

LAPORAN TUGAS AKHIR

ANALISA PERFORMA *OPTIMIZER* ALGORITMA PEMBELAJARAN MESIN DALAM PREDIKSI IRADIASI MATAHARI BERDASARKAN LUAS PENERANGAN (LUX) DI ATAP GEDUNG JAKARTA SELATAN

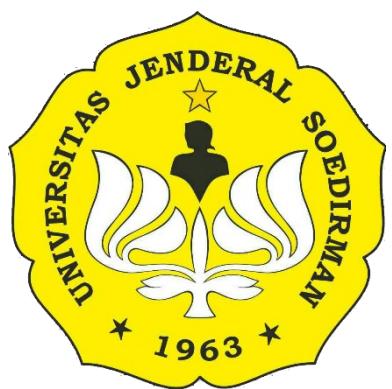
Disusun untuk memenuhi sebagian persyaratan memperoleh
gelar Sarjana Teknik di Jurusan Teknik Elektro
Universitas Jenderal Soedirman



Disusun oleh:

Rangga Wibisana Putra Pamungkas
H1A021058

**KEMENTERIAN PENDIDIKAN TINGGI, SAINS, DAN
TEKNOLOGI
UNIVERSITAS JENDERAL SOEDIRMAN
FAKULTAS TEKNIK
JURUSAN/PROGRAM STUDI TEKNIK ELEKTRO
PURBALINGGA
2025**



LAPORAN TUGAS AKHIR

**ANALISA PERFORMA *OPTIMIZER ALGORITMA*
PEMBELAJARAN MESIN DALAM PREDIKSI IRADIASI
MATAHARI BERDASARKAN LUAS PENERANGAN (LUX)
DI ATAP GEDUNG JAKARTA SELATAN**

Disusun untuk memenuhi sebagian persyaratan memperoleh
gelar Sarjana Teknik di Jurusan Teknik Elektro
Universitas Jenderal Soedirman



Disusun oleh:

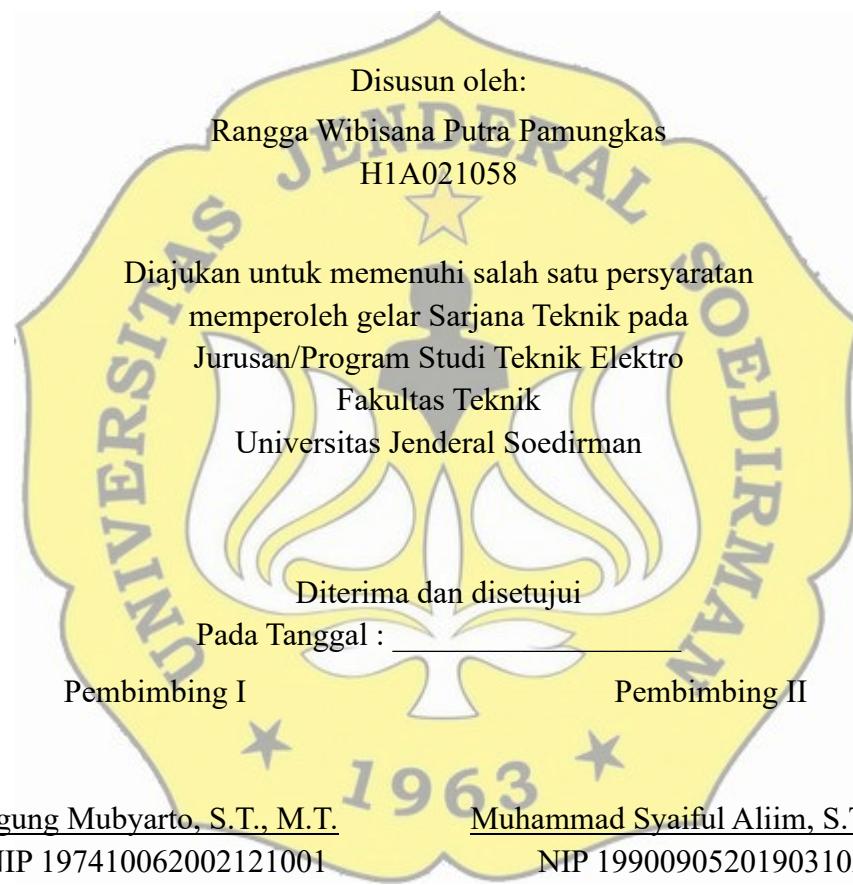
Rangga Wibisana Putra Pamungkas
H1A021058

**KEMENTERIAN PENDIDIKAN TINGGI, SAINS DAN
TEKNOLOGI
UNIVERSITAS JENDERAL SOEDIRMAN
FAKULTAS TEKNIK
JURUSAN/PROGRAM STUDI TEKNIK ELEKTRO
PURBALINGGA
2025**

HALAMAN PENGESAHAN

Tugas Akhir dengan Judul:

ANALISA PERFORMA *OPTIMIZER ALGORITMA* PEMBELAJARAN MESIN DALAM PREDIKSI IRADIASI MATAHARI BERDASARKAN LUAS PENERANGAN (LUX) DI ATAP GEDUNG JAKARTA SELATAN



Mengetahui:
Dekan Fakultas Teknik

Prof. Dr. Eng. Ir. Agus Maryoto, S.T., M.T., IPU.
NIP 197109202006041001

HALAMAN PERNYATAAN

Dengan ini saya menyatakan bahwa dalam Laporan Tugas Akhir dengan judul "***ANALISA PERFORMA OPTIMIZER ALGORITMA PEMBELAJARAN MESIN DALAM PREDIKSI IRADIASI MATAHARI BERDASARKAN LUAS PENERANGAN (LUX) DI ATAP GEDUNG JAKARTA SELATAN***" ini tidak terdapat karya yang pernah diajukan untuk memperoleh gelar kesarjanaan di suatu Perguruan Tinggi, dan sepanjang pengetahuan saya juga tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis diacu dalam naskah ini dan disebutkan dalam daftar pustaka.

Purbalingga, 24 Maret 2023

Rangga Wibisana Putra Pamungkas

NIM H1A021058

HALAMAN MOTTO DAN PERSEMBAHAN

MOTTO

“Don't be pushed around by the fears in your mind.

Be led by the dreams in your heart.”

— Roy T. Bennett, *The Light in the Heart.*

PERSEMBAHAN

Dengan penuh rasa syukur kepada Allah Swt., penulis menyadari bahwa keberhasilan dalam menyelesaikan penelitian dan laporan tugas akhir ini tidak terlepas dari dukungan, doa, dan bantuan dari berbagai pihak. Oleh karena itu, penulis mempersembahkan laporan tugas akhir ini kepada:

1. Segala puji dan syukur penulis panjatkan kepada Allah Swt. atas rahmat, karunia, dan hidayah-Nya yang telah membimbing penulis menjadi pribadi yang berpikir, berilmu, beriman, dan bersabar. Semoga keberhasilan ini menjadi langkah awal dalam meraih cita-cita besar penulis.
2. Kepada kedua orang tua tercinta, Ayahanda Maman Rusmana dan Ibunda Mini Rukmini yang telah memberikan cinta, kasih sayang, dan doa yang tiada henti sejak penulis lahir hingga saat ini. Terima kasih atas pengorbanan, dukungan, dan segala hal terbaik yang telah Ayahanda dan Ibunda berikan.
3. Kakak-kakak tercinta saya: Fery Irwan Firmansyah, Fitry Sukmawati Buana, Reny Rian Marliana, dan Krisna Rizky Rismawan. Terima kasih atas segala doa, dukungan, dan motivasi yang senantiasa menjadi penyemangat bagi penulis.
4. Kepada Bapak Agung Mubyarto, S.T., M.T. dan Bapak Muhammad Syaiful Aliim, S.T., M.T. sebagai dosen pembimbing tugas akhir yang dengan sabar

dan penuh dedikasi telah memberikan arahan, bimbingan, dan ilmu yang sangat berarti selama proses penelitian ini.

5. Para Dosen dan Staf di Jurusan Teknik Elektro, Fakultas Teknik Universitas Jenderal Soedirman. Ucapan terima kasih yang mendalam untuk seluruh bapak/ibu dosen yang telah mendidik, membimbing, dan berbagi ilmu selama masa studi.
6. Rekan-rekan magang maupun mentor di Divisi RIMS, Telkom CorpU. Terima kasih atas pengalaman berharga, dukungan, dan kebersamaan selama masa magang yang telah membantu memperkaya wawasan dan kemampuan penulis.
7. Teman-teman SEKGUS JAYA. Terima kasih atas persahabatan, kebersamaan, dan dukungan yang selalu memberikan semangat dalam perjalanan ini.
8. Untuk Dewi Ayu Pratiwi, terima kasih atas kehadirannya yang senantiasa mendukung selama proses penyelesaian laporan tugas akhir ini. Kehadiranmu adalah anugerah, yang melengkapi setiap detail perjalanan akademik ini dengan kebahagiaan dan motivasi.
9. Teman-teman Teknik Elektro Universitas Jenderal Soedirman. Terima kasih atas kenangan, kerja sama, dan bantuan selama masa studi ini.
10. Semua pihak yang tidak dapat disebutkan satu per satu. Terima kasih atas segala bentuk kebaikan, dukungan, dan doa yang telah diberikan kepada penulis. Semoga Tuhan senantiasa membalsas setiap kebaikan yang telah kalian berikan dan melimpahkan keberkahan dalam hidup kalian.

RINGKASAN

ANALISA PERFORMA *OPTIMIZER* ALGORITMA PEMBELAJARAN MESIN DALAM PREDIKSI IRADIASI MATAHARI BERDASARKAN LUAS PENERANGAN (LUX) DI ATAP GEDUNG JAKARTA SELATAN

Rangga Wibisana Putra Pamungkas

Prediksi iradiasi matahari yang akurat merupakan faktor krusial untuk optimalisasi sistem *fotovoltaik*, namun performa model prediktif sangat dipengaruhi oleh pemilihan *optimizer*. Penelitian ini bertujuan untuk menganalisis secara sistematis performa dari tiga *optimizer*—SGD, RMSProp, dan Adam—dalam melatih model *Long Short-Term Memory* (LSTM) untuk prediksi iradiasi matahari harian. Studi ini menggunakan *dataset* deret waktu univariat yang dikumpulkan dari pengukuran intensitas cahaya (lux) di atap sebuah gedung di Jakarta Selatan selama satu tahun, yang kemudian dikonversi menjadi iradiasi (W/m^2). Arsitektur *Stacked LSTM* yang konsisten digunakan sebagai variabel kontrol untuk mengisolasi dampak dari 12 skenario konfigurasi *optimizer* yang berbeda.

Evaluasi dilakukan melalui analisis kualitatif terhadap histori pelatihan dan analisis kuantitatif menggunakan metrik *Mean Absolute Error* (MAE), *Root Mean Squared Error* (RMSE), serta *Mean Absolute Percentage Error* (MAPE). Hasil penelitian menunjukkan bahwa *optimizer* Adam dengan *learning rate* rendah (0.001) menunjukkan kemampuan belajar paling superior pada data latih. Namun, pada tahap pengujian akhir, sebuah temuan menarik muncul: model dengan *optimizer* SGD secara tak terduga menghasilkan metrik *error* kuantitatif terendah. Meskipun demikian, analisis kualitatif mengungkap bahwa prediksi SGD cenderung datar dan gagal menangkap volatilitas data (*underfitting*). Sebaliknya, model dengan *optimizer* Adam (Percobaan 12; $\eta = 0.001, \beta_1 = 0.5, \beta_2 = 0.85$) terbukti paling unggul dalam meniru fluktuasi dinamis dari data aktual.

Berdasarkan sintesis antara performa kuantitatif yang kompetitif dan kemampuan generalisasi kualitatif yang superior, *optimizer* Adam dengan konfigurasi tersebut dipilih sebagai model terbaik. Penelitian ini memberikan panduan praktis mengenai pemilihan *optimizer* dan menegaskan pentingnya evaluasi kualitatif dalam pemodelan deret waktu yang bersifat stokastik.

Kata kunci : Pembelajaran Mesin, Prediksi Deret Waktu, LSTM, Lux, Iradiasi Matahari, *Optimizer*, Adam, RMSProp, SGD, Analisis Performa.

SUMMARY

PERFORMANCE ANALYSIS OF MACHINE LEARNING ALGORITHM OPTIMIZERS IN PREDICTING SOLAR IRRADIATION BASED ON ILLUMINATION (LUX) ON A BUILDING ROOF IN SOUTH JAKARTA

Rangga Wibisana Putra Pamungkas

Accurate solar irradiance forecasting is a critical factor for optimizing photovoltaic systems, yet the performance of predictive models is significantly influenced by the choice of optimizer. This research aims to systematically analyze the performance of three optimizers—SGD, RMSProp, and Adam—in training a Long Short-Term Memory (LSTM) model for daily solar irradiance prediction. The study utilizes a univariate time-series dataset collected from illuminance (lux) measurements on a building rooftop in South Jakarta over one year, which was subsequently converted to irradiance (W/m^2). A consistent Stacked LSTM architecture was used as a control variable to isolate the impact of 12 different optimizer configuration scenarios.

Evaluation was conducted through a qualitative analysis of the training history and a quantitative analysis using Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and Mean Absolute Percentage Error (MAPE) metrics. The results indicate that the Adam optimizer with a low learning rate (0.001) demonstrated the most superior learning capability on the training data. However, a compelling finding emerged during the final testing phase: the model with the SGD optimizer unexpectedly yielded the lowest quantitative error metrics. Despite this, qualitative analysis revealed that SGD's predictions were predominantly flat and failed to capture the data's volatility (underfitting). Conversely, the model with the Adam optimizer (Experiment 12; $\eta=0.001$, $\beta_1=0.5$, $\beta_2=0.85$) proved to be qualitatively superior in mimicking the dynamic fluctuations of the actual data.

Based on a synthesis of competitive quantitative performance and superior qualitative generalization, the Adam optimizer with the specified configuration was selected as the best model. This study provides practical guidance on optimizer selection and underscores the importance of qualitative evaluation in modeling stochastic time series.

Keywords : Machine Learning, Time Series Forecasting LSTM, Lux, Solar Irradiance, Optimizer, Performance Analysis.

PRAKATA

Segala puji dan syukur senantiasa penulis panjatkan ke Hadirat Allah Swt. yang telah melimpahkan rahmat, karunia dan hidayah-Nya sehingga tugas akhir ini dapat selesai disusun sebagai salah satu persyaratan untuk menyelesaikan Program Studi Sarjana Teknik Elektro Unsoed.

Penulis menyadari banyak pihak yang memberikan dukungan dan bantuan selama menyelesaikan studi dan tugas akhir ini. Oleh karena itu, sudah sepantasnya penulis dengan penuh hormat mengucapkan terima kasih kepada semua pihak yang telah membantu penulis dalam menyelesaikan tugas akhir ini yang tidak dapat penulis sebutkan satu persatu.

Penulis juga menyadari bahwa tidak ada yang sempurna, penulis masih melakukan kesalahan dalam penyusunan tugas akhir. Oleh karena itu, segala kritik dan saran dibutuhkan untuk dapat menyempurnakan tugas akhir ini. Penulis berharap semoga tugas akhir ini dapat bermanfaat bagi pembaca dan dapat dijadikan referensi demi pengembangan ke arah yang lebih baik.

Purbalingga, 23 Juli 2025
Penulis

Rangga Wibisana Putra Pamungkas

DAFTAR ISI

HALAMAN JUDUL.....	i
HALAMAN PENGESAHAN.....	ii
HALAMAN PERNYATAAN	iii
HALAMAN MOTTO DAN PERSEMBAHAN.....	iv
RINGKASAN	vi
<i>SUMMARY</i>	vii
PRAKATA.....	viii
DAFTAR ISI.....	ix
DAFTAR TABEL.....	xii
DAFTAR GAMBAR	xiii
DAFTAR LAMPIRAN	xv
DAFTAR ISTILAH DAN SINGKATAN	xvi
BAB 1 PENDAHULUAN	19
1.1 Latar Belakang	19
1.2 Rumusan Masalah	22
1.3 Batasan Masalah.....	22
1.4 Tujuan dan Manfaat	23
1.4.1 Tujuan.....	23
1.4.2 Manfaat	23
1.5 Sistematika Penulisan	24
BAB 2 TINJAUAN PUSTAKA.....	26
2.1 Penelitian Terdahulu.....	26
2.2 Iluminasi (Lux).....	29
2.3 Iradiasi Matahari	30
2.4 Hubungan antara Lux dan Iradiasi Matahari.....	32
2.5 <i>Time series</i>	32
2.6 <i>Deep Learning</i>	35
2.6.1 <i>Artificial Neural network</i>	36
2.6.2 <i>Recurrent Neural Network</i>	37
2.6.3 <i>Long Short-Term Memory</i>	38
2.6.4 Arsitektur <i>Long Short-Term Memory</i>	39
2.7 <i>Optimizer</i>	44

2.7.1	<i>Stochastic Gradient Descent</i>	48
2.7.2	<i>Root Mean Square Propagation</i>	49
2.7.3	<i>Adaptive Moment Estimation</i>	51
2.8	Metrik Evaluasi Model.....	54
2.8.1	<i>Mean Absolute Error</i>	54
2.8.2	<i>Root Mean Square Error</i>	54
2.8.3	<i>Mean Absolute Percentage Error</i>	55
2.9	Python	56
2.10	Tensorflow.....	56
2.11	Google Colab	57
BAB 3	METODE PENELITIAN.....	58
3.1	Waktu dan Tempat.....	58
3.2	Alat dan Bahan.....	58
3.2.1	Alat.....	58
3.2.2	Bahan.....	58
3.3	Alur dan Tahapan Penelitian	59
3.3.1	Tahap Persiapan	59
3.3.2	Tahap Pengumpulan Data	59
3.3.3	Tahap <i>Exploratory Data Analysis</i>	59
3.3.4	Tahap Perancangan Model	61
3.3.5	Tahap <i>Data Preprocessing</i>	62
1.	Pembersihan Data.....	62
2.	Konversi Lux Ke Iradiasi Matahari.....	62
3.	Normalisasi Data.....	63
4.	Seleksi Fitur	63
5.	Pembagian Data	64
6.	Segmentasi Data.....	64
3.3.6	Tahap Pelatihan Model.....	65
3.3.7	Tahap Evaluasi dan Pemilihan Model Terbaik.....	66
3.4	Diagram Alur Penelitian.....	67
3.5	Jadwal Penelitian.....	67
BAB 4	HASIL PENELITIAN DAN PEMBAHASAN.....	68
4.1	<i>Exploratory Data Analysis</i> (EDA)	68
4.1.1	Deskripsi Variabel	68

4.1.2	Deskripsi Statistik	69
4.1.3	<i>Missing value</i> dan <i>Outlier</i>	70
4.1.4	Distribusi Data	72
4.2	Perancangan Model.....	74
4.3	<i>Data Preprocessing</i>	77
4.3.1	Penanganan <i>Missing value</i> dan <i>Outlier</i>	77
4.3.2	Dekomposisi Deret Waktu	79
4.3.3	Seleksi Fitur	81
4.3.4	Konversi Lux ke Iradiasi.....	82
4.3.5	Normalisasi Data.....	83
4.3.6	Pembagian dan Segmentasi Data	84
4.4	Pelatihan dan Evaluasi Kinerja Model.....	85
4.4.1	Hasil Pelatihan <i>Optimizer SGD</i>	87
4.4.2	Hasil Pelatihan <i>Optimizer RMSProp</i>	91
4.4.3	Hasil Pelatihan <i>Optimizer Adam</i>	95
4.5	Pemilihan Model Terbaik	99
BAB 5	KESIMPULAN DAN SARAN.....	104
5.1	Kesimpulan	104
5.2	Saran.....	105
	DAFTAR PUSTAKA	107
	LAMPIRAN	110
	BIODATA PENULIS	114

DAFTAR TABEL

Tabel 2.1 Penelitian terdahulu.....	27
Tabel 3.1 Jadwal penelitian	67
Tabel 4.1 Deskripsi Statistik Lux	70
Tabel 4.2 Hasil pemeriksaan <i>missing values</i> pada <i>dataset</i> sudut 75°	71
Tabel 4.3 Ringkasan arsitektur model <i>Stacked LSTM</i>	76
Tabel 4.4 <i>Threshold</i> metode imputasi data.....	78
Tabel 4.5 Kombinasi percobaan berdasarkan <i>optimizer</i> dan <i>hyperparameter</i>	86
Tabel 4.6 Empat Percobaan dengan metrik terbaik.....	99

DAFTAR GAMBAR

Gambar 2.1 Metode umum prediksi iradiasi matahari[7].	26
Gambar 2.2 Komponen iradiasi matahari	31
Gambar 2.3 <i>Time series</i> iradiasi matahari di Kota Manokwari[14].....	33
Gambar 2.4 Berbagai teknik <i>time series forecasting</i>	34
Gambar 2.5 Hubungan kecerdasan buatan, <i>machine learning</i> , dan <i>deep learning</i>	35
Gambar 2.6 Arsitektur jaringan saraf feedforward	37
Gambar 2.7 Arsitektur dasar <i>recurrent neural network</i>	38
Gambar 2.8 Jaringan saraf LSTM[17], [18], [19].....	39
Gambar 2.9 Arsitektur blok sel LSTM[20].....	39
Gambar 2.10 Ilustrasi titik kerja <i>optimizer</i> pada arsitektur LSTM	44
Gambar 2.11 Diagram blok fungsi kerugian.....	45
Gambar 2.12 Ilustrasi proses <i>gradient descent</i>	46
Gambar 2.13 Ilustrasi pengaruh <i>learning rate</i>	47
Gambar 2.14 Logo Python	56
Gambar 2.15 Logo Tensorflow	57
Gambar 2.16 Logo Google Colab	57
Gambar 3.1 Contoh <i>Boxplot</i> sempurna tanpa kemiringan	60
Gambar 3.2 Ilustrasi segmentasi data	65
Gambar 3.3 Diagram alur penelitian.....	67
Gambar 4.1 Deskripsi variabel.....	68
Gambar 4.2 <i>Missing values</i> pada <i>dataset</i> sudut 75°	70
Gambar 4.3 <i>Boxplot</i> intensitas cahaya	71
Gambar 4.4 <i>Boxplot</i> distribusi data bulanan	72
Gambar 4.5 <i>Boxplot</i> distribusi data harian.....	73
Gambar 4.6 Arsitektur model <i>Stacked LSTM</i>	75
Gambar 4.7 <i>Boxplot</i> intensitas cahaya setelah imputasi	78
Gambar 4.8 Hasil dekomposisi time plot lux.....	79
Gambar 4.9 Hasil konversi lux ke iradiasi	82
Gambar 4.10 Hasil normalisasi iradiasi	83
Gambar 4.11 Pembagian <i>dataset</i>	84
Gambar 4.12 Sampel segmentasi data 7 <i>timestep</i>	85

Gambar 4.13 Grafik hasil pelatihan <i>optimizer</i> SGD percobaan 1	87
Gambar 4.14 Grafik hasil pelatihan <i>optimizer</i> SGD percobaan 2.....	88
Gambar 4.15 Grafik hasil pelatihan <i>optimizer</i> SGD percobaan 3.....	88
Gambar 4.16 Grafik hasil pelatihan <i>optimizer</i> SGD percobaan 4.....	89
Gambar 4.17 Grafik perbandingan hasil pelatihan <i>optimizer</i> SGD	89
Gambar 4.18 Perbandingan metrik hasil pelatihan <i>optimizer</i> SGD	90
Gambar 4.19 Grafik hasil pelatihan <i>optimizer</i> RMSProp percobaan 5.....	91
Gambar 4.20 Grafik hasil pelatihan <i>optimizer</i> RMSProp percobaan 6.....	91
Gambar 4.21 Grafik hasil pelatihan <i>optimizer</i> RMSProp percobaan 7.....	92
Gambar 4.22 Grafik hasil pelatihan <i>optimizer</i> RMSProp percobaan 8.....	92
Gambar 4.23 Grafik perbandingan hasil pelatihan <i>optimizer</i> RMSProp	93
Gambar 4.24 Perbandingan metrik hasil pelatihan <i>optimizer</i> RMSProp	94
Gambar 4.25 Grafik hasil pelatihan <i>optimizer</i> Adam percobaan 9	95
Gambar 4.26 Grafik hasil pelatihan <i>optimizer</i> Adam percobaan 10	95
Gambar 4.27 Grafik hasil pelatihan <i>optimizer</i> Adam percobaan 11	96
Gambar 4.28 Grafik hasil pelatihan <i>optimizer</i> Adam percobaan 12	96
Gambar 4.29 Grafik perbandingan hasil pelatihan <i>optimizer</i> Adam.....	97
Gambar 4.30 Perbandingan metrik hasil pelatihan <i>optimizer</i> Adam	98
Gambar 4.31 Grafik perbandingan metrik 4 model terbaik pada data uji.....	100
Gambar 4.32 Grafik hasil prediksi percobaan 1.....	100
Gambar 4.33 Grafik hasil prediksi percobaan 8.....	101
Gambar 4.34 Grafik hasil prediksi percobaan 11.....	101
Gambar 4.35 Grafik hasil prediksi percobaan 12.....	102
Gambar 4.36 Grafik perbandingan hasil prediksi 4 model terbaik	102

DAFTAR LAMPIRAN

Lampiran 1 Alat Pengumpulan <i>Dataset</i>	110
Lampiran 2 Kegiatan Pengukuran Lux	111
Lampiran 3 <i>Dataset</i> Pengukuran Lux	112

DAFTAR ISTILAH DAN SINGKATAN

- Adam : *Adaptive Moment Estimation* – Algoritma optimasi yang menggabungkan keunggulan dari RMSProp dan *Momentum* untuk menyesuaikan laju pembelajaran (*learning rate*) secara adaptif bagi setiap parameter model.
- ANN : *Artificial Neural Network* – Jaringan Saraf Tiruan; sebuah model komputasi yang terinspirasi dari struktur dan fungsi otak manusia, terdiri dari unit-unit pemrosesan (*neuron*) yang saling terhubung untuk tugas-tugas seperti prediksi dan klasifikasi.
- CNN : *Convolutional Neural Network* – Jenis jaringan saraf dalam yang efektif untuk pengolahan data visual atau spasial menggunakan lapisan konvolusi.
- DI : *Diffuse Irradiance* – Iradiasi Difus; komponen radiasi matahari yang telah tersebar oleh partikel di atmosfer (seperti awan dan debu) sebelum mencapai permukaan bumi.
- DNI : *Direct Normal Irradiance* – Iradiasi Normal Langsung; komponen radiasi matahari yang datang langsung dari matahari dalam garis lurus tanpa hambatan ke permukaan yang tegak lurus terhadapnya.
- EDA : *Exploratory Data Analysis* – Analisis Data Eksploratif; proses investigasi awal pada *dataset* untuk memahami karakteristik utama, menemukan pola, mendeteksi anomali, dan merangkum statistik penting, seringkali dengan bantuan visualisasi data.
- FFNN : *Feedforward Neural Network* – Arsitektur jaringan saraf tanpa *loop* umpan balik; aliran data hanya satu arah dari *input* ke *output*.
- GRU : *Gated Recurrent Unit* – Varian dari arsitektur RNN yang mirip dengan LSTM tetapi memiliki struktur gerbang yang lebih sederhana, digunakan untuk memproses data sekuensial.

- LSTM : *Long Short-Term Memory* – Memori Jangka Pendek Panjang; arsitektur jaringan saraf berulang (RNN) yang dirancang khusus untuk memproses, mempelajari, dan mengingat pola dalam data sekuensial (deret waktu) dalam jangka waktu yang panjang.
- Lux (lx) : Satuan SI (Sistem Internasional) untuk iluminasi atau tingkat pencahayaan, yang mengukur kerapatan fluks cahaya yang jatuh pada suatu permukaan per satuan luas (lumen/m^2).
- MAE : *Mean Absolute Error* – Rata-rata Kesalahan Absolut; metrik evaluasi yang mengukur rata-rata dari selisih absolut (nilai mutlak) antara nilai prediksi dan nilai aktual.
- MAPE : *Mean Absolute Percentage Error* – Rata-rata Kesalahan Persentase Absolut; metrik evaluasi yang mengukur rata-rata kesalahan prediksi dalam bentuk persentase terhadap nilai aktual, berguna untuk memahami skala kesalahan secara relatif dan mudah diinterpretasikan.
- ML : *Machine Learning* – Pembelajaran Mesin; sebuah cabang dari kecerdasan buatan di mana komputer belajar dari data untuk membuat prediksi atau keputusan tanpa diprogram secara eksplisit.
- MSE : *Mean Squared Error* – Rata-rata Kesalahan Kuadrat; fungsi kerugian (*loss function*) yang mengukur rata-rata dari kuadrat selisih antara nilai prediksi dan nilai aktual, memberikan bobot lebih besar pada kesalahan yang besar.
- PV : *Photovoltaic* – Fotovoltaik; teknologi yang mengubah energi cahaya matahari secara langsung menjadi energi listrik menggunakan sel surya.
- RMSE : *Root Mean Squared Error* – Akar Rata-rata Kesalahan Kuadrat; metrik yang merupakan akar kuadrat dari MSE. Metrik ini

memberikan bobot lebih besar pada kesalahan yang besar dan hasilnya memiliki satuan yang sama dengan data asli.

RMSProp : *Root Mean Square Propagation* – Algoritma optimasi yang menyesuaikan laju pembelajaran (*learning rate*) secara adaptif untuk setiap parameter berdasarkan rata-rata pergerakan dari kuadrat gradien, efektif untuk menangani data yang fluktuatif.

RNN : *Recurrent Neural Network* – Jaringan Saraf Berulang; jenis jaringan saraf yang dirancang untuk data sekuensial, di mana koneksi antar *neuron* membentuk siklus yang memungkinkannya memiliki "memori" dari *input* sebelumnya.

SGD : *Stochastic Gradient Descent* – Penurunan Gradien Stokastik; algoritma optimasi yang memperbarui parameter model menggunakan gradien dari satu sampel data atau sebagian kecil data (*mini-batch*) secara acak, membuatnya efisien secara komputasi untuk *dataset* besar.

SI : *Système International d'Unités* – Sistem satuan internasional yang digunakan untuk memastikan konsistensi pengukuran secara global.

BAB 1

PENDAHULUAN

1.1 Latar Belakang

Energi merupakan kebutuhan vital bagi keberlangsungan dan perkembangan komunitas manusia. Seiring dengan kemajuan peradaban, kebutuhan akan energi telah meningkat secara bertahap. Dalam beberapa dekade terakhir, lonjakan populasi dunia dan ketergantungan pada teknologi modern semakin memperbesar kebutuhan akan energi. Memenuhi kebutuhan energi saat ini memang krusial, tapi kelestarian lingkungan untuk generasi mendatang jauh lebih penting. Oleh karena itu, menyediakan energi berkelanjutan dan ramah lingkungan sama pentingnya dengan memenuhi kebutuhan energi itu sendiri[1].

Sejak Revolusi Industri, bauran energi sebagian besar negara didominasi oleh bahan bakar fosil. Di Asia Tenggara, sekitar 77% bauran energi didominasi oleh sumber energi tersebut, Sedangkan kontribusi energi terbarukan hanya 17,5% dan penggunaan biomassa tradisional 5,5%[2]. Bahan bakar fosil, seperti batu bara, minyak, dan gas sejauh ini juga merupakan kontributor terbesar terhadap perubahan iklim global serta kesehatan manusia. Bahan bakar fosil menyumbang lebih dari 75% emisi gas rumah kaca global dan hampir 90% dari semua emisi karbon dioksida. Hal ini menyebabkan setidaknya 5 juta kematian dini setiap tahunnya[3].

Di tengah meningkatnya kebutuhan energi dan dampak negatif bahan bakar fosil, energi terbarukan menawarkan solusi yang menjanjikan. Salah satu sumber energi terbarukan yang paling potensial adalah energi surya. Energi surya adalah energi yang dihasilkan dari matahari melalui sistem *Photovoltaic* (PV) atau

yang lebih dikenal dengan panel surya. Sistem PV digunakan untuk mengubah iradiasi matahari menjadi tenaga listrik, sistem ini juga dapat digunakan untuk berbagai macam aplikasi[4].

Salah satu faktor penting dalam pemanfaatan energi surya adalah prediksi iradiasi matahari. Iradiasi matahari merupakan besaran energi yang diterima bumi (W/m^2) dan menjadi parameter kunci dalam sistem *fotovoltaik*. Di sisi lain, luas penerangan atau iluminasi (lux) adalah kuantitas cahaya yang jatuh pada suatu permukaan. Meskipun iradiasi matahari adalah parameter standar untuk sistem *fotovoltaik*, pengukurannya seringkali memerlukan sensor yang lebih mahal. Oleh karena itu, penelitian ini memanfaatkan data lux yang lebih mudah diperoleh, untuk kemudian dikonversi menjadi nilai iradiasi, sehingga prediksi iradiasi matahari yang akurat dapat dikembangkan secara lebih efisien dan hemat biaya untuk memaksimalkan kinerja sistem PV[5].

Prediksi iradiasi matahari telah banyak dipelajari dalam literatur untuk mendapatkan akurasi prediksi yang lebih tinggi dan pemodelan yang lebih efisien. Upaya penelitian dalam memprediksi iradiasi matahari meliputi penggunaan model fisik, statistik, dan pembelajaran mesin (*Machine Learning/ML*) yang menggunakan citra awan, iradiasi matahari historis, serta data cuaca eksogen untuk memprediksi iradiasi matahari dalam jangka pendek maupun panjang[6].

Pemanfaatan algoritma pembelajaran mesin merevolusi prediksi energi matahari dengan memanfaatkan teknik komputasi yang canggih. Algoritma pembelajaran mesin adalah sebuah metode di mana komputer belajar dari data untuk membuat prediksi atau keputusan, tanpa diprogram secara eksplisit untuk

melakukan tugas tersebut. Algoritma pembelajaran mesin dapat secara otomatis belajar dan beradaptasi dari data dalam jumlah besar, mengidentifikasi pola, tren, dan hubungan kompleks yang mungkin tidak terlihat oleh analisa manusia[4].

Algoritma pembelajaran mesin dapat dikategorikan menjadi tiga jenis utama: *supervised learning*, *unsupervised learning*, dan *reinforcement learning*. *Long Short-Term Memory* (LSTM) merupakan salah satu algoritma paling populer yang termasuk *supervised learning*, LSTM adalah jenis arsitektur jaringan saraf tiruan yang dirancang khusus untuk memproses data sekuensial seperti data *time series*. Keunggulan LSTM terletak pada kemampuannya mengingat informasi jangka panjang dan menangkap dependensi temporal dalam data. Oleh karena itu, LSTM sangat cocok untuk memprediksi fenomena yang dipengaruhi oleh peristiwa masa lalu, seperti iradiasi matahari. Dengan memanfaatkan data historis iradiasi, LSTM dapat mempelajari pola musiman, tren jangka panjang, dan fluktuasi harian, sehingga menghasilkan prediksi yang lebih akurat

Meskipun banyak penelitian telah menerapkan LSTM untuk prediksi iradiasi, analisis mendalam mengenai dampak spesifik dari pemilihan *optimizer* terhadap performa model—terutama pada data yang memiliki fluktuasi tinggi seperti di Jakarta—masih sangat terbatas. Pemilihan *optimizer* yang tidak tepat dapat menyebabkan konvergensi model yang lambat, hasil prediksi yang kurang akurat, atau bahkan kegagalan dalam menangkap dinamika cuaca, sehingga mengurangi efektivitas sistem prediksi secara keseluruhan.

Oleh karena itu, penelitian ini bertujuan untuk menganalisis dan membandingkan performa *optimizer* yang biasa digunakan dalam prediksi iradiasi

matahari, yaitu *Adaptive Moment Estimation* (Adam), *Root Mean Square Propagation* (RMSProp), dan *Stochastic Gradient Descent* (SGD) dalam algoritma LSTM. Penelitian ini diharapkan dapat memberikan informasi mengenai performa dari setiap *optimizer*, sehingga membantu dalam pemilihan *optimizer* yang optimal dan meningkatkan performa pemodelan algoritma pembelajaran mesin LSTM.

1.2 Rumusan Masalah

1. Bagaimana penerapan algoritma pembelajaran mesin untuk memprediksi iradiasi matahari berdasarkan lux di atap Gedung Jakarta Selatan?
2. Bagaimana performa setiap *optimizer* algoritma pembelajaran mesin dalam memprediksi iradiasi matahari berdasarkan lux di atap Gedung Jakarta Selatan?
3. Manakah *optimizer* pembelajaran mesin yang paling optimal dalam memprediksi iradiasi matahari berdasarkan lux di atap Gedung Jakarta Selatan?

1.3 Batasan Masalah

Batasan berikut dibuat untuk menentukan ruang lingkup penelitian:

1. Algoritma pembelajaran mesin yang digunakan pada penelitian ini terbatas pada algoritma *Long Short-Term Memory* yang menggunakan perangkat lunak dari *Library tensorflow keras*.
2. *Optimizer* yang digunakan pada penelitian ini terbatas pada *optimizer* Adam, RMSprop, dan SGD.

3. Penelitian ini menggunakan *dataset* (himpunan data) nilai lux siang hari pada sudut 75 derajat, yang dikumpulkan dari atap gedung di Jakarta Selatan selama periode Maret 2023 hingga Februari 2024.
4. Metrik evaluasi model yang digunakan pada penelitian ini terbatas pada metrik MAE, RMSE, dan MAPE.
5. Implementasi algoritma menggunakan bahasa pemrograman Python didalam platform Google Colab.

1.4 Tujuan dan Manfaat

1.4.1 Tujuan

Adapun tujuan dalam penyusunan tugas akhir ini, selain untuk memperoleh gelar kesarjanaan diantaranya adalah:

1. Mengetahui bagaimana penerapan algoritma pembelajaran mesin untuk memprediksi iradiasi matahari berdasarkan lux di atap Gedung Jakarta Selatan.
2. Menganalisis performa setiap *optimizer* algoritma pembelajaran mesin dalam memprediksi iradiasi matahari berdasarkan lux di atap Gedung Jakarta Selatan.
3. Mengetahui *optimizer* pembelajaran mesin yang paling optimal dalam memprediksi iradiasi matahari berdasarkan lux di atap Gedung Jakarta Selatan.

1.4.2 Manfaat

Penelitian ini diharapkan dapat memberikan informasi komprehensif serta perbandingan performa antara *optimizer* Adam, RMSprop, dan SGD dalam

memprediksi iradiasi matahari. Hasil dari penelitian ini diharapkan dapat menjadi referensi yang berharga bagi pengembangan model prediksi iradiasi matahari selanjutnya. Selain itu, temuan penelitian ini dapat menjadi panduan bagi para insinyur atau pengembang sistem PV di Indonesia dalam memilih *optimizer* yang paling efisien untuk implementasi model prediksi di lapangan, sehingga dapat meningkatkan keandalan operasional dan manajemen energi pada instalasi panel surya.

1.5 Sistematika Penulisan

Dalam penulisan laporan tugas akhir ini, dibuat sistematika penulisan sebagai berikut.

BAB I PENDAHULUAN

Bab ini berisi uraian tentang latar belakang penelitian, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, dan sistematika penulisan laporan.

BAB II TINJAUAN PUSTAKA

Bab ini memaparkan tinjauan pustaka tentang teori-teori fundamental yang mendasari penelitian ini. Teori-teori yang dikaji meliputi konsep luas penerangan, iradiasi matahari, algoritma pembelajaran mesin, dan temuan penelitian terdahulu yang relevan. Pemahaman menyeluruh mengenai teori-teori ini menjadi landasan penting dalam pelaksanaan penelitian dan membantu penulis dalam menyelesaikan tugas akhir.

BAB III METODE PENELITIAN

Bab ini menjelaskan tentang metode dan langkah-langkah yang dilakukan untuk melakukan penelitian ini seperti waktu dan tempat penelitian, alat dan bahan, tahapan penelitian, diagram alur penelitian, dan jadwal penelitian.

BAB IV HASIL DAN PEMBAHASAN

Bab ini memaparkan hasil analisis performa dari model pembelajaran mesin yang telah dirancang dengan menggunakan tiga *optimizer* berbeda. Evaluasi performa model dilakukan dengan menggunakan metrik evaluasi MAPE, RMSE, dan MAE.

BAB V PENUTUP

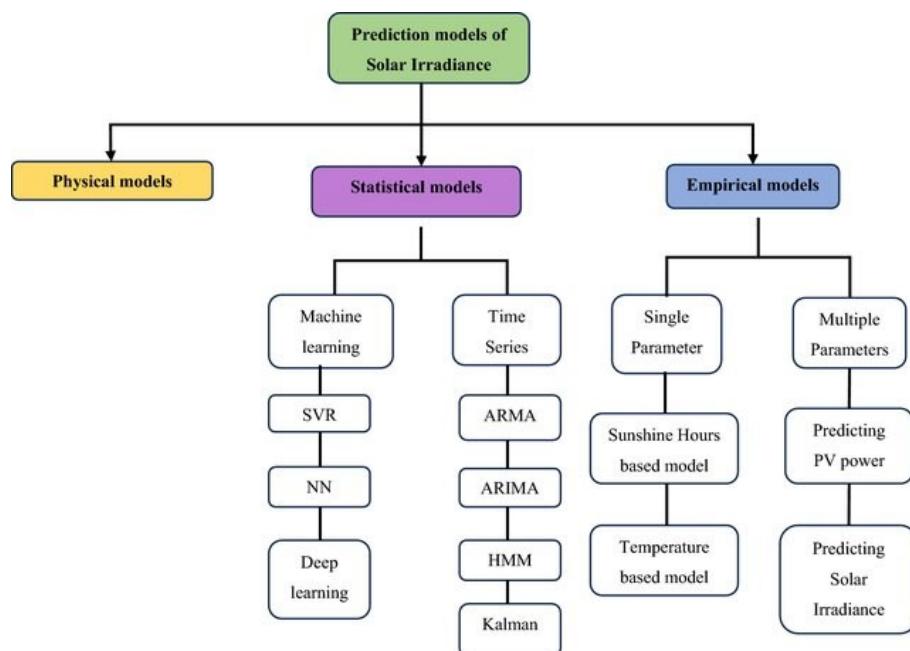
Bab ini menguraikan kesimpulan yang ditarik dari hasil penelitian, serta saran-saran untuk pengembangan penelitian selanjutnya.

BAB 2

TINJAUAN PUSTAKA

2.1 Penelitian Terdahulu

Prediksi iradiasi matahari telah banyak dipelajari dalam literatur untuk mendapatkan akurasi prediksi yang lebih tinggi dan pemodelan yang lebih efisien. Upaya penelitian dalam memprediksi iradiasi matahari meliputi penggunaan model fisik, statistik, dan pembelajaran mesin. Gambar 2.1 berikut mengilustrasikan metode umum yang digunakan untuk memprediksi iradiasi matahari.



Gambar 2.1 Metode umum prediksi iradiasi matahari[7].

Untuk memposisikan penelitian ini dalam lanskap keilmuan yang ada, telah dilakukan tinjauan terhadap beberapa studi terdahulu yang relevan. Tinjauan ini berfokus pada penelitian yang menerapkan algoritma *deep learning*, khususnya *Long Short-Term Memory* (LSTM) dan variannya, untuk prediksi iradiasi matahari. Analisis ini bertujuan untuk mengidentifikasi metode yang umum digunakan,

variabel yang berpengaruh, serta celah penelitian yang masih dapat dieksplorasi lebih lanjut. Ringkasan dari penelitian-penelitian kunci disajikan pada Tabel 2.1.

Tabel 2.1 Penelitian terdahulu

No.	Judul Penelitian	Dataset Yang Digunakan	Tujuan Penelitian	Hasil Penelitian
1.	<i>Day-ahead to week-ahead solar irradiance prediction using convolutional Long Short-Term Memory networks</i> (Cheng dkk., 2021)	- Data iradiasi matahari per jam dan data cuaca selama 14 bulan dari dua lokasi di Taiwan.	<ul style="list-style-type: none"> - Mengembangkan dan menguji dua skema model LSTM, yaitu Conv-LSTM-A dan Conv-LSTM-B, serta membandingkannya dengan beberapa model lain. - Menganalisis Pengaruh penambahan data cuaca (suhu, kelembaban, curah hujan) terhadap akurasi prediksi. - Menguji pengaruh panjang jendela <i>input</i>, panjang filter dan jumlah filter terhadap performa model. 	<ul style="list-style-type: none"> - Model Conv-LSTM-B-R secara konsisten mengungguli model lainnya dengan akurasi terbaik. - Menambahkan fitur cuaca secara umum meningkatkan akurasi prediksi. - Model dapat dilatih hanya dengan 2 bulan data dan tetap memberikan hasil akurat.
2.	<i>Hour-Ahead Solar Irradiance Forecasting Using Multivariate Gated Recurrent Units</i> (Wojtkiewicz dkk., 2019)	- Data iradiasi matahari per jam dan data cuaca selama 11 tahun di Phoenix, Arizona.	<ul style="list-style-type: none"> - Menguji dan mengembangkan tiga model GRU (univariat dan multivariat dengan tambahan variabel eksogen) - Membandingkan GRU dengan LSTM. - Menganalisis dampak tutupan awan terhadap akurasi prediksi. 	<ul style="list-style-type: none"> - LSTM lebih akurat daripada GRU, tetapi GRU lebih cepat dalam komputasi. - Penambahan variabel cuaca meningkatkan akurasi hingga 18,33%. - Model kesulitan memprediksi perubahan mendadak akibat tutupan awan.
3.	<i>Day-Ahead Solar Irradiance Forecasting for Microgrids Using a Long Short-Term Memory Recurrent Neural Network A</i>	- Enam dataset dari empat negara berbeda (Jerman, AS, Swiss, dan Korea Selatan) yang berisi data iradiasi, suhu	<ul style="list-style-type: none"> - Mengembangkan model prediksi day-ahead menggunakan LSTM-RNN. - Tidak bergantung pada data historis irradiance, hanya 	<ul style="list-style-type: none"> - LSTM-RNN lebih akurat dibandingkan FFNN dan model persistence dengan RMSE 60.31 W/m².

	<i>Deep Learning Approach (Husein dan Chung, 2019)</i>	udara, kelembaban, kecepatan angin, arah angin, curah hujan, serta tutupan awan.	<ul style="list-style-type: none"> - menggunakan data cuaca. - Menguji model di berbagai lokasi dengan kondisi iklim yang berbeda. 	<ul style="list-style-type: none"> - Model dapat digunakan tanpa data irradiance historis. - Prediksi yang lebih akurat meningkatkan efisiensi energi microgrid hingga 2% per tahun.
4.	<i>Hourly day-ahead solar irradiance prediction using weather forecasts by LSTM (Qing dan Niu, 2018)</i>	<ul style="list-style-type: none"> - Data iradiasi dan data cuaca selama 30 bulan dari pembangkit listrik tenaga surya di Pulau Santiago, Tanjung Verde. 	<ul style="list-style-type: none"> - Menggunakan LSTM dengan data prakiraan cuaca untuk prediksi hourly day-ahead. - Menguji structured <i>output</i> prediction untuk meningkatkan akurasi. - Menganalisis pengaruh jumlah data latih terhadap performa model. 	<ul style="list-style-type: none"> - LSTM mengurangi <i>error</i> hingga 42,9% dibandingkan BPNN dalam <i>dataset</i> besar. - Structured <i>output</i> prediction meningkatkan akurasi. - Model lebih stabil dan memiliki generalisasi lebih baik dibandingkan metode lain.
5.	<i>Solar Irradiance Forecasting Using Deep Neural networks (Alzahrani dkk., 2017)</i>	<ul style="list-style-type: none"> - Data iradiasi matahari per jam beresolusi tinggi (dicatat pada frekuensi 100 Hz) diperoleh dari sumber daya alam di Kanada. - <i>Dataset</i> ini mencakup empat hari, yang mewakili kondisi cuaca berbeda 	<ul style="list-style-type: none"> - Menggunakan DRNN untuk meningkatkan akurasi prediksi dibandingkan metode konvensional. - Mengatasi tantangan big data dalam prediksi irradiance. - Membandingkan DRNN dengan FNN dan SVR. 	<ul style="list-style-type: none"> - DRNN memiliki RMSE sebesar 0.086, lebih baik dari FNN (0.16) dan SVR (0.11). - Model lebih efisien dalam menangani big data. - DRNN lebih stabil dalam berbagai kondisi cuaca dan unggul dalam prediksi short-term irradiance.

Dari tinjauan pada Tabel 2.1 dan analisis mendalam terhadap penelitian-penelitian tersebut, terlihat sebuah konsensus bahwa model sekuensial berbasis *Recurrent Neural Network* (RNN) secara konsisten menunjukkan performa superior. Studi oleh Cheng dkk. (2021) dan Alzahrani dkk. (2017), misalnya, berfokus pada inovasi arsitektur dengan mengintegrasikan *Convolutional Neural Network* (Conv-LSTM) atau membangun jaringan yang dalam (*Deep RNN*) untuk

mengekstraksi fitur temporal secara lebih efektif. Kelompok penelitian lain lebih menekankan pada strategi penggunaan data *input*. Husein dan Chung (2019) serta Qing dan Niu (2018) berhasil membuktikan bahwa model LSTM dapat dilatih secara efektif hanya dengan menggunakan data prakiraan cuaca eksternal, sebuah pendekatan yang sangat berharga untuk lokasi tanpa data iradiasi historis. Sementara itu, Wojtkiewicz dkk. (2019) secara komparatif menunjukkan bahwa penambahan variabel cuaca secara signifikan meningkatkan akurasi, baik pada model LSTM maupun *Gated Recurrent Unit* (GRU).

Namun, terlihat jelas bahwa fokus utama dari penelitian-penelitian tersebut adalah pada perbandingan arsitektur model atau rekayasa fitur *input*. Meskipun setiap penelitian menggunakan sebuah *optimizer* (misalnya Adam pada penelitian Cheng dkk.), analisis komparatif yang sistematis mengenai dampak pemilihan *optimizer* (seperti Adam, RMSProp, dan SGD) sebagai variabel independen utama terhadap stabilitas pelatihan dan akurasi akhir prediksi LSTM pada data iradiasi matahari masih sangat terbatas. Cela inilah yang menjadi fokus utama dalam penelitian ini, di mana performa dari berbagai *optimizer* akan dianalisis secara mendalam untuk menentukan konfigurasi yang paling optimal.

2.2 Iluminasi (Lux)

Illuminating Engineering Society of North America (IESNA) mendefinisikan iluminasi (E), yang dikenal juga sebagai luas penerangan, sebagai kerapatan fluks cahaya yang jatuh pada suatu permukaan. Dengan kata lain, iluminasi adalah ukuran kuantitatif jumlah cahaya yang diterima oleh suatu area tertentu. Secara matematis, iluminasi dapat dihitung dengan membagi fluks cahaya

(lumen) dengan luas permukaan (meter persegi). Satuan SI untuk iluminasi adalah lux (lx atau $lumens/m^2$), sedangkan satuan non-SI yang umum digunakan adalah *foot-candle* (*ft-c*). Satu *foot-candle* setara dengan satu lumen per kaki persegi[8].

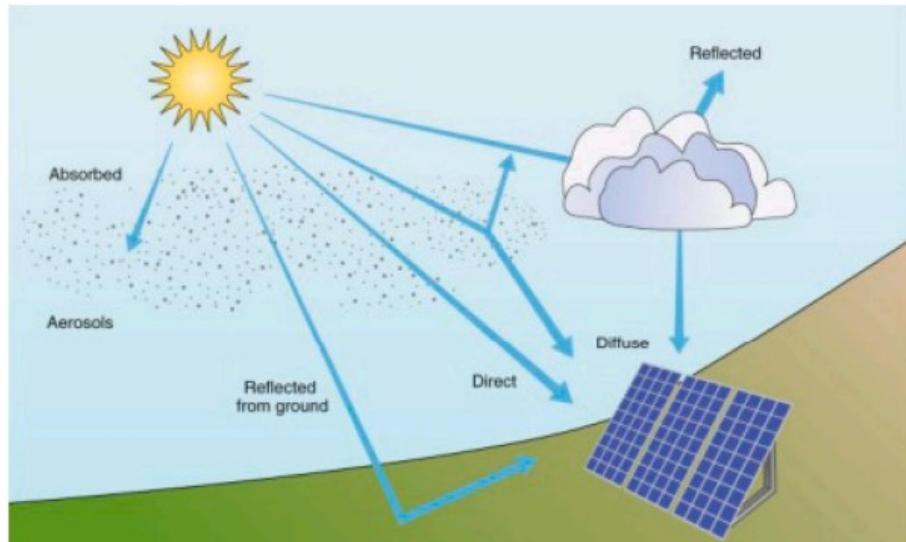
Konsep iluminasi mencakup cahaya langsung yang berasal dari sumber cahaya (seperti lampu) dan cahaya tidak langsung yang dihasilkan dari pantulan pada permukaan benda di sekitar. Faktor-faktor yang mempengaruhi iluminasi meliputi jenis sumber cahaya, jarak antara sumber cahaya dengan permukaan, sudut datang cahaya, dan sifat reflektansi permukaan. Semakin tinggi tingkat iluminasi, potensial energi listrik yang dapat dihasilkan oleh panel surya juga semakin besar.[9].

2.3 Iradiasi Matahari

Iradiasi matahari, yang sering juga disebut insolasi, adalah besaran fisika yang mengukur jumlah energi matahari yang jatuh pada suatu permukaan dalam satuan waktu tertentu. Secara sederhana, iradiasi matahari menunjukkan intensitas iradiasi matahari yang mencapai permukaan bumi. Satuan SI yang umum digunakan untuk mengukur iradiasi matahari adalah watt per meter persegi (W/m^2). Besaran ini mengindikasikan jumlah daya yang diterima setiap meter persegi permukaan dalam satu detik. Semakin tinggi nilai iradiasi, semakin besar pula potensi energi listrik yang dapat dihasilkan oleh sistem tenaga surya. Faktor-faktor seperti lokasi geografis, waktu, kondisi cuaca, dan kemiringan permukaan berpengaruh signifikan terhadap variasi nilai iradiasi matahari.

Gambar 2.2 mengilustrasikan komponen-komponen utama iradiasi matahari yang mencapai permukaan bumi. Secara umum, iradiasi matahari dapat

dibagi menjadi tiga jenis utama: *Direct Normal Irradiance*, *Diffuse Irradiance*, *Reflected Irradiance*.



Gambar 2.2 Komponen iradiasi matahari

Direct Normal Irradiance (DNI) merupakan komponen iradiasi yang datang langsung dari matahari tanpa mengalami hambatan atau pembelokan oleh atmosfer. DNI diukur pada permukaan yang tegak lurus terhadap arah datangnya sinar matahari. Besarnya DNI dipengaruhi oleh jarak antara bumi dan matahari serta kondisi atmosfer seperti keberadaan awan.

Diffuse Irradiance (DI) adalah komponen iradiasi yang telah tersebar ke segala arah akibat interaksi dengan partikel-partikel di atmosfer, seperti uap air, debu, dan aerosol. Iradiasi difus ini mencapai permukaan bumi dari seluruh bagian langit.

Reflected Irradiance merupakan komponen iradiasi yang dipantulkan oleh permukaan bumi, seperti tanah, air, atau bangunan. Besarnya iradiasi reflektif dipengaruhi oleh sifat permukaan (albedo) dan sudut datang sinar matahari. Ketiga

komponen ini saling melengkapi dan berkontribusi terhadap total iradiasi matahari yang diterima oleh panel surya[10], [11].

2.4 Hubungan antara Lux dan Iradiasi Matahari

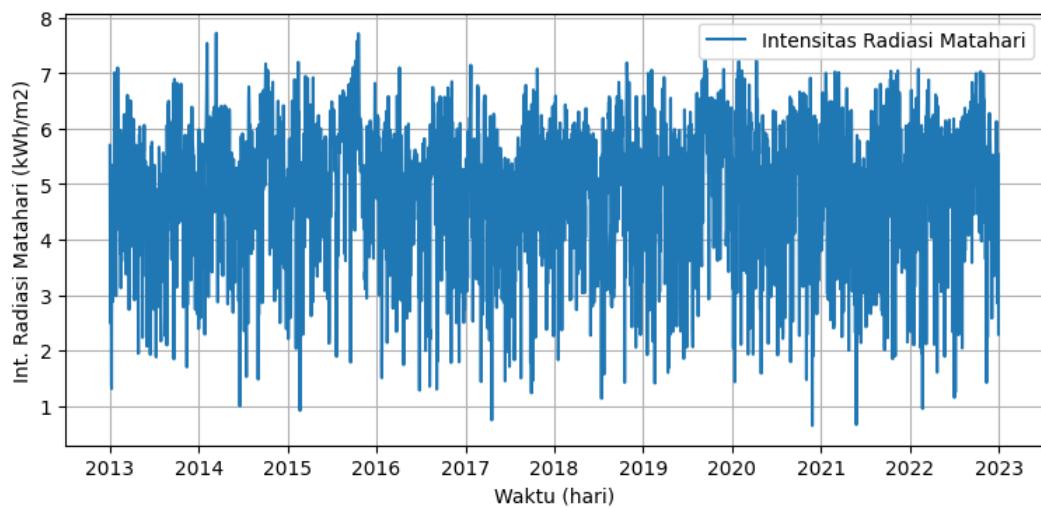
Berbagai studi menunjukkan adanya variasi yang signifikan dalam nilai hubungan antara lux dan W/m^2 , yang disebabkan oleh perbedaan dalam spektrum cahaya sumber, metode pengukuran, serta kondisi lingkungan. Namun, penelitian terbaru oleh Michael dkk. (2020) telah memberikan wawasan penting mengenai keterkaitan antara kedua bobot ini.

Dengan mengombinasikan pendekatan teoretis dan eksperimen laboratorium, studi tersebut menemukan adanya korelasi yang kuat antara nilai lux dan W/m^2 . Hasil penelitian menunjukkan bahwa dalam kondisi tertentu, 1 W/m^2 setara dengan $116 \pm 3 \text{ lx}$ untuk sumber cahaya dari simulator surya berbasis LED di dalam ruangan, sedangkan untuk sinar matahari alami di luar ruangan, nilai konversinya adalah sekitar $122 \pm 1 \text{ lx}$. Berdasarkan temuan tersebut, penelitian ini akan menggunakan nilai konversi $1 \text{ W/m}^2 \approx 122 \pm 1 \text{ lx}$ sebagai acuan utama[12].

2.5 Time series

Time series atau deret waktu adalah rangkaian observasi yang disusun berdasarkan urutan waktu. Titik-titik data dalam *time series* umumnya dicatat pada interval waktu yang konstan dan berurutan. Dengan kata lain, *time series* merupakan urutan data yang diindeks atau digambarkan menurut waktu. Secara sederhana, data *time series* merupakan kumpulan data yang merekam suatu objek atau fenomena secara berkala dari waktu ke waktu[13].

Data *time series* banyak dijumpai dalam kehidupan sehari-hari, seperti harga komoditas, harga saham, serta data iradiasi matahari. Iradiasi matahari dapat direpresentasikan sebagai data *time series* karena pencatatannya dilakukan secara periodik (harian, bulanan, atau per jam), sehingga membentuk rangkaian observasi yang teratur. Contoh data *time series* harian iradiasi matahari ditunjukkan pada Gambar 2.3.



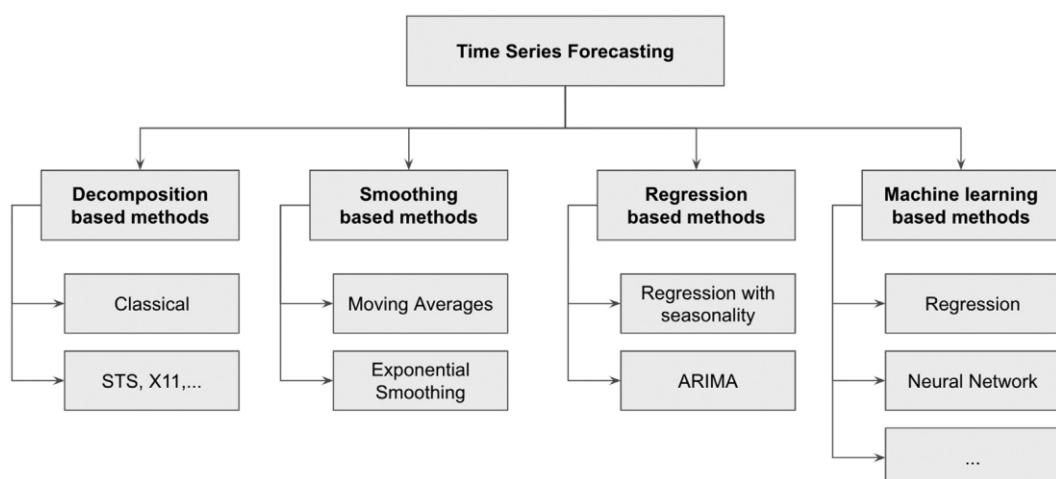
Gambar 2.3 Time series iradiasi matahari di Kota Manokwari[14].

Time series iradiasi matahari menunjukkan bagaimana energi matahari yang diterima permukaan bumi berfluktuasi sepanjang waktu. Analisis terhadap data ini dapat mengungkap pola musiman, tren jangka panjang, dan variasi acak, yang sangat penting dalam berbagai aplikasi seperti perencanaan energi surya, manajemen sistem kelistrikan, hingga studi perubahan iklim.

Secara umum, analisis *time series* terbagi menjadi dua pendekatan utama, yaitu analisis deskriptif (*time series analysis*) dan peramalan prediktif (*time series forecasting*). *Time series analysis* adalah proses mengekstraksi informasi yang bermakna dari data deret waktu, seperti tren jangka panjang, pola musiman, serta

komponen residual (kesalahan atau *noise*). Proses ini sering melibatkan dekomposisi data menjadi komponen-komponen dasar untuk memahami struktur data secara menyeluruh. Agar hasil analisis valid dan akurat, diperlukan jumlah data yang cukup besar agar pola-pola dalam data dapat dikenali dengan konsisten dan tidak terpengaruh oleh nilai pencilan (*outlier*).

Sedangkan *time series forecasting* merupakan proses memprediksi nilai-nilai masa depan berdasarkan pola historis dan data sebelumnya. Teknik ini merupakan salah satu pendekatan prediktif tertua dalam analitik data dan banyak digunakan dalam berbagai bidang karena memiliki landasan statistik yang kuat dan mampu mendukung pengambilan keputusan jangka pendek maupun jangka panjang. Secara umum, metode *time series forecasting* dapat dikelompokkan ke dalam empat kategori utama, yaitu: Teknik berbasis dekomposisi (*decomposition-based*), Teknik perataan (*smoothing-based*), Teknik berbasis regresi (*regression-based*), Teknik berbasis *Machine Learning*. Klasifikasi teknik peramalan ini dapat dilihat pada Gambar 2.4.

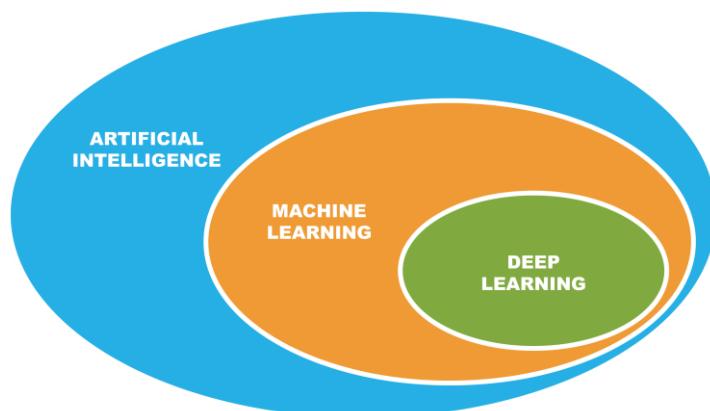


Gambar 2.4 Berbagai teknik time series forecasting

Salah satu pendekatan modern yang banyak digunakan dalam *time series forecasting* waktu adalah metode *Machine Learning* dan *deep learning*, khususnya model *Long Short-Term Memory* (LSTM). LSTM merupakan arsitektur dari jaringan saraf berulang (*Recurrent Neural Network* atau RNN) yang dirancang untuk memproses data berurutan dan memiliki kemampuan untuk menyimpan serta memanfaatkan informasi jangka panjang secara efektif.

Untuk mengimplementasikan model LSTM, data deret waktu perlu diubah ke dalam format *windowing*, yaitu membagi data ke dalam jendela waktu tertentu, di mana beberapa nilai sebelumnya digunakan sebagai *input* untuk memprediksi nilai berikutnya. Pendekatan ini serupa dengan model autoregresif, akan tetapi LSTM memiliki keunggulan dalam mengenali pola-pola nonlinier serta ketergantungan jangka panjang yang lebih kompleks dalam data[13].

2.6 Deep Learning



Gambar 2.5 Hubungan kecerdasan buatan, machine learning, dan deep learning

Deep learning merupakan salah satu cabang dari kecerdasan buatan yang termasuk dalam ranah pembelajaran mesin (*machine learning*). Pendekatan ini memanfaatkan jaringan saraf tiruan (*artificial neural networks*) yang terinspirasi

oleh cara kerja otak manusia untuk secara otomatis mengekstraksi fitur-fitur penting dari *dataset* berskala besar. Melalui penggunaan banyak lapisan pemrosesan, *deep learning* mampu mempelajari representasi data pada berbagai tingkat kompleksitas.

Proses pembelajaran dalam *deep learning* didukung oleh algoritma *backpropagation*, yang memungkinkan penyesuaian bobot internal secara bertahap guna mengenali pola-pola kompleks dalam data yang masif. Beberapa arsitektur *deep learning* yang umum digunakan antara lain *Feedforward Neural Network* (FFNN), *Recurrent Neural Network* (RNN), *Deep Belief Network* (DBN), dan *Restricted Boltzmann Machine* (RBM), dengan FFNN dan RNN sebagai model yang paling sering diaplikasikan.

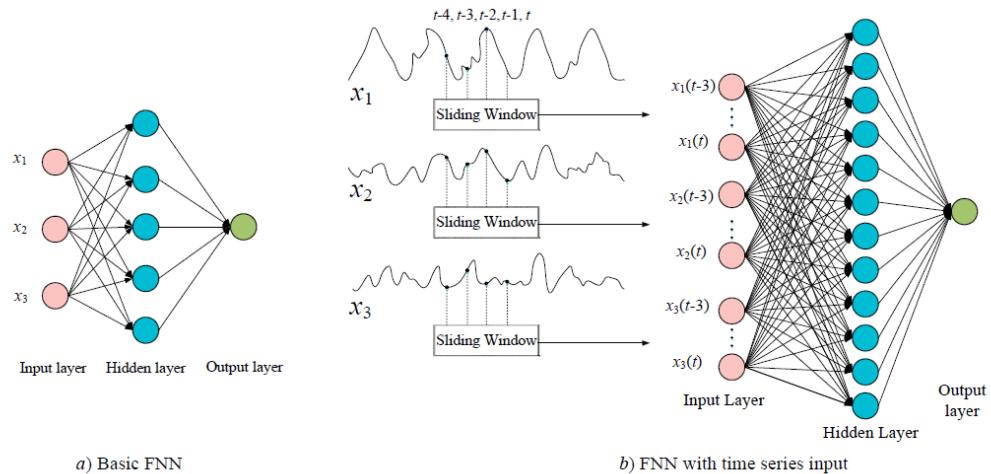
Varian dari FFNN seperti *Convolutional Neural Network* (CNN) sangat efektif dalam mengolah data visual seperti citra dan video, serta data audio. Sedangkan *Long Short-Term Memory* (LSTM), yang merupakan jenis RNN, sangat cocok digunakan untuk memproses data berurutan seperti teks, suara, maupun data *time series*[15].

2.6.1 Artificial Neural network

Artificial Neural Network (ANN) atau jaringan saraf tiruan merupakan metode komputasi yang meniru prinsip kerja otak manusia melalui sejumlah besar unit pemrosesan buatan (*neuron*). Unit-unit ini dihubungkan oleh koneksi yang memiliki bobot, menyerupai cara kerja sinapsis pada sistem saraf biologis. ANN banyak diaplikasikan di berbagai bidang, seperti pembelajaran mesin, pemrosesan citra, pemrosesan sinyal, dan ilmu komputer. Selain itu, ANN juga telah

dimanfaatkan untuk pengendalian konverter elektronika daya dalam sistem PV dan untuk pemodelan sumber energi terbarukan.

Struktur dasar ANN umumnya terdiri atas tiga komponen utama: *neuron* fungsi aktivasi, dan bias. *Neuron* dapat dikategorikan sebagai *neuron* masukan (*input neuron*), *neuron* tersembunyi (*hidden neuron*), atau *neuron* keluaran (*output neuron*). Gambar 2.6 (a) menampilkan struktur jaringan saraf *feedforward* sederhana dengan satu lapisan tersembunyi. Sedangkan Gambar 2.6 (b) mengilustrasikan jaringan *feedforward* dengan masukan yang diterapkan menggunakan teknik *sliding window*[16].

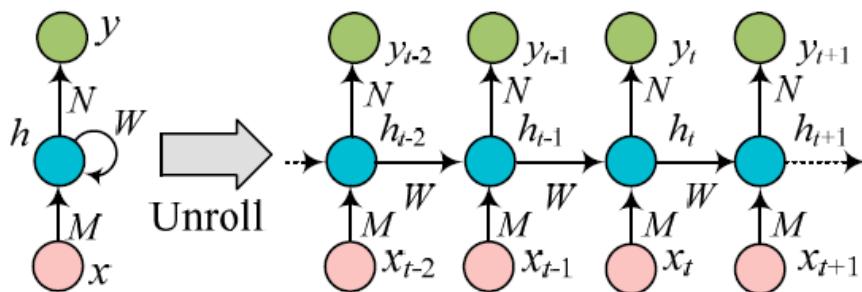


Gambar 2.6 Arsitektur jaringan saraf *feedforward*

2.6.2 Recurrent Neural Network

Recurrent Neural Network (RNN) atau jaringan saraf tiruan berulang adalah jenis ANN yang secara khusus dirancang untuk memproses data sekuensial atau berurutan. Karakteristik utama RNN adalah bahwa keluaran pada suatu *timesteps* bergantung pada masukan saat itu dan informasi dari *timesteps* sebelumnya. Keunggulan mendasar RNN terletak pada keberadaan memori

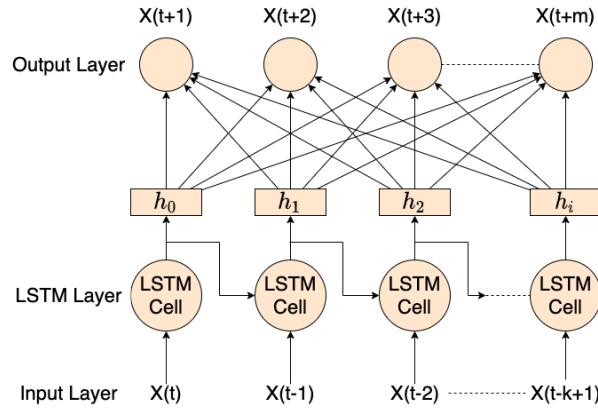
internalnya, yang memungkinkan jaringan untuk menyimpan informasi relevan dari perhitungan-perhitungan sebelumnya guna memproses urutan data dengan mempertimbangkan konteks historis. Gambar 2.7 menunjukkan struktur dasar RNN, di mana *hidden neuron* (h) menerima umpan balik dari *neuron* pada *timesteps* sebelumnya melalui bobot koneksi (w). Ketika representasi jaringan ini dibuka (*unfolded*), terlihat bahwa setiap *neuron* tersembunyi menerima masukan dari keluaran proses pada *timesteps* sebelumnya, membentuk struktur yang efektif untuk pemrosesan data sekuensial[16].



Gambar 2.7 Arsitektur dasar recurrent neural network

2.6.3 Long Short-Term Memory

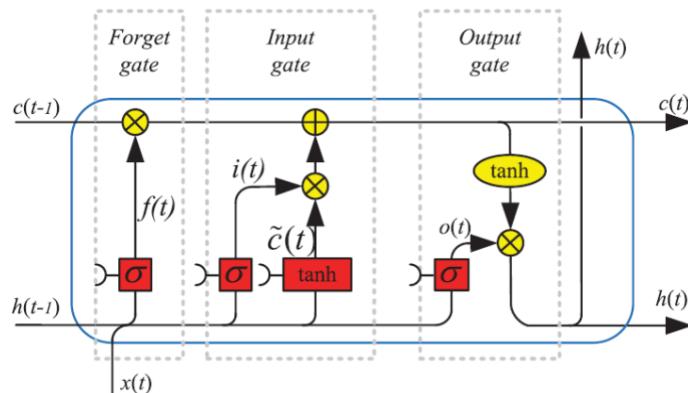
Long Short-Term Memory (LSTM) adalah jenis algoritma RNN yang dirancang secara spesifik untuk mengatasi permasalahan gradien menghilang (*vanishing gradient*) dalam data sekuensial dengan dependensi jangka panjang. Permasalahan ini sering kali muncul pada RNN tradisional ketika berupaya mempelajari ketergantungan antara elemen data yang terpisah jauh dalam urutan. LSTM mengatasi kendala ini dengan mengimplementasikan mekanisme sel memori (*memory cell*) yang memungkinkan jaringan untuk "mengingat" informasi relevan dalam rentang waktu yang signifikan, sebagaimana diilustrasikan pada Gambar 2.8.



Gambar 2.8 Jaringan saraf LSTM[17], [18], [19].

2.6.4 Arsitektur *Long Short-Term Memory*

Arsitektur LSTM terdiri atas beberapa komponen utama, yaitu sel memori (*memory cell*) dan tiga gerbang (*gate*) utama. Sel memori merupakan inti dari arsitektur LSTM yang berfungsi sebagai unit penyimpanan informasi. Sel memori memiliki kemampuan untuk mempertahankan informasi dalam beberapa langkah waktu (*timesteps*), bahkan ketika terdapat informasi baru yang masuk. Tiga gerbang utama pada LSTM adalah gerbang masukan (*input gate*), gerbang pelupa (*forget gate*), dan gerbang keluaran (*output gate*). Gerbang-gerbang ini berfungsi untuk mengontrol aliran informasi yang masuk dan keluar dari sel memori.



Gambar 2.9 Arsitektur blok sel LSTM[20].

1. Forget gate (f_t)

Forget gate bertugas menentukan informasi mana yang perlu "dilupakan" atau dihapus dari memori sel. Komponen ini merupakan lapisan sigmoid yang menerima dua masukan: *hidden state* pada waktu sebelumnya ($t - 1$) dan *input* pada waktu t . Kedua masukan tersebut digabungkan, kemudian diproses menggunakan fungsi aktivasi sigmoid. Karena sifat fungsi sigmoid, *output* dari *gate* ini berada dalam rentang 0 hingga 1. Jika $f_t = 0$, maka keadaan (*state*) sebelumnya akan dihapus, sedangkan jika $f_t = 1$, keadaan sebelumnya dipertahankan sepenuhnya.

Secara intuitif, *forget gate* bekerja dengan "melihat" informasi dari langkah sebelumnya (h_{t-1}) dan masukan saat ini (x_t) untuk memutuskan bagian mana dari memori jangka panjang yang sudah tidak relevan dan perlu dibuang. Persamaan untuk f_t adalah sebagai berikut:

$$f_t = \sigma(w_f \cdot h_{t-1} + w_f \cdot x_t + b_f) \quad (2.1)$$

dengan:

- f_t = *forget gate* pada *timestep* ke- t
- σ = fungsi aktivasi sigmoid
- w_f = bobot untuk *forget gate*
- x_t = *input* data untuk *timestep* ke- t .
- h_{t-1} = *hidden state* pada *timestep* ke- $(t - 1)$
- b_f = bias pada *forget gate*

2. Input Gate (i_t)

Input gate berfungsi menentukan informasi baru yang akan disimpan dalam *memory cell*. Proses ini mencakup pengolahan informasi untuk memutuskan data mana yang akan diperbarui dan disimpan dalam status sel c_t . Secara intuitif, *input gate* bekerja seperti filter dua tahap. Tahap pertama adalah lapisan sigmoid

yang memutuskan 'seberapa penting' setiap informasi baru yang masuk. Tahap kedua adalah lapisan *tanh* yang menciptakan 'kandidat' informasi baru itu sendiri. Keduanya kemudian digabungkan untuk memperbarui memori sel hanya dengan informasi baru yang dianggap relevan.

Peran utama *input gate* adalah mendeteksi bagian sel yang perlu diperbarui dan memastikan informasi yang relevan diteruskan ke tahap berikutnya. *Gate* ini menerima *output* dari waktu sebelumnya dan *input* baru, lalu memprosesnya melalui lapisan sigmoid. Hasilnya berupa nilai dalam rentang 0 hingga 1, yang menentukan sejauh mana informasi tersebut akan diperbarui. Rumus dari i_t adalah sebagai berikut:

$$i_t = \sigma (w_i \cdot h_{t-1} + w_i \cdot x_t + b_i) \quad (2.2)$$

dengan:

- i_t = *input gate* untuk *timestep* ke- t .
- σ = fungsi aktivasi sigmoid .
- w_i = bobot pada *input gate*.
- x_t = *input* data untuk *timestep* ke- t .
- h_{t-1} = *hidden state* pada *timestep* ke- $(t - 1)$.
- b_i = bias untuk *input gate*.

Selanjutnya, *input gate* memperbarui informasi dengan menghasilkan vektor kandidat baru \tilde{c}_t . Lapisan ini menerapkan fungsi tangen hiperbolik pada kombinasi *input* saat ini dan *output* sebelumnya. Hasilnya adalah vektor kandidat yang mewakili informasi baru yang akan ditambahkan ke *state*. Proses pembaruan *state* dilakukan menggunakan rumus berikut:

$$\tilde{c}_t = \tanh(w_c \cdot h_{t-1} + w_c \cdot x_t + b_c) \quad (2.3)$$

dengan:

- $\tilde{c}_{(t)}$ = kandidat *cell state* baru pada *timestep* ke- t

\tanh = fungsi hiperbolik \tan
 w_c = bobot pada *cell state*
 h_{t-1} = *hidden state* pada *timestep* ke- $(t - 1)$
 x_t = *input* data untuk *timestep* ke- t .
 b_c = bias untuk *cell state*

3. Cell State (c_t)

Pada tahap ini, nilai *cell state* sebelumnya (c_{t-1}) diperbarui menjadi nilai *cell state* baru (c_t). Proses ini melibatkan perkalian *state* sebelumnya dengan nilai *forget gate*, yang menentukan informasi mana yang akan dipertahankan. Selanjutnya, hasil tersebut ditambahkan dengan nilai kandidat baru yang diperoleh melalui *input gate*. Proses ini diatur oleh persamaan berikut:

$$c_t = f_t \cdot c_{t-1} + i_t \cdot \tilde{c}_t \quad (2.4)$$

dengan:

c_t = *cell state* baru pada *timestep* ke- t
 c_{t-1} = nilai *cell state* pada *timestep* ke- $(t - 1)$
 f_t = nilai *forget gate* pada waktu t
 i_t = nilai *input gate* pada waktu t
 \tilde{c}_t = nilai kandidat *cell state* baru pada waktu t

4. Output Gate (o_t)

Output gate berfungsi menentukan informasi yang akan diteruskan ke lapisan berikutnya. *Gate* ini mengontrol seberapa banyak *state* yang diteruskan ke *output* menggunakan mekanisme yang serupa dengan *gate* lainnya. Secara sederhana, *output gate* bertugas menyaring informasi dari memori sel (c_t) untuk menghasilkan keluaran jangka pendek (h_t). Kemudian memutuskan bagian mana dari memori jangka panjang yang relevan untuk dijadikan prediksi atau untuk diteruskan ke langkah waktu berikutnya.

Output gate akhirnya menghasilkan *hidden state* baru (h_t). Tujuan utama dari *output gate* adalah mengatur sejauh mana informasi dari keadaan sel saat ini digunakan. Proses ini dimulai dengan menentukan informasi *output* melalui lapisan sigmoid. Kemudian, *cell state* diproses menggunakan fungsi aktivasi *tanh*, dan hasilnya dikalikan dengan *output* lapisan sigmoid untuk menghasilkan *output* akhir. Persamaan untuk o_t dan h_t adalah sebagai berikut:

$$o_t = \sigma(w_o \cdot h_{t-1} + w_o \cdot x_t + b_o) \quad (2.5)$$

dengan:

- o_t = *output gate* untuk *timestep* ke- t
- σ = fungsi aktivasi sigmoid
- w_o = bobot pada *output gate*
- x_t = *input* data untuk *timestep* ke- t .
- h_{t-1} = *hidden state* pada *timestep* ke- $(t - 1)$
- b_o = bias untuk *output gate*

Nilai *output* akhir dari *cell state* yang baru (h_t) didefinisikan pada persamaan berikut.

$$h_t = o_t \cdot \tanh(c_t) \quad (2.6)$$

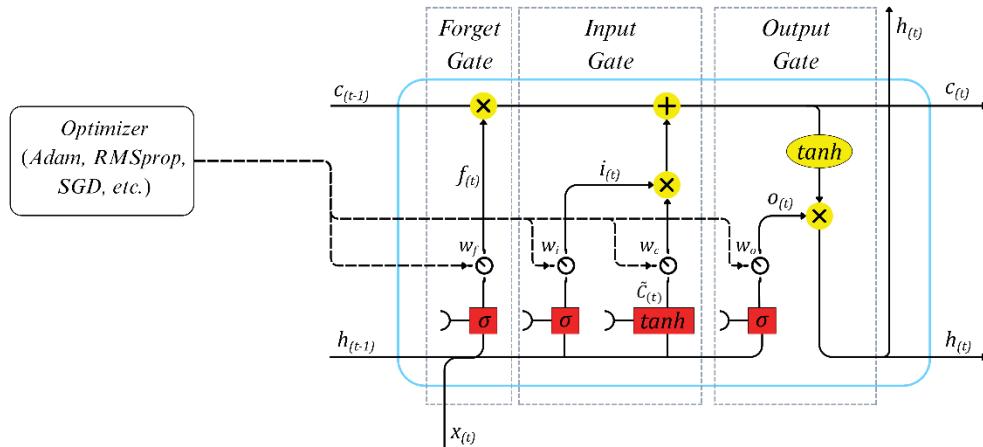
dengan:

- h_t = *hidden state* baru pada *timestep* ke- t
- o_t = *output gate* untuk *timestep* ke- t
- \tanh = fungsi hiperbolik tangen
- c_t = *cell state* baru pada *timestep* ke- t

Mekanisme kerja LSTM melibatkan interaksi antara *memory cell* dan gerbang-gerbangnya. Informasi baru yang masuk akan diproses oleh *input gate* dan ditambahkan ke *memory cell*. *Memory cell* kemudian akan menghasilkan *output* yang dipengaruhi oleh informasi yang tersimpan di dalamnya dan oleh keputusan

output gate. *Forget gate* akan secara berkala menghapus informasi yang sudah tidak relevan dari *memory cell*[21].

2.7 Optimizer



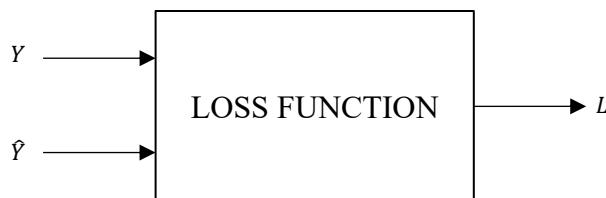
Gambar 2.10 Ilustrasi titik kerja optimizer pada arsitektur LSTM

Pengoptimal atau *optimizer* adalah algoritma yang secara adaptif menyesuaikan bobot dan bias dalam jaringan saraf selama proses pelatihan. Tujuannya adalah untuk meminimalkan nilai fungsi kerugian (*loss function*) dan meningkatkan kinerja model secara keseluruhan. Pemahaman mendasar mengenai mekanisme pengoptimal memerlukan pemahaman terhadap tiga komponen utama, yaitu fungsi objektif atau fungsi kerugian, gradien (*gradient*) atau penurunan gradien (*gradient descent*), dan laju pembelajaran (*learning rate*).

Dalam konteks pembelajaran mesin, fungsi kerugian berperan krusial dalam mengevaluasi tingkat kesalahan prediksi model dan menjadi landasan bagi proses perbaikan diri. Fungsi kerugian adalah fungsi matematika yang mengukur disparitas antara nilai prediksi dan nilai aktual dalam model pembelajaran mesin. Fungsi ini juga dikenal sebagai fungsi biaya (*cost function*) atau fungsi kesalahan (*error function*) dan berfungsi untuk menguantifikasi kinerja model selama fase

pelatihan. Dengan kata lain, fungsi kerugian mengestimasi seberapa akurat suatu algoritma dalam memodelkan data yang diberikan. Berdasarkan jenis tugas pembelajaran, fungsi kerugian dapat diklasifikasikan menjadi dua kategori utama: model regresi dan model klasifikasi.

Untuk model regresi, yang bertujuan memprediksi nilai kontinu, fungsi kerugian yang umum digunakan antara lain *Mean Squared Error* (MSE), *Mean Absolute Error* (MAE), dan *Root Mean Squared Error* (RMSE). Sementara itu, pada model klasifikasi, di mana tujuannya adalah memprediksi keluaran dari serangkaian nilai kategorikal yang terbatas, fungsi kerugian yang sering diterapkan adalah *Binary Cross-Entropy* untuk skenario dua kelas, serta *Categorical Cross-Entropy* untuk klasifikasi multi-kelas.



Gambar 2.11 Diagram blok fungsi kerugian

Input dari fungsi kerugian adalah prediksi yang dihasilkan oleh model (\hat{Y}) dan nilai aktual atau *ground truth* (Y), sedangkan *output* nya adalah nilai skalar yang dikenal sebagai kerugian atau *loss* (L). Nilai kerugian ini merefleksikan seberapa baik atau buruk prediksi model. Oleh karena itu, bentuk spesifik fungsi kerugian sangat bergantung pada karakteristik permasalahan yang dihadapi. Setiap jenis fungsi kerugian memiliki sifat yang berbeda dan lebih sesuai untuk jenis masalah dan distribusi data tertentu.

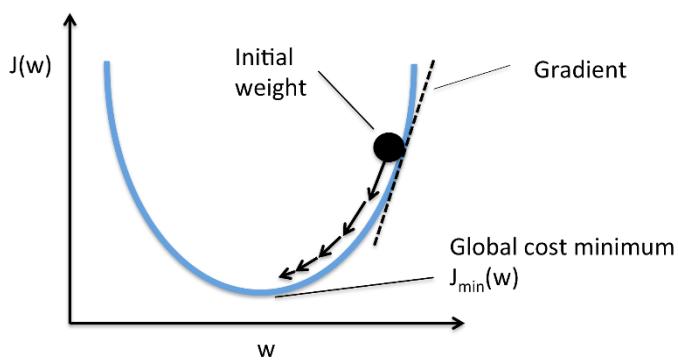
Dalam penelitian ini, fungsi kerugian yang digunakan adalah *Mean Squared Error* (MSE) karena relevan untuk permasalahan *time series forecasting*. MSE memberikan penalti yang lebih besar terhadap kesalahan prediksi yang signifikan, sehingga membantu model untuk mempelajari pola numerik dalam data deret waktu dengan lebih akurat. Fungsi kerugian MSE dapat dirumuskan sebagai:

$$L(w) = \frac{1}{n} \sum_{i=1}^n (y_i - f(x_i; w))^2 \quad (2.7)$$

dengan:

- $L(w)$ = fungsi kerugian terhadap bobot
- y_i = nilai aktual untuk sampel data ke-i.
- n = jumlah sampel
- $f(x_i; w)$ = prediksi model untuk *input* x_i dengan bobot w

Gradient Descent (GD) atau Penurunan Gradien adalah algoritma optimasi fundamental yang secara luas digunakan dalam pembelajaran mesin dan pembelajaran mendalam untuk meminimalkan fungsi biaya melalui pembaruan bobot model secara iteratif. Mekanisme algoritma ini melibatkan perhitungan gradien (turunan pertama) dari fungsi kerugian terhadap setiap bobot model. Selanjutnya, bobot diperbarui dengan bergerak ke arah yang berlawanan dengan arah gradien tersebut. Proses ini diulang hingga algoritma mencapai titik minimum lokal atau global, di mana nilai fungsi kerugian mencapai titik terendahnya.



Gambar 2.12 Ilustrasi proses gradient descent

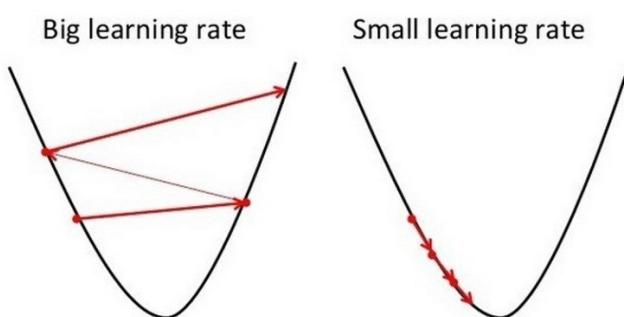
Secara intuitif, gradien merepresentasikan arah dan besarnya tingkat kenaikan paling curam dari permukaan fungsi kerugian. Mengingat tujuan optimasi adalah untuk meminimalkan kesalahan, pembaruan bobot dilakukan dengan mengambil langkah proporsional ke arah negatif gradien. Aturan pembaruan bobot umumnya didefinisikan sebagai:

$$w_t = w_{t-1} - \eta \nabla L(w_{t-1}) \quad (2.8)$$

dengan:

- w_t = bobot baru
- w_{t-1} = bobot sebelumnya
- η = laju pembelajaran (*learning rate*)
- $\nabla L(w_{t-1})$ = gradien fungsi kerugian terhadap bobot pada langkah t-1.

Laju pembelajaran (*learning rate*) atau ukuran langkah (*step size*) adalah *hyperparameter* krusial yang mengontrol besarnya langkah yang diambil menuju minimum pada setiap iterasi. Pemilihan nilai *learning rate* memerlukan pertimbangan yang cermat. Jika laju pembelajaran terlalu kecil, proses konvergensi akan berjalan lambat dan memerlukan banyak iterasi untuk mencapai. Sebaliknya, jika laju pembelajaran terlalu besar, algoritma dapat melampaui titik minimum atau bahkan mengalami divergensi, di mana nilai fungsi biaya justru meningkat[22].



Gambar 2.13 Ilustrasi pengaruh learning rate

2.7.1 Stochastic Gradient Descent

Stochastic Gradient Descent (SGD) merupakan algoritma optimasi yang sangat populer dalam pelatihan model *deep learning*, terutama untuk *dataset* berukuran besar. Karakteristik utama SGD terletak pada elemen "stochastic" atau keacakan dalam proses pembaruan bobot model. Berbeda dengan metode penurunan gradien *batch* (*batch gradient descent*) yang menghitung gradien fungsi kerugian (*loss function*) berdasarkan seluruh *dataset* pada setiap iterasi, SGD secara acak memilih satu sampel data atau sejumlah kecil sampel (dikenal sebagai *mini-batch*) untuk mengestimasi gradien. Pendekatan ini memungkinkan optimasi yang lebih efisien dan mengurangi beban komputasi, sehingga mempercepat pelatihan model *deep learning* dan membuatnya lebih praktis dalam menangani *dataset* berukuran besar[23].

Secara teknis, langkah-langkah operasional *Stochastic Gradient Descent* adalah sebagai berikut:

1. Inisialisasi bobot w secara acak.
2. Pilih secara acak satu sampel atau *mini-batch* dari data pelatihan.
3. Hitung gradien fungsi kerugian terhadap bobot w menggunakan sampel terpilih:

$$\nabla L_i(w) = \frac{\partial L_i(w)}{\partial w} \quad (2.9)$$

dengan:

$$\begin{aligned}\nabla L_i(w) &= \text{gradien fungsi kerugian untuk satu titik data } i \\ L_i(w) &= \text{nilai fungsi kerugian untuk satu titik data } i \\ w &= \text{bobot}\end{aligned}$$

4. Perbarui bobot w menggunakan gradien yang telah dihitung.

Pada metode *Gradient Descent* biasa, pembaruan bobot didasarkan pada gradien seluruh *dataset*, seperti yang ditunjukkan pada Persamaan (2.2). Namun, pada *Stochastic Gradient Descent* pembaruan dilakukan berdasarkan gradien dari satu sampel data atau *mini-batch*:

$$w_t = w_{t-1} - \eta \nabla L_i(w_{t-1}) \quad (2.10)$$

dengan:

w_t = bobot baru
 w_{t-1} = bobot sebelumnya
 η = *learning rate*

$\nabla L_i(w_{t-1})$ = gradien fungsi kerugian untuk satu titik data i

5. Ulangi langkah 2 sampai 4 hingga kriteria penghentian terpenuhi, seperti mencapai jumlah epoch tertentu atau nilai fungsi kerugian yang telah konvergen[24].

2.7.2 Root Mean Square Propagation

Root Mean Square Propagation (RMSprop) adalah algoritma *Optimizer* yang populer dalam pelatihan model *deep learning*, dikembangkan untuk mengatasi masalah gradien yang bervariasi dan penurunan *learning rate* yang terlalu cepat pada algoritma seperti *Adaptive Gradient* (AdaGrad). Algoritma ini secara adaptif menyesuaikan *learning rate* untuk setiap bobot berdasarkan rata-rata pergerakan eksponensial dari kuadrat gradien. Dengan demikian, RMSprop mampu mengatasi masalah gradien yang fluktuatif dan memungkinkan pelatihan yang lebih stabil serta efektif, terutama dengan penggunaan *mini-batch*. RMSprop lebih berfokus pada adaptasi berdasarkan magnitudo gradien dan dapat dianggap sebagai perbaikan dari AdaGrad. Selain itu, RMSprop juga menjadi salah satu algoritma

yang menginspirasi pengembangan Adam, yang mengintegrasikan konsep dari RMSprop dan *momentum*[23], [25].

Secara teknis, langkah-langkah operasional *Root Mean Square Propagation* adalah sebagai berikut:

1. Hitung gradien pada posisi saat ini.

$$g_t = \nabla L(w_{t-1}) \quad (2.11)$$

dengan:

$$\begin{aligned} g_t &= \text{gradien pada waktu t} \\ \nabla L(w_{t-1}) &= \text{gradien fungsi kerugian terhadap bobot} \end{aligned}$$

2. Selanjutnya, hitung rata-rata bergerak dari kuadrat gradien. Tingkat peluruhan (*decay rate*) informasi lampau dikontrol oleh *hyperparameter* β , yang umumnya ditetapkan sekitar 0.9 atau 0.95.

$$v_t = \beta \cdot v_{t-1} + (1 - \beta) \cdot g_t^2 \quad (2.12)$$

dengan:

$$\begin{aligned} v_t &= \text{rata-rata bergerak dari kuadrat gradien} \\ \beta &= \text{decay rate} \\ v_{t-1} &= \text{rata-rata bergerak dari kuadrat gradien sebelumnya} \\ g_t &= \text{gradien pada waktu t} \end{aligned}$$

3. Perbarui bobot dengan bergerak ke arah yang berlawanan dengan gradien yang telah dinormalisasi.

$$w_t = w_{t-1} - \frac{\eta}{\sqrt{v_t + \epsilon}} g_t \quad (2.13)$$

dengan:

$$\begin{aligned} w_t &= \text{bobot baru} \\ w_{t-1} &= \text{bobot sebelumnya} \\ \eta &= \text{learning rate} \\ g_t &= \text{gradien pada waktu t} \\ v_t &= \text{rata-rata bergerak dari kuadrat gradien pada waktu t} \end{aligned}$$

ϵ = konstanta kecil mencegah pembagian dengan nol

4. Ulangi langkah 1 sampai 4 hingga kriteria penghentian terpenuhi, seperti mencapai jumlah epoch tertentu atau nilai fungsi kerugian yang telah konvergen[26].

2.7.3 *Adaptive Moment Estimation*

Adaptive Moment Estimation (Adam) adalah algoritma *optimizer* yang banyak digunakan dalam *deep learning*, dan merupakan perluasan dari algoritma SGD dengan mekanisme adaptif. Algoritma ini secara adaptif menyesuaikan *learning rate* untuk setiap bobot jaringan berdasarkan gradien masa lalu serta dua momen statistiknya. Berbeda dengan SGD yang menggunakan *learning rate* tetap, Adam mengintegrasikan keunggulan dari AdaGrad dan RMSprop dengan mempertimbangkan rata-rata gradien pertama (momen pertama) serta varians gradien yang tidak terpusat (momen kedua). Melalui pendekatan ini, Adam mampu menyesuaikan *learning rate* secara dinamis, mempercepat konvergensi, dan meningkatkan stabilitas optimasi. Secara keseluruhan, Adam merupakan algoritma yang efisien dan adaptif dalam memperbarui bobot jaringan selama pelatihan pembelajaran mendalam[23], [25].

Secara teknis, langkah-langkah operasional *Adaptive Moment Estimation* adalah sebagai berikut:

1. Inisialisasi bobot model (w), *learning rate* (η), dan *hyperparameters* (β_1 , β_2 , dan ϵ). Nilai bawaan yang umum digunakan adalah $\eta = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$

2. Hitung gradien (g) dari fungsi kerugian (L) terhadap bobot model menggunakan persamaan 2.11.

3. Perbarui estimasi momen pertama (m):

Estimasi momen pertama ini pada dasarnya adalah rata-rata bergerak dari gradien (mirip dengan konsep *momentum*), yang membantu model mengetahui arah umum pergerakan untuk mempercepat konvergensi.

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t \quad (2.14)$$

dengan:

- m_t = estimasi momen pertama pada langkah t
- m_{t-1} = estimasi momen pertama pada langkah t-1
- β_1 = *decay rate* untuk estimasi momen pertama
- g_t = gradien pada langkah t

4. Perbarui estimasi momen kedua (v):

Estimasi momen kedua adalah rata-rata bergerak dari kuadrat gradien (mirip dengan konsep RMSProp), yang membantu mengadaptasi laju pembelajaran untuk setiap parameter secara individual.

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2)(g_t \odot g_t) \quad (2.15)$$

dengan:

- v_t = estimasi momen kedua pada langkah t
- v_{t-1} = estimasi momen kedua pada langkah t-1
- β_2 = *decay rate* untuk estimasi momen kedua
- \odot = operasi perkalian elemen-wise (Hadamard product)
- g_t = gradien pada langkah t

5. Koreksi bias pada estimasi momen pertama (\hat{m}_t) dan kedua (\hat{v}_t) untuk iterasi saat ini (t):

$$\hat{m}_t = \frac{m_t}{(1 - \beta_1)} \quad (2.16)$$

dengan:

- \hat{m}_t = estimasi momen pertama terkoreksi bias pada langkah t
- m_t = estimasi momen pertama pada langkah t
- β_1 = *decay rate* untuk estimasi momen pertama

$$\hat{v}_t = \frac{v_t}{(1-\beta_2)} \quad (2.17)$$

dengan:

\hat{v}_t = estimasi momen kedua terkoreksi bias pada langkah t

v_t = estimasi momen kedua pada langkah t

β_2 = decay rate untuk estimasi momen kedua

6. Hitung laju pembelajaran adaptif (η_t):

$$\eta_t = \frac{\eta_{t-1}\sqrt{(1-\beta_2)}}{(1-\beta_1)} \quad (2.18)$$

dengan:

η_t = learning rate adaptif pada langkah t

η_{t-1} = learning rate sebelumnya

β_1 = decay rate estimasi momen pertama

β_2 = decay rate estimasi momen kedua

7. Perbarui bobot model menggunakan laju pembelajaran adaptif:

$$w_t = w_{t-1} - \frac{\eta_t \hat{m}_t}{\sqrt{\hat{v}_t + \epsilon}} \quad (2.19)$$

dengan:

w_t = bobot baru

w_{t-1} = bobot sebelumnya

η_t = learning rate adaptif pada langkah t

\hat{m}_t = estimasi momen pertama terkoreksi bias pada t

\hat{v}_t = estimasi momen kedua terkoreksi bias pada t

ϵ = konstanta kecil mencegah pembagian dengan nol

8. Ulangi langkah 2 sampai 7 hingga kriteria penghentian terpenuhi, seperti mencapai jumlah epoch tertentu atau nilai fungsi kerugian yang telah konvergen[27].

2.8 Metrik Evaluasi Model

Membangun model pembelajaran mesin yang efektif membutuhkan proses berkelanjutan. Model awal dibuat, kinerjanya dievaluasi, dan kemudian diperbaiki secara berulang hingga mencapai tingkat akurasi yang diinginkan. Dalam siklus ini, evaluasi model memainkan peran krusial untuk mengukur performa model dan mengidentifikasi area yang perlu ditingkatkan. Di ranah industri AI, ada berbagai jenis metrik untuk mengevaluasi model pembelajaran mesin. Berikut adalah beberapa metrik yang digunakan pada penelitian ini.

2.8.1 Mean Absolute Error

Mean Absolute Error (MAE) adalah nilai absolut selisih rata-rata antara prediksi model dan nilai aktual. Sederhananya, MAE mengukur seberapa dekat hasil prediksi model dengan nilai aktual. Kelemahan MAE adalah tidak memberikan informasi tentang arah kesalahan. Artinya, MAE tidak menunjukkan apakah prediksi kita lebih rendah atau lebih tinggi dari nilai aktual.

$$MAE = \frac{\sum_{i=1}^n |y_i - \hat{y}_i|}{n} \quad (2.20)$$

dengan:

n = jumlah total observasi atau sampel.

y_i = nilai aktual atau nilai sebenarnya dari observasi ke- i .

\hat{y}_i = nilai prediksi model untuk observasi ke- i [28], [29].

2.8.2 Root Mean Square Error

Root Mean Square Error (RMSE) adalah metrik yang dapat diperoleh hanya dengan mengambil akar kuadrat dari nilai *Mean Square Error* (MSE) yang merupakan kuadrat dari rata-rata antara nilai prediksi dan nilai asli. Metrik MSE

tidak tahan terhadap *outlier*, begitu pula dengan nilai RMSE. Hal ini memberikan bobot yang lebih tinggi pada kesalahan yang besar dalam prediksi.

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2} \quad (2.21)$$

dengan:

- n = jumlah total observasi atau sampel.
- y_i = nilai aktual atau nilai sebenarnya dari observasi ke- i .
- \hat{y}_i = nilai prediksi model untuk observasi ke- i [28], [29].

2.8.3 Mean Absolute Percentage Error

Mean Absolute Percentage Error (MAPE) adalah ukuran statistik yang umum digunakan untuk mengevaluasi akurasi model peramalan. MAPE sangat populer dalam analisis deret waktu, keuangan, ekonomi, dan pembelajaran mesin. MAPE mengukur rata-rata besarnya kesalahan yang dihasilkan oleh suatu model, atau seberapa jauh rata-rata prediksi dari nilai aktual. Hal ini membuat MAPE menjadi ukuran yang intuitif dan mudah diinterpretasikan, karena memberikan wawasan tentang kesalahan rata-rata dalam bentuk persentase, sehingga memungkinkan pemahaman yang jelas tentang kinerja model. Nilai MAPE yang lebih rendah menunjukkan prediksi yang lebih akurat.

$$MAPE = \frac{\sum_{i=1}^n \left| \frac{y_i - \hat{y}_i}{y_i} \right| \times 100\%}{n} \quad (2.22)$$

dengan:

- n = jumlah total observasi atau sampel.
- y_i = nilai aktual atau nilai sebenarnya dari observasi ke- i .
- \hat{y}_i = nilai prediksi model untuk observasi ke- i [30], [31].

2.9 Python



Gambar 2.14 Logo Python

Python merupakan salah satu bahasa pemrograman yang sangat populer dan dikenal luas dalam pengembangan perangkat lunak. Bahasa ini dikategorikan sebagai bahasa tingkat tinggi (*high-level*), diterjemahkan langsung (*interpreted*), dan serbaguna (*general-purpose*).

Sebagai bahasa tingkat tinggi, Python membebaskan pengembang dari kerumitan detail teknis perangkat keras, memungkinkan mereka untuk fokus pada logika program. Sifat *interpreted* memungkinkan eksekusi kode baris per baris tanpa proses kompilasi menjadi bahasa mesin. Keserbagunaannya menjadikan Python relevan untuk berbagai keperluan, mulai dari pengembangan web, aplikasi desktop, otomatisasi sistem (*system scripting*), pengembangan backend, hingga penelitian. Dalam bidang *Machine Learning*, Python menjadi pilihan utama berkat kemudahan penggunaan serta dukungan ekosistem pustaka (*library*) dan *framework* yang luas[32].

2.10 Tensorflow

TensorFlow adalah *library* perangkat lunak bersifat *open source* yang dikembangkan oleh tim Google Brain. *Library* ini dirancang untuk mendukung komputasi numerik berskala besar, khususnya dalam bidang *Machine Learning*. TensorFlow menyediakan kerangka kerja komprehensif untuk pembangunan, pelatihan, hingga implementasi model *Machine Learning*.

Keunggulan utama TensorFlow terletak pada kemampuannya untuk diaplikasikan pada berbagai bidang, seperti pengolahan citra, pemrosesan bahasa alami, dan pengenalan suara. Selain Python, TensorFlow juga mendukung berbagai bahasa pemrograman lain seperti C++, JavaScript, Java, Swift, dan Go. Fleksibilitas ini memudahkan pengembang untuk mengintegrasikannya dengan bahasa yang mereka kuasai dalam mengembangkan aplikasi berbasis *Machine Learning*[33].



Gambar 2.15 Logo Tensorflow

2.11 Google Colab



Gambar 2.16 Logo Google Colab

Google Colab merupakan sebuah platform yang serupa dengan Jupyter Notebook dan dikembangkan oleh Google Research. Namun, berbeda dengan Jupyter Notebook, Google Colab menyediakan integrasi langsung dengan perangkat keras khusus, seperti GPU, yang difasilitasi oleh Google. Fitur ini memudahkan pengembang maupun peneliti dalam melakukan eksplorasi dan analisis di bidang data science, tanpa harus mempersiapkan atau memasang infrastruktur perangkat keras secara manual[34].

BAB 3

METODE PENELITIAN

3.1 Waktu dan Tempat

Penelitian ini dilakukan pada semester ganjil tahun akademik 2024/2025 di Jakarta Selatan. Penyusunan Laporan Tugas Akhir dilaksanakan di Jakarta Selatan dan Purbalingga.

3.2 Alat dan Bahan

Alat dan bahan yang digunakan dalam penelitian tugas akhir ini adalah sebagai berikut:

3.2.1 Alat

1. Laptop HP Victus
2. Digital Lux Meter AS823
3. Laser Distance Meter SNDWAY SW-100G
4. Microsoft Word
5. Google Colab
6. Google Form
7. Google Spreadsheet

3.2.2 Bahan

Data lux harian yang diambil di atap sebuah Gedung di Jakarta Selatan selama satu tahun mulai dari Maret 2023 hingga Februari 2024. Data lux meter ini berisi nilai lux pada pagi, siang dan sore hari mulai dari sudut 0 derajat menghadap selatan hingga 180 derajat menghadap utara, dengan selisih pengukuran 15 derajat. Selain itu terdapat juga informasi cuaca pada saat pengukuran yaitu Cerah, Cerah Berawan, Berawan, Mendung dan Hujan.

3.3 Alur dan Tahapan Penelitian

Penelitian ini dibagi menjadi beberapa tahapan untuk memudahkan dalam penelitian. Tahapan penelitian dibagi sebagai berikut:

3.3.1 Tahap Persiapan

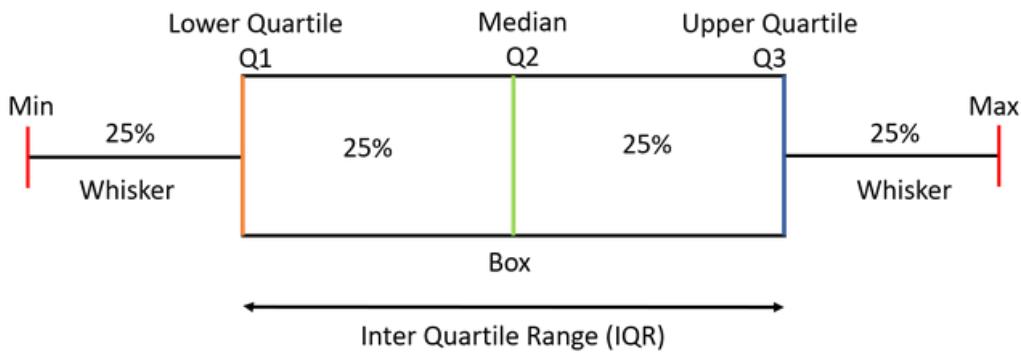
Pada tahap ini dilakukan studi literatur untuk memahami konsep dan teori yang mendasari penelitian ini. Studi mencakup penelitian sebelumnya terkait konversi antara lux dan iradiasi matahari, penggunaan algoritma pembelajaran mesin—khususnya *Long Short-Term Memory* (LSTM)—penggunaan *optimizer* dalam pembelajaran mesin, serta metode pengumpulan dan pengolahan data yang relevan.

3.3.2 Tahap Pengumpulan Data

Tahap pengumpulan data melibatkan akuisisi data lux di atap gedung di wilayah Jakarta Selatan. Data dikumpulkan secara berkala pada pagi, siang, dan sore hari, dengan sudut pengukuran sensor dari 0° hingga 180° dengan selisih 15° pada setiap pengukuran. Sensor ditempatkan di lokasi tetap dan diarahkan dari selatan (0°) hingga utara (180°).

3.3.3 Tahap *Exploratory Data Analysis*

Pada tahap ini, dilakukan *Exploratory Data Analysis* (EDA) untuk memahami karakteristik dasar, mendeteksi potensi *outlier*, dan menemukan pola dari *dataset*. Proses ini meliputi analisis statistik deskriptif dan visualisasi data. Salah satu teknik visualisasi utama yang digunakan adalah *Boxplot* (Diagram Kotak Garis). *Boxplot* adalah metode grafis yang sangat efektif untuk menyajikan ringkasan statistik kunci dari data secara visual, seperti pemusatan, sebaran, dan ada atau tidaknya *outlier*.



Gambar 3.1 Contoh Boxplot sempurna tanpa kemiringan

Berdasarkan Gambar 3.1 sebuah *boxplot* dapat dipecah menjadi beberapa komponen sebagai berikut:

- Minimum dan Maksimum: Merupakan nilai data terkecil dan terbesar dalam *dataset*, tidak termasuk *outlier*. Batas ini ditandai oleh ujung dari garis sumbu (*whisker*).
- Kuartil Pertama (Q1): Merepresentasikan persentil ke-25, di mana 25% dari total data berada di bawah nilai ini. Ini adalah batas bawah dari kotak.
- Median (Q2): Merupakan titik tengah dari *dataset*, di mana garis ini membagi data menjadi dua bagian.
- Kuartil Ketiga (Q3): Merepresentasikan persentil ke-75, di mana 75% dari total data berada di bawah nilai ini. Ini adalah batas atas dari kotak.
- *Inter Quartile Range (IQR)*: Merupakan bagian kotak utama yang merepresentasikan rentang interkuartil ($IQR = Q3 - Q1$). Bagian ini menunjukkan sebaran dari 50% data yang berada di tengah.

- Sumbu (*Whiskers*): Adalah garis yang memanjang dari kotak. Berdasarkan standar Tukey, garis atas memanjang hingga $Q3 + 1.5 * IQR$, dan garis bawah memanjang hingga $Q1 - 1.5 * IQR$.
- *Outliers*: Setiap titik data yang berada di luar jangkauan *whisker* dianggap sebagai *outliers*. Alat ini sangat efektif untuk mendeteksi *outliers*[35].

Dalam penelitian ini, *boxplot* akan digunakan untuk menganalisis distribusi data lux dan mengidentifikasi keberadaan *outliers*. Selain itu, akan dibuat juga visualisasi lain seperti *time series plot* untuk mengamati pola data seiring waktu.

3.3.4 Tahap Perancangan Model

Pada tahap ini, dilakukan perancangan arsitektur model pembelajaran mesin yang akan menjadi fondasi bagi seluruh percobaan. Model yang dirancang menggunakan arsitektur *Stacked Long Short-Term Memory* (LSTM bertumpuk). Pemilihan arsitektur ini sejalan dengan praktik terbaik dan temuan dalam penelitian terdahulu. Studi oleh Husein dan Chung (2019) serta Alzahrani dkk. (2017) secara spesifik menunjukkan bahwa penggunaan arsitektur RNN dengan beberapa lapisan tersembunyi (dua hingga tiga lapis) mampu meningkatkan akurasi prediksi secara signifikan. Alzahrani dkk. (2017) bahkan menegaskan bahwa jaringan yang lebih dalam lebih efisien dalam merepresentasikan fungsi yang kompleks.

Arsitektur bertumpuk memungkinkan model untuk mempelajari representasi data pada berbagai tingkat abstraksi: lapisan pertama dapat menangkap pola temporal jangka pendek, sementara lapisan-lapisan berikutnya dapat

mensintesis informasi tersebut untuk mengenali pola jangka panjang yang lebih kompleks, seperti siklus musiman yang teridentifikasi dalam data iradiasi matahari. Perancangan meliputi penentuan jumlah lapisan LSTM, jumlah unit *neuron* pada setiap lapisan, serta penggunaan lapisan Dense sebagai *output layer* untuk menghasilkan nilai prediksi akhir. Arsitektur dasar yang sama ini akan digunakan secara konsisten untuk menguji ketiga jenis *optimizer*: Adam, RMSprop, dan SGD.

3.3.5 Tahap *Data Preprocessing*

Pada tahap ini, data lux yang telah dikumpulkan diolah dan dirapikan. Proses *preprocessing* meliputi pembersihan data dari nilai-nilai anomali, pengisian nilai yang hilang, normalisasi, dan transformasi data agar siap digunakan dalam pembuatan model pembelajaran mesin. Preprocessing data sangat penting untuk meningkatkan akurasi dan kinerja model. Tahapan preprocessing ini meliputi:

1. Pembersihan Data

Pembersihan data atau data cleaning adalah metode yang digunakan untuk menangani data yang hilang (*missing value*) atau mengandung *outlier*. *Missing value* merujuk pada situasi di mana *dataset* memiliki data yang tidak lengkap atau hilang. Proses pembersihan data ini dapat dilakukan dengan dua pendekatan, yaitu dengan menghapus seluruh baris yang mengandung data hilang atau mengganti nilai yang hilang (*imputation*) menggunakan nilai tetap (konstanta), nilai rata-rata, nilai median, atau metode imputasi lainnya[36].

2. Konversi Lux Ke Iradiasi Matahari

Proses konversi data lux ke iradiasi melibatkan penerapan model matematis yang menghubungkan antara intensitas cahaya yang diukur oleh sensor lux dengan jumlah energi matahari yang diterima permukaan. Model matematis yang

digunakan pada penelitian ini didasarkan pada hasil penelitian terbaru oleh Michael dkk. (2020) yang dapat dirumuskan ke persamaan berikut:

$$Iradiasi = \frac{lux}{122} \quad (3.1)$$

3. Normalisasi Data

Normalisasi data atau *scalling* data merupakan proses transformasi data yang mengubah data asli ke dalam bentuk lain. Proses normalisasi perlu dilakukan agar menghasilkan nilai *error* yang sekecil mungkin. Teknik normalisasi yang digunakan untuk penelitian ini adalah *MinMaxScaler* atau bisa disebut juga dengan *Min-Max Scaling*. Teknik ini mentransformasikan data aktual ke dalam nilai dengan rentang [0,1]. Teknik *Min-Max Scaling* dirumuskan ke dalam persamaan berikut [37]:

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}} \quad (3.2)$$

dengan:

- x' = data hasil normalisasi.
- x = data aktual.
- x_{min} = nilai minimum data.
- x_{max} = nilai maksimum data.

4. Seleksi Fitur

Tahapan seleksi fitur bertujuan untuk memilih fitur-fitur yang paling relevan dan berkontribusi signifikan terhadap prediksi *time series*. Proses ini melibatkan penghapusan fitur yang redundan atau tidak relevan. Dengan memilih fitur yang tepat, kita dapat meningkatkan akurasi model dan mengurangi kompleksitas komputasi.

5. Pembagian Data

Pembagian data merupakan tahap krusial untuk mengevaluasi performa model secara objektif. Pada penelitian ini, *dataset* akan dibagi secara kronologis menjadi tiga bagian dengan rasio 70% data latih, 15% data validasi, dan 15% data uji.

Data Latih (*Training Set*) adalah data yang digunakan oleh model untuk mempelajari pola dan hubungan dalam deret waktu. Data Validasi (*Validation Set*) adalah data yang digunakan untuk memonitor performa model selama proses pelatihan. Metrik pada data validasi menjadi acuan untuk menyimpan versi model terbaik (*checkpointing*) dan mencegah overfitting. Sedangkan, Data Uji (*Testing Set*) adalah data yang sepenuhnya "baru" bagi model dan hanya digunakan satu kali pada tahap akhir untuk memberikan evaluasi yang tidak bias terhadap kemampuan generalisasi model.

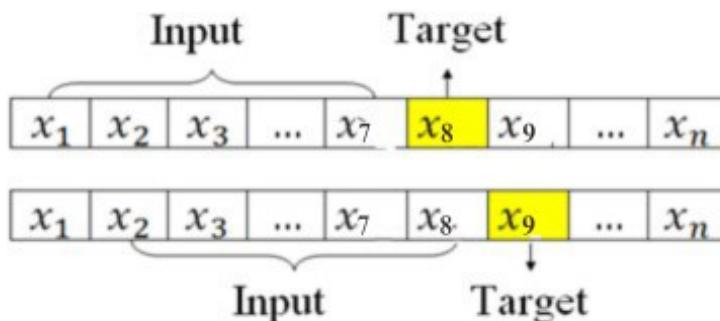
Pembagian secara kronologis ini memastikan bahwa model dilatih menggunakan data masa lalu untuk memprediksi data masa depan, sesuai dengan skenario penggunaan di dunia nyata[38].

6. Segmentasi Data

Segmentasi adalah proses memisahkan serta mengelompokkan data menjadi data yang diperlukan oleh sistem. Segmentasi data dilakukan dengan pembentukan pola *time series* pada data. Proses segmentasi untuk penelitian ini diilustrasikan pada gambar berikut.

Pada Gambar 3.2, proses segmentasi diilustrasikan menggunakan jendela waktu (*sliding window*). Ukuran jendela atau *timestep* sepanjang 7 hari dipilih

secara spesifik untuk memungkinkan model menangkap dan mempelajari potensi pola atau siklus mingguan (*weekly Seasonality*) dalam data. Dengan pendekatan ini, 7 data berurutan (misalnya, data hari ke-1 hingga ke-7) digunakan sebagai bagian dari *input* untuk memprediksi data pada hari ke-8 yang menjadi target-nya. Proses ini digeser satu hari ke depan secara berulang untuk seluruh *dataset*.



Gambar 3.2 Ilustrasi segmentasi data

3.3.6 Tahap Pelatihan Model

Pada tahap ini, dilakukan pelatihan model untuk menganalisis dan membandingkan performa dari ketiga *optimizer*. Untuk memastikan perbandingan yang adil dan terkontrol, arsitektur model *Stacked LSTM* yang sama (seperti yang dirancang pada sub-bab 3.3.4) akan dilatih menggunakan tiga jenis *optimizer* yang berbeda: Adam, RMSprop, dan SGD.

Masing-masing *optimizer* akan diuji dalam beberapa skenario konfigurasi *hyperparameter* untuk mengevaluasi dampaknya terhadap performa model. Skenario ini mencakup variasi pada laju belajar (*learning rate*), seperti laju belajar tinggi (0.01) dan rendah (0.001), serta penyesuaian parameter internal spesifik lainnya seperti *momentum* untuk SGD atau *decay rate* untuk Adam dan RMSprop. Desain eksperimental yang sistematis ini memungkinkan analisis yang terkontrol

untuk menentukan kombinasi *optimizer* dan *hyperparameter* yang paling optimal untuk kasus prediksi iradiasi matahari.

3.3.7 Tahap Evaluasi dan Pemilihan Model Terbaik

Pada tahap akhir, model yang telah dibuat dievaluasi untuk menilai kinerjanya. Evaluasi dilakukan dengan mengujinya ke data test. Hasil pengujian tersebut kemudian di denormalisasi sebelum dievaluasi menggunakan metrik evaluasi MAE, RMSE, serta MAPE.

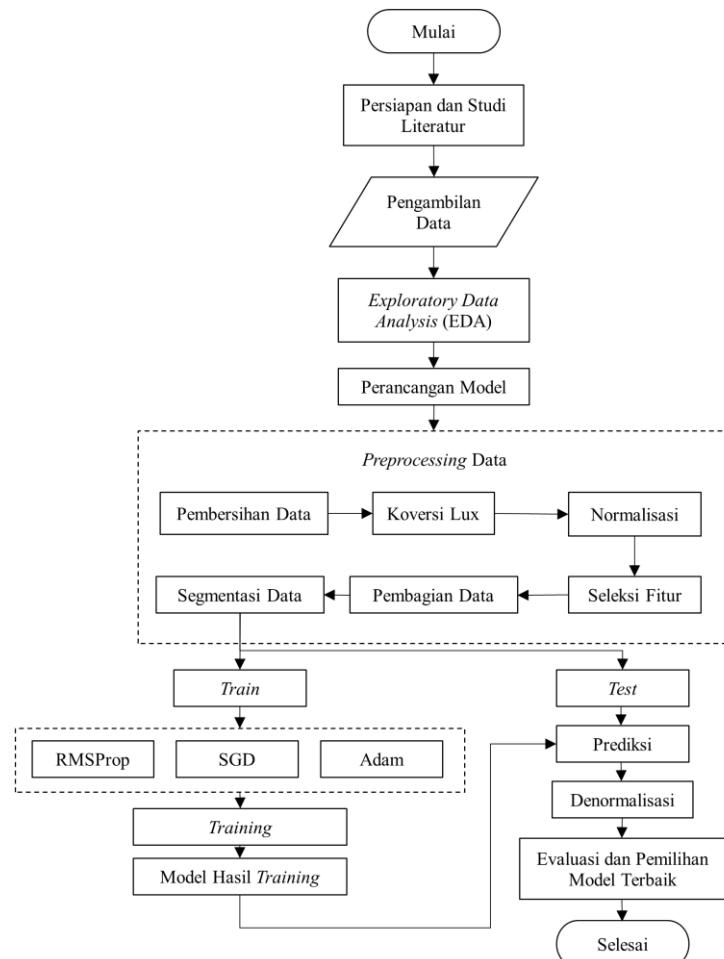
Denormalisasi merupakan proses merubah hasil prediksi yang diperoleh dari data yang telah dinormalisasi kembali ke dalam bentuk data aslinya. Proses ini dilakukan agar dapat membandingkan nilai hasil prediksi dengan data aktual guna mengevaluasi kinerja model yang digunakan. Jika normalisasi sebelumnya dilakukan dalam rentang [0,1], maka proses denormalisasi dapat dijelaskan melalui persamaan berikut[21]:

$$y = y'(x_{max} - x_{min}) + x_{min} \quad (3.3)$$

dengan:

- y = Nilai hasil denormalisasi
- y' = Nilai data normalisasi
- x_{min} = Nilai minimum data aktual
- x_{max} = Nilai maksimum data aktual

3.4 Diagram Alur Penelitian



Gambar 3.3 Diagram alur penelitian

3.5 Jadwal Penelitian

Tabel 3.1 Jadwal penelitian

BAB 4

HASIL PENELITIAN DAN PEMBAHASAN

4.1 *Exploratory Data Analysis (EDA)*

Tahap analisis diawali dengan *Exploratory Data Analysis* (EDA) untuk memahami data secara menyeluruh sebelum memasuki tahap pemodelan. Tujuan utama EDA adalah mengidentifikasi karakteristik dasar, mendeteksi potensi anomali seperti *missing values* dan *outliers*, serta memahami distribusi data.

Dataset yang dianalisis dalam penelitian ini adalah hasil pengukuran intensitas cahaya (lux) pada siang hari, yang dikumpulkan menggunakan Digital Lux Meter AS823 dari atap sebuah gedung di Jakarta Selatan. Periode pengumpulan data berlangsung selama satu tahun, dari 1 Maret 2023 hingga 29 Februari 2024.

4.1.1 Deskripsi Variabel

Analisis struktur data awal menunjukkan bahwa *dataset* mentah terdiri dari 4758 entri dan enam kolom. *Output* fungsi `info()` dari *library* pandas pada Gambar 4.1 menunjukkan bahwa sebagian besar kolom memiliki 4758 data non-null, akan tetapi kolom 'Lux' hanya memiliki 4056 entri non-null. Hal ini mengindikasikan adanya *missing values* pada variabel target yang akan menjadi fokus pada tahap pra-pemrosesan.

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 4758 entries, 2023-03-01 to 2024-02-29
Data columns (total 6 columns):
 #   Column   Non-Null Count  Dtype  
--- 
 0   No        4758 non-null   int64  
 1   Bulan     4758 non-null   object  
 2   Sudut     4758 non-null   int64  
 3   Lux       4056 non-null   float64 
 4   Cuaca     4758 non-null   object  
 5   Tampak    4758 non-null   object  
dtypes: float64(1), int64(2), object(3)
memory usage: 260.2+ KB
```

Gambar 4.1 Deskripsi variabel

Dataset yang digunakan dalam penelitian ini terdiri dari beberapa variabel. Variabel ‘No.’ berfungsi sebagai nomor urut data, sementara ‘Bulan’ adalah variabel kategorikal yang menandai bulan dilakukannya pengukuran. Terdapat pula variabel numerik ‘Sudut’ yang mencatat sudut pengukuran sensor dengan rentang 0 hingga 180 derajat. Variabel target utama adalah ‘Lux’, sebuah nilai numerik (float) yang merepresentasikan intensitas cahaya. Selain itu, *dataset* ini diperkaya dengan dua variabel kategorikal deskriptif: ‘Cuaca’, yang menjelaskan kondisi cuaca (seperti Berawan, Mendung, Hujan, Cerah, dan Cerah Berawan), dan ‘Tampak’, yang menggambarkan visibilitas matahari saat pengukuran.

4.1.2 Deskripsi Statistik

Analisis statistik deskriptif dilakukan untuk mendapatkan gambaran umum mengenai tendensi sentral, sebaran, dan variabilitas data. Dalam penelitian ini, analisis difokuskan pada variabel target utama, yaitu intensitas cahaya (Lux), untuk memahami karakteristiknya pada setiap sudut pengukuran sebelum masuk ke tahap pemodelan.

Berdasarkan Tabel 4.1, dapat diamati beberapa pola penting. Nilai rata-rata (Mean) menunjukkan tren peningkatan nilai Lux dari sudut 0° , mencapai puncaknya pada sudut 75° dengan nilai 49.794 lux, kemudian menurun secara bertahap hingga sudut 180° . Pola ini mengindikasikan bahwa intensitas cahaya rata-rata tertinggi diterima pada sudut yang mendekati 75° . Simpangan baku (Std), yang mencerminkan sebaran data, juga menunjukkan pola serupa, dengan variabilitas tertinggi terjadi pada sudut 75° (34.947), yang berarti pada sudut ini rentang nilai Lux paling beragam.

Tabel 4.1 Deskripsi Statistik Lux

Sudut	Mean	Std	Min	25%	50%	75%	Max
0	19747	12852	203	14311	18944	25571	75420
15	29988	19816	364	17805	29988	40014	95460
30	37373	24358	499	20159	37580	52656	115900
45	43701	29815	635	20125	40760	67860	130800
60	49789	33102	771	22163	50128	76296	135700
75	49794	34947	706	21101	45046	78690	151700
90	46714	33905	613	19620	41105	71950	155000
105	44609	30481	659	20244	41131	67929	129100
120	40625	26722	590	21277	40182	59195	113600
135	33082	21208	520	19091	32387	47417	91300
150	25906	16190	449	16201	25223	35610	74740
165	19396	11949	379	12509	19338	27161	93430
180	17259	11441	233	11152	16800	24147	90740

4.1.3 Missing value dan Outlier

Analisis pada sub-bab ini berfokus pada identifikasi *missing values* dan *outliers* pada dataset sudut 75° . Pemeriksaan ini krusial karena keberadaan anomali data dapat memengaruhi kualitas dan akurasi model prediksi yang akan dibangun. Untuk identifikasi *missing values*, dilakukan pemeriksaan pada setiap variabel seperti yang terlihat pada Gambar 4.2 dan diringkas pada Tabel 4.2.

No	Bulan	Sudut	Lux	Cuaca	Tampak
Tanggal					
2023-03-04	45	March	75	Nan	Berawan Utara
2023-03-23	292	March	75	Nan	Cerah Berawan Utara
2023-03-25	318	March	75	Nan	Berawan Utara
2023-04-09	513	April	75	Nan	Cerah Berawan Utara
2023-04-18	630	April	75	Nan	Cerah Berawan Utara
...
2024-02-11	4517	February	75	Nan	Cerah Utara
2024-02-14	4556	February	75	Nan	Berawan Utara
2024-02-17	4595	February	75	Nan	Cerah Berawan Utara
2024-02-21	4647	February	75	Nan	Cerah Berawan Utara
2024-02-28	4738	February	75	Nan	Berawan Utara

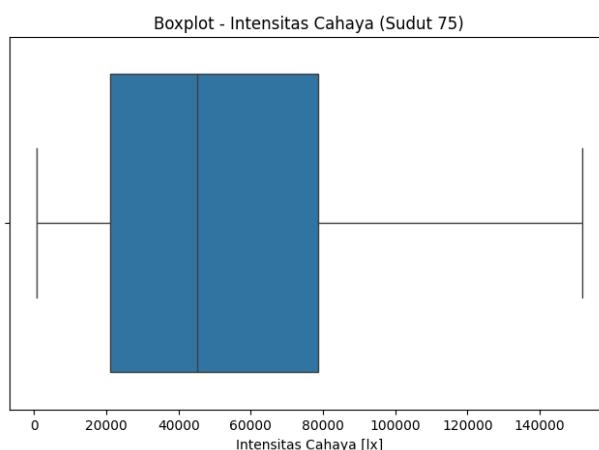
Gambar 4.2 Missing values pada dataset sudut 75°

Tabel 4.2 Hasil pemeriksaan missing values pada dataset sudut 75°

Variabel	Jumlah Missing value
No.	0
Bulan	0
Sudut	0
Lux	54
Cuaca	0
Tampak	0

Berdasarkan Tabel 4.2, teridentifikasi bahwa hanya variabel Lux yang mengandung *missing values*, yaitu sebanyak 54 entri. Mengingat data dikumpulkan secara harian selama satu tahun (sekitar 366 hari), jumlah ini merepresentasikan sekitar 14.7% dari total data pada sudut 75° , yang menandakan adanya hari-hari tanpa pencatatan. Adanya proporsi data hilang yang cukup signifikan ini akan menjadi perhatian utama yang harus ditangani pada tahap pra-pemrosesan data.

Selanjutnya, untuk mengidentifikasi *outliers* pada variabel Lux, digunakan metode visualisasi berupa *boxplot* seperti yang ditunjukkan pada Gambar 4.3.



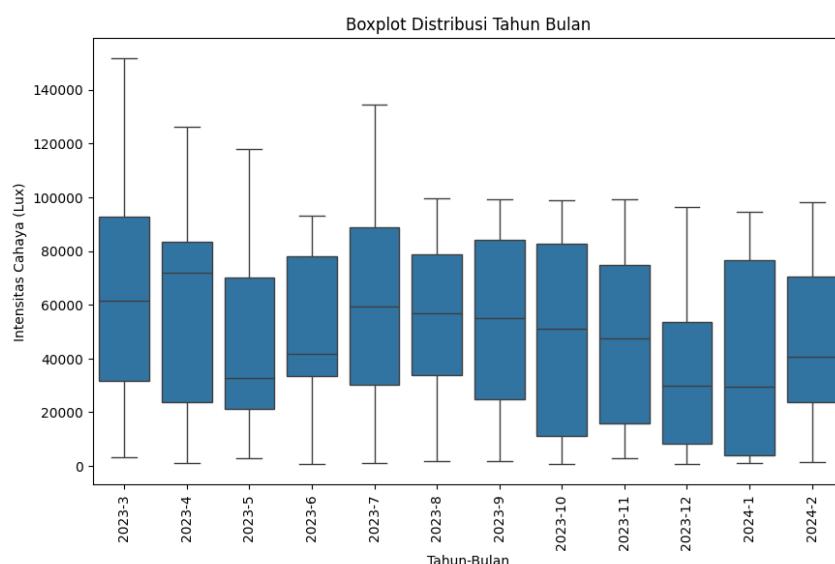
Gambar 4.3 Boxplot intensitas cahaya

Berdasarkan boxplot tersebut, dapat diamati bahwa distribusi data Lux memiliki rentang yang sangat luas, dengan *whisker* yang panjang mengindikasikan

keberadaan nilai-nilai ekstrem yang secara statistik dapat dianggap sebagai *outlier*. Namun, dalam konteks data iklim, nilai-nilai ekstrem ini bukanlah anomali data atau kesalahan pengukuran, melainkan representasi valid dari kondisi cuaca nyata di Jakarta Selatan. Nilai yang sangat tinggi merepresentasikan hari cerah tanpa tutupan awan, sementara nilai yang sangat rendah mencerminkan kondisi mendung tebal atau hujan lebat. Menghapus *outlier* ini justru akan menghilangkan informasi krusial dan menciptakan model yang bias, yang hanya mampu memprediksi kondisi cuaca ideal. Oleh karena itu, untuk membangun model prediksi yang tangguh dan memiliki kemampuan generalisasi yang baik di dunia nyata, seluruh titik data, termasuk nilai-nilai ekstrem ini, akan dipertahankan dalam *dataset*.

4.1.4 Distribusi Data

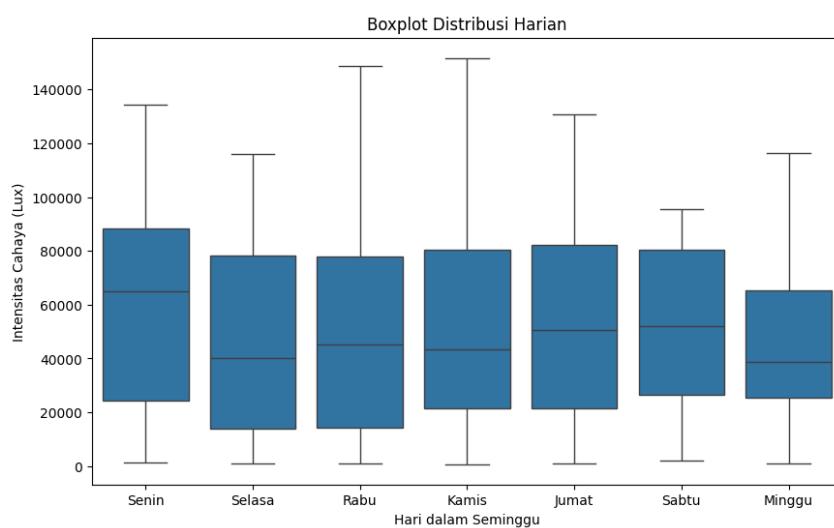
Analisis distribusi data bertujuan untuk memahami bagaimana nilai intensitas cahaya (Lux) tersebar. Informasi ini penting untuk mengidentifikasi pola temporal, seperti tren musiman dan mingguan, yang krusial untuk pemodelan deret waktu. Analisis ini ditinjau dari dua dimensi: distribusi bulanan dan harian.



Gambar 4.4 Boxplot distribusi data bulanan

Gambar 4.4 menyajikan perbandingan distribusi intensitas cahaya untuk setiap bulan dari Maret 2023 hingga Februari 2024. Analisis terhadap *boxplot* bulanan menunjukkan bahwa terdapat indikasi pola musiman pada intensitas cahaya. Median cenderung lebih tinggi pada bulan-bulan yang umumnya merupakan musim kemarau, seperti Maret 2023 dan Juli 2023. Sebaliknya, bulan-bulan yang identik dengan puncak musim hujan seperti Desember 2023 dan Januari 2024 menunjukkan median dan rentang data yang lebih rendah.

Variabilitas intensitas cahaya (direpresentasikan oleh ukuran kotak/IQR) juga beragam. Bulan Maret 2023 dan April 2023 menunjukkan variabilitas yang sangat tinggi, menandakan fluktuasi kondisi cuaca yang signifikan dalam bulan-bulan tersebut. Beberapa bulan, terutama Maret 2023 juga memiliki *whisker* atas yang sangat panjang, mengindikasikan adanya hari-hari dengan intensitas cahaya yang sangat tinggi (ekstrem) pada periode tersebut.



Gambar 4.5 Boxplot distribusi data harian

Distribusi intensitas cahaya juga dianalisis berdasarkan hari dalam seminggu, seperti terlihat pada Gambar 4.5. Analisis terhadap *boxplot* harian

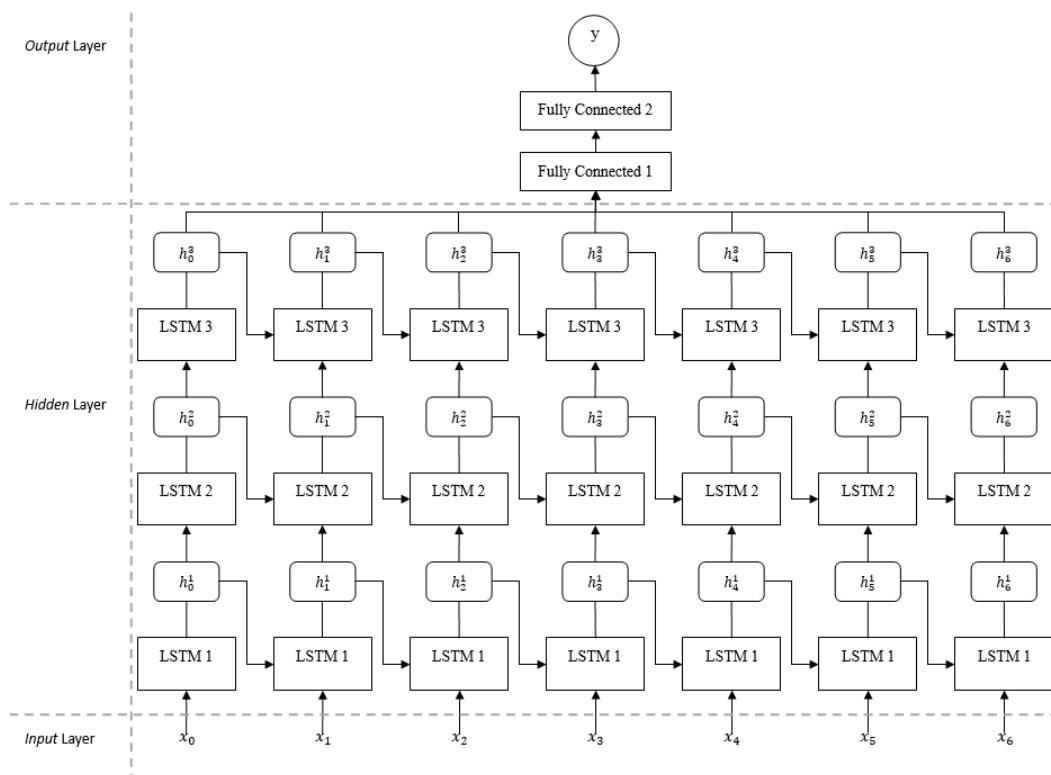
menunjukkan terdapat pola variasi antar hari. Hari Senin dan Jumat tercatat memiliki median intensitas cahaya yang cenderung lebih tinggi dibandingkan hari lainnya. Hari Minggu secara konsisten menunjukkan median dan jangkauan data yang paling rendah, yang mungkin mengindikasikan adanya pola aktivitas atau kondisi atmosferik yang berbeda pada hari tersebut. Variabilitas data tertinggi terlihat pada hari Minggu dan Senin. Sementara itu, nilai ekstrem tertinggi (puncak *whisker*) tercatat pada hari Rabu dan Kamis.

Secara keseluruhan, hasil dari *Exploratory Data Analysis* (EDA) ini memberikan dua justifikasi fundamental untuk metodologi penelitian yang akan diterapkan. Pertama, identifikasi adanya pola temporal yang kuat, baik secara bulanan maupun mingguan, secara eksplisit mendukung pemilihan model sekuensial seperti LSTM, yang dirancang khusus untuk menangkap dependensi waktu dalam data. Kedua, karakteristik data yang sangat fluktuatif dan non-stasioner, yang dikonfirmasi melalui simpangan baku yang tinggi dan keberadaan nilai-nilai ekstrem yang valid, menegaskan pentingnya analisis komparatif terhadap *optimizer*. Kemampuan *optimizer* untuk menavigasi permukaan *loss function* yang kompleks pada data seperti ini akan menjadi faktor penentu keberhasilan model. Dengan pemahaman mendalam terhadap karakteristik data ini, langkah selanjutnya adalah melakukan pra-pemrosesan data untuk persiapan tahap pemodelan.

4.2 Perancangan Model

Sesuai dengan kerangka metodologi yang telah dipaparkan pada Bab 3, perancangan model dalam penelitian ini diimplementasikan menggunakan arsitektur Stacked *Long Short-Term Memory* (LSTM). Pemilihan arsitektur

bertumpuk ini didasarkan pada temuan dari tahap EDA, yang mengungkap adanya pola temporal berlapis dan kompleks pada data iradiasi matahari. Dengan menempatkan tiga lapisan LSTM secara berurutan, model dirancang untuk mampu mengekstraksi fitur pada berbagai tingkat abstraksi—mulai dari fluktuasi harian hingga tren musiman jangka panjang—sehingga dapat memodelkan data yang non-stasioner secara lebih efektif.



Gambar 4.6 Arsitektur model Stacked LSTM.

Sebelum diproses oleh model, data *input* dibentuk menggunakan teknik jendela geser (*sliding window*) dengan panjang 7 timesteps (7 hari) untuk memprediksi nilai pada *timestep* berikutnya (hari ke-8). Proses ini diilustrasikan secara konseptual oleh *input* x_0 hingga x_6 pada Gambar 4.6, yang bersama-sama membentuk satu sampel *input* untuk model. Dengan demikian, bentuk data

masukan (*input shape*) untuk model adalah sekuens dengan 7 timesteps dan 1 fitur (nilai iradiasi). Bentuk data masukan (*input shape*) ini, yaitu sekuens dengan 7 timesteps dan 1 fitur, dirancang secara spesifik untuk menangkap potensi siklus mingguan dalam data.

Detail teknis dari arsitektur yang diimplementasikan dirangkum pada Tabel 4.3, yang merupakan ringkasan model dari *library* TensorFlow Keras.

Tabel 4.3 Ringkasan arsitektur model Stacked LSTM

<i>Layer</i>	<i>Output Shape</i>	<i>Trainable parameters</i>
lstm	(None, 7, 128)	66.560
lstm_1	(None, 7, 64)	49.408
lstm_2	(None, 32)	12.416
dense	(None, 32)	10.56
dense_1	(None, 16)	528
dense_2	(None, 1)	17

Arsitektur ini terdiri dari dua blok fungsional utama: blok pemrosesan sekuensial dan blok regresi. Blok pertama, yang terdiri dari tiga lapisan LSTM (lstm, lstm_1, lstm_2) dengan jumlah unit menurun (128, 64, 32), bertugas untuk memproses sekuens *input* dan merangkum informasi temporal menjadi sebuah vektor fitur tunggal. Vektor ini kemudian diumpulkan ke blok regresi, yang terdiri dari tiga lapisan Dense (dense, dense_1, dense_2), untuk memetakan representasi yang telah dipelajari menjadi satu nilai prediksi akhir.

Secara keseluruhan, arsitektur ini memiliki total 129.985 parameter yang dapat dilatih (*trainable parameters*). Penting untuk ditekankan bahwa arsitektur yang sama dan konsisten ini digunakan sebagai fondasi atau variabel kontrol dalam

seluruh percobaan. Dengan menjaga arsitektur model tetap konstan, penelitian ini dapat secara objektif mengisolasi dan menganalisis dampak dari variabel yang diuji, yaitu performa dari ketiga *optimizer* (SGD, RMSProp, dan Adam) dalam melatih model.

4.3 Data Preprocessing

Tahap pra-pemrosesan data merupakan langkah krusial untuk mentransformasi data mentah menjadi format yang bersih, terstruktur, dan siap untuk pemodelan *Machine Learning*. Proses ini mencakup serangkaian langkah yang sistematis untuk memastikan kualitas dan kesesuaian data.

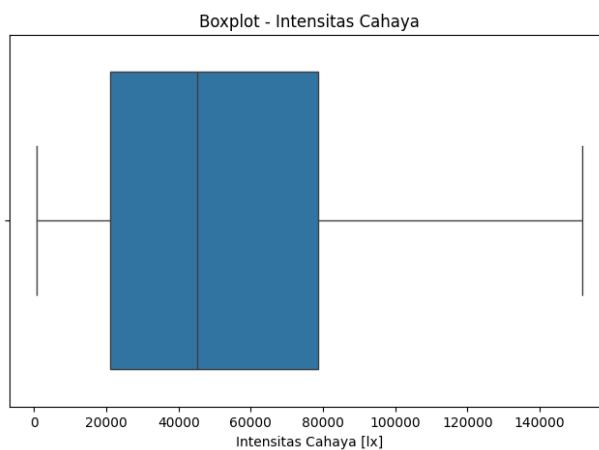
4.3.1 Penanganan *Missing value* dan *Outlier*

Sebagaimana teridentifikasi pada tahap EDA, terdapat 54 entri data yang hilang (*missing values*) pada variabel Lux. Untuk mengatasinya, diterapkan metode imputasi acak bersyarat (*conditional random imputation*). Metode ini dipilih karena kemampuannya untuk mempertahankan distribusi statistik dan variabilitas alami data, tidak seperti metode imputasi sederhana (misalnya, pengisian dengan nilai rata-rata atau median) yang dapat mereduksi varians dan menghilangkan informasi penting mengenai fluktuasi cuaca. Prosesnya adalah dengan mengisi nilai Lux yang hilang menggunakan angka acak dari rentang nilai minimum dan maksimum yang pernah tercatat, berdasarkan kondisi Bulan dan Cuaca yang sesuai, seperti yang dirincikan pada Tabel 4.4.

Kemudian terkait *outliers*, sesuai dengan analisis pada sub-bab 4.1.3, nilai-nilai ekstrem dianggap sebagai data valid yang merepresentasikan kondisi cuaca nyata. Oleh karena itu, tidak dilakukan penghapusan atau modifikasi *outlier* untuk menjaga integritas dan representasi data yang sebenarnya.

Tabel 4.4 Threshold metode imputasi data

Bulan	Cerah		Cerah Berawan		Berawan		Mendung		Hujan	
	Min	Max	Min	Max	Min	Max	Min	Max	Min	Max
Januari	69350	78870	3550	94624	2154	77990	14370	50770	974	4395
Februari	67970	98040	1373	95300	21030	50770	21030	50770	2817	3287
Maret	33100	151700	40930	96810	2154	77990	14370	50220	3409	4193
April	45760	126100	1105	94804	21370	77990	14370	50770	845	4395
Mei	60460	117900	2972	34380	14300	17340	14370	50770	845	4395
Juni	82660	93150	3564	45360	18130	73649	14370	50770	845	3260
Juli	57830	134300	1085	89074	19663	59413	14370	50770	845	4395
Agustus	14330	99460	1813	95442	2154	77990	14370	50770	845	4395
September	10170	99130	1847	86091	2154	77990	14370	50770	845	4395
Oktober	1284	98830	706	90262	2154	77990	14370	50770	845	4395
November	10870	99370	2845	82061	3047	43667	14370	50770	845	4395
Desember	906	55620	923	96227	3262	72303	14370	50770	845	4395

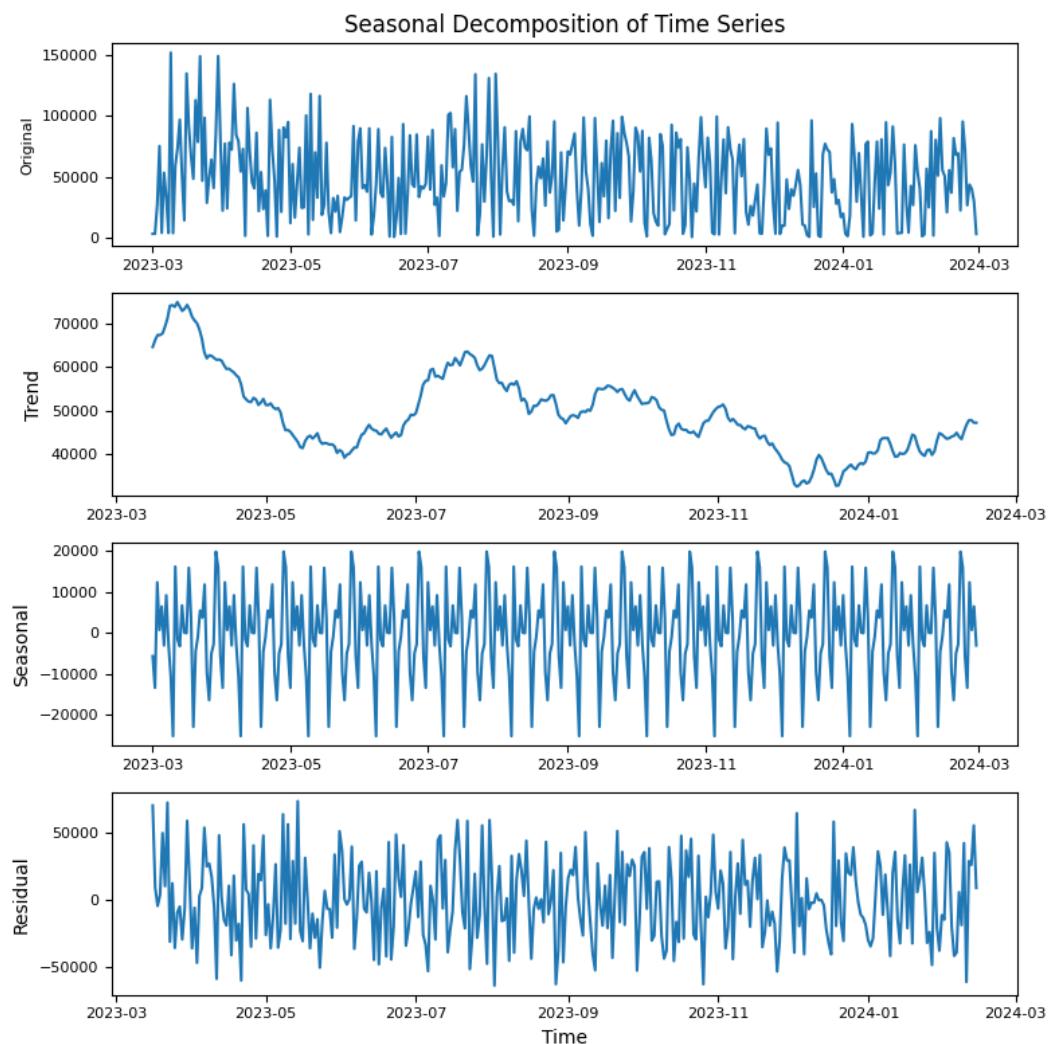


Gambar 4.7 Boxplot intensitas cahaya setelah imputasi

Hasil visualisasi setelah imputasi pada Gambar 4.7 menunjukkan profil distribusi yang sangat mirip dengan kondisi data sebelum imputasi. Hal ini mengonfirmasi bahwa metode imputasi berhasil mengisi *missing values* tanpa merusak distribusi alaminya.

4.3.2 Dekomposisi Deret Waktu

Untuk memahami dinamika data secara lebih mendalam, dilakukan analisis dekomposisi deret waktu. Proses ini, yang diimplementasikan menggunakan model dekomposisi aditif, memisahkan data Lux (setelah imputasi) menjadi tiga komponen utama utama: tren jangka panjang (*Trend*), pola berulang musiman (*Seasonal*), variasi acak (*Residual*), seperti yang ditunjukkan pada Gambar 4.8. Analisis ini krusial untuk mengonfirmasi keberadaan pola-pola yang dapat dipelajari oleh model LSTM.



Gambar 4.8 Hasil dekomposisi time plot lux

Komponen Tren (*Trend*) merepresentasikan arah atau pergerakan jangka panjang dari data, setelah fluktuasi jangka pendek dihilangkan. Secara teknis, tren ini diekstrak dengan menghitung rata-rata bergerak (*moving average*) pada data asli. Hasilnya (panel kedua) secara jelas menunjukkan adanya siklus tahunan, di mana intensitas cahaya cenderung meningkat dari awal tahun, mencapai puncaknya sekitar Agustus 2023, dan kemudian menurun menuju awal tahun 2024.

Komponen Musiman (*Seasonal*) menangkap pola-pola yang berulang secara periodik dalam interval waktu yang tetap. Setelah komponen tren dihilangkan dari data asli (proses *deTrending*), nilai rata-rata untuk setiap periode (dalam kasus ini, periode bulanan) dihitung untuk mengisolasi pola musiman ini. Panel ketiga berhasil mengungkap adanya pola periodik dengan frekuensi tinggi yang konsisten sepanjang tahun merefleksikan siklus cuaca lokal atau bulanan.

Komponen Sisa (Residual) adalah sisa dari data asli setelah komponen Tren dan Musiman dihilangkan (Data Asli - Tren - Musiman). Idealnya, residual ini merepresentasikan derau acak (*random noise*) yang tidak memiliki pola. Panel terakhir menunjukkan bahwa sebagian besar nilai residual berfluktuasi di sekitar nol, yang menandakan model dekomposisi berhasil menangkap sebagian besar struktur data. Namun, adanya beberapa puncak tajam mengindikasikan keberadaan variasi acak atau pengaruh faktor tak terduga, seperti kondisi cuaca ekstrem sesaat.

Analisis dekomposisi ini berfungsi sebagai validasi akhir sebelum pemodelan. Keberadaan komponen Tren dan Musiman yang jelas dan kuat secara eksplisit memvalidasi pemilihan model sekuensial seperti LSTM, karena model ini

dirancang secara khusus untuk dapat mempelajari dan memprediksi pola-pola temporal yang kompleks tersebut.

4.3.3 Seleksi Fitur

Tahap seleksi fitur bertujuan untuk menyederhanakan model dan memfokuskan pembelajaran pada informasi yang paling signifikan. Dalam penelitian ini, diputuskan bahwa model yang akan dibangun adalah model deret waktu univariat. Artinya, model hanya akan menggunakan data historis dari Lux (yang kemudian dikonversi menjadi iradiasi) untuk memprediksi nilai di masa depan. Fitur-fitur lain yang ada dalam *dataset* dieliminasi berdasarkan pertimbangan berikut:

- Sudut: Variabel ini dihilangkan karena nilainya konstan (75°) setelah data disaring pada tahap sebelumnya. Fitur dengan nilai konstan tidak memberikan informasi prediktif apa pun kepada model.
- Tampak: Sama seperti Sudut, variabel ini juga dieliminasi karena memiliki nilai yang seragam ("Utara") di seluruh data yang digunakan, sehingga tidak memiliki daya prediksi.
- Bulan: Informasi temporal dari variabel Bulan sudah secara implisit terkandung di dalam *DateTimeIndex dataset*. Model sekuensial seperti LSTM dirancang untuk menangkap pola musiman langsung dari urutan data tanpa memerlukan fitur bulan secara eksplisit.
- Cuaca: Meskipun kondisi cuaca berpengaruh terhadap intensitas cahaya, variabel Cuaca sengaja tidak disertakan. Keputusan ini diambil untuk menguji hipotesis bahwa nilai Lux itu sendiri secara

implisit sudah merefleksikan kondisi cuaca (misalnya, nilai Lux akan rendah saat cuaca mendung dan tinggi saat cerah). Dengan demikian, model ditantang untuk belajar hanya dari data historis Lux tanpa bergantung pada data eksternal tambahan.

Dengan strategi ini, pemodelan menjadi lebih efisien dan fokus pada kemampuan LSTM dalam mengekstrak pola dari satu deret waktu tunggal.

4.3.4 Konversi Lux ke Iradiasi

	Lux	SolarIrradiation
Tanggal		
2023-03-01	3409.0	27.94
2023-03-02	3412.0	27.97
2023-03-03	27230.0	223.20
2023-03-04	39819.0	326.39
2023-03-05	4193.0	34.37
...
2024-02-25	26761.0	219.35
2024-02-26	43650.0	357.79
2024-02-27	40102.0	328.70
2024-02-28	37286.0	305.62
2024-02-29	3287.0	26.94

Gambar 4.9 Hasil konversi lux ke iradiasi

Setelah tahap seleksi fitur, langkah pra-pemrosesan selanjutnya adalah mengonversi variabel target dari satuan intensitas cahaya (lux) menjadi satuan iradiasi matahari (W/m^2). Proses konversi ini dilakukan berdasarkan konstanta yang diacu dari studi oleh Michael, dkk. (2020). Studi tersebut merekomendasikan bahwa untuk kondisi sinar matahari alami di luar ruangan, nilai konversi yang dapat digunakan adalah $1 \text{ W/m}^2 \approx 122 \text{ lx}$. Gambar 4.9 menampilkan sampel *dataset* setelah proses konversi, di mana setiap nilai pada kolom Lux telah memiliki padanan nilai pada kolom *SolarIrradiation*.

4.3.5 Normalisasi Data

SolarIrradiation	
Tanggal	
2023-03-01	0.017897
2023-03-02	0.017921
2023-03-03	0.175664
2023-03-04	0.259039
2023-03-05	0.023092
...	...
2024-02-25	0.172553
2024-02-26	0.284410
2024-02-27	0.260906
2024-02-28	0.242258
2024-02-29	0.017089

Gambar 4.10 Hasil normalisasi iradiasi

Tahap pra-pemrosesan selanjutnya adalah normalisasi. Tujuan dari langkah ini adalah untuk mengubah skala nilai pada variabel *SolarIrradiation* agar berada dalam rentang yang seragam, tanpa mengubah distribusi relatifnya.

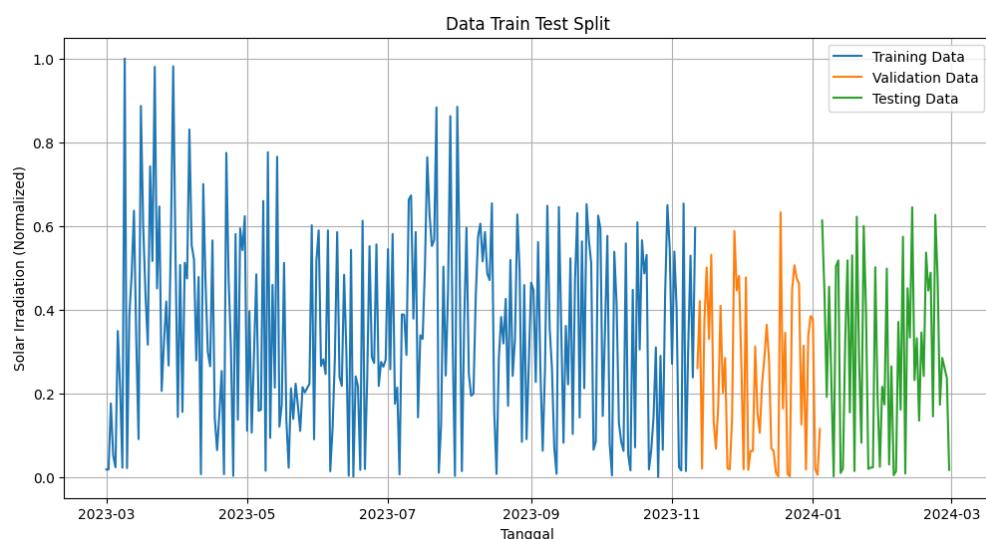
Teknik normalisasi yang diterapkan dalam penelitian ini adalah *Min-Max Scaling*. Teknik ini mentransformasikan setiap nilai data ke dalam rentang antara [0, 1]. Normalisasi sangat penting untuk model berbasis *gradient descent* seperti LSTM, karena dapat meningkatkan stabilitas numerik, mempercepat laju konvergensi selama proses pelatihan, dan mencegah nilai dengan magnitudo besar mendominasi proses pembelajaran. Gambar 4.10 menampilkan sampel data *SolarIrradiation* setelah dinormalisasi, di mana terlihat semua nilainya kini berada dalam rentang [0, 1].

4.3.6 Pembagian dan Segmentasi Data

Tahap akhir dari pra-pemrosesan adalah membagi dan menstrukturkan data untuk proses pelatihan dan pengujian. Proses ini terdiri dari dua langkah yaitu pembagian data menjadi set pelatihan, validasi dan pengujian, diikuti oleh segmentasi data menggunakan teknik *sliding window*.

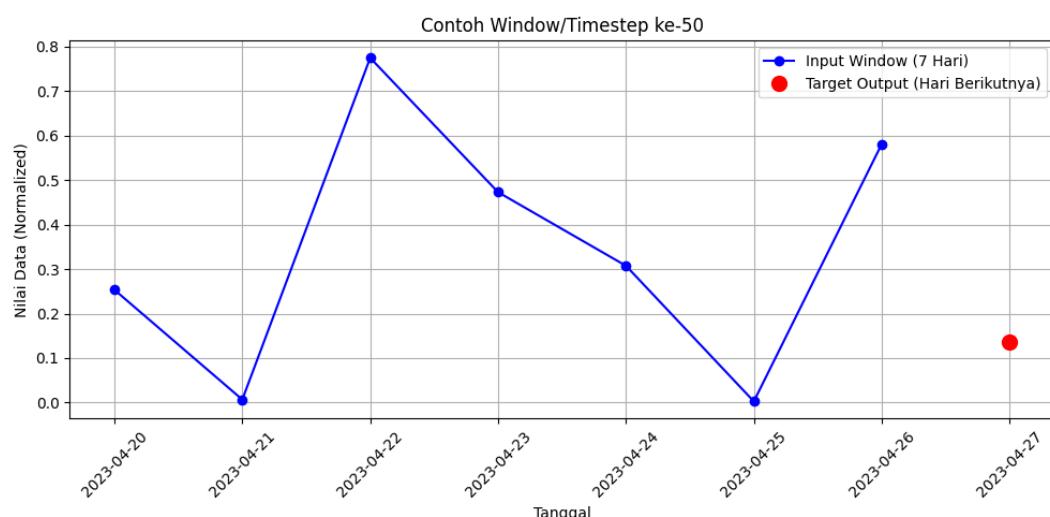
Dataset yang terdiri dari 366 data harian dibagi secara kronologis menjadi tiga set berbeda dengan rasio 70% untuk data latih, 15% untuk data validasi, dan 15% untuk data uji. Pembagian ini penting untuk mengevaluasi kemampuan generalisasi model secara objektif pada data yang belum pernah dilihat sebelumnya (*unseen data*).

Data Latih (*Training Set*) terdiri dari 256 hari pertama, yaitu dari periode 1 Maret 2023 – 11 November 2023. Data Validasi (*Validation Set*) terdiri dari 55 hari berikutnya, yaitu dari periode 12 November 2023 – 5 Januari 2024. Dan Data Uji (*Testing Set*) terdiri dari 55 hari terakhir, yaitu dari periode 6 Januari 2024 – 29 Februari 2024. Pembagian secara kronologis ini penting untuk simulasi kondisi nyata, di mana model memprediksi masa depan berdasarkan data masa lalu.



Gambar 4.11 Pembagian dataset

Setelah dibagi, ketiga set data tersebut disegmentasi untuk menciptakan sekuens *input-output* yang dapat dipelajari oleh model LSTM. Sesuai dengan rancangan model, ukuran jendela (*window size*) yang digunakan adalah 7 *timesteps*. Artinya, tujuh data historis berurutan digunakan sebagai *input* (X) untuk memprediksi data pada hari ke-8 sebagai *output* (y). Proses segmentasi ini menghasilkan 249 sampel untuk set pelatihan, 48 sampel untuk set validasi, dan 48 untuk set pengujian.



Gambar 4.12 Sampel segmentasi data 7 timestep

4.4 Pelatihan dan Evaluasi Kinerja Model

Setelah arsitektur model final dirancang, dan data telah melalui tahap pra-pemrosesan. Tahap selanjutnya adalah melatih model dan mengevaluasi kinerjanya secara sistematis. Proses evaluasi dilakukan dalam dua tahap utama untuk menganalisis kinerja setiap konfigurasi.

Tahap evaluasi pertama adalah analisis kualitatif terhadap histori pelatihan (*training history*). Proses ini difokuskan pada pemantauan grafik *training loss* dan *Mean Absolute Error* (MAE) untuk setiap *epoch*. Tahap kedua adalah evaluasi

kuantitatif berdasarkan metrik kinerja model pada data latih. Setelah pelatihan selesai, kinerja dari model diukur menggunakan *Mean Absolute Error* (MAE), *Root Mean Squared Error* (RMSE), dan *Mean Absolute Percentage Error* (MAPE).

Untuk memastikan perbandingan yang adil dan terkontrol, dirancang 12 skenario percobaan. Dalam semua skenario ini, beberapa variabel dijaga konstan (*controlled variables*), yaitu:

- Arsitektur Model: Arsitektur *Stacked LSTM* yang sama (sub-bab 4.2) digunakan untuk semua percobaan.
- Fungsi Kerugian (*Loss function*): Semua model menggunakan *Mean Squared Error* (MSE).
- Jumlah *Epoch*: Setiap model dilatih selama 150 *epoch*.
- *Batch* Pelatihan: *Batch* dijaga konstan pada ukuran *batch* 1.

Sementara itu, variabel yang diuji (*tested variables*) adalah konfigurasi *optimizer*. Tiga jenis *optimizer* dievaluasi dengan masing-masing empat variasi *hyperparameter*, yang mencakup kombinasi *learning rate* (tinggi: 0.01 dan rendah: 0.001) dan parameter internalnya. Seluruh kombinasi skenario percobaan dirangkum pada Tabel 4.5.

Tabel 4.5 Kombinasi percobaan berdasarkan optimizer dan hyperparameter

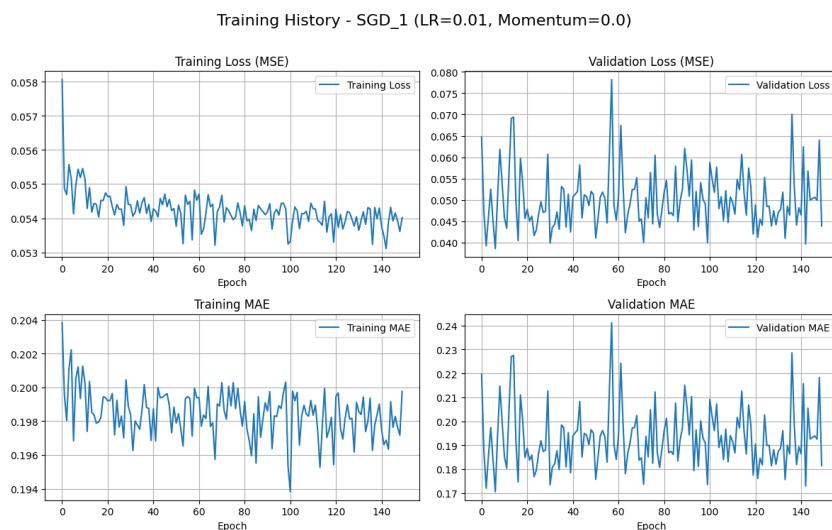
Percobaan	Optimizer	Learning rate	Parameter
Percobaan 1	SGD	0.01	<i>Momentum</i> = 0
Percobaan 2	SGD	0.01	<i>Momentum</i> = 0.9
Percobaan 3	SGD	0.001	<i>Momentum</i> = 0
Percobaan 4	SGD	0.001	<i>Momentum</i> = 0.9
Percobaan 5	RMSProp	0.01	β = 0.9
Percobaan 6	RMSProp	0.01	β = 0.5

Percobaan 7	RMSProp	0.001	$\beta = 0.9$
Percobaan 8	RMSProp	0.001	$\beta = 0.5$
Percobaan 9	Adam	0.01	$\beta_1 = 0.9, \beta_2 = 0.99$
Percobaan 10	Adam	0.01	$\beta_1 = 0.5, \beta_2 = 0.85$
Percobaan 11	Adam	0.001	$\beta_1 = 0.9, \beta_2 = 0.99$
Percobaan 12	Adam	0.001	$\beta_1 = 0.5, \beta_2 = 0.85$

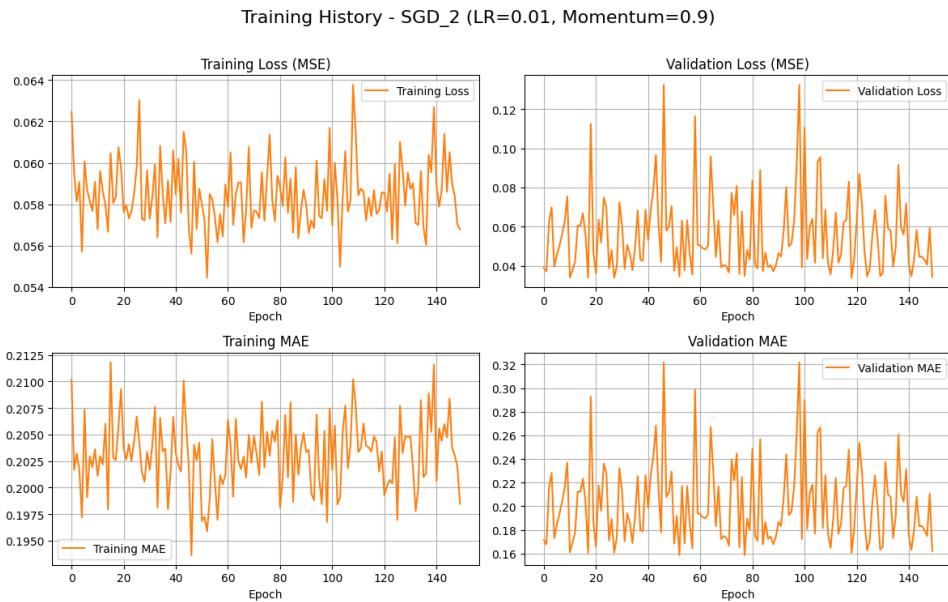
4.4.1 Hasil Pelatihan *Optimizer SGD*

Analisis pertama berfokus pada empat skenario percobaan yang menggunakan *optimizer Stochastic Gradient Descent* (SGD) dengan variasi *learning rate* dan parameter internalnya yaitu *momentum*. Analisis dilakukan secara kualitatif melalui pengamatan grafik histori pelatihan dan secara kuantitatif melalui metrik evaluasi.

Percobaan 1 dan Percobaan 2 yang menggunakan *learning rate* tinggi (0.01) menunjukkan kurva loss dan MAE yang sangat fluktuatif (Gambar 4.13 dan 4.14). Khususnya pada Percobaan 2, penambahan *momentum* dengan *learning rate* tinggi menyebabkan ketidakstabilan yang parah, terlihat dari lonjakan-lonjakan tajam pada grafik validation loss. Hal ini mengindikasikan bahwa *optimizer* "melompati" titik minimum optimal karena langkah pembaruan bobot yang terlalu besar.

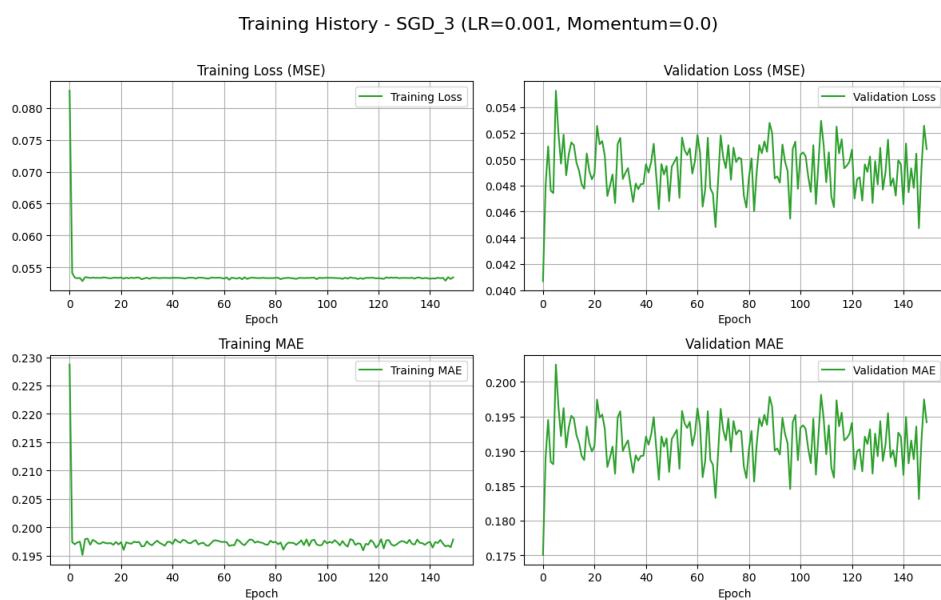


Gambar 4.13 Grafik hasil pelatihan *optimizer SGD* percobaan 1

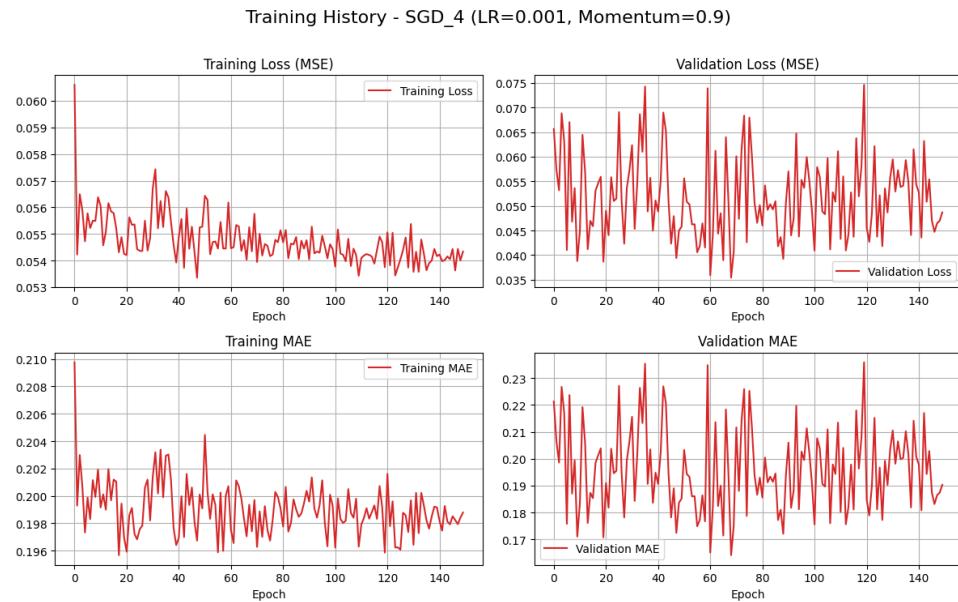


Gambar 4.14 Grafik hasil pelatihan optimizer SGD percobaan 2

Percobaan 3 pada Gambar 4.15 menunjukkan penurunan loss yang sangat tajam pada epoch awal, kemudian cepat mengalami stagnasi atau plateau. Ini menandakan model belajar dengan cepat di awal namun kemudian kesulitan untuk menemukan perbaikan lebih lanjut, kemungkinan terjebak di sebuah minimum lokal.

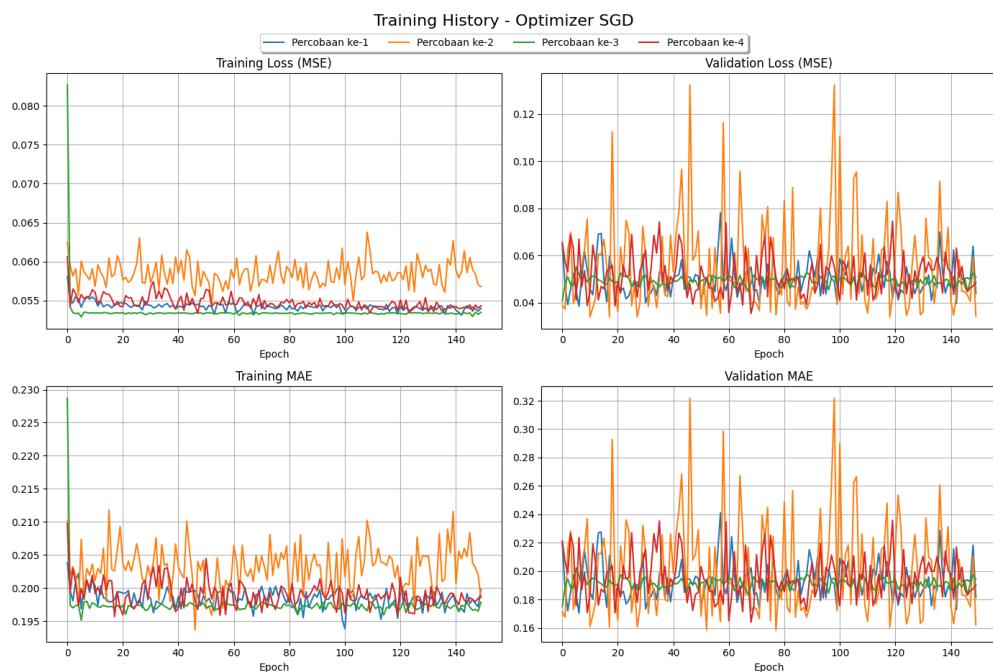


Gambar 4.15 Grafik hasil pelatihan optimizer SGD percobaan 3



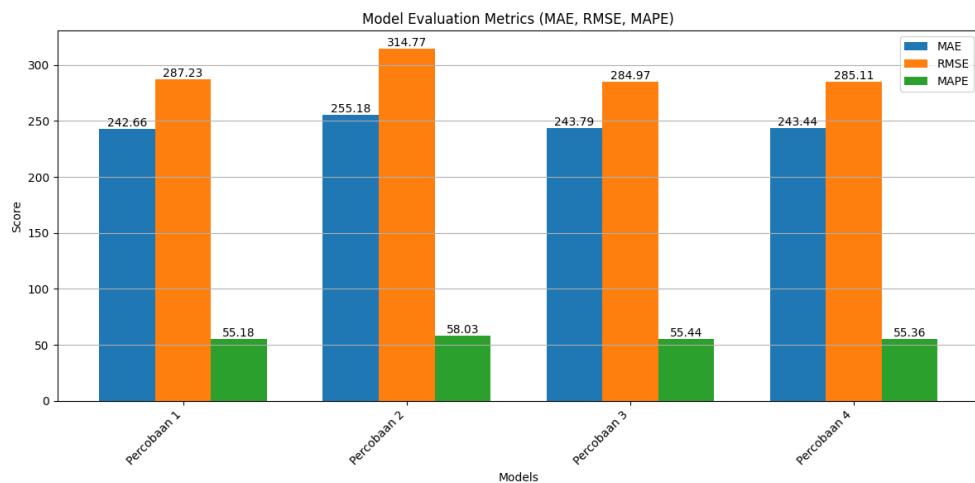
Gambar 4.16 Grafik hasil pelatihan optimizer SGD percobaan 4

Percobaan 4 pada Gambar 4.16 memperlihatkan kurva loss yang lebih stabil dibandingkan dengan skenario *learning rate* tinggi. Penambahan *momentum* tampak membantu model untuk terus belajar secara perlahan-lahan tanpa mengalami stagnasi separah Percobaan 3.



Gambar 4.17 Grafik perbandingan hasil pelatihan optimizer SGD

Untuk mendapatkan gambaran yang lebih pasti, kinerja keempat model dievaluasi secara kuantitatif menggunakan metrik MAE, RMSE, dan MAPE pada data latih, seperti yang dirangkum pada Gambar 4.18.



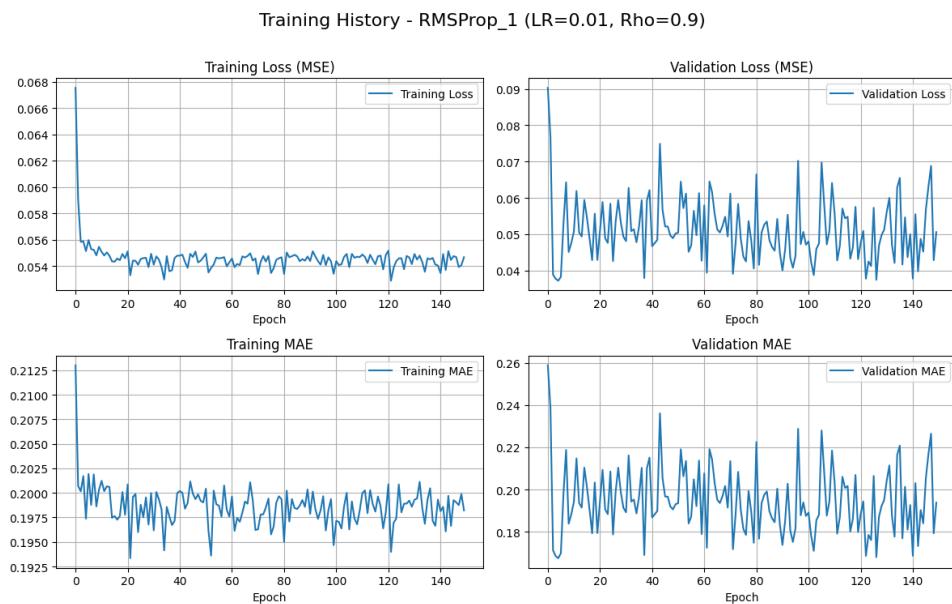
Gambar 4.18 Perbandingan metrik hasil pelatihan optimizer SGD

Berdasarkan diagram batang tersebut, hasil kuantitatif menunjukkan bahwa tidak ada satu percobaan pun yang unggul secara absolut di semua metrik. Percobaan 1 berhasil meraih nilai terendah untuk MAE (*Mean Absolute Error*) dengan skor 242.66 dan MAPE (*Mean Absolute Percentage Error*) sebesar 55.18%. Di sisi lain, Percobaan 3 mencatatkan nilai RMSE (*Root Mean Squared Error*) terendah dengan skor 284.97.

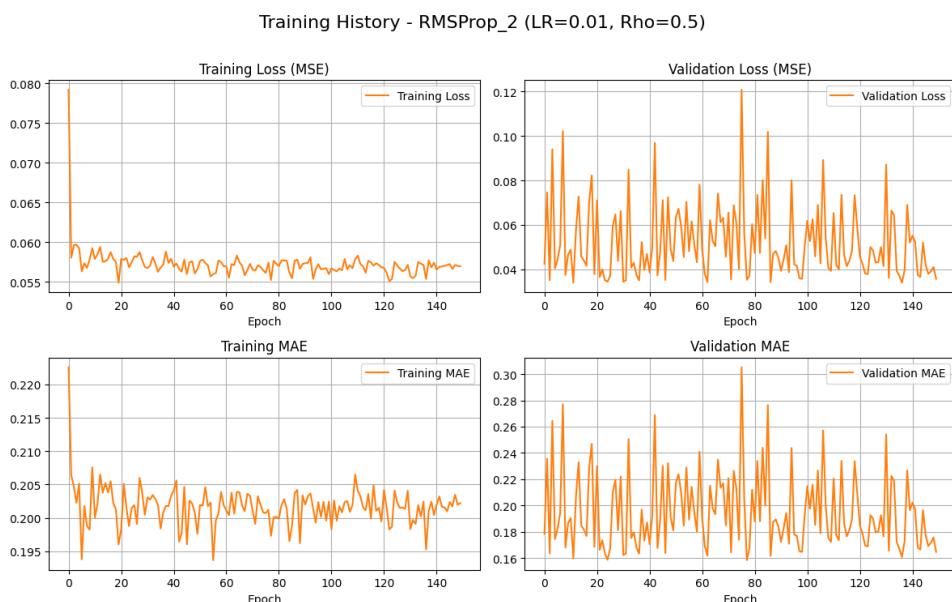
Dari hasil ini, dapat disimpulkan bahwa pemilihan *hyperparameter* pada *optimizer SGD* memiliki dampak yang sangat signifikan. Untuk *dataset* ini, SGD menunjukkan kinerja yang lebih baik dengan *learning rate* yang lebih tinggi (0.01) tetapi tanpa *momentum* (Percobaan 1), meskipun stabilitas pada data validasi masih rendah.

4.4.2 Hasil Pelatihan Optimizer RMSProp

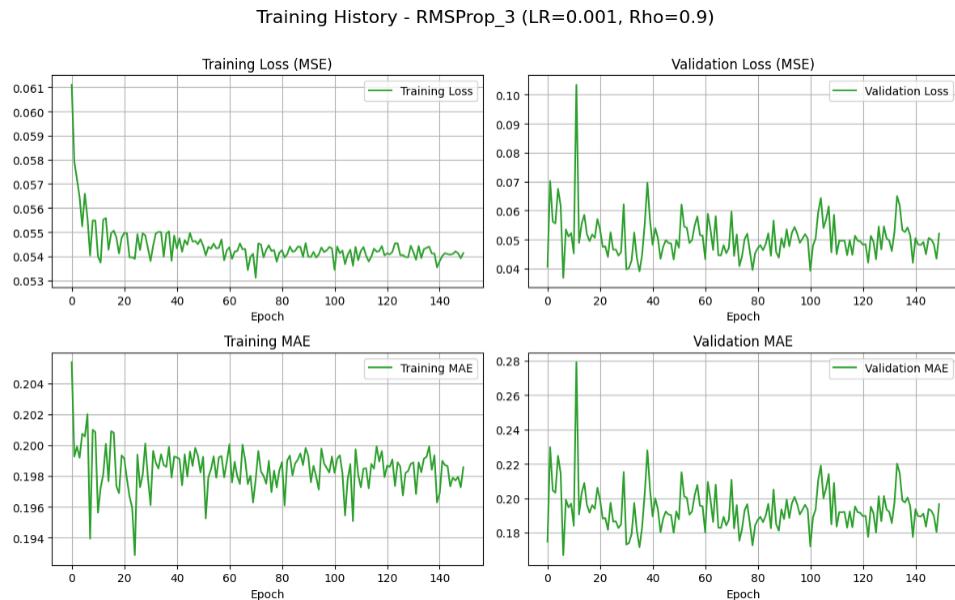
Selanjutnya, analisis dilakukan pada *optimizer Root Mean Square Propagation* (RMSProp). Empat skenario percobaan dirancang dengan memvariasikan *learning rate* dan parameter internalnya yaitu *decay rate* (B) yang juga dikenal sebagai rho.



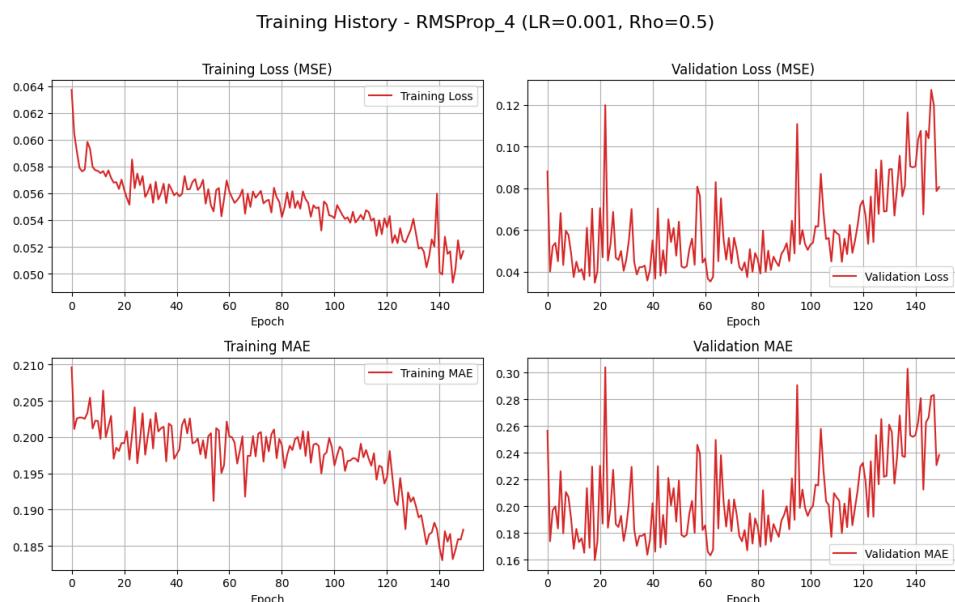
Gambar 4.19 Grafik hasil pelatihan optimizer RMSProp percobaan 5



Gambar 4.20 Grafik hasil pelatihan optimizer RMSProp percobaan 6



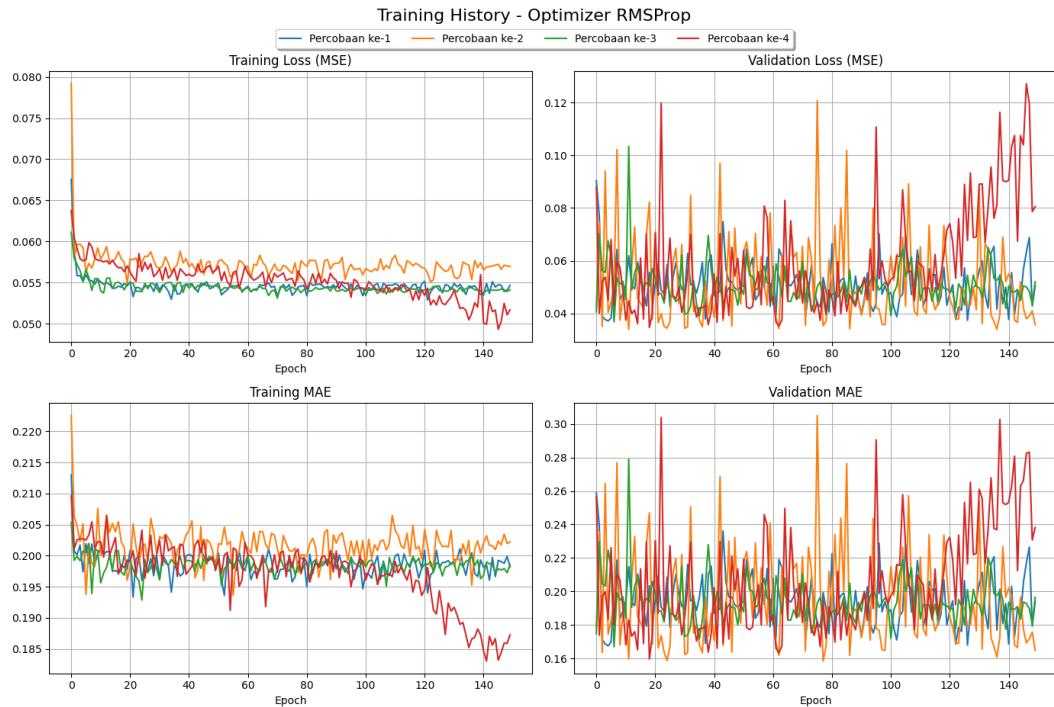
Gambar 4.21 Grafik hasil pelatihan optimizer RMSProp percobaan 7



Gambar 4.22 Grafik hasil pelatihan optimizer RMSProp percobaan 8

Serupa dengan *optimizer* sebelumnya, kurva validasi untuk semua percobaan RMSProp juga menunjukkan volatilitas yang tinggi. Namun, *learning rate* yang lebih rendah (0.001) pada Percobaan 7 dan 8 cenderung menghasilkan kurva validation loss yang lebih stabil dibandingkan *learning rate* tinggi (0.01) pada Percobaan 5 dan 6. Pengaruh parameter β , yang mengontrol bobot gradien

masa lalu, juga terlihat signifikan. Pada *learning rate* tinggi (0.01), percobaan 5 ($\beta=0.9$) menghasilkan pelatihan yang lebih stabil dibandingkan percobaan 6 ($\beta=0.5$) yang sangat fluktuatif.



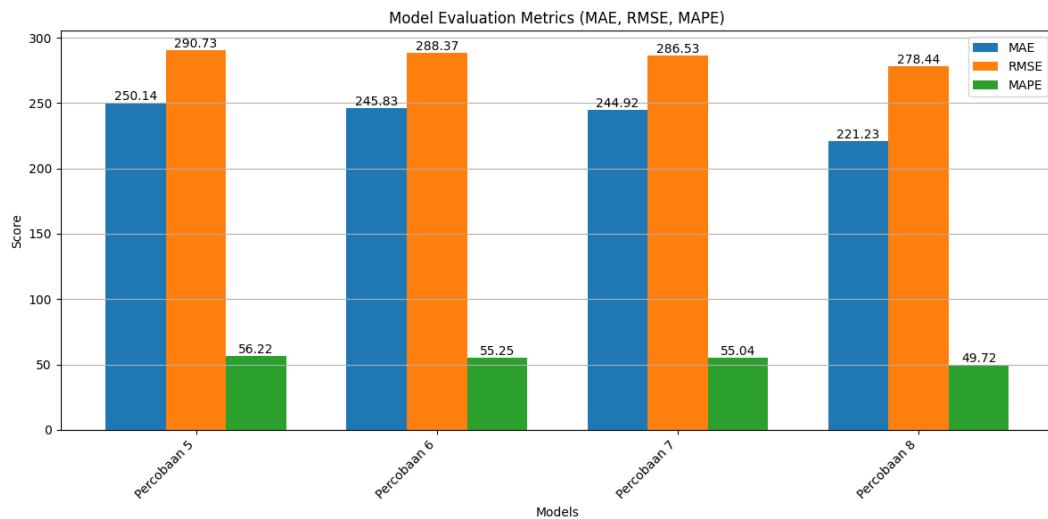
Gambar 4.23 Grafik perbandingan hasil pelatihan optimizer RMSProp

Dari grafik histori pelatihan (Gambar 4.19 - 4.22) dan perbandingannya

(Gambar 4.23), terlihat bahwa semua konfigurasi RMSProp menunjukkan konvergensi awal yang lebih cepat dibandingkan SGD. Nilai loss dan MAE pada data latih turun secara signifikan dalam sekitar 20 epoch pertama sebelum mulai bergerak stabil.

Namun, temuan paling menarik terjadi pada *learning rate* rendah. Percobaan 8 ($\eta = 0.001, \beta = 0.5$) menunjukkan perilaku unik di mana kurva *training loss* dan MAE terus menurun secara konsisten hingga akhir pelatihan, tidak seperti model lain yang cenderung stagnan setelah 40 epoch. Hal ini

mengindikasikan bahwa kombinasi *hyperparameter* tersebut memungkinkan model untuk terus belajar dan memperbaiki diri sepanjang proses pelatihan.



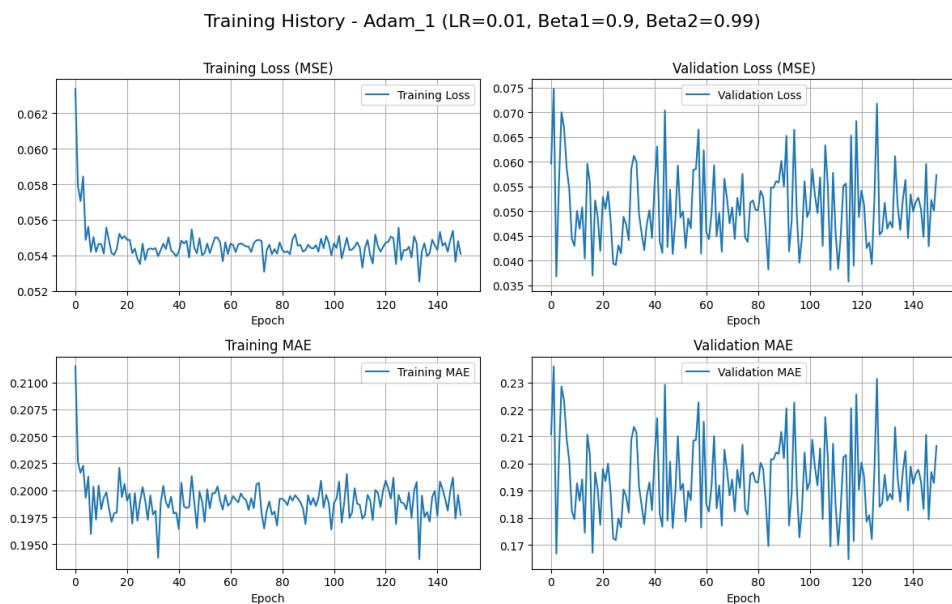
Gambar 4.24 Perbandingan metrik hasil pelatihan optimizer RMSProp

Analisis kuantitatif pada Gambar 4.24 memperkuat temuan dari analisis kualitatif. Dari diagram tersebut, Percobaan 8 secara meyakinkan menunjukkan keunggulan mutlak dibandingkan tiga konfigurasi lainnya, berhasil meraih nilai terendah di semua metrik: MAE sebesar 221.23, RMSE sebesar 278.44, dan MAPE sebesar 49.72%.

Dari hasil ini, dapat disimpulkan untuk *optimizer* RMSProp, Percobaan 8 dengan konfigurasi *learning rate* 0.001 dan $\beta = 0.5$ terbukti menjadi yang paling superior. Analisis kualitatif menunjukkan kemampuannya untuk terus belajar secara konsisten selama 150 epoch, sebuah karakteristik yang tidak dimiliki oleh konfigurasi lainnya. Keunggulan ini dikonfirmasi oleh hasil kuantitatif, di mana model ini berhasil mencatatkan nilai *error* terendah pada ketiga metrik evaluasi (MAE, RMSE, dan MAPE).

4.4.3 Hasil Pelatihan *Optimizer Adam*

Selanjutnya, analisis dilakukan pada *optimizer Adaptive Moment Estimation* (Adam), yang dikenal dengan kemampuannya menggabungkan keunggulan dari RMSProp dan *Momentum*. Histori pelatihan dari empat skenario dengan *optimizer* Adam disajikan pada Gambar 4.25 hingga 4.28.

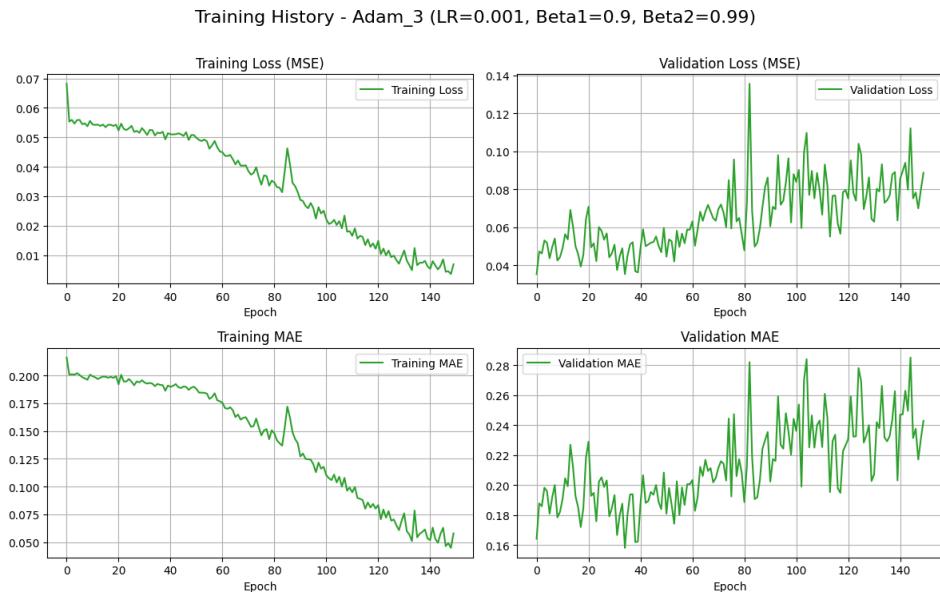


Gambar 4.25 Grafik hasil pelatihan optimizer Adam percobaan 9

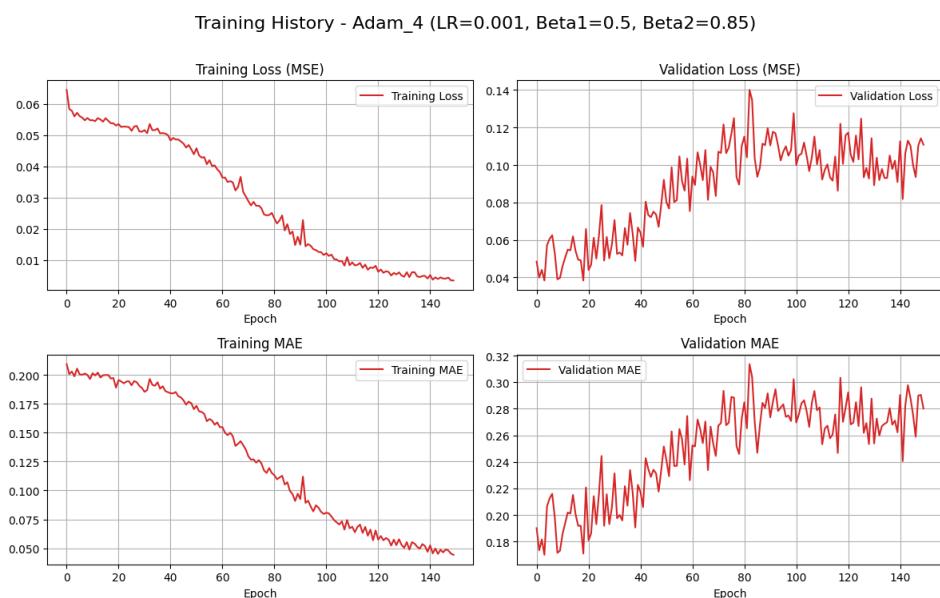


Gambar 4.26 Grafik hasil pelatihan optimizer Adam percobaan 10

Mirip dengan *optimizer* lainnya, penggunaan *learning rate* tinggi (0.01) pada percobaan 9 dan percobaan 10 menyebabkan model cepat mengalami stagnasi. Kurva *training loss* menurun di awal namun kemudian menjadi datar, menandakan tidak banyak perbaikan yang terjadi setelahnya. Kurva *validation loss* juga cenderung fluktuatif tanpa menunjukkan tren penurunan yang jelas.

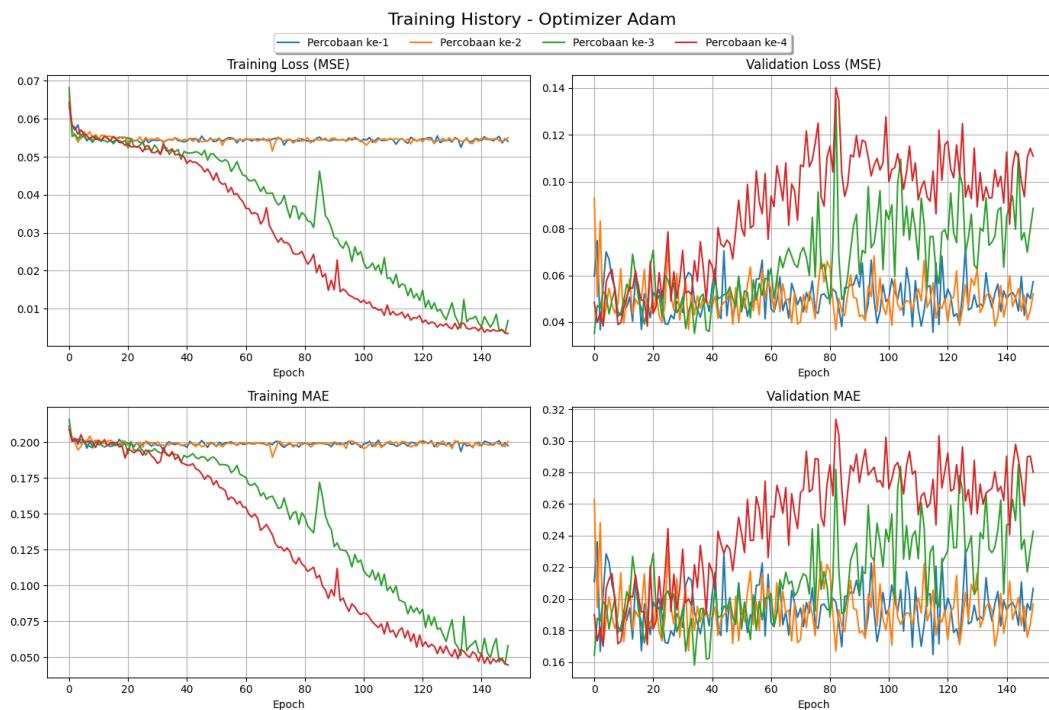


Gambar 4.27 Grafik hasil pelatihan optimizer Adam percobaan 11



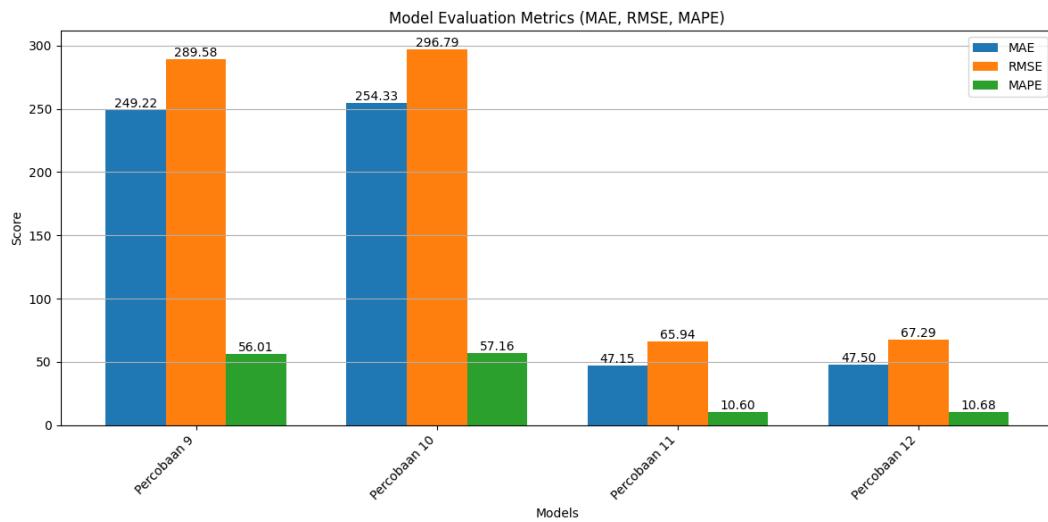
Gambar 4.28 Grafik hasil pelatihan optimizer Adam percobaan 12

Hasil yang sangat berbeda dan signifikan ditunjukkan oleh kedua percobaan dengan *learning rate* rendah (0.001). Baik Percobaan 11 (dengan parameter B default) maupun Percobaan 12 menunjukkan kemampuan belajar yang luar biasa. Kurva *training loss* dan training MAE terus menurun secara drastis dan konsisten sepanjang 150 epoch. Ini menunjukkan Adam dengan *learning rate* rendah sangat efektif dalam meminimalkan *error* pada data latih.



Gambar 4.29 Grafik perbandingan hasil pelatihan optimizer Adam
Namun, sebuah fenomena penting teridentifikasi pada plot validasi. Untuk

Percobaan 11 dan 12, kurva validation loss mulai menunjukkan tren kenaikan setelah sekitar epoch ke-80 (Gambar 4.27 dan 4.28). Ini adalah indikasi klasik dari overfitting, di mana model menjadi terlalu "hafal" dengan data latih sehingga performanya pada data validasi mulai menurun. Meskipun terjadi indikasi overfitting, metrik kuantitatif yang diukur pada akhir pelatihan (Gambar 4.30) menunjukkan hasil yang sangat baik untuk skenario *learning rate* rendah.



Gambar 4.30 Perbandingan metrik hasil pelatihan optimizer Adam

Terdapat perbedaan performa yang sangat besar antara kelompok *learning rate*. Percobaan 11 dan 12, yang menggunakan *learning rate* rendah, menunjukkan kinerja yang hampir identik dan lebih unggul dari semua percobaan dalam penelitian ini. Keunggulan ini terlihat jelas pada semua metrik: nilai MAE anjlok ke 47.15 (Percobaan 11) dan 47.50 (Percobaan 12), sebuah lompatan besar dari skor terbaik sebelumnya yaitu ~221 pada kelompok RMSProp. Hasil luar biasa ini juga terlihat pada nilai RMSE sebesar 65.94 (Percobaan 11) dan 67.29 (Percobaan 12), serta MAPE yang sangat rendah yaitu 10.60% (Percobaan 11) dan 10.68% (Percobaan 12).

Dari hasil ini, dapat disimpulkan untuk *optimizer* Adam dengan *learning rate* rendah (Percobaan 11 dan 12) secara drastis mengungguli semua konfigurasi *optimizer* lainnya dalam hal meminimalkan *error* pada data latih. Tidak ada perbedaan signifikan antara penggunaan parameter beta *default* dan alternatif pada *learning rate* ini.

Dari keseluruhan analisis, empat model menunjukkan kinerja paling menonjol dan dianggap sebagai kandidat terbaik. Keempat konfigurasi model ini dirangkum pada Tabel 4.6 dan akan dibawa ke tahap evaluasi final untuk menentukan satu model terbaik berdasarkan kemampuannya dalam melakukan generalisasi pada data uji.

Tabel 4.6 Empat Percobaan dengan metrik terbaik

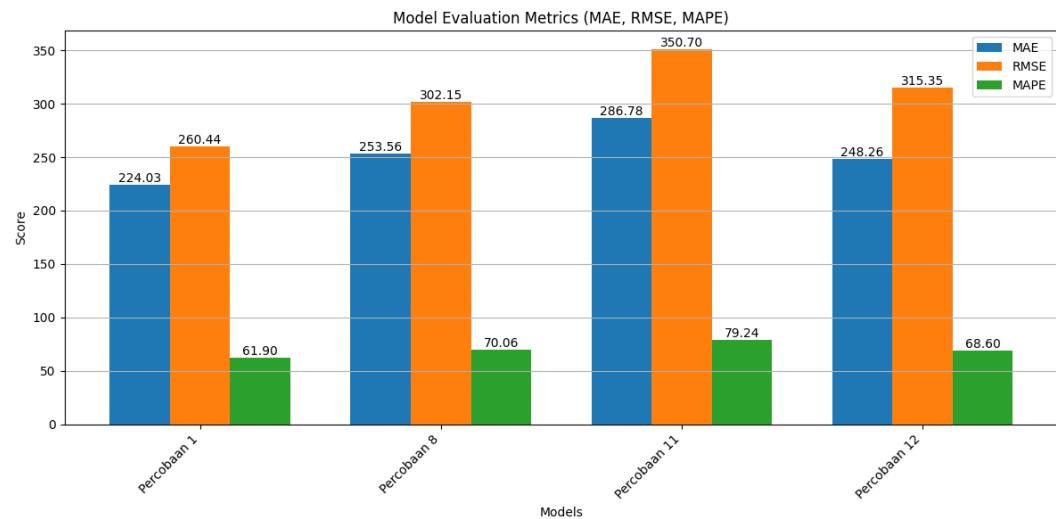
Percobaan	Optimizer	Konfigurasi	MAE	RMSE	MAPE (%)
Percobaan 1	SGD	$\eta = 0.01, mom = 0$	243.55	283.83	54.76
Percobaan 8	RMSProp	$\eta = 0.001, \beta = 0.5$	211.40	248.83	59.61
Percobaan 11	Adam	$\eta = 0.001, \beta_1 = 0.9, \beta_2 = 0.99$	47.15	65.94	10.60
Percobaan 12	Adam	$\eta = 0.001, \beta_1 = 0.5, \beta_2 = 0.85$	47.50	67.29	10.68

4.5 Pemilihan Model Terbaik

Tahap akhir dalam penelitian ini merupakan proses evaluasi komprehensif untuk menentukan konfigurasi model yang paling optimal. Evaluasi dilakukan terhadap empat model kandidat terbaik (Tabel 4.6), dengan menggunakan data uji (*test set*) yang sebelumnya tidak pernah dilibatkan dalam proses pelatihan. Pemilihan model didasarkan pada kombinasi analisis kuantitatif, yang mengacu pada metrik kinerja, serta analisis kualitatif melalui interpretasi visual hasil prediksi.

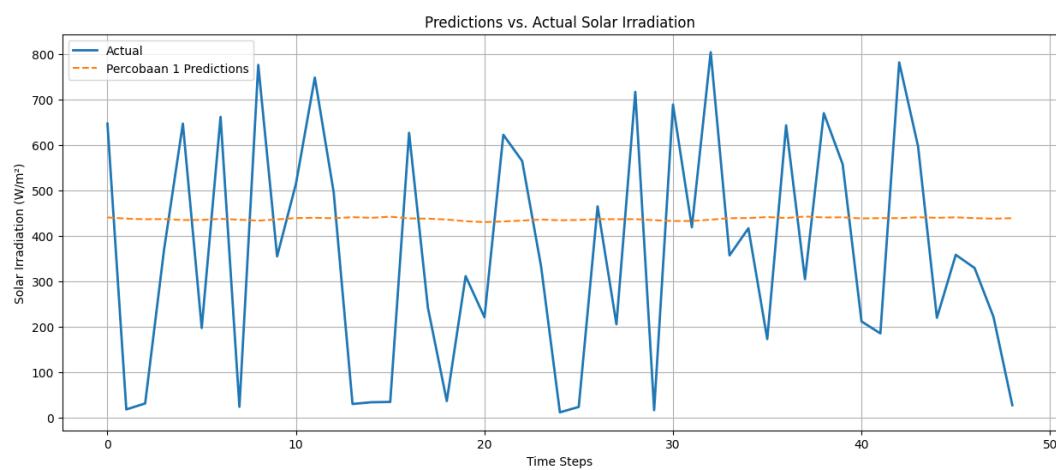
Berdasarkan analisis kuantitatif pada Gambar 4.31, ditemukan hasil yang cukup mengejutkan dan tidak sesuai ekspektasi awal. Model pada Percobaan 1 (menggunakan *optimizer* SGD) mencatatkan nilai *error* terendah, yaitu MAE sebesar 224,03 dan RMSE sebesar 260,44. Sementara itu, model dengan *optimizer*

Adam pada Percobaan 11 dan 12, yang sebelumnya menunjukkan performa unggul pada data latih, justru menghasilkan nilai *error* yang lebih tinggi pada data uji.



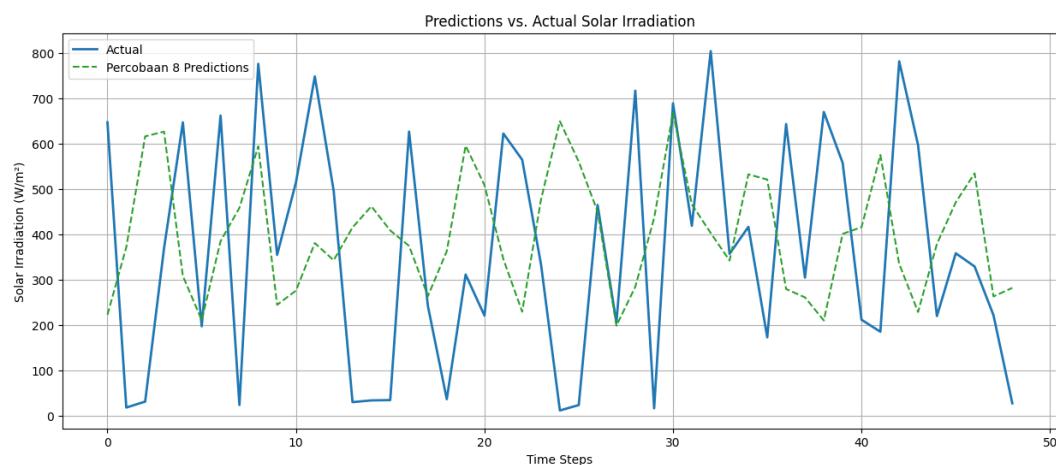
Gambar 4.31 Grafik perbandingan metrik 4 model terbaik pada data uji

Namun, analisis kualitatif terhadap plot prediksi mengungkap gambaran yang sangat berbeda. Visualisasi hasil prediksi pada Gambar 4.32 hingga Gambar 4.35 menunjukkan kelemahan fundamental dari model yang hanya unggul secara metrik.

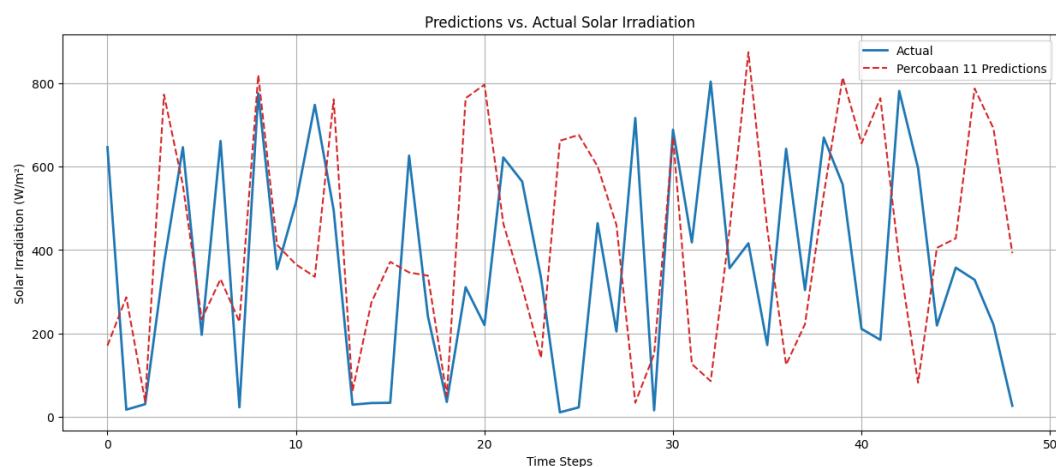


Gambar 4.32 Grafik hasil prediksi percobaan 1

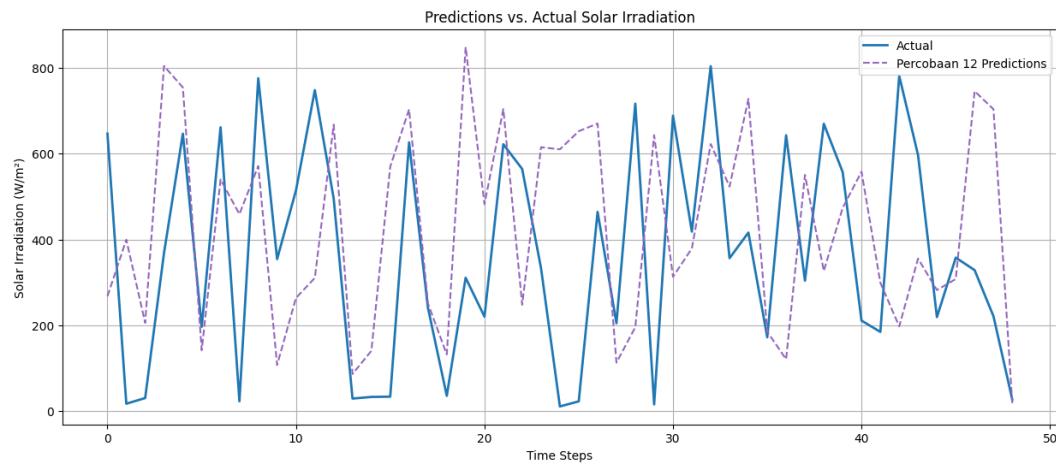
Model dengan *optimizer* SGD (Percobaan 1, garis oranye putus-putus) menghasilkan prediksi yang nyaris konstan dan tidak mampu menangkap dinamika iradiasi matahari. Karakteristik ini merupakan contoh klasik dari *underfitting*, di mana model terlalu sederhana dan hanya menghasilkan estimasi rata-rata. Meskipun menghasilkan *error* yang rendah, model semacam ini tidak memiliki kegunaan praktis karena gagal merepresentasikan perubahan kondisi cuaca secara dinamis.



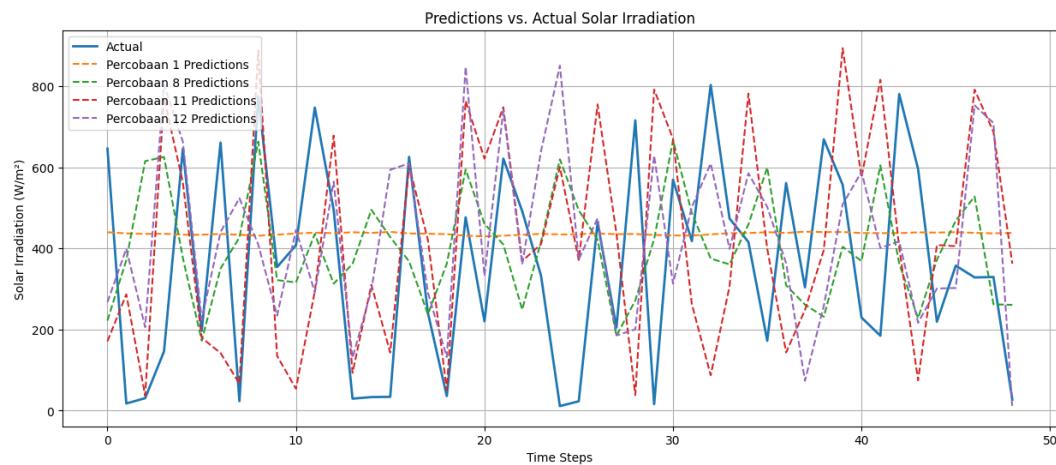
Gambar 4.33 Grafik hasil prediksi percobaan 8



Gambar 4.34 Grafik hasil prediksi percobaan 11



Gambar 4.35 Grafik hasil prediksi percobaan 12



Gambar 4.36 Grafik perbandingan hasil prediksi 4 model terbaik

Model RMSprop (Percobaan 8, garis hijau putus-putus) menunjukkan perbaikan dibanding model SGD, namun prediksinya masih cenderung teredam dan kurang responsif terhadap puncak dan lembah data aktual. Sebaliknya, model Adam pada Percobaan 11 dan 12 (garis merah dan ungu) menunjukkan performa kualitatif yang unggul. Walaupun nilai *error*-nya relatif lebih tinggi, model ini berhasil mengikuti pola fluktuasi data aktual lebih baik termasuk mendekripsi sebagian besar puncak dan lembah, yang mengindikasikan kemampuan dalam mempelajari pola

kompleks. Kemampuan ini sangat penting dalam konteks aplikasi praktis, karena memungkinkan prediksi yang lebih representatif terhadap dinamika kondisi nyata.

Secara metrik, Percobaan 1 memang mencatatkan performa kuantitatif terbaik dengan nilai MAE, RMSE, dan MAPE terendah. Namun, keunggulan ini diperoleh melalui strategi prediksi rata-rata, yang dalam konteks praktis sangat terbatas manfaatnya. Model semacam ini cenderung mengabaikan fluktuasi data dan tidak mampu menginformasikan pola iradiasi matahari secara bermakna.

Sebaliknya, model pada Percobaan 12 memperlihatkan kinerja yang jauh lebih menjanjikan. Meskipun secara metrik sedikit lebih tinggi dibanding Percobaan 1, model ini menempati peringkat kedua terbaik untuk MAE (248,26) dan MAPE (68,60%). Yang lebih penting, analisis kualitatif pada visualisasi hasil prediksinya menunjukkan bahwa model ini mampu mempelajari pola fluktuatif secara konsisten dan merepresentasikannya dalam prediksi.

Berdasarkan pertimbangan komprehensif antara kinerja kuantitatif dan kualitatif, model terbaik adalah yang mampu menawarkan keseimbangan optimal antara akurasi metrik dan kemampuan merepresentasikan dinamika data. Model pada Percobaan 12 menunjukkan kinerja metrik yang sangat kompetitif (peringkat kedua terbaik pada MAE dan MAPE) serta memiliki kemampuan yang unggul dalam menangkap tren, puncak, dan lembah dari data aktual. Hal ini mencerminkan kapasitas generalisasi yang baik dan relevansi praktis yang tinggi untuk kebutuhan prediksi jangka panjang. Oleh karena itu, model Percobaan 12 (*Optimizer Adam*, $\eta = 0.001$, $\beta_1 = 0.5$, $\beta_2 = 0.85$) dipilih sebagai model terbaik dalam penelitian ini.

BAB 5

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil penelitian mengenai analisis performa *optimizer* algoritma pembelajaran mesin untuk prediksi iradiasi matahari, dapat ditarik tiga kesimpulan utama yang menjawab rumusan masalah penelitian:

1. Penerapan algoritma pembelajaran mesin untuk prediksi iradiasi matahari berhasil dilakukan melalui serangkaian tahapan sistematis. Proses ini meliputi pra-pemrosesan data (penanganan *missing values*, konversi lux ke W/m², dan normalisasi), perancangan arsitektur Stacked *Long Short-Term Memory* (LSTM), serta segmentasi data menjadi format deret waktu univariat dengan teknik *sliding window* berukuran 7 *timesteps*.
2. Terdapat perbedaan performa yang signifikan antara ketiga *optimizer* yang diuji (SGD, RMSProp, dan Adam). Selama fase pelatihan, Adam menunjukkan kecepatan konvergensi dan stabilitas terbaik. Pada tahap pengujian, model-model yang menggunakan *optimizer* SGD dan RMSProp cenderung mengalami *underfitting* dan gagal menangkap fluktuasi data secara efektif. Sebaliknya, model yang menggunakan *optimizer* Adam terbukti paling mampu mengikuti pola dinamis dari data aktual.
3. *Optimizer* yang paling optimal untuk kasus prediksi iradiasi matahari dalam penelitian ini adalah Adam. Konfigurasi terbaik diraih pada Percobaan 12, yang dipilih bukan hanya karena metrik *error*-nya yang

kompetitif, tetapi karena kemampuannya yang superior secara kualitatif dalam menangkap volatilitas data pada data uji. Model ini menghasilkan performa akhir pada data uji dengan nilai MAE 248.26, RMSE 315.35, dan MAPE 68.60%.

5.2 Saran

Berdasarkan temuan dan keterbatasan dalam penelitian ini, berikut adalah beberapa saran untuk pengembangan di masa depan:

1. Mengingat model dengan *optimizer* SGD dan RMSProp menunjukkan kecenderungan *underfitting*, penelitian selanjutnya disarankan untuk mengeksplorasi arsitektur *deep learning* yang lebih kompleks. Beberapa alternatif yang dapat diuji antara lain menambah jumlah lapisan atau unit pada model LSTM, atau menggunakan arsitektur sekuensial lain seperti *Gated Recurrent Unit* (GRU) atau model berbasis Transformer.
2. Penelitian ini sengaja difokuskan pada model univariat untuk menguji kemampuan prediksi berdasarkan data Lux saja. Untuk meningkatkan akurasi prediksi, penelitian mendatang dapat mengembangkan model multivariat dengan menyertakan kembali fitur cuaca (setelah melalui proses encoding) atau menambahkan data meteorologi relevan lainnya seperti suhu udara, kelembapan, dan kecepatan angin.
3. Proses pencarian *hyperparameter* dalam penelitian ini dilakukan secara manual. Untuk menemukan kombinasi yang lebih optimal, disarankan untuk menggunakan teknik optimasi *hyperparameter* otomatis seperti Grid Search, Random Search, atau Bayesian Optimization.

4. Penelitian ini menggunakan ukuran batch 1 untuk seluruh percobaan.

Penelitian selanjutnya dapat mengeksplorasi pengaruh penggunaan ukuran batch yang lebih besar (*mini-batch*), misalnya 8, 16, atau 32.

Ukuran batch yang berbeda dapat memengaruhi kecepatan konvergensi, stabilitas proses pelatihan, dan efisiensi komputasi, sehingga analisis ini dapat memberikan wawasan tambahan dalam menemukan konfigurasi pelatihan yang paling efisien.

5. Model terbaik yang dihasilkan dari penelitian ini (Percobaan 12) memiliki potensi untuk dikembangkan lebih lanjut menjadi sistem prediksi praktis. Sistem ini dapat diintegrasikan dengan *Energy Management System* (EMS) pada instalasi panel surya untuk membantu optimasi penjadwalan penggunaan dan penyimpanan energi.

DAFTAR PUSTAKA

- [1] A. O. Maka dan J. M. Alabid, “Solar energy technology and its roles in sustainable development,” *Clean Energy*, vol. 6, no. 3, hlm. 476–483, Jun 2022, doi: 10.1093/ce/zkac023.
- [2] “Key findings – Southeast Asia Energy Outlook 2022 – Analysis,” IEA. Diakses: 17 Juni 2024. [Daring]. Tersedia pada: <https://www.iea.org/reports/southeast-asia-energy-outlook-2022/key-findings>
- [3] H. Ritchie, M. Roser, dan P. Rosado, “Renewable Energy,” *Our World in Data*, Mar 2024, Diakses: 15 Juni 2024. [Daring]. Tersedia pada: <https://ourworldindata.org/renewable-energy>
- [4] Y. Ledmaoui, A. El Maghraoui, M. El Aroussi, R. Saadane, A. Chebak, dan A. Chehri, “Forecasting solar energy production: A comparative study of Machine Learning algorithms,” *Energy Reports*, vol. 10, hlm. 1004–1012, Nov 2023, doi: 10.1016/j.egyr.2023.07.042.
- [5] H.-Y. Cheng, C.-C. Yu, dan C.-L. Lin, “Day-ahead to week-ahead solar irradiance prediction using convolutional Long Short-Term Memory networks,” *Renewable Energy*, vol. 179, hlm. 2300–2308, Des 2021, doi: 10.1016/j.renene.2021.08.038.
- [6] J. Wojtkiewicz, M. Hosseini, R. Gottumukkala, dan T. L. Chambers, “Hour-Ahead Solar Irradiance Forecasting Using Multivariate Gated Recurrent Units,” *Energies*, vol. 12, no. 21, hlm. 4055, Okt 2019, doi: 10.3390/en12214055.
- [7] I. M. Mfetoum dkk., “A multilayer perceptron neural network approach,” *Scientific Reports*.
- [8] M. S. Alrubaih, M. F. M. Zain, M. A. Alghoul, N. L. N. Ibrahim, M. A. Shameri, dan O. Elayeb, “Research and development on aspects of daylighting fundamentals,” *Renewable and Sustainable Energy Reviews*, vol. 21, hlm. 494–505, Mei 2013, doi: 10.1016/j.rser.2012.12.057.
- [9] D. L. Loe, “Light, colour and human response,” dalam *Colour Design*, Elsevier, 2017, hlm. 349–369. doi: 10.1016/B978-0-08-101270-3.00015-1.
- [10] Z. Allal, H. N. Noura, dan K. Chahine, “Machine Learning Algorithms for Solar Irradiance Prediction: A Recent Comparative Study,” *e-Prime - Advances in Electrical Engineering, Electronics and Energy*, vol. 7, hlm. 100453, Mar 2024, doi: 10.1016/j.prime.2024.100453.
- [11] M. Paulescu, E. Paulescu, P. Gravila, dan V. Badescu, “Solar Radiation Measurements,” dalam *Weather Modeling and Forecasting of PV Systems Operation*, dalam Green Energy and Technology. , London: Springer London, 2013, hlm. 17–42. doi: 10.1007/978-1-4471-4649-0_2.
- [12] P. R. Michael, D. E. Johnston, dan W. Moreno, “A conversion guide: solar irradiance and lux illuminance,” *J. meas. eng.*, vol. 8, no. 4, hlm. 153–166, Des 2020, doi: 10.21595/jme.2020.21667.
- [13] V. Kotu dan B. Deshpande, “Time Series Forecasting,” dalam *Data Science*, Elsevier, 2019, hlm. 395–445. doi: 10.1016/B978-0-12-814761-0.00012-5.
- [14] E. Padang, “PREDIKSI INTENSITAS RADIASI MATAHARI BERBASIS MODEL DEEP NEURAL NETWORKS (DNN),” 2024.

- [15] M. Husein dan I.-Y. Chung, “Day-Ahead Solar Irradiance *Forecasting* for Microgrids Using a *Long Short-Term Memory* Recurrent Neural Network: A Deep Learning Approach,” *Energies*, vol. 12, no. 10, hlm. 1856, Mei 2019, doi: 10.3390/en12101856.
- [16] A. Alzahrani, P. Shamsi, C. Dagli, dan M. Ferdowsi, “Solar Irradiance *Forecasting* Using Deep Neural Networks,” *Procedia Computer Science*, vol. 114, hlm. 304–313, 2017, doi: 10.1016/j.procs.2017.09.045.
- [17] A. Karpathy, J. Johnson, dan L. Fei-Fei, “Visualizing and Understanding Recurrent Networks,” 17 November 2015, *arXiv*: arXiv:1506.02078. Diakses: 16 November 2024. [Daring]. Tersedia pada: <http://arxiv.org/abs/1506.02078>
- [18] Z. Zhao, W. Chen, X. Wu, P. C. Y. Chen, dan J. Liu, “LSTM network: a deep learning approach for short-term traffic forecast,” *IET Intelligent Transport Systems*, vol. 11, no. 2, hlm. 68–75, 2017, doi: <https://doi.org/10.1049/iet-its.2016.0208>.
- [19] R. Chandra, S. Goyal, dan R. Gupta, “Evaluation of deep learning models for multi-step ahead time series prediction,” *IEEE Access*, vol. 9, hlm. 83105–83123, 2021, doi: 10.1109/ACCESS.2021.3085085.
- [20] Y. Yu, X. Si, C. Hu, dan J. Zhang, “A Review of Recurrent Neural Networks: LSTM Cells and Network Architectures,” *Neural Computation*, vol. 31, no. 7, hlm. 1235–1270, Jul 2019, doi: 10.1162/neco_a_01199.
- [21] L. Wiranda dan M. Sadikin, “PENERAPAN LONG SHORT TERM MEMORY PADA DATA TIME SERIES UNTUK MEMPREDIKSI PENJUALAN PRODUK PT. METISKA FARMA,” vol. 8, 2019.
- [22] Sumbatilinda, “Deep Learning(Part 2). *Loss function* and Gradient Function,” Medium. Diakses: 10 Mei 2025. [Daring]. Tersedia pada: <https://medium.com/@sumbatilinda/deep-learning-part-2-loss-function-and-gradient-function-2f64c566a1d6>
- [23] “*Optimizers* in Deep Learning: A Comprehensive Guide.” Diakses: 9 Juli 2024. [Daring]. Tersedia pada: <https://www.analyticsvidhya.com/blog/2021/10/a-comprehensive-guide-on-deep-learning-optimizers/>
- [24] M. Jamhuri, “Understanding *Stochastic Gradient Descent* (SGD),” Medium. Diakses: 7 Mei 2025. [Daring]. Tersedia pada: <https://jamhuri.medium.com/understanding-stochastic-gradient-descent-sgd-c31454313a9a>
- [25] V. Efimov, “Understanding Deep Learning *Optimizers*: *Momentum*, AdaGrad, RMSProp & Adam,” Medium. Diakses: 25 Juni 2024. [Daring]. Tersedia pada: <https://towardsdatascience.com/understanding-deep-learning-optimizers-momentum-adagrad-rmsprop-adam-e311e377e9c2>
- [26] P. Kashyap, “Understanding RMSProp: A Simple Guide to One of Deep Learning’s Powerful *Optimizers*,” Medium. Diakses: 7 Mei 2025. [Daring]. Tersedia pada: <https://medium.com/@piyushkashyap045/understanding-rmsprop-a-simple-guide-to-one-of-deep-learnings-powerful-optimizers-403baeed9922>
- [27] M. Jamhuri, “Understanding the Adam Optimization Algorithm: A Deep Dive into the Formulas,” Medium. Diakses: 7 Mei 2025. [Daring]. Tersedia pada: <https://medium.com/@piyushkashyap045/understanding-the-adam-optimization-algorithm-a-deep-dive-into-the-formulas-403baeed9922>

- pada: <https://jamhuri.medium.com/understanding-the-adam-optimization-algorithm-a-deep-dive-into-the-formulas-3ac5fc5b7cd3>
- [28] “Evaluation Metrics in *Machine Learning*,” GeeksforGeeks. Diakses: 25 Juni 2024. [Daring]. Tersedia pada: <https://www.geeksforgeeks.org/metrics-for-machine-learning-model/>
- [29] S. Tiwari, “Complete Guide to *Machine Learning* Evaluation Metrics,” Analytics Vidhya. Diakses: 25 Juni 2024. [Daring]. Tersedia pada: <https://medium.com/analytics-vidhya/complete-guide-to-machine-learning-evaluation-metrics-615c2864d916>
- [30] apoovakumar169, “*Mean Absolute Percentage Error (MAPE)*: What You Need To Know,” Origins AI. Diakses: 23 Maret 2025. [Daring]. Tersedia pada: <https://originshq.com/blog/what-is-mean-absolute-percentage-error/>
- [31] admin, “*Mean Absolute Percentage Error (MAPE)*: A Overview,” Pusat Penelitian, Pengabdian kepada Masyarakat dan Publikasi Internasional (P3MPI). Diakses: 23 Maret 2025. [Daring]. Tersedia pada: <https://p3mpi.uma.ac.id/2024/09/18/mean-absolute-percentage-error-mape-a-overview/>
- [32] “What is Python? Executive Summary,” Python.org. Diakses: 16 April 2025. [Daring]. Tersedia pada: <https://www.python.org/doc/essays/blurb/>
- [33] M. Abadi dkk., “TensorFlow: Large-Scale *Machine Learning* on Heterogeneous Distributed Systems”.
- [34] “What is Google Colab: A Beginner’s Guide - ByteXD.” Diakses: 21 April 2025. [Daring]. Tersedia pada: <https://bytexd.com/what-is-google-colab-a-beginner-guide/>
- [35] “*Boxplot*,” GeeksforGeeks. Diakses: 28 Juni 2025. [Daring]. Tersedia pada: <https://www.geeksforgeeks.org/machine-learning/box-plot/>
- [36] B. A. Aprian, Y. Azhar, dan V. R. S. Nastiti, “Prediksi Pendapatan Kargo Menggunakan Arsitektur Long Short Term Memory,” *JKT*, vol. 6, no. 2, hlm. 148–157, Nov 2020, doi: 10.35143/jkt.v6i2.3621.
- [37] M. Rizki, S. Basuki, dan Y. Azhar, “Implementasi Deep Learning Menggunakan Arsitektur Long Short Term Memory(LSTM) Untuk Prediksi Curah Hujan Kota Malang,” *JR*, vol. 2, no. 3, Jan 2024, doi: 10.22219/repositor.v2i3.30499.
- [38] R. Cahyadi, A. Damayanti, dan D. Aryadani, “*RECURRENT NEURAL NETWORK (RNN) DENGAN LONG SHORT TERM MEMORY (LSTM) UNTUK ANALISIS SENTIMEN DATA INSTAGRAM*,” vol. 5, no. 1, 2020.

LAMPIRAN

Lampiran 1 Alat Pengumpulan Dataset



Lampiran 2 Kegiatan Pengukuran Lux



Pencatatan Lux Meter

ranggawibisanapp@gmail.com Switch account

✉ Not shared

☁

* Indicates required question

Cuaca *

Hujan

Mendung

Berawan

Cerah Berawan

Cerah

Derajat *

0

15

30

45

Lampiran 3 Dataset Pengukuran Lux

No	Bulan	Tanggal	Sudut	Lux	Cuaca	Tampak
1	March	3/1/2023	0	0	Hujan	Utara
2	March	3/1/2023	15	0	Hujan	Utara
3	March	3/1/2023	30	0	Hujan	Utara
4	March	3/1/2023	45	0	Hujan	Utara
5	March	3/1/2023	60	0	Hujan	Utara
6	March	3/1/2023	75	0	Hujan	Utara
7	March	3/1/2023	90	0	Hujan	Utara
8	March	3/1/2023	105	0	Hujan	Utara
9	March	3/1/2023	120	0	Hujan	Utara
10	March	3/1/2023	135	0	Hujan	Utara
11	March	3/1/2023	150	0	Hujan	Utara
12	March	3/1/2023	165	0	Hujan	Utara
13	March	3/1/2023	180	0	Hujan	Utara
14	March	3/2/2023	0	0	Hujan	Utara
15	March	3/2/2023	15	0	Hujan	Utara
16	March	3/2/2023	30	0	Hujan	Utara
17	March	3/2/2023	45	0	Hujan	Utara
18	March	3/2/2023	60	0	Hujan	Utara
19	March	3/2/2023	75	0	Hujan	Utara
20	March	3/2/2023	90	0	Hujan	Utara
21	March	3/2/2023	105	0	Hujan	Utara
22	March	3/2/2023	120	0	Hujan	Utara
23	March	3/2/2023	135	0	Hujan	Utara
24	March	3/2/2023	150	0	Hujan	Utara
25	March	3/2/2023	165	0	Hujan	Utara
26	March	3/2/2023	180	0	Hujan	Utara
27	March	3/3/2023	0	9674	Mendung	Utara
28	March	3/3/2023	15	12540	Mendung	Utara
29	March	3/3/2023	30	16540	Mendung	Utara
30	March	3/3/2023	45	20130	Mendung	Utara
31	March	3/3/2023	60	23840	Mendung	Utara
32	March	3/3/2023	75	27230	Mendung	Utara
33	March	3/3/2023	90	33070	Mendung	Utara
34	March	3/3/2023	105	22440	Mendung	Utara
35	March	3/3/2023	120	22540	Mendung	Utara
36	March	3/3/2023	135	20900	Mendung	Utara

37	March	3/3/2023	150	18640	Mendung	Utara
38	March	3/3/2023	165	15550	Mendung	Utara
39	March	3/3/2023	180	12160	Mendung	Utara
40	March	3/4/2023	0		Berawan	Utara
41	March	3/4/2023	15		Berawan	Utara
42	March	3/4/2023	30		Berawan	Utara
43	March	3/4/2023	45		Berawan	Utara
44	March	3/4/2023	60		Berawan	Utara
45	March	3/4/2023	75		Berawan	Utara
46	March	3/4/2023	90		Berawan	Utara
47	March	3/4/2023	105		Berawan	Utara
48	March	3/4/2023	120		Berawan	Utara
49	March	3/4/2023	135		Berawan	Utara
50	March	3/4/2023	150		Berawan	Utara
...
4736	February	2/28/2024	45		Berawan	Utara
4737	February	2/28/2024	60		Berawan	Utara
4738	February	2/28/2024	75		Berawan	Utara
4739	February	2/28/2024	90		Berawan	Utara
4740	February	2/28/2024	105		Berawan	Utara
4741	February	2/28/2024	120		Berawan	Utara
4742	February	2/28/2024	135		Berawan	Utara
4743	February	2/28/2024	150		Berawan	Utara
4744	February	2/28/2024	165		Berawan	Utara
4745	February	2/28/2024	180		Berawan	Utara
4746	February	2/29/2024	0	0	Hujan	Utara
4747	February	2/29/2024	15	0	Hujan	Utara
4748	February	2/29/2024	30	0	Hujan	Utara
4749	February	2/29/2024	45	0	Hujan	Utara
4750	February	2/29/2024	60	0	Hujan	Utara
4751	February	2/29/2024	75	0	Hujan	Utara
4752	February	2/29/2024	90	0	Hujan	Utara
4753	February	2/29/2024	105	0	Hujan	Utara
4754	February	2/29/2024	120	0	Hujan	Utara
4755	February	2/29/2024	135	0	Hujan	Utara
4756	February	2/29/2024	150	0	Hujan	Utara
4757	February	2/29/2024	165	0	Hujan	Utara
4758	February	2/29/2024	180	0	Hujan	Utara

BIODATA PENULIS



A. Identitas

Nama : Rangga Wibisana Putra Pamungkas
NIM : H1A021058
Tempat, tanggal lahir : Kuningan, 11 Desember 2003
Alamat : Jalan Dalam Desa Kertawangan, Blok Paleben
Rt. 03/Rw. 01, Kec. Sindangagung, Kab. Kuningan,
Prov. Jawa Barat, Indonesia 45573
No. Telp. : 081214014304
Alamat e-mail : ranggawpp.work@gmail.com

B. Riwayat Pendidikan Akademik

Periode	Jenjang	Institusi
2021 – Sekarang	S1	Teknik Elektro Universitas Jenderal Soedirman
2018 – 2021	SMA	SMA Negeri 2 Kuningan
2015 – 2018	SMP	SMP Negeri 4 Kuningan