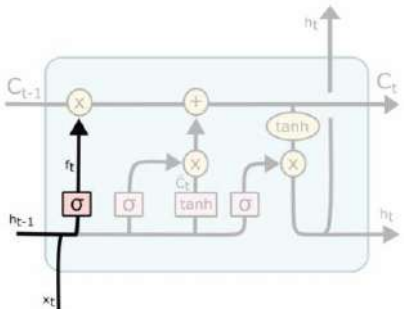
Long Short-Term Memory (LSTM) merupakan salah satu jenis arsitektur *Recurrent Neural Network* (RNN) yang dikenalkan pertama kali oleh Hochreiter dan Schmidhuber pada tahun 1997. Hal yang membedakan antara LSTM dan RNN adalah LSTM mampu mempelajari informasi mana yang harus dibuang dan informasi mana yang harus tetap disimpan di *memory cell* (Andiyantama et al., 2021). LSTM diusulkan sebagai solusi untuk mengatasi terjadinya *vanishing gradient* pada RNN saat memproses data *sequential* yang panjang.



Gambar 2. Arsitektur LSTM (Andiyantama et al., 2021)

Pada LSTM terdapat mekanisme gerbang yang bertujuan untuk menghapus dan memodifikasi informasi dari *cell state*. *Cell state* mengandung struktur yang disebut *cell gates. Cell gates* memperbarui informasi menggunakan empat gerbang, yaitu *input gate* (i_t)*, forget gate* (f_t)*, memory-cell state gate* (C_t)*,* dan *output gate* (o_t)*.* Ilustrasi untuk setiap gerbang dapat dilihat pada Gambar 2 (Budiprasetyo et al., 2023). Input merupakan gate yang digunakan untuk mengontrol *input* data yang layak disimpan atau tidak. *Forget gate* digunakan untuk mengontrol informasi tersembunyi sebelumnya yang akan disimpan di *cell memory* informasi tersembunyi saat ini. *Memory-cell state gate* digunakan untuk perbarui data berdasarkan informasi dari *input gate* dan *forget gate*. *Output gate* digunakan untuk menghitung *output* data dari jaringan berdasarkan *memory-cell state.* Berikut langkah-langkah dari LSTM dan gerbangnya (Winda Kurnia Sari et al., 2020).

1.6.6.1 *Forget Gate*



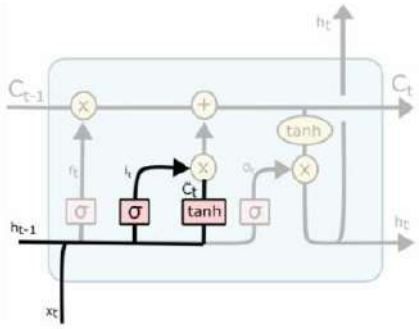
Gambar 3. *Forget gate* (Andiyantama et al., 2021)

Pada Gambar 3, *forget gate* memutuskan informasi mana yang harus dibuang dan mana yang harus disimpan. Keputusan dibuat oleh lapisan *sigmoid* yang disebut lapisan *forget gate* di mana *output* angka antara 0 dan 1. Persamaan *forget gate* ditunjukkan pada persamaan 6.

$$f_t = \sigma(W_f . [h_{t-1}, x_t] + b_f) \quad (6)$$

Keterangan:
Notasi $[h_{t-1}, x_t]$ adalah operasi konkatenasi yang artinya menambahkan baris dari x_t dengan baris dari h_{t-1}.
W_f = bobot dari *forget gate*
h_{t-1} = *state* sebelumnya
x_t = *input* pada waktu t
σ = fungsi aktivasi *sigmoid*
b_f = bias *forget gate*

1.6.6.2 *Input Gate*



Gambar 4. *Input gate* (Andiyantama et al., 2021)

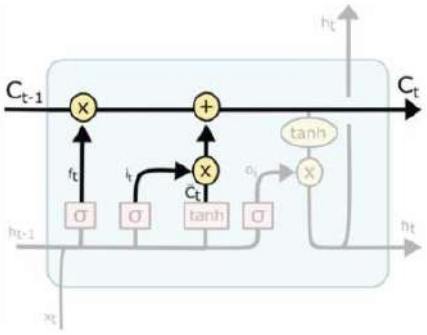
Pada Gambar 4, *input gate* memutuskan nilai mana yang akan diperbarui dengan transformasi nilai antara 0 dan 1. Gerbang ini memiliki dua bagian. Pertama, lapisan *sigmoid* yang disebut lapisan *input gate* memutuskan nilai mana yang harus diperbarui, sebagaimana yang ditunjukkan pada persamaan 7. Kedua, lapisan tanh membuat vektor dari kandidat baru $\hat{C}_t$ yang dapat ditambah kedalam *cell state*, yang ditunjukkan pada persamaan 8.

$$i_t = \sigma(W_i . [h_{t-1}, x_t] + b_i) \quad (7)$$

$$\tilde{C}_t = tanh(W_c . [h_{t-1}, x_t] + b_c) \quad (8)$$

Keterangan:
σ = fungsi aktivasi *sigmoid*
$\tilde{C}_t$ = *intermediate cell state*
W_i = bobot dari *input gate*
W_c = bobot dari *cell state*
h_{t-1} = state sebelumnya
x_t = *input* pada waktu t
σ = fungsi aktivasi *sigmoid*
$tanh$ = fungsi aktivasi *tanh*
b_i = bias *input gate*
b_c = bias *cell state*

1.6.6.3 *Memory Gate*



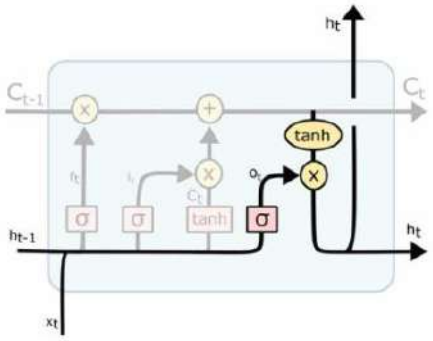
Gambar 5. *Memory-cell state gate* (Andiyantama et al., 2021)

Pada Gambar 5, *memory-cell state gate* bertugas memperbarui *cell state* dengan menghapus sebagian informasi lama dan menambahkan informasi baru sesuai kebutuhan jaringan. Berikut persamaan dari *memory-cell state gate.*

$$C_t = (f_t * C_{t-1} + i_t * \tilde{C}_t) \quad (9)$$

Keterangan:
f_t = *forget gate*
i_t = *input gate*
C_{t-1} = *state* sebelumnya
$\tilde{C}_t$ = *intermediate cell state*

1.6.6.4 *Output Gate*



Gambar 6. *Output gat*e (Andiyantama et al., 2021)

Pada Gambar 6, *output gate* menentukan nilai apa yang akan diteruskan sebagai *hidden state* berikutnya. *Hidden state* sendiri mengandung informasi yang berasal dari *input* pada waktu sebelumnya. Proses ini dimulai dengan lapisan *sigmoid* yang memutuskan bagian mana dari *cell state* yang akan menjadi *output*, sebagaimana ditunjukkan pada persamaan 10. Selanjutnya, *cell state* diaktifkan menggunakan fungsi *tanh*, seperti terlihat pada persamaan 11.

$$o_t = \sigma(W_o\,[h_{t-1}, x_t] + b_o\,) \quad (10)$$

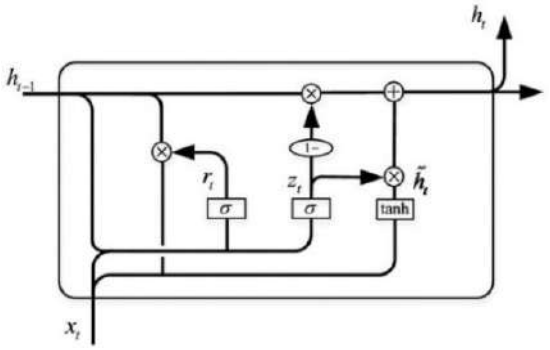
$$h_t = o_t * tanh\,(C_t) \quad (11)$$

Keterangan:
σ = fungsi aktivasi *sigmoid*
$tanh$ = fungsi aktivasi *tanh*
W_o = bobot dari *output gate*
h_{t-1} = *state* sebelumnya
x_t = *input* pada waktu t
b_o = bias *output gate*
h_t = *hidden state output*
C_t = *cell state*

1.6.7 *Gated Recurrent Unit* (GRU)

LSTM mempunyai banyak varian, salah satunya adalah *Gated Recurrent Unit* (GRU) ditemukan oleh Kyunghyun Cho pada tahun 2014 yang memiliki kelebihan pada proses komputasi yang lebih sederhana dibandingkan dengan model LSTM, akan tetapi memiliki akurasi yang setara serta cukup efektif dalam mengurangi permasalahan *vanishing gradient* (Tholib et al., 2023).

GRU merupakan sel dengan kandungan 2 *gate,* yaitu *update gate* dan *reset gate* serta 3 fungsi aktivasi, yaitu dua fungsi aktivasi *sigmoid* dan satu fungsi aktivasi *tanh*. Dengan *gate* dan fungsi aktivasi yang minim ini tentunya akan mempercepat proses pengolahan data yang umumnya berjumlah sangat besar. Kemampuan GRU dirancang untuk menjadi lebih baik dari LSTM terutama untuk dataset yang jumlahnya sedikit (Hastomo et al., 2021). Dalam GRU, komponen pengatur alur informasi akan disebut sebagai *gate.*

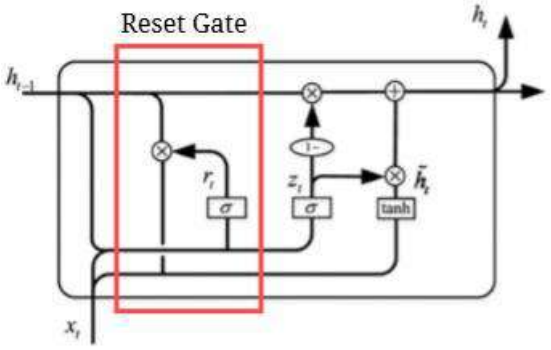


Gambar 7. Arsitektur GRU (Mienye et al., 2024)

Gambar 7 menunjukkan arsitektur GRU. Gerbang-gerbang ini mengontrol aliran informasi untuk memastikan bahwa informasi yang relevan dipertahankan dan informasi yang tidak relevan dibuang. Serupa dengan LSTM, GRU mengandalkan rekurensi internal dalam setiap unit saat mempertahankan dan memperbarui status tersembunyi di seluruh langkah waktu untuk menangkap dependensi temporal (Mienye et al., 2024).

1.6.7.1 *Reset Gate*

Reset gate merupakan gerbang yang digunakan untuk menentukan seberapa banyak informasi tidak relevan yang harus diatur ulang atau dihapus. Pada proses ini *reset gate* akan menentukan berapa banyak informasi dari *time step* terdahulu yang dapat dilupakan menggunakan fungsi aktivasi *sigmoid. Output* dari *reset gate* bernilai 0 dan 1. Jika *output* semakin mendekati nilai 0 berarti informasi dari *time step* sebelumnya tidak terlalu berpengaruh dan akan dihapus. Sedangkan jika mendekati nilai 1 maka informasi dari *time step* terdahulu berpengaruh dan akan disimpan (Prissy, 2022). Proses pada *reset gate* dapat dilihat pada Gambar 8.



Gambar 8. *Reset gate* (Mienye et al., 2024)

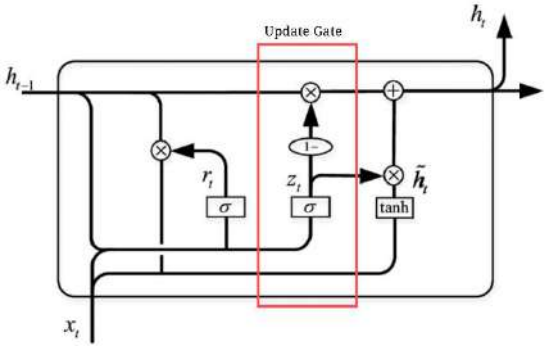
Persamaan *reset gate* diuraikan sebagai berikut.

$$R_t = \sigma(X_t W_r + b_{xr} + H_{t-1} U_r + b_{hr}) \quad (12)$$

Keterangan:
R_t = *reset gate*
σ = fungsi *sigmoid*
X_t = nilai data *input*
H_{t-1} = *hidden state* dari *time step* sebelumnya
W_r , U_r = nilai *weight* pada *reset gate*
b_{xr} , b_{hr} = nilai *bias* pada *reset gate*

1.6.7.2 *Update Gate*

Update Gate berfungsi menentukan seberapa banyak status sel yang akan diperbarui, dengan rentang nilai antara 0 hingga 1. Apabila nilai *update gate* mendekati 0, maka sebagian besar informasi dari *time step* sebelumnya akan dipertahankan. Sebaliknya, jika nilainya mendekati 1 maka informasi lama akan digantikan sepenuhnya oleh informasi baru. Proses kerja *update gate* mirip dengan *input gate* dan *forget gate* pada LSTM (Prissy, 2022). Proses pada *update gate* dapat dilihat pada Gambar 9.



Gambar 9. *Update gate* (Mienye et al., 2024)

Persamaan *update gate* diuraikan sebagai berikut.

$$Z_t = \sigma(X_t W_z + b_{xz} + H_{t-1} U_z + b_{hz}) \quad (13)$$

Keterangan:
Z_t = *update gate*
σ = fungsi *sigmoid*
X_t = nilai data *input*
H_{t-1} = *hidden state* dari *time step* sebelumnya
W_z , U_z = nilai *weight* pada *update gate*
b_{xz} , b_{hz} = nilai bias pada *update gate*

1.6.8 *Pre-processing Data*

1.6.8.1 Data *Outlier*

Data *outlier* adalah fenomena yang wajar dan sering dijumpai dalam analisis data, termasuk pada data *time series*. Pengaruh dari adanya data *outlier* bisa menyebabkan bias atau salah prediksi pada model data *time series* tersebut. *Outlier* merupakan data yang kemunculannya sulit diprediksi karena dapat dipengaruhi oleh berbagai faktor penyebab. Kehadiran *outlier* dapat memberikan dampak besar terhadap proses identifikasi pola, estimasi parameter, maupun hasil peramalan (Aulia & Atok, 2017). Penanganan *outlier* dapat dilakukan melalui dua pendekatan utama, yaitu dengan menghapus data *outlier* apabila terbukti merupakan kesalahan pencatatan atau tidak relevan, serta dengan mengganti nilai *outlier* menggunakan nilai lain yang masih merepresentasikan karakteristik data sehingga informasi penting tetap terjaga.

Pada penelitian ini, deteksi *outlier* dilakukan dengan mengombinasikan analisis *Z-Score* dan visualisasi *boxplot*. Metode *Z-Score* digunakan untuk mengukur sejauh mana suatu nilai menyimpang dari nilai rata-rata data dalam satuan standar deviasi, yang terdapat pada persamaan berikut.

$$Z = \frac{x - \mu}{\sigma} \quad (14)$$

Keterangan:
x = nilai asli data
μ = nilai rata-rata
σ = standar deviasi

Data dengan nilai *Z-Score* lebih besar dari 3 atau lebih kecil dari -3 dikategorikan sebagai kandidat *outlier* karena menunjukkan penyimpangan yang signifikan dari pola umum distribusi data. Hasil deteksi *outlier* menggunakan *Z-Score* umumnya diperkuat dengan visualisasi *boxplot* untuk memberikan gambaran distribusi data secara visual. *Boxplot* memudahkan identifikasi nilai ekstrem yang berada di luar rentang utama data, yang mengindikasikan keberadaan pengamatan dengan karakteristik yang tidak biasa, baik bernilai sangat tinggi maupun sangat rendah dibandingkan dengan mayoritas data lainnya (Tahta Phudjashakty et al., 2026).

Untuk mengurangi pengaruh nilai ekstrem tanpa harus menghilangkan data observasi dari dataset, penelitian ini menerapkan teknik *winsorization.* Secara konsep, nilai yang berada di bawah persentil ke-5 (P_5) dan di atas persentil ke-95 (P_{95}) disesuaikan agar berada pada batas tersebut. Penyesuaian nilai dapat dinyatakan dalam persamaan berikut.

$$X_{new} = \begin{cases} P_5, & x < P_5 \\ x, & P_5 \leq x \leq P_{95} \\ P_{95}, & x > P_{95} \end{cases} \quad (15)$$

Keterangan:
x = nilai asli data
P_5, P_{95} = persentil ke-5 dan ke-95

1.6.8.2 Normalisasi Data Model *Min-Max*

Normalisasi merupakan proses membuat beberapa variabel memiliki rentang nilai yang sama, tidak ada nilai yang terlalu besar maupun terlalu kecil sehingga dapat membuat analisis statistik menjadi lebih mudah (Wenny, 2024). Pada penelitian ini menggunakan normalisasi *min-max.* Normalisasi *min-max* adalah metode proses data yang menggunakan nilai maksimum dan minimum dari suatu atribut untuk mentransformasikan data ke rentang baru secara linear dengan rentang nilai 0 sampai dengan 1 sehingga akan menghasilkan perbandingan antar data yang seimbang, baik sebelum atau sesudah proses normalisasi (Dwididanti & Anggoro, 2022). Persamaan normalisasi *min-max* ditunjukkan pada persamaan 16.

$$X_{new} = \frac{X_{old} - X_{min}}{X_{max} - X_{min}} \quad (16)$$

Keterangan:
X_{new} = nilai ternormalisasi
X_{old} = nilai pada sebuah fitur
X_{min} = nilai minimum
X_{max} = nilai maksimum

1.6.9 Denormalisasi

Denormalisasi adalah sebuah proses mengembalikan data yang telah dinormalisasi ke data sebenarnya untuk dibandingkan dengan data hasil prediksi (Prissy, 2022). Denormalisasi memungkinkan hasil prediksi untuk diinterpretasikan secara lebih realistis dan digunakan dalam pengambilan keputusan. Adapun persamaan denormalisasi yang digunakan adalah sebagai berikut.

$$X_t = y\,(X_{max} - X_{min}) + X_{min} \quad (17)$$

Keterangan:
X_t = nilai data denormalisasi
y = nilai *output* yang dinormalisasi
X_{min} = nilai minimum data aktual
X_{max} = nilai maksimum data aktual

1.6.10 *Streamlit*

Streamlit merupakan sebuah *framework* berbasis *python* dan bersifat *open-source* yang dibuat untuk memudahkan dalam membangun aplikasi web interaktif di bidang data *sience* dan *machine learning. Framework* ini dirancang agar pengguna dapat menampilkan hasil analisis data dan model pembelajaran mesin dengan cepat tanpa memerkulan keahlian dalam pemrograman web. Fitur-fitur yang disediakan oleh *Streamlit* seperti tampilan visualisasi data, input interaktif, serta integrasi dengan berbagai *library python,* membantu mempermudah proses pengembangan melalui antarmuka interaktif yang memungkinkan komunikasi langsung dengan *user* (Hermawan et al., 2024). Dengan menggunakan *framework* ini hanya perlu *install library Streamlit* dan membuat *file python* dalam merancang *website* sehingga tidak perlu membuat *file* HTML, CSS, *JavaScript, Flask* ataupun *tools* lainnya.

1.7 Penelitian Terdahulu

Dalam penyusunan penelitian ini, penulis menggunakan acuan dari berbagai hasil penelitian terdahulu sebagai referensi. Berikut disajikan tabel ringkasan penelitian terdahulu yang relevan dengan topik penelitian ini, yang dapat dilihat pada Tabel 2.

Tabel 2. Ringkasan penelitian terdahulu

No	Tahun	Penulis	Judul	Identifikasi Masalah	Metode Penelitian	Hasil
1.	2020	(Satyo et al., 2020)	Optimalisasi Data Terbatas Prediksi Jangka Panjang Covid-19 dengan Kombinasi Lstm dan Gru	Jumlah data terbatas membuat model sulit memprediksi COVID-19 secara akurat untuk jangka panjang.	Kombinasi metode LSTM dan GRU dengan pengaturan *hidden layer* dan pengurangan *time step* untuk memperbanyak data.	LSTM lebih akurat untuk prediksi kasus baru (RMSE=57,497) sedangkan GRU lebih baik untuk prediksi kematian (RMSE=3,264).
2.	2023	(Carnegie & Chairani, 2023)	Perbandingan *Long Short-Term Memory* (LSTM) dan *Gated Recurrent Unit* (GRU) Untuk Memprediksi Curah Hujan	Curah hujan sulit diprediksi karena pola temporal kompleks dan *noise* tinggi.	Membandingkan model LSTM dan GRU menggunakan data cuaca.	LSTM memiliki performa terbaik dengan RMSE 16.81, sedangkan GRU unggul dalam klasifikasi hujan atau tidak hujan dengan akurasi 62%.

No	Tahun	Penulis	Judul	Identifikasi Masalah	Metode Penelitian	Hasil
3.	2023	(Tholib et al., 2023)	Prediksi Harga Emas Menggunakan Metode Lstm dan Gru	Harga emas memiliki pola *time series* yang kompleks dan belum banyak membandingkan performa LSTM dan GRU secara spesifik.	Model *deep learning* (LSTM dan GRU), evaluasi dengan MAE, RMSE, dan MAPE.	Model LSTM lebih akurat dengan MAE = 0,0389, RMSE = 0,0475, dan MAPE = 5,2047% dibandingkan GRU dalam memprediksi harga emas.
4.	2024	(Fikriaziz et al., 2024)	Metode *Long Short-Term Memory* Untuk Memprediksi Konsumsi Energi Listrik di Kabupaten Kebumen Tahun 2023	Perlu metode akurat untuk memprediksi konsumsi energi listrik di Kabupaten Kebumen.	Menggunakan model LSTM untuk data *time series* konsumsi listrik.	Hasil penelitian ini menunjukkan bahwa LSTM mampu memberikan tingkat prediksi yang baik dengan nilai MAPE sebesar 4.07%.
5.	2024	(Dewi & Riwurohi, 2024)	*Forecasting The Electricity Consumptions Of PLN UP3 Cengkareng Using Deep Learning*	Peramalan kebutuhan listrik pelanggan PLN berdasarkan data historis 10 tahun.	Model LSTM dengan pembagian data 70% *training* dan 30% data *validation*.	LSTM mampu memprediksi kebutuhan listrik dengan akurasi tinggi (32 neuron, 10 *epoch*, nilai *error* terkecil 96.689).

No	Tahun	Penulis	Judul	Identifikasi Masalah	Metode Penelitian	Hasil
6.	2025	(Lim & Handhayani, 2025)	Penerapan Lstm dan Gru Untuk Prediksi Harga Cabai Merah di Kota Jawa Timur	Fluktuasi harga cabai merah di Jawa Timur yang dipengaruhi musim tanam, cuaca, dan permintaan pasar sehingga membutuhkan metode prediksi yang akurat untuk menjaga stabilitas ekonomi.	Membandingkan LSTM dan GRU pada dua skenario data *training* (70% dan 80%) dengan 50 *epoch*.	Pada skenario 80%, LSTM menghasilkan MAE sebesar 1458,764, RMSE sebesar 2596,010, dan R^2 sebesar 0,978. GRU unggul pada 70% data *training* dengan MAE 1742,027, RMSE 2820,462 dan R^2 0,969.
7.	2025	(Perdana et al., 2025)	Analisis Perbandingan Model Gru dan Lstm Untuk Prediksi Harga Saham Bank Rakyat Indonesia	Fluktuasi dan noise tinggi dalam data harga saham, serta kurangnya studi perbandingan LSTM-GRU pada saham Indonesia.	Membandingkan GRU dan LSTM pada data saham BRI (Feb 2023-Nov 2024)	Model GRU lebih unggul dengan peningkatan R^2 sebesar 10,7% dan penurunan MAPE 18,5%; RMSE = 80,731 dan MAE = 62,662.

No	Tahun	Penulis	Judul	Identifikasi Masalah	Metode Penelitian	Hasil
8.	2025	(Hijriani Putri, 2025)	Prediksi Kebutuhan Beras Di Jawa Timur Menggunakan Metode *Gated Recurrent Unit* (GRU)	Ketidakpastian perencanaan pangan menyebabkan ketidakseimbangan produksi dan konsumsi beras.	Model GRU dengan dua lapisan (64 dan 32 neuron), *optimizer* Adam, *loss* MSE, 100 *epoch*, dan evaluasi dengan MAE, MSE, RMSE, R^2.	Model GRU menunjukkan performa prediksi yang sangat baik dengan nilai MAE 0,0103, MSE 0,0001, RMSE 0,0116, dan R^2 0,9935 dibanding LSTM.

Berdasarkan tinjauan terhadap penelitian-penelitian terdahulu yang telah dijelaskan sebelumnya, dapat disimpulkan bahwa LSTM dan GRU merupakan dua model *deep learning* yang banyak digunakan dan terbukti efektif dalam memprediksi data *time series* pada berbagai bidang. Model LSTM umumnya menunjukkan ketepatan prediksi yang baik pada pola data yang kompleks, sementara GRU menawarkan performa yang kompetitif dengan struktur arsitektur yang lebih sederhana. Namun, sebagian besar penelitian tersebut dilakukan pada konteks di luar sektor kelistrikan Indonesia atau tidak secara khusus membandingkan kedua model dalam memprediksi konsumsi listrik.

Oleh karena itu, diperlukan penelitian yang dapat mengevaluasi dan membandingkan kinerja LSTM dan GRU pada data konsumsi listrik di tingkat lokal, khususnya di Kota Makassar. Keterbatasan kajian sebelumnya dalam memberikan analisis perbandingan kedua model pada konteks tersebut menjadi landasan penting dilakukannya penelitian ini untuk memperoleh model prediksi yang lebih relevan, akurat, dan dapat mendukung kebutuhan perencanaan energi di wilayah setempat.

BAB II
METODE PENELITIAN

2.1 Pendekatan Penelitian

Pendekatan ini menggunakan pendekatan kuantitatif sebagai landasan utama dalam pengumpulan dan analisis data. Pendekatan ini dipilih karena mampu memberikan gambaran objektif dan terukur mengenai hasil prediksi konsumsi listrik di Kota Makassar. Pendekatan kuantitatif berfokus pada pengolahan data numerik berupa data historis konsumsi listrik yang bersifat *time series,* sehingga memungkinkan analisis pola dan tren penggunaan listrik dari waktu ke waktu. Selain itu, pendekatan ini mendukung penerapan metode *Long Short-Term Memory* (LSTM) dan *Gated Recurrent Unit* (GRU) yang digunakan untuk mempelajari pola data historis dan menghasilkan prediksi konsumsi listrik secara lebih akurat.

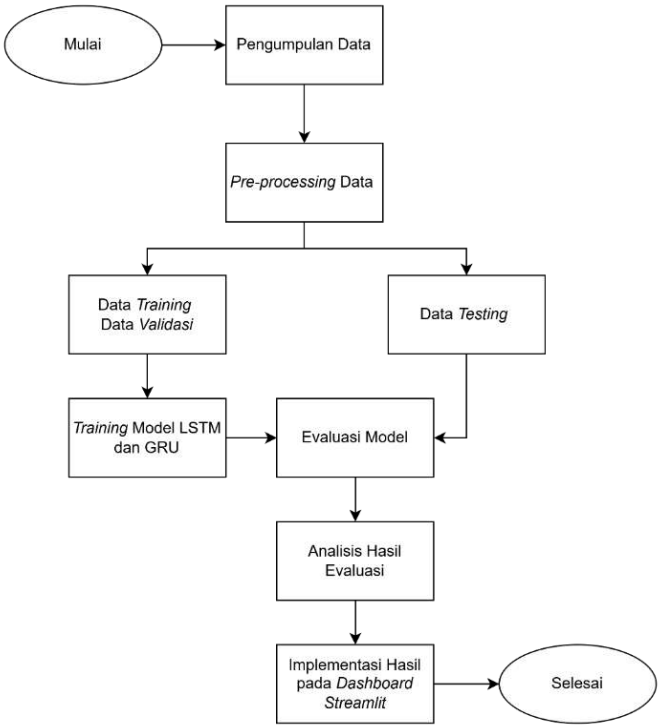
2.2 Waktu dan Tempat Penelitian

Penelitian ini dilaksanakan selama kurang lebih 3 bulan, dimulai dari bulan September hingga bulan November 2025. Penelitian ini bertempat di PT PLN (Persero) Induk Unit Wilayah Sulselrabar sehingga data yang diperoleh sesuai dengan tujuan penelitian. Adapun untuk detail waktu penelitian dapat dilihat pada Tabel 3.

Tabel 3. Waktu penelitian

Kegiatan	September				Oktober				November			
	1	2	3	4	1	2	3	4	1	2	3	4
Studi Literatur												
Pengumpulan Data												
Pre-processing Data												
Training Model												
Evaluasi Model												
Analisis Hasil												
Deploy Model												

Untuk memberikan gambaran yang lebih jelas mengenai tahapan yang dilakukan selama proses penelitian, berikut disajikan alur penelitian yang menggambarkan langkah-langkah mulai dari pengumpulan data hingga evaluasi hasil model yang dapat dilihat pada Gambar 10.



Gambar 10. Diagram alur penelitian

2.3 Deskripsi Data dan Sumber Data

Data yang digunakan dalam penelitian ini merupakan data historis konsumsi listrik pelanggan PLN di Kota Makassar. Data tersebut bersifat *time series,* karena dicatat secara berkala berdasarkan periode waktu bulanan. Setiap entri data mencakup informasi mengenai tahun, bulan, tanggal, dan jumlah konsumsi listrik (kWh). Sumber data diperoleh dari PT PLN (Persero) Unit Induk Wilayah Sulselrabar. Data dibagi menjadi dua kelompok berdasarkan Unit Pelaksana Pelayanan Pelanggan (UP3), yaitu UP3 Makassar Selatan dan UP3 Makassar Utara dengan rentang waktu data Januari 2019 sampai Agustus 2024 dengan total 68 baris data per wilayah, sehingga keseluruhan data berjumlah 136 baris data.

Metode pengumpulan data yang dilakukan dengan mengumpulkan data sekunder yang telah tersedia dari instansi atau sumber resmi. Penggunaan data sekunder dipilih karena informasi tersebut telah tercatat secara sistematis serta dapat diandalkan untuk mendukung proses analisis serta penerapan model prediksi konsumsi listrik. Untuk memberikan gambaran yang lebih jelas mengenai data yang digunakan dalam penelitian ini, berikut disajikan tabel yang menampilkan beberapa baris data konsumsi listrik dari masing-masing wilayah.

Tabel 4. Data konsumsi listrik bulanan UP3 Makassar Selatan

tahun	bulan	tanggal	konsumsi_kWh
2019	Januari	2019-01-01 00:00:00	171792789
2019	Februari	2019-02-01 00:00:00	111438251
2019	Maret	2019-03-01 00:00:00	125652134
2019	April	2019-04-01 00:00:00	128265149
2019	Mei	2019-05-01 00:00:00	137017830
…	…	…	…
2024	Juni	2024-06-01 00:00:00	162330095
2024	Juli	2024-07-01 00:00:00	160461104
2024	Agustus	2024-08-01 00:00:00	165458752

Tabel 5. Data konsumsi listrik bulanan UP3 Makassar Utara

tahun	bulan	tanggal	konsumsi_kWh
2019	Januari	2019-01-01 00:00:00	165439955
2019	Februari	2019-02-01 00:00:00	152450538
2019	Maret	2019-03-01 00:00:00	162244934
2019	April	2019-04-01 00:00:00	155772534
2019	Mei	2019-05-01 00:00:00	151874687
…	…	…	…
2024	Juni	2024-06-01 00:00:00	193975757
2024	Juli	2024-07-01 00:00:00	205425976
2024	Agustus	2024-08-01 00:00:00	194979348

Berdasarkan Tabel 4 dan Tabel 5, dapat diketahui bahwa data konsumsi listrik di kedua wilayah memiliki pola yang relatif fluktuatif dari bulan ke bulan. Nilai konsumsi listrik di UP3 Makassar Utara umumnya lebih tinggi dibandingkan dengan UP3 Makassar Selatan. Variasi nilai konsumsi listrik ini yang menjadi dasar penting bagi penelitian untuk melakukan prediksi konsumsi listrik.

2.4 Instrumen Penelitian

Dalam penelitian ini terdapat beberapa instrumen dalam membantu proses penelitian agar berjalan secara lancar. Instrumen tersebut terbagi menjadi dua yaitu perangkat keras (*hardware*) dan perangkat lunak (*software*).

2.4.1 Perangkat Keras (*Hardware*)

Perangkat keras beserta dengan spesifikasi yang digunakan dalam penelitian kali ini dapat dilihat pada Tabel 6.

Tabel 6. Perangkat keras (*hardware*)

Jenis	Spesifikasi
Sistem Operasi	Windows 10 (64-bit)
Prosesor	AMD Ryzen 7 4800H with Radeon Graphics @ 2.90 GHz
Memori (RAM)	8 GB DDR4
GPU	NVIDIA GeForce GTX 1650

2.4.2 Perangkat Lunak (*Software*)

Terdapat beberapa perangkat lunak atau *software* yang digunakan dalam penelitian ini yang bertujuan untuk mempermudah proses kerja. Perangkat lunak yang digunakan dapat dilihat pada Tabel 7.

Tabel 7. Perangkat lunak *(software)*

Perangkat Lunak	Fungsi Utama
Python	Bahasa pemrograman utama yang digunakan dalam penelitian ini.
NumPy	*Library python* untuk komputasi numerik, menyediakan dukungan terhadap operasi *array* dan matriks multidimensi.
Pandas	*Library python* untuk pengolahan dataset yang berupa *dataframe*.
Matplotlib & Seaborn	*Library* visualisasi data untuk menampilkan grafik, diagram, dan pola tren secara informatif.
Scikit-learn	*Library* pembelajaran mesin yang digunakan untuk *preprocessing* data, pembagian dataset, dan evaluasi model.
TensorFlow / Keras	*Framework deep learning* yang digunakan untuk membangun dan melatih model jaringan saraf seperti LSTM dan GRU.
Streamlit	*Framework* yang membantu dalam membuat *dashboard* atau membagikan visual data ke *website* sederhana.

Perangkat Lunak	Fungsi Utama
Jupyter Notebook	Lingkungan interaktif untuk menjalankan kode *python*, eksplorasi data, dan dokumentasi hasil eksperimen.
Visual Studio Code	Editor kode yang digunakan untuk pengembangan program, integrasi *library*, dan implementasi model penelitian.

2.5 Teknik Analisis Data dan Pemodelan

Teknik analisis data pada penelitian ini bertujuan untuk menghasilkan model prediksi konsumsi listrik di Kota Makassar menggunakan metode *Long Short-Term Memory* (LSTM) dan *Gated Recurrent Unit* (GRU). Proses analisis meliputi tahap *pre-processing* data, pembangunan model, pelatihan dan pengujian, evaluasi performa, serta pembuatan *dashboard* interaktif menggunakan *Streamlit,*

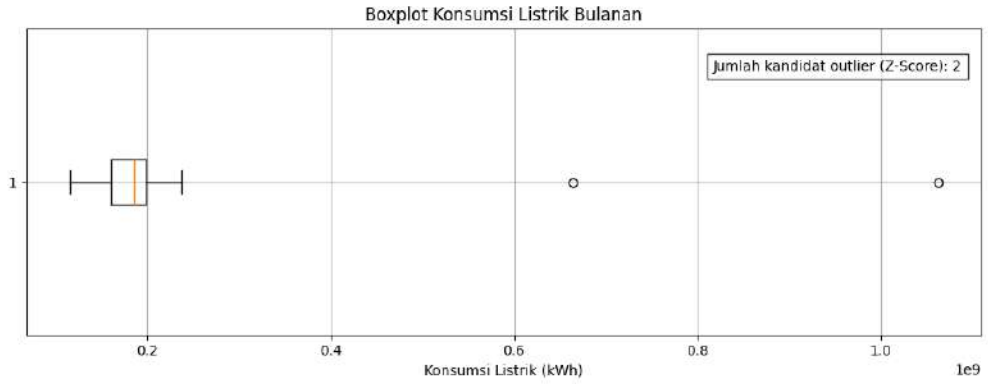
2.5.1 Tahapan *Pre-Processing* Data

Setelah data terkumpul, data tersebut akan berupa *raw* dataset atau dataset mentah yang mungkin masih mengandung beberapa anomali atau kekurangan. Sebelum data dapat dianalisis dan digunakan untuk pemodelan, *raw* dataset akan melalui serangkaian tahapan *pre-processing*. Tahapan *pre-processing* data dilakukan untuk menyiapkan data mentah agar siap digunakan pada proses pelatihan model. Proses ini bertujuan meningkatkan kualitas data serta memastikan model dapat mengenali pola dengan lebih akurat. Adapun tahapan *pre-processing* dalam penelitian ini meliputi deteksi dan penanganan *oulier* data, normalisasi data, serta pembagian data (*splitting* data).

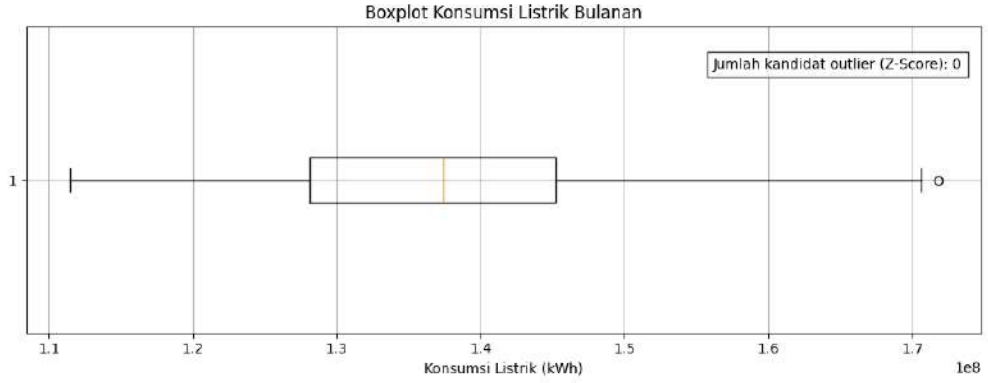
2.5.1.1 Deteksi dan Penanganan *Outlier* Data

Pada tahap *pre-processing*, dilakukan deteksi dan penanganan *outlier* pada data konsumsi listrik wilayah UP3 Makassar Utara dan UP3 Makassar Selatan. Berdasarkan hasil eksplorasi awal, ditemukan sejumlah nilai konsumsi yang menyimpang secara signifikan dari pola historis pada wilayah UP3 Makassar Utara, seperti lonjakan konsumsi yang terlalu tinggi pada bulan tertentu. Sedangkan pada wilayah UP3 Makassar Selatan tidak ditemukan nilai ekstrem yang menyimpang secara signifikan. Nilai-nilai ekstrem pada wilayah UP3 Makassar Utara berpotensi mengganggu proses normalisasi dan dapat menyebabkan model LSTM atau GRU mempelajari pola yang keliru, sehingga menghasilkan prediksi yang kurang stabil. Proses deteksi *outlier* dilakukan menggunakan metode *Z-Score*, yang mengukur tingkat penyimpangan suatu nilai terhadap rata-rata data dalam satuan standar deviasi. Data dengan nilai *Z-Score* lebih besar dari 3 atau lebih kecil dari −3 dikategorikan sebagai kandidat *outlier* karena berada

jauh di luar pola umum distribusi data. Hasil deteksi ini kemudian diperkuat melalui visualisasi *boxplot*, yang digunakan untuk mengonfirmasi keberadaan nilai ekstrem secara visual dan mempermudah interpretasi distribusi data. Hasil deteksi *outlier* menggunakan metode *Z-Score* dan *boxplot* pada data konsumsi listrik wilayah UP3 Makassar Utara ditunjukkan pada Gambar 11, sedangkan visualisasi *boxplot* untuk data konsumsi listrik wilayah UP3 Makassar Selatan ditunjukkan pada Gambar 12.



Gambar 11. Hasil deteksi *outlier* data UP3 Makassar Utara

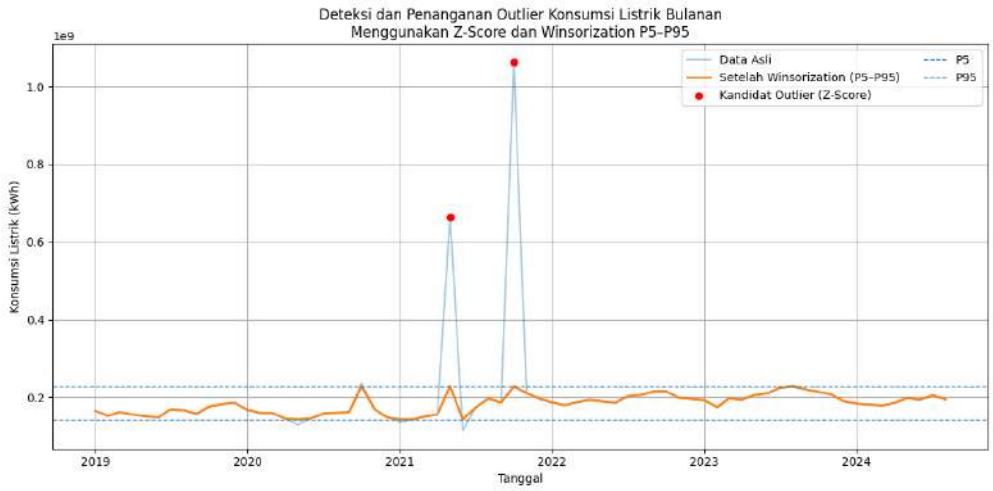


Gambar 12. Hasil deteksi *outlier* data UP3 Makassar Selatan

Berdasarkan visualisasi pada gambar di atas, data wilayah UP3 Makassar Utara menunjukkan adanya beberapa nilai ekstrem yang berada di luar batas rentang utama data. Nilai-nilai tersebut merepresentasikan lonjakan konsumsi listrik pada bulan tertentu yang menyimpang dari pola historis secara umum dan dikategorikan sebagai *outlier* serta memerlukan penanganan lebih lanjut. Sedangkan data wilayah UP3 Makassar Selatan tidak menunjukkan adanya nilai ekstrem yang berada di luar batas rentang utama data, sehingga dapat disimpulkan bahwa tidak terdapat *outlier* yang signifikan pada wilayah tersebut.

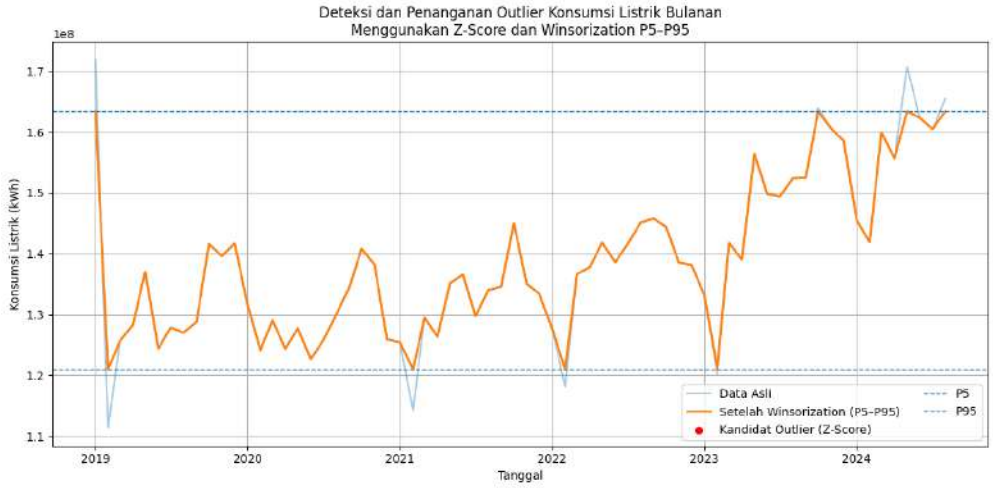
Selanjutnya, untuk menangani *outlier* tanpa menghilangkan data, penelitian ini menerapkan teknik *winsorization* dengan batas persentil ke-5 (P_5) dan persentil ke-95

(P_{95}). Pada tahap ini, nilai konsumsi yang berada di bawah P_5 disesuaikan menjadi nilai P_5, sedangkan nilai yang berada di atas P_{95} disesuaikan menjadi nilai P_{95}. Pendekatan ini bertujuan untuk mengurangi pengaruh nilai ekstrem terhadap persebaran data, sekaligus mempertahankan karakteristik utama dan kontinuitas pola historis konsumsi listrik. Hasil penerapan *winsorization* terhadap data konsumsi listrik bulanan wilayah UP3 Makassar Utara ditunjukkan pada Gambar 13 yang memperlihatkan perbandingan data sebelum dan sesudah dilakukan penanganan *outlier*.



Gambar 13. Penerapan *winsorization* data UP3 Makassar Utara

Pada data wilayah UP3 Makassar Selatan, meskipun tidak ditemukan *outlier* berdasarkan analisis *Z-Score* dan *boxplot*, teknik *winsorization* tetap diterapkan secara konsisten untuk menjaga keseragaman tahapan *pre-processing* antar wilayah. Penerapan *winsorization* pada data ini bersifat minimal dan tidak mengubah pola distribusi data secara signifikan. Hasil penerapan winsorization pada data konsumsi listrik wilayah UP3 Makassar Selatan ditunjukkan pada Gambar 14, yang memperlihatkan perbandingan data sebelum dan sesudah dilakukan *winsorization.*



Gambar 14 Penerapan *winsorization* data UP3 Makassar Selatan

2.5.1.2 Normalisasi Data

Normalisasi data merupakan langkah dalam proses *pre-processing* yang bertujuan untuk menyeragamkan skala nilai pada setiap variabel agar berada dalam rentang tertentu. Dalam penelitian ini, proses normalisasi dilakukan menggunakan metode *Min-Max Scaler* yang telah dijelaskan pada bagian landasan teori agar data berada pada interval 0 sampai 1. Langkah ini sangat penting karena dapat meminimalkan tingkat kesalahan (*error*) pada proses pelatihan model. Penerapan metode ini memastikan setiap nilai data memiliki skala yang sebanding, sehingga metode LSTM dan GRU dapat mengenali pola data dengan lebih efisien dan menghasilkan proses pelatihan yang stabil.

2.5.1.3 Pembagian Data (*Splitting* Data)

Setelah proses normalisasi selesai, tahap selanjutnya adalah melakukan pembagian data (*splitting* data) untuk keperluan pelatihan dan evaluasi model. Sebelum dilakukan *splitting*, data terlebih dahulu dibentuk menjadi *sequence* agar sesuai dengan karakteristik model berbasis *time series*. Pada penelitian ini digunakan *sequence length* sebesar 12, sehingga setiap sampel input memuat data konsumsi listrik dari 12 bulan sebelumnya yang digunakan untuk memprediksi konsumsi listrik pada bulan berikutnya.

Dari total 68 baris data pada masing-masing wilayah (UP3 Makassar Selatan dan UP3 Makassar Utara), proses pembentukan *sequence* dengan *window* sepanjang 12 bulan menghasilkan 56 *sequence*. *Sequence* tersebut terbentuk mulai dari indeks ke-12 hingga ke-67 sesuai dengan panjang jendela *input* yang digunakan. Setelah *sequence* diperoleh, dilakukan *spilitting* data menggunakan proporsi 80% untuk *training*, 10% untuk *validation*, dan 10% untuk *testing*. Dari total 56 *sequence*, sebanyak 44 digunakan sebagai data *training*, 5 sebagai data *validation*, dan 7 sebagai data *testing*. Pembagian ini dilakukan secara terpisah untuk setiap wilayah agar model dapat

mempelajari pola konsumsi listrik yang spesifik pada masing-masing wilayah tersebut. Berikut disajikan tabel pembagian data *sequence* untuk kedua wilayah penelitian.

Tabel 8. Hasil pembagian data (*splitting* data)

Wilayah	Total Data	*Sequence* Data	Data *Train* 80%	Data *Validation* 10%	Data *Test* 10%
UP3 Makassar Selatan	68	56	44	5	7
UP3 Makassar Utara	68	56	44	5	7

2.5.2 Arsitektur Model (*Setup Training*)

Dalam penelitian ini, proses pemodelan dilakukan menggunakan *Keras* dengan *TensorFlow* sebagai *backend*. Dua arsitektur utama yang digunakan adalah *Long Short-Term Memory* (LSTM) dan *Gated Recurrent Unit* (GRU). Kedua model tersebut dirancang untuk prediksi *time series* dengan pendekatan regresi untuk memperkirakan konsumsi listrik di Kota Makassar berdasarkan data historis.

2.5.2.1 *Long Short-Term Memory* (LSTM)

Model LSTM yang digunakan dalam penelitian ini terdiri atas beberapa *layer* berurutan yang dirancang untuk menangkap pola dalam data *time series.* Arsitektur model yang digunakan terdiri atas dua lapisan LSTM, dengan lapisan pertama memiliki 64 unit yang berfungsi mengekstraksi pola sekuensial dari data historis. *Output* dari lapisan ini diteruskan ke lapisan kedua yang terdiri atas 32 unit untuk menangkap hubungan temporal yang lebih kompleks.

Untuk mencegah *overfitting,* setiap lapisan LSTM diikuti dengan lapisan *Dropout* dengan *rate* 0,2 yang secara acak menonaktifkan sebagian neuron selama pelatihan. Selanjutnya hasil *output* diproses oleh lapisan *Dense* dengan 16 neuron dan fungsi aktivasi *ReLU,* yang memungkinkan model mempelajari hubungan nonlinear antar fitur. Lapisan *output Dense* terdiri dari satu neuron dengan fungsi aktivasi linear, yang menghasilkan prediksi kontinu berupa konsumsi listrik pada periode berikutnya. Model dikompilasi menggunakan *optimizer* Adam dengan *learning rate* 0,0001 dan fungsi *loss* yaitu *Mean Squared Error* (MSE), yang sesuai untuk masalah regresi deret waktu. Rincian arsitektur dari model LSTM yang digunakan ditunjukkan pada Tabel 9.

Tabel 9. Arsitektur model LSTM

Nama *Layer (Type)*	Jumlah Unit / Neuron	Fungsi Aktivasi	*Output Shape*	Jumlah Parameter	Keterangan
LSTM	64	-	(None, 12, 64)	16.896	Mengekstraksi pola sekuensial dari data historis
Dropout	*rate* = 0,2	-	(None, 12, 64)	0	Regulasi untuk mencegah *overfitting*
LSTM	32	-	(None, 32)	12.416	Menangkap hubungan temporal yang lebih kompleks
Dropout	*rate* = 0,2	-	(None, 32)	0	Regulasi tambahan untuk meningkatkan generalisasi model.
Dense	16	*ReLU*	(None, 16)	528	Menghubungkan fitur nonlinear antar *layer*
Dense (Output)	1	Linear	(None, 1)	17	Menghasilkan prediksi kontinu
Total parameter				29.857	Seluruh parameter bersifat *trainable*

2.5.2.2 *Gated Recurrent Unit* (GRU)

Model *Gated Recurrent Unit* (GRU) merupakan varian dari *Recurrent Neural Network* (RNN) yang memiliki struktur serupa dengan LSTM, namun lebih sederhana karena tidak memiliki memori sel (*cell state*) terpisah. Arsitektur GRU yang digunakan dalam penelitian ini terdiri atas dua lapisan GRU berturut-turut, di mana lapisan pertama memiliki 64 unit yang berfungsi untuk menangkap pola sekuensial jangka panjang, dan lapisan kedua memiliki 32 unit untuk mempelajari hubungan temporal yang lebih kompleks.

Sama seperti model LSTM, setiap lapisan GRU diikuti oleh lapisan *Dropout* dengan *rate* 0,2 untuk mengurangi risiko *overfitting*. Hasil *output* kemudian diteruskan ke lapisan *Dense* berisi 16 neuron dengan aktivasi ReLU agar model mampu menangkap hubungan nonlinear antar fitur, dan diakhiri dengan lapisan *output Dense* satu neuron dengan aktivasi linear untuk menghasilkan prediksi kontinu. Model GRU juga dikompilasi menggunakan *optimizer* Adam dengan *learning rate* 0,0001 dan fungsi *loss* yaitu *Mean Squared Error* (MSE). Keunggulan utama GRU dibandingkan LSTM adalah efisiensi

komputasinya yang lebih tinggi, sehingga pelatihan model cenderung lebih cepat tanpa menurunkan akurasi secara signifikan. Rincian lengkap arsitektur model GRU yang digunakan dalam penelitian ini dapat dilihat pada Tabel 10.

Tabel 10. Arsitektur model GRU

Nama *Layer* *(Type)*	**Jumlah Unit / Neuron**	**Fungsi Aktivasi**	***Output Shape***	**Jumlah Parameter**	**Keterangan**
GRU	64	-	(None, 12, 64)	16.896	Mengekstraksi pola sekuensial dari data historis
Dropout	*rate* = 0,2	-	(None, 12, 64)	0	Regulasi untuk mencegah *overfitting*
GRU	32	-	(None, 32)	12.416	Menangkap hubungan temporal yang lebih kompleks
Dropout	*rate* = 0,2	-	(None, 32)	0	Regulasi tambahan untuk meningkatkan generalisasi model.
Dense	16	*ReLU*	(None, 16)	528	Menghubungkan fitur nonlinear antar *layer*
Dense (Output)	1	Linear	(None, 1)	17	Menghasilkan prediksi kontinu
Total parameter				29.857	Seluruh parameter bersifat *trainable*

2.5.3 Pengaturan *Hyperparameter* Pelatihan Model

Pada tahapan ini, model yang telah dirancang akan melalui proses *training* dan *testing* untuk mengevaluasi performa prediksi konsumsi listrik. Proses ini dilakukan setelah data telah dinormalisasi serta dibagi menjadi data *training,* data *validation,* dan data *testing.* Eksperimen dilakukan dengan variasi *batch size* 32 dan 64, serta jumlah *epoch* maksimum sebanyak 500. Proses pelatihan dijalankan hingga mencapai 500 *epoch* untuk setiap kombinasi *hyperparameter*. Selama pelatihan, nilai *loss* pada data *training* dan *validation* dicatat pada setiap *epoch*, kemudian *epoch* terbaik ditentukan berdasarkan nilai *validation loss* terendah. Untuk kombinasi *hyperparameter* yang digunakan dalam penelitian ini dapat dilihat pada Tabel 11.

Tabel 11. Kombinasi *hyperparameter* pelatihan model

Hyperparameter	Konfigurasi
Batch size	32, 64
Epoch maksimum	500
Optimizer	Adam
Learning rate	0,0001
Loss function	*Mean Squared Error* (MSE)

Setiap kombinasi *hyperparameter* menghasilkan model yang berbeda, sehingga memungkinkan analisis performa berdasarkan pengaturan yang digunakan. Untuk setiap kombinasi, model dibangun kembali agar pelatihan tidak dipengaruhi oleh model lainnya, kemudian dilatih menggunakan data *training* dan divalidasi menggunakan data *validation*. Selama *training*, nilai *loss* dicatat pada setiap *epoch* dan divisualisasikan untuk memantau konvergensi serta mendeteksi kemungkinan *overfitting*. Setelah proses *training* selesai, model diuji menggunakan data *testing* untuk mengevaluasi performanya.

2.5.4 Metrik Evaluasi Kinerja Model

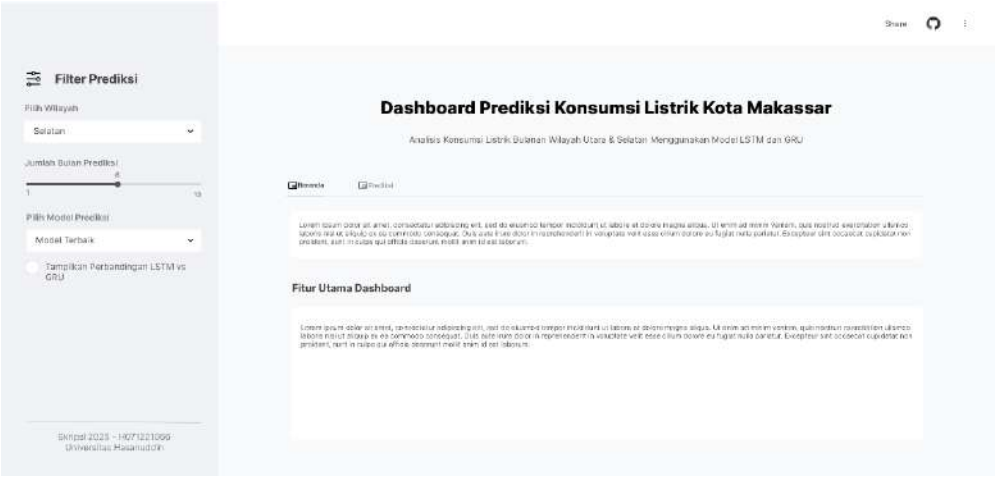
Untuk menilai kinerja model LSTM dan GRU dalam prediksi konsumsi listrik, digunakan metrik evaluasi yang umum diterapkan pada prediksi *time series*. Metrik tersebut adalah *Mean Absolute Percentage Error* (MAPE), yang menghitung rata-rata selisih absolut antara nilai aktual dan prediksi dalam bentuk persentase. Sebelum menghitung MAPE, nilai hasil prediksi dan nilai aktual perlu dikembalikan ke skala aslinya (denormalisasi). Proses ini dilakukan karena data awal telah melalui tahap normalisasi menggunakan *MinMaxScaler* untuk menyesuaikan rentang nilai antara 0 hingga 1. Denormalisasi dilakukan dengan menggunakan kembali nilai minimum dan maksimum dari data asli sehingga diperoleh nilai konsumsi listrik dalam satuan kWh. Hasil evaluasi ini akan menentukan sejauh mana model layak digunakan untuk memprediksi konsumsi listrik di Kota Makassar.

2.6 Rancangan Dashboard Streamlit

Dalam penelitian ini, dibangun sebuah antarmuka pengguna atau *dashboard* menggunakan *Streamlit,* yang merupakan *framework open source* untuk membangun aplikasi web *data science* secara cepat dan efisien menggunakan *Python*. Halaman tampilan yang dikembangkan pada *website* ada dua. Pertama, halaman beranda yang menjadi tampilan utama saat *website* diakses. Kedua, halaman prediksi yang menampilkan hasil prediksi konsumsi listrik beserta visualisasi tren, pemilihan model, serta informasi performa model sehingga pengguna dapat melakukan analisis secara lebih mudah dan interaktif.

2.6.1 Halaman Beranda

Sebelum proses implementasi dilakukan, rancangan antarmuka *dashboard* terlebih dahulu divisualisasikan dalam bentuk *wireframe*. Pembuatan *wireframe* ini bertujuan untuk memberikan representasi awal mengenai penyusunan elemen tampilan, alur interaksi pengguna, serta tata letak fitur yang akan diimplementasikan pada *website*. Dengan adanya *wireframe*, proses perancangan dapat dilakukan secara lebih terarah dan sistematis, serta meminimalkan perubahan signifikan pada tahap pengembangan. Rincian *wireframe* untuk halaman beranda ditampilkan pada gambar berikut.



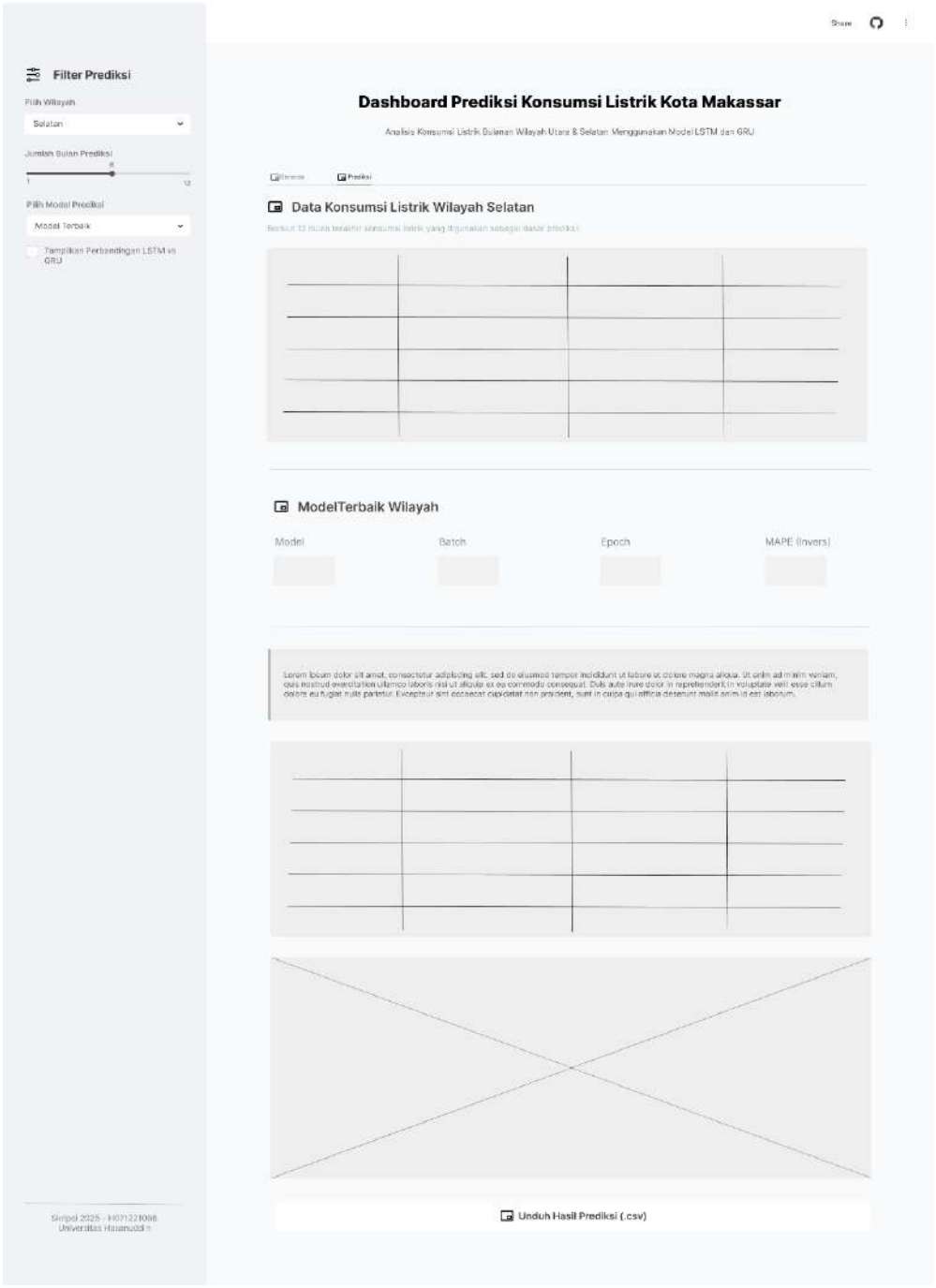
Gambar 15. *Wireframe* - halaman beranda

Gambar 15 menampilkan *wireframe* halaman beranda dari *dashboard* prediksi konsumsi listrik. Pada bagian kiri halaman ditampilkan panel navigasi yang berfungsi sebagai elemen kontrol utama, berisi beberapa opsi seperti pemilihan wilayah, penentuan jumlah bulan prediksi, pemilihan model prediksi, serta opsi untuk menampilkan perbandingan performa model LSTM dan GRU. Sementara itu, pada bagian kanan halaman ditampilkan judul *dashboard*, deskripsi singkat mengenai tujuan dan fungsi aplikasi, serta informasi mengenai fitur utama yang dapat diakses oleh pengguna. Selain itu, pada bagian kanan atas tersedia ikon berbagi yang mengindikasikan bahwa pengguna dapat membagikan konten aplikasi atau mengakses *source code* melalui tautan *GitHub* yang disediakan.

2.6.2 Halaman Prediksi

Pada *wireframe* halaman prediksi, antarmuka ini dirancang sebagai pusat interaksi utama antara pengguna dan sistem prediksi konsumsi listrik. Seluruh komponen analisis serta *output* model ditata secara terstruktur sehingga informasi dapat dipahami dengan jelas dan digunakan secara efektif. Perancangan tampilan ini juga mempertimbangkan aspek kemudahan penggunaan (*usability*), sehingga pengguna dapat mengakses dan

memanfaatkan fitur yang tersedia tanpa memerlukan pemahaman teknis yang kompleks. Adapun rancangan *wireframe* untuk halaman prediksi ditampilkan pada gambar berikut.



Gambar 16. *Wireframe* - halaman prediksi

Gambar 16 menunjukkan *wireframe* halaman prediksi yang menjadi komponen inti dari *dashboard*. Pada bagian kiri, disediakan panel pengaturan prediksi yang ditampilkan secara konsisten untuk memudahkan pengguna dalam melakukan konfigurasi parameter, seperti pemilihan wilayah, jumlah bulan prediksi, pemilihan model, serta opsi perbandingan model tanpa perlu berpindah halaman. Panel ini berfungsi sebagai elemen kontrol utama dalam menjalankan proses prediksi sesuai kebutuhan pengguna.

Pada bagian kanan halaman, ditampilkan hasil analisis dan visualisasi data secara berurutan. Informasi yang ditampilkan mencakup data konsumsi listrik historis berdasarkan wilayah yang dipilih serta informasi terkait model terbaik, meliputi jenis model, konfigurasi *batch size*, jumlah *epoch*, dan nilai MAPE sebagai indikator evaluasi performa. Selanjutnya, *website* menampilkan tabel hasil prediksi untuk periode waktu berikutnya dan grafik perbandingan antara nilai aktual dan nilai prediksi untuk memberikan representasi visual yang lebih komprehensif. Selain itu, halaman ini juga dilengkapi dengan fitur unduhan hasil prediksi dalam format *.csv* sehingga pengguna dapat menyimpan dan memanfaatkan data tersebut untuk keperluan analisis lebih lanjut atau penyusunan laporan.