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

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

Rangga Wibisana P. P.[*1], Agung Mubyarto, S.T., M.T.[*2], Muhammad Syaiful Aliim, S.T., M.T.[*3]

***** Email: rangga.pamungkas@mhs.unsoed.ac.id

Mahasiswa Pemakalah[*1], Dosen Pembimbing 1[*2], Dosen Pembimbing 2[*3]

[1,2,3]Jurusan Teknik Elektro, Fakultas Teknik, Universitas Jenderal Soedirman, Purwokerto

***Abstrak* -** Pemanfaatan energi surya membutuhkan prediksi iradiasi matahari yang akurat untuk mengoptimalkan kinerja sistem *photovoltaic* (PV). Penelitian ini membangun model prediksi iradiasi matahari berdasarkan data *lux* harian menggunakan algoritma *Long Short-Term Memory* (LSTM), serta menganalisis secara sistematis performa tiga *optimizer*—Adam, RMSProp, dan SGD—dalam pelatihan model tersebut. *Dataset* yang digunakan diperoleh dari pengukuran intensitas cahaya di atap sebuah gedung di Jakarta Selatan selama satu tahun (Maret 2023 – Februari 2024) pada sudut pengukuran 75 derajat. Data mentah dikonversi dari *lux* ke iradiasi (W/m²), kemudian dinormalisasi dan disegmentasi menjadi format deret waktu untuk diolah oleh arsitektur *Stacked LSTM*.

Evaluasi performa model dilakukan menggunakan metrik MAE, RMSE, dan MAPE. Hasil penelitian menunjukkan perbedaan kinerja yang signifikan di antara ketiga *optimizer*. Pada data uji, model yang dilatih dengan SGD secara tak terduga menghasilkan *error* kuantitatif terendah. Namun, analisis kualitatif menunjukkan bahwa prediksinya cenderung datar dan gagal menangkap volatilitas data (*underfitting*). Sebaliknya, model yang dilatih dengan *optimizer* Adam ($\eta = 0.001, \beta_1 = 0.5, \beta_2 = 0.85$) terbukti paling unggul secara kualitatif dalam meniru fluktuasi dinamis dari data aktual. Berdasarkan sintesis antara performa kuantitatif yang kompetitif dan kemampuan generalisasi kualitatif yang *superior*, *optimizer* Adam dipilih sebagai konfigurasi terbaik, yang sekaligus menegaskan pentingnya evaluasi visual dalam pemodelan deret waktu yang stokastik.

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

***Abstract* –** *Accurate solar irradiance forecasting is essential for optimizing photovoltaic (PV) system performance. This study develops a solar irradiance prediction model based on daily lux data using the Long Short-Term Memory (LSTM) algorithm and systematically analyzes the performance of three optimizers—Adam, RMSProp, and SGD—in training the model. The dataset was obtained from illuminance measurements on a building rooftop in South Jakarta over one year (March 2023 – February 2024) at a 75-degree measurement angle. The raw data was converted from lux to irradiance (W/m²), then normalized and segmented into a time-series format to be processed by a Stacked LSTM architecture.*

Model performance was evaluated using MAE, RMSE, and MAPE metrics. The results revealed significant performance differences among the three optimizers. On the test data, the model trained with SGD unexpectedly yielded the lowest quantitative error. However, a qualitative analysis showed that its predictions were predominantly flat and failed to capture the data's volatility (underfitting). Conversely, the model trained with the Adam optimizer (η=0.001, β1=0.5, β2=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 was selected as the best configuration, which also highlights the importance of visual evaluation in stochastic time-series modeling.

Keywords : Machine Learning, Time series Forecasting, LSTM, Lux, Solar Irradiance, Optimizer, Adam, RMSProp, SGD, Performance Analysis.

I. PENDAHULUAN

Seiring dengan perkembangan peradaban dan kemajuan teknologi, kebutuhan energi global mengalami peningkatan signifikan yang didorong oleh pertumbuhan populasi. Saat ini, bauran energi global masih didominasi oleh bahan bakar fosil, yang merupakan kontributor utama lebih dari 75% emisi gas rumah kaca dan hampir 90% dari total emisi karbon dioksida. Ketergantungan ini telah menjadi pemicu utama perubahan iklim serta menimbulkan dampak negatif terhadap kesehatan masyarakat[1], [2], [3]. Sebagai respons terhadap tantangan tersebut, pengembangan energi terbarukan menjadi solusi krusial untuk mewujudkan keberlanjutan lingkungan.

Salah satu sumber energi terbarukan yang menjanjikan adalah energi surya, yang dapat dimanfaatkan melalui sistem *fotovoltaik* (PV). Namun, tantangan utama dari sistem ini terletak pada sifatnya yang intermiten dan sangat dipengaruhi oleh kondisi cuaca[4]. Untuk menjamin stabilitas dan efisiensi sistem, prediksi iradiasi matahari—yakni energi yang diterima per satuan luas (W/m^2)—secara akurat menjadi sangat penting. Salah satu pendekatan praktis untuk memprediksi iradiasi adalah dengan mengonversi data iluminasi (*lux*) yang lebih mudah diperoleh[5].

Dalam beberapa dekade terakhir, metode berbasis *Machine Learning* (ML) telah menunjukkan kemajuan yang signifikan dalam tugas prediksi iradiasi berkat kemampuannya dalam mengenali pola non-linear yang kompleks[6]. Di antara berbagai algoritma, *Long Short-Term Memory* (LSTM), yang merupakan pengembangan dari arsitektur *Recurrent Neural Network* (RNN), terbukti sangat efektif untuk data deret waktu. Kemampuannya dalam menangkap dependensi temporal jangka panjang menjadikannya kandidat ideal untuk prediksi iradiasi matahari yang lebih akurat[4].

Berbagai penelitian terdahulu telah menunjukkan potensi besar algoritma sekuensial dalam prediksi iradiasi matahari. Cheng dkk. (2021) mengembangkan model prediksi iradiasi menggunakan jaringan *Conv-LSTM*, yang menunjukkan peningkatan akurasi yang signifikan, bahkan ketika model yang diusulkan dilatih hanya dengan data selama dua bulan[5]. Wojtkiewicz dkk. (2019) membandingkan model *Gated Recurrent Unit* (GRU) multivariat dengan LSTM, dan menemukan bahwa meskipun GRU unggul dalam kecepatan pelatihan, LSTM memberikan akurasi lebih tinggi. Penambahan variabel cuaca juga terbukti meningkatkan performa pada kedua model[6]. Husein dan Chung (2019) merancang model LSTM-RNN untuk prediksi *day-ahead* yang hanya menggunakan data cuaca, tanpa memerlukan data historis iradiasi, dan memperoleh akurasi yang lebih tinggi dibandingkan dengan *Feedforward Neural Network* (FFNN) dan model *persistence*[7]. Alzahrani dkk. (2017) menerapkan *Deep Recurrent Neural Network* (DRNN) dan menunjukkan performa yang lebih baik dibandingkan FNN dan *Support Vector Regression* (SVR), serta lebih efisien dalam menangani data berukuran besar[8].

Secara umum, 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, Cheng dkk. (2021)), analisis komparatif yang sistematis mengenai pengaruh pemilihan *optimizer*—seperti *Adaptive Moment Estimation* (Adam), *Root Mean Square Propagation* (RMSProp), dan *Stochastic Gradient Descent* (SGD)—terhadap stabilitas dan akurasi hasil prediksi masih sangat terbatas. Padahal, pemilihan *optimizer* yang tidak tepat dapat menyebabkan konvergensi yang lambat atau prediksi yang tidak akurat, sehingga mengurangi efektivitas sistem prediksi secara keseluruhan.

Menjawab celah tersebut, penelitian ini bertujuan untuk (1) menerapkan algoritma pembelajaran mesin untuk memprediksi iradiasi matahari berdasarkan *lux*, (2) menganalisis dan membandingkan secara sistematis performa dari tiga *optimizer*—Adam, RMSProp, dan SGD, serta (3) menentukan konfigurasi *optimizer* yang paling optimal untuk kasus ini.

Ruang lingkup penelitian ini dibatasi pada penggunaan arsitektur *Long Short-Term Memory* yang diimplementasikan menggunakan *library* Tensorflow-Keras dengan bahasa pemrograman Python pada platform Google Colab. *Dataset* yang digunakan terdiri dari data *lux* harian dari atap gedung di Jakarta Selatan selama periode Maret 2023 hingga Februari 2024. Evaluasi performa model difokuskan pada tiga metrik

utama, yaitu *Mean Absolute Error* (MAE), *Root Mean Square Error* (RMSE), dan *Mean Absolute Percentage Error* (MAPE). Hasil dari penelitian ini diharapkan dapat menjadi referensi yang berharga dalam pemilihan *optimizer* untuk meningkatkan performa pemodelan prediksi iradiasi matahari.

II. TINJAUAN PUSTAKA

A. Iluminasi (*Lux*) dan Iradiasi Matahari

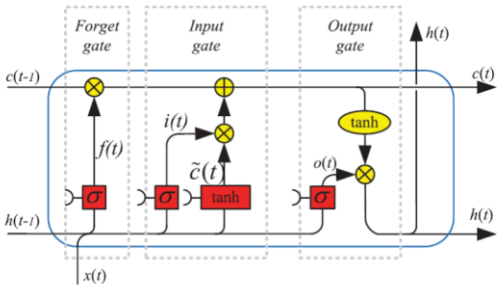
Iluminasi (E), atau luas penerangan, didefinisikan sebagai kerapatan fluks cahaya yang jatuh pada suatu permukaan. Satuan Sistem Internasional (SI) untuk iluminasi adalah *lux* (lx), yang setara dengan satu lumen per meter persegi (lumen/m²)[9]. Sedangkan iradiasi matahari merujuk pada besaran energi radiasi matahari yang diterima per satuan luas dalam periode waktu tertentu, dengan satuan SI watt per meter persegi (W/m²)[10].

Meskipun keduanya berkaitan dengan cahaya matahari, terdapat perbedaan mendasar dalam aspek yang diukur. *Lux* mengukur intensitas cahaya yang terlihat oleh mata manusia berdasarkan sensitivitas spektral penglihatan manusia (fotopik), sedangkan iradiasi mengukur total daya energi yang masuk, yang lebih relevan untuk aplikasi konversi energi seperti sistem *fotovoltaik* (PV).

Studi oleh Michael dkk. (2020), menunjukkan adanya korelasi yang kuat antara iluminasi dan iradiasi untuk kondisi pencahayaan alami di luar ruangan. Dalam penelitian tersebut, diperoleh nilai konversi empiris sekitar 1 W/m² ≈ 122 lx, yang digunakan secara luas dalam berbagai pendekatan praktis[11]. Hubungan ini menjadi dasar konversi data *lux* ke iradiasi dalam penelitian ini, mengingat kemudahan akuisisi data *lux* melalui sensor optik komersial.

B. *Long Short Term Memory*

Long Short-Term Memory (LSTM) merupakan arsitektur dari *Recurrent Neural Network* (RNN) yang dirancang secara khusus untuk mengatasi permasalahan *vanishing gradient*, yaitu kesulitan dalam mempelajari dependensi jangka panjang pada data sekuensial. Masalah ini sering terjadi pada RNN konvensional ketika informasi penting dalam data berada jauh dalam urutan waktu. LSTM mengatasi kendala tersebut dengan mengimplementasikan mekanisme sel memori (*memory cell*) yang memungkinkan jaringan untuk "mengingat" informasi penting dalam rentang waktu yang panjang[12], [13], [14].



Gambar 1 Arsitektur blok LSTM[15].

Inovasi utama LSTM terletak pada unit memorinya yang terdiri atas *cell state* (C_t) dan mekanisme gerbang (*gating mechanism*) untuk meregulasi aliran informasi. *Cell state* berfungsi sebagai "memori" jangka panjang, sedangkan tiga gerbang utama—*forget gate, input gate, dan output gate*—secara dinamis mengontrol informasi yang dihapus, ditambahkan, dan dikeluarkan dari sel pada setiap langkah waktu (*timestep*)[15].

Mekanisme ini memungkinkan LSTM untuk secara selektif mengingat informasi relevan dari masa lalu dan menggunakannya untuk melakukan prediksi pada masa depan. Proses internal LSTM pada setiap *timestep* t dapat dirangkum sebagai berikut:

1. *Forget gate* (f_t)

Forget gate menentukan informasi apa dari *cell state* sebelumnya (C_{t-1}) yang akan dilupakan, berdasarkan *hidden state* sebelumnya (h_{t-1}) dan *input* saat ini (x_t).

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

2. *Input gate* (i_t)

Input gate memutuskan informasi baru mana yang akan disimpan dalam *cell state*. Gerbang ini menghasilkan vektor kandidat nilai baru ($\tilde{C}_t$) untuk ditambahkan ke *cell state*.

$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] + b_i\right) \quad (2)$$

$$\tilde{c}_t = tanh(W_c \cdot [h_{t-1}, x_t] + b_C) \quad (3)$$

3. Pembaruan *Cell State* (C_t)

Cell state (C_t) diperbarui dengan menggabungkan informasi lama yang dipertahankan dan informasi baru yang relevan.

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t \quad (4)$$

4. *Output gate* (o_t)

Output gate menentukan *output* atau *hidden state* berikutnya (h_t) berdasarkan *cell state* yang telah diperbarui.

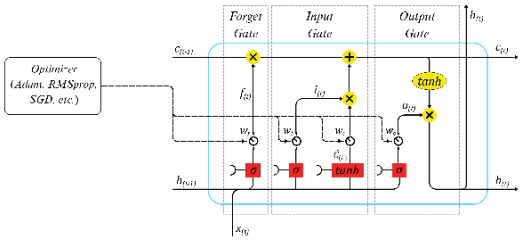
$$o_t = \sigma\left(W_o.[h_{t-1}, x_t] + b_o\right) \quad (5)$$

$$h_t = o_t \cdot tanh(C_t) \quad (6)$$

Di mana σ adalah fungsi aktivasi sigmoid, $tanh$ adalah fungsi tangen hiperbolik, W dan b adalah parameter bobot dan bias yang dipelajari, dan $(\cdot)$ melambangkan perkalian antar-elemen(*element-wise multiplication*)[16].

C. *Optimizer*

Proses pelatihan model *deep learning* seperti LSTM pada dasarnya merupakan suatu masalah optimasi, di mana tujuannya adalah untuk menemukan serangkaian parameter (bobot dan bias) yang dapat meminimalkan fungsi kerugian (*loss function).* Prinsip dasar dari proses ini adalah *Gradient Descent*, yaitu metode pembaruan parameter model dengan bergerak ke arah yang berlawanan dari gradien *loss function*.



Gambar 2 Ilustrasi titik kerja optimizer

Optimizer adalah algoritma yang secara adaptif menyesuaikan bobot dan bias jaringan saraf selama proses pelatihan dengan tujuan meminimalkan *loss function* serta meningkatkan kinerja model secara keseluruhan. Aturan pembaruan bobot umumnya didefinisikan sebagai:

$$W_t = W_{t-1} - \eta \nabla L(W_{t-1}) \qquad (7)$$

Di mana W_t adalah bobot pada iterasi atau *timestep* ke- t, η adalah *learning rate*, $\nabla L(W)$ adalah gradien dari fungsi kerugian terhadap parameter W.

Loss Function atau fungsi kerugian merupakan fungsi matematika yang mengukur selisih antara nilai prediksi model $(f(x_i; W))$ dan nilai aktual (y_i). Dalam tugas regresi seperti prediksi deret waktu, fungsi kerugian yang umum digunakan adalah *Mean Squared Error* (MSE), yang memberikan penalti lebih besar terhadap kesalahan yang besar[17]. Fungsi kerugian MSE dapat dirumuskan sebagai:

$$L(w) = \frac{1}{n}\sum_{i=1}^{n}(y_i - f(x_i; w))^2 \qquad (8)$$

1. *Stochastic Gradient Descent*

Stochastic Gradient Descent (SGD) adalah salah satu algoritma optimasi yang paling populer, terutama untuk pelatihan model *deep learning* dengan *dataset* berukuran besar. Berbeda dengan *gradient descent* biasa yang menghitung *gradient* berdasarkan seluruh himpunan data pada setiap iterasi, SGD secara acak memilih satu sampel data atau sejumlah kecil sampel (*mini-batch*) untuk mengestimasi *gradient*. Pendekatan ini memungkinkan proses pelatihan yang lebih cepat dan efisien, sekaligus mengurangi beban komputasi[18].

2. *Root Mean Square Propagation*

Root Mean Square Propagation (RMSprop) adalah algoritma *optimizer* yang dikembangkan untuk mengatasi masalah *gradient* yang bervariasi dan penurunan *learning rate* yang terlalu cepat pada algoritma seperti *Adaptive Gradient* (AdaGrad). RMSprop secara adaptif menyesuaikan *learning rate* untuk setiap bobot berdasarkan rata-rata pergerakan eksponensial dari kuadrat *gradient*. Dengan demikian, RMSprop mampu mengatasi masalah *gradient* yang fluktuatif dan memungkinkan pelatihan yang lebih stabil serta efektif, terutama dengan penggunaan *mini-batch*. RMSprop juga menjadi salah satu algoritma yang menginspirasi pengembangan Adam[18], [19].

3. *Adaptive Moment Estimation*

Adaptive Moment Estimation (Adam) merupakan algoritma *optimizer* yang banyak digunakan dalam pelatihan model *deep learning*. Algoritma ini secara adaptif menyesuaikan *learning rate* untuk setiap bobot jaringan berdasarkan *gradient* masa lalu serta dua momen statistiknya (rata-rata *gradient* pertama dan varians *gradient* tidak terpusat). Adam mengintegrasikan keunggulan dari AdaGrad dan RMSprop, 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 *deep learning*[18], [19].

III. METODE PENELITIAN

Penelitian ini mengikuti alur kerja sistematis yang ditunjukkan pada Gambar 10, yang mencakup tahap-tahap: persiapan, pengumpulan data, analisis eksplorasi (EDA), perancangan model, pra-pemrosesan, pelatihan, dan evaluasi model.

A. *Exploratory Data Analysis*

Exploratory Data Analysis (EDA) dilakukan untuk memahami karakteristik *dataset* intensitas cahaya (*lux*) yang

dikumpulkan dari atap gedung di Jakarta Selatan selama 1 Maret 2023 hingga 29 Februari 2024. *Dataset* terdiri dari 4.758 entri dengan enam variabel, yaitu: No, Bulan, Sudut, *Lux*, Cuaca, dan Tampak.

Tabel 1 Deskripsi Statistik Dataset Lux

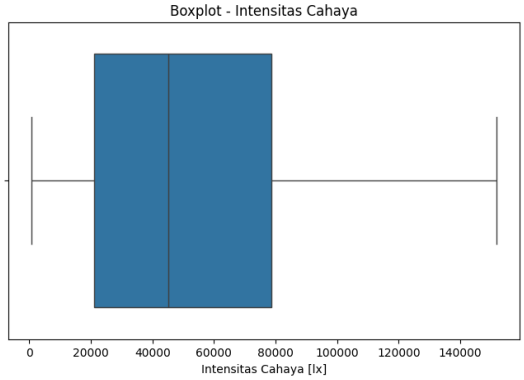
Sudut	Mean	Std	Min	25%	50%	75%	Max
0	19747	12852	203	14311	18944	25571	75420
15	29988	19816	364	17805	29988	40014	95460
30	37373	24358	499	20159	37580	52656	115900
45	43701	29815	635	20125	40760	67860	130800
60	49789	33102	771	22163	50128	76296	135700
75	49794	34947	706	21101	45046	78690	151700
90	46714	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

Analisis difokuskan pada sudut 75° karena menunjukkan nilai rata-rata tertinggi (49.794 *lux*) dan simpangan baku terbesar (34.947). Nilai-nilai ini menunjukkan variabilitas data yang tinggi, yang menjadikan sudut ini representatif untuk pengujian performa model prediksi.

Tabel 2 Hasil pemeriksaan data hilang

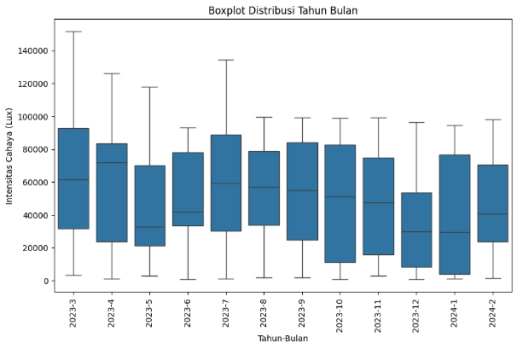
Variabel	Jumlah Missing Value
No.	0
Bulan	0
Sudut	0
Lux	54
Cuaca	0
Tampak	0

Pemeriksaan terhadap *missing values* menunjukkan bahwa hanya variabel *Lux* yang memiliki nilai hilang, yaitu sebanyak 54 entri atau sekitar 14,7% dari total data pada sudut 75°, sedangkan variabel lain tidak memiliki nilai hilang.



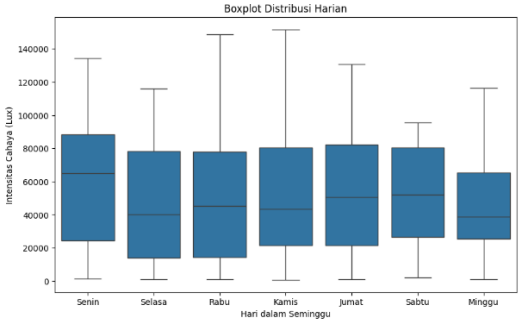
Gambar 3 Boxplot intensitas cahaya

Analisis *outlier* menggunakan *boxplot* menunjukkan adanya nilai-nilai ekstrem pada variabel *Lux*. Namun, nilai-nilai ekstrem tersebut tidak dihapus karena dianggap mencerminkan kondisi cuaca ekstrem yang nyata, seperti siang hari yang sangat cerah atau hujan lebat.



Gambar 4 Boxplot distribusi data bulanan

Distribusi data juga dianalisis secara bulanan dan harian menggunakan *boxplot*. Secara bulanan, median nilai *Lux* cenderung lebih tinggi pada musim kemarau (misalnya Maret dan Juli 2023) dan lebih rendah pada musim hujan (Desember 2023 dan Januari 2024). Secara harian, nilai median tertinggi terjadi pada hari Senin dan Jumat, sedangkan Minggu menunjukkan variabilitas tertinggi dengan median terendah.

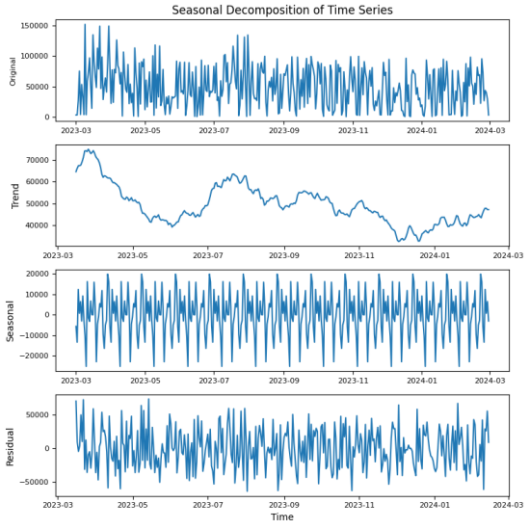


Gambar 5 Boxplot distribusi data harian

Untuk mendalami pola deret waktu, dilakukan dekomposisi deret waktu terhadap data *lux* untuk memisahkan komponen Tren, Musiman, dan Residualnya. Komponen Tren menunjukkan penurunan bertahap dari Maret 2023 kemudian menaik kembali di sekitar Agustus 2023. Komponen musiman menunjukkan pola periodik yang konsisten dari bulan ke bulan, sedangkan residual berfluktuasi di sekitar nol, menunjukkan struktur data telah ditangkap dengan baik.

Temuan-temuan hasil EDA ini menegaskan bahwa data memiliki pola temporal yang kuat, baik dalam jangka pendek (harian), maupun panjang (musiman). Oleh karena itu, model sekuensial seperti LSTM dipilih sebagai pendekatan utama. Selain itu, variabilitas data yang tinggi dan

ketidakteraturan musiman menjadi dasar untuk mengevaluasi kinerja *optimizer* dalam kondisi data yang tidak sepenuhnya stasioner.

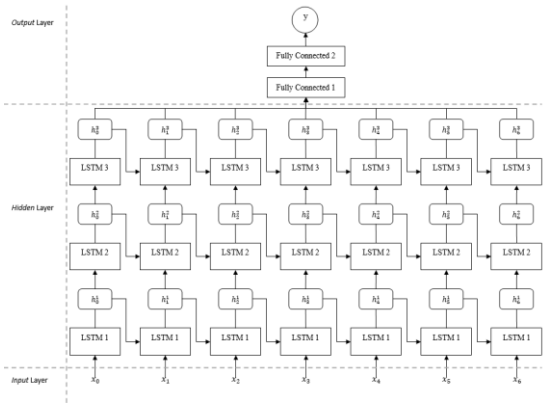


Gambar 6 Timeplot Lux

B. Arsitektur Model dan Desain Eksperimen

Penelitian ini menggunakan arsitektur *Stacked Long Short-Term Memory (LSTM)* dengan tiga lapisan bertingkat. Pemilihan arsitektur bertumpuk ini didasarkan pada hasil analisis EDA yang mengungkap adanya pola temporal berlapis dan kompleks dalam data *lux* harian. Dengan menumpuk beberapa lapisan LSTM, model dirancang untuk dapat mempelajari representasi data pada berbagai tingkat abstraksi: lapisan pertama dapat menangkap pola temporal jangka pendek, sedangkan lapisan-lapisan berikutnya dapat mensintesis informasi tersebut untuk mengenali pola jangka panjang yang lebih kompleks, seperti pola musiman. Pendekatan ini sejalan dengan temuan Alzahrani dkk. (2017) yang menunjukkan bahwa jaringan dengan kedalaman yang lebih tinggi cenderung lebih efisien dalam merepresentasikan fungsi yang kompleks[8].

Arsitektur spesifik yang digunakan dalam penelitian ini terdiri dari tiga lapisan LSTM dengan jumlah unit yang menurun bertahap (128, 64, 32) yang berfungsi sebagai blok pemrosesan sekuensial, diikuti oleh tiga lapisan *Dense (fully connected)* yang bertugas sebagai blok regresi untuk menghasilkan satu nilai prediksi akhir. *Input* data diproses menggunakan pendekatan *Sliding Window* sepanjang 7 hari (*timesteps*) untuk memprediksi nilai iradiasi pada hari ke-8.



Gambar 7 Arsitektur model Stacked LSTM.

Untuk memastikan perbandingan yang adil dan konsisten, arsitektur ini digunakan sebagai *baseline architecture* pada seluruh skenario eksperimen yang dirancang untuk mengevaluasi pengaruh dari tiga *optimizer* berbeda: SGD, RMSprop, dan Adam.

Sebanyak 12 skenario eksperimen dirancang secara sistematis. Variabel yang diuji adalah konfigurasi *optimizer*, sedangkan komponen lainnya seperti arsitektur model, fungsi kerugian (MSE), jumlah *epoch* (150), dan ukuran *batch* pelatihan (1) dijaga tetap (*constant*). Eksperimen dilakukan dengan variasi *hyperparameter* sebagai berikut:

- *Learning Rate (η):* Diuji pada dua nilai, yaitu 0.01 (tinggi) untuk mengamati kecepatan konvergensi dan 0.001 (rendah) untuk mengamati stabilitas pelatihan.
- Parameter Internal:
  - Untuk SGD, diuji dengan *momentum* 0 dan 0.9.
  - Untuk RMSProp dan Adam, diuji dengan kombinasi nilai parameter *decay rate* (β) *default* dan alternatif, untuk mengamati pengaruh "memori" gradien terhadap proses optimasi pada data yang sangat fluktuatif.

Tabel 3 Rangkuman 12 skenario percobaan

Percobaan	*Optimizer*	*Learning Rate*	*Parameter*
Percobaan 1	SGD	0.01	*Momentum = 0*
Percobaan 2	SGD	0.01	*Momentum = 0.9*
Percobaan 3	SGD	0.001	*Momentum = 0*
Percobaan 4	SGD	0.001	*Momentum = 0.9*
Percobaan 5	RMSProp	0.01	$\beta = 0.9$
Percobaan 6	RMSProp	0.01	$\beta = 0.5$
Percobaan 7	RMSProp	0.001	$\beta = 0.9$
Percobaan 8	RMSProp	0.001	$\beta = 0.5$
Percobaan 9	Adam	0.01	$\beta1 = 0.9, \beta2 = 0.99$
Percobaan 10	Adam	0.01	$\beta1 = 0.5, \beta2 = 0.85$
Percobaan 11	Adam	0.001	$\beta1 = 0.9, \beta2 = 0.99$
Percobaan 12	Adam	0.001	$\beta1 = 0.5, \beta2 = 0.85$

C. Data Preprocessing

Data *lux* yang telah dikumpulkan diolah melalui serangkaian proses *data preprocessing* untuk mempersiapkan data ke dalam format yang sesuai bagi pemodelan deret waktu menggunakan LSTM. Proses ini mencakup pembersihan data, transformasi satuan, normalisasi, seleksi fitur, serta pembagian dan segmentasi data.

1. Pembersihan Data

Sebanyak 54 *missing values* pada variabel *Lux* diatasi dengan menggunakan metode imputasi acak bersyarat (*conditional random imputation*). Metode ini mengisi nilai hilang dengan angka acak yang diambil dari rentang minimum-maksimum (*min-max*) berdasarkan kombinasi kondisi Bulan dan Cuaca, sehingga distribusi dan variabilitas alami data tetap terjaga.

Tabel 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

Sedangkan untuk *outlier* atau nilai-nilai ekstrem yang ada pada data tidak dihapus atau dimodifikasi karena dianggap sebagai representasi valid dari kondisi cuaca ekstrem di Jakarta Selatan. Oleh karena itu, seluruh nilai tetap dipertahankan guna menjaga integritas dan kelengkapan data historis.

2. Konversi *Lux* Ke Iradiasi Matahari

Variabel target *Lux* dikonversi ke dalam satuan iradiasi matahari (W/m²) menggunakan hubungan konversi dari studi Michael, dkk. (2020) yang menyatakan untuk kondisi sinar matahari alami di luar ruangan 1 W/m² ≈ 122 lx[11]. Rumus konversi yang digunakan adalah:

$$Iradiasi = \frac{lux}{122} \qquad (9)$$

Konversi ini memungkinkan interpretasi model terhadap nilai *Lux* dalam konteks energi yang lebih relevan terhadap performa panel surya.

3. Normalisasi Data

Setelah konversi satuan, variabel target dinormalisasi menggunakan teknik *Min-Max Scaling* untuk mentransformasi setiap nilai ke dalam rentang [0, 1]. Normalisasi ini bertujuan untuk mempercepat konvergensi model dan menghindari dominasi nilai numerik yang besar. Teknik *Min-Max Scaling* dirumuskan sebagai berikut[20]:

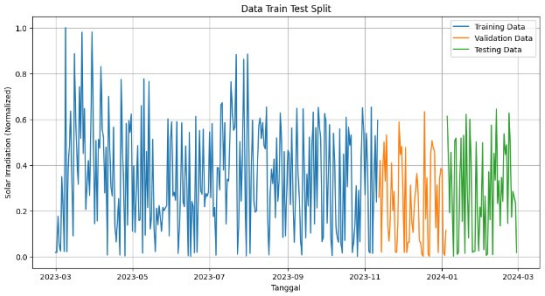
$$x' = \frac{x - x_{min}}{x_{max} - x_{min}} \qquad (10)$$

4. Seleksi Fitur

Model kemudian disederhanakan menjadi model deret waktu univariat yang hanya menggunakan data historis *Lux* (setelah dikonversi menjadi iradiasi dan dinormalisasi). Model disederhanakan dengan menghapus fitur Sudut dan Tampak karena nilainya konstan dan tidak prediktif. Variabel Bulan karena informasi temporalnya sudah terekspresikan secara implisit dalam *DateTimeIndex*. Variabel Cuaca juga sengaja tidak disertakan untuk menguji hipotesis bahwa nilai *Lux* itu sendiri secara implisit merefleksikan kondisi cuaca, sehingga model ditantang untuk belajar hanya dari informasi historis *Lux*.

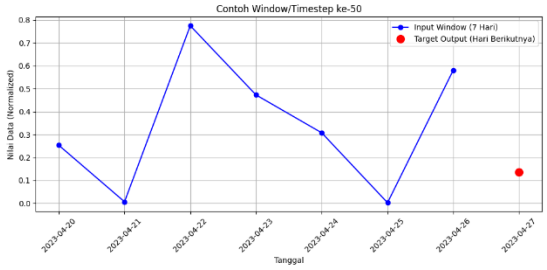
5. Pembagian dan Segmentasi Data

Dataset 366 hari dibagi secara kronologis menjadi data latih (70%), validasi (15%), dan uji (15%). Data latih (256 hari) dari 1 Maret 2023 – 11 November 2023; data validasi (55 hari) dari 12 November 2023 – 5 Januari 2024; dan data uji (55 hari) dari 6 Januari 2024 – 29 Februari 2024.



Gambar 8 Pembagian data

Setelah dibagi, ketiga *subset* data tersebut disegmentasi menggunakan pendekatan *Sliding Window* sepanjang 7 *timesteps*. Dalam setiap segmen, tujuh data historis berurutan digunakan sebagai *input* (x) untuk memprediksi nilai pada hari ke-8 sebagai *output* (y). Proses ini menghasilkan 249 sampel latih, 48 sampel validasi, dan 48 sampel uji.



Gambar 9 Sampel segmentasi data

D. Metrik Evaluasi

Untuk menilai akurasi model dalam memprediksi iradiasi matahari, kinerja model dievaluasi secara kuantitatif menggunakan tiga metrik evaluasi standar, yaitu *Mean Absolute Error* (MAE), *Root Mean Square Error* (RMSE), dan *Mean Absolute Percentage Error* (MAPE). Evaluasi dilakukan terhadap hasil prediksi pada data latih dan data uji.

Sebelum perhitungan metrik dilakukan, seluruh hasil prediksi yang telah dinormalisasi dikembalikan terlebih dahulu ke skala asli menggunakan rumus invers dari *Min-Max Scaling* berikut[16]:

$$y = y'(x_{max} - x_{min}) + x_{min} \quad (11)$$

Dengan y' adalah nilai prediksi hasil normalisasi, dan x_{min}, x_{max} adalah nilai minimum dan maksimum dari data aktual.

1. *Mean Absolute Error* (MAE)

MAE mengukur rata-rata dari selisih absolut antara nilai prediksi dan nilai aktual. Metrik ini memberikan gambaran umum mengenai besar kesalahan tanpa memperhatikan arah kesalahan (positif atau negatif)[21], [22].

$$MAE = \frac{\sum_{i=1}^{n} |y_i - \hat{y}_i|}{n} \quad (12)$$

2. *Root Mean Square Error (RMSE)*

RMSE mengkuadratkan selisih antara nilai prediksi dan aktual sebelum dirata-rata, sehingga memberikan bobot lebih besar terhadap kesalahan yang ekstrem. Oleh karena itu, metrik ini cocok digunakan ketika kesalahan besar perlu mendapatkan perhatian lebih[21], [22].

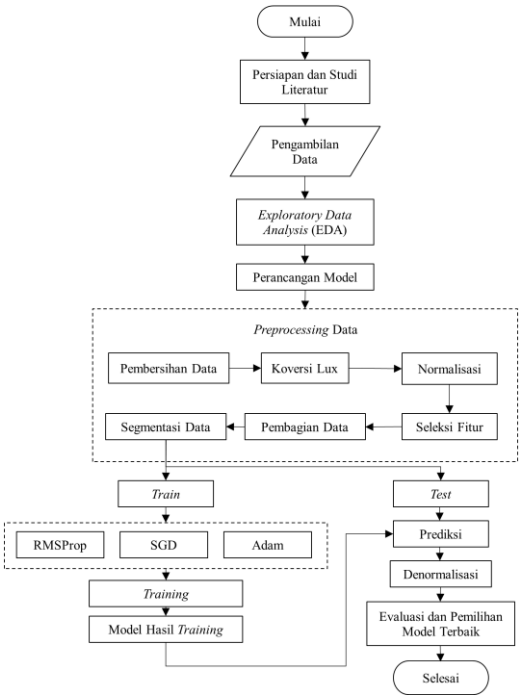
$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \quad (13)$$

3. *Mean Absolute Percentage Error (MAPE)*

MAPE mengukur rata-rata kesalahan absolut dalam bentuk persentase terhadap nilai aktual. Metrik ini berguna untuk menilai akurasi model dalam skala relatif dan lebih mudah diinterpretasikan[23], [24].

$$MAPE = \frac{\sum_{i=1}^{n} \left| \frac{y_i - \hat{y}_i}{y_i} \right| \times 100\%}{n} \quad (14)$$

Dengan n adalah jumlah total observasi atau sampel, y_i adalah nilai aktual observasi ke-i, dan $\hat{y}_i$ adalah nilai prediksi observasi ke-i.



Gambar 10 Diagram Alur Penelitian

IV. HASIL PENELITIAN DAN PEMBAHASAN

A. Analisis Kinerja Selama Pelatihan Model

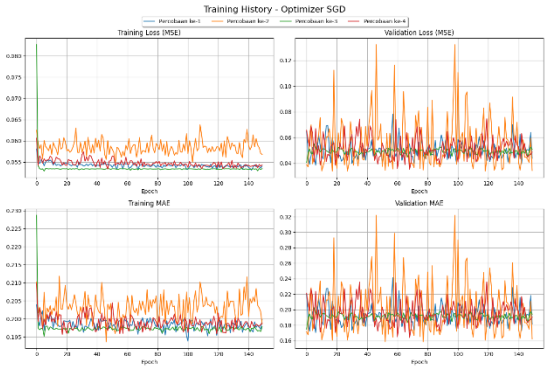
Evaluasi performa masing-masing algoritma *optimizer* dilakukan melalui analisis histori pelatihan secara kualitatif (melalui kurva *loss*) dan kuantitatif (berdasarkan metrik evaluasi pada data latih). Analisis ini bertujuan untuk memahami dinamika pembelajaran model selama proses pelatihan.

1. Hasil Pelatihan *Optimizer* SGD

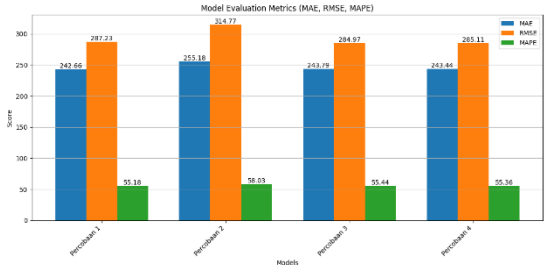
Optimizer SGD menunjukkan sensitivitas yang tinggi terhadap nilai *learning rate* dan parameter *momentum*. Seperti yang ditunjukkan pada Gambar 11, penggunaan *learning rate* tinggi ($\eta = 0.01$) pada percobaan 1 menghasilkan kurva *validation loss* yang berfluktuasi secara signifikan, menandakan proses pelatihan yang kurang stabil. Ketidakstabilan ini semakin meningkat pada Percobaan 2, dimana penambahan *momentum* sebesar 0.9 membuat pembaruan bobot menjadi lebih besar dan agresif, sehingga memperparah fluktuasi.

Sebaliknya, pada Percobaan 3 *learning rate* yang lebih rendah ($\eta = 0.001$) menyebabkan pelatihan cepat mengalami stagnasi (*plateau*), dimana kurva *loss* tidak menunjukkan penurunan yang berarti setelah beberapa *epoch* awal. Sedangkan penambahan *momentum* sebesar 0.9 pada Percobaan 4 tampak membantu model untuk terus belajar secara perlahan-lahan tanpa mengalami stagnasi separah Percobaan 3.

Secara kuantitatif, ditunjukkan pada Gambar 12, Percobaan 1 ($\eta = 0.01$, $momentum = 0$) menghasilkan nilai MAE dan MAPE terendah pada data latih dibandingkan percobaan lainnya. Hal ini menunjukkan bahwa dalam konteks data ini, konfigurasi SGD dengan *learning rate* yang lebih tinggi dan tanpa *momentum* mampu memberikan hasil prediksi terbaik secara rata-rata, meskipun stabilitas pelatihannya kurang optimal.



Gambar 11 Grafik perbandingan hasil pelatihan optimizer SGD



Gambar 12 Perbandingan metrik hasil pelatihan optimizer SGD

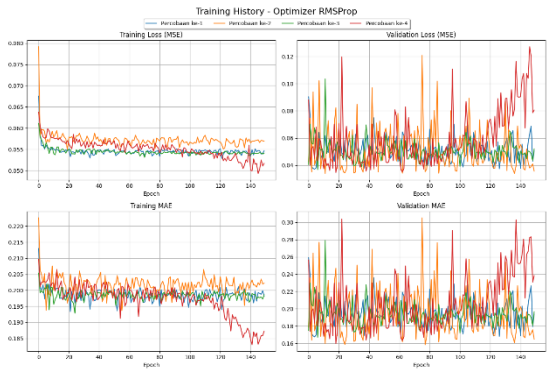
2. Hasil Pelatihan *Optimizer* RMSProp

Optimizer RMSProp menunjukkan kemampuan konvergensi awal yang lebih cepat dibandingkan dengan SGD. Meskipun demikian, performanya sangat bergantung pada pengaruh *hyperparameter*, khususnya parameter *decay rate* (β) yang mengatur besarnya "memori" terhadap kuadrat gradien dari langkah-langkah sebelumnya.

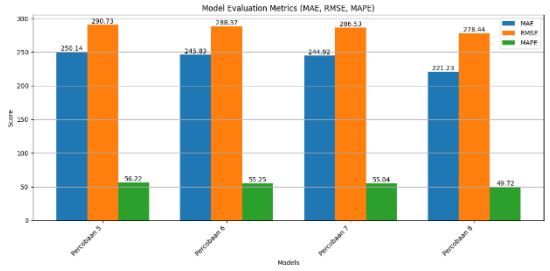
Pada konfigurasi dengan *learning rate* tinggi ($\eta = 0.01$), nilai β yang rendah (0.5) pada Percobaan 6 menyebabkan respon berlebihan terhadap gradien terkini, yang mengakibatkan fluktuasi *loss* yang tinggi selama pelatihan. Hal ini mengindikasikan bahwa pada *learning rate* besar, *decay rate* yang rendah membuat pembelajaran menjadi terlalu sensitif dan tidak stabil.

Sebaliknya, temuan signifikan muncul pada *learning rate* rendah ($\eta = 0.001$). Pada Percobaan 8 ($\beta = 0.5$), kurva *training loss* yang ditunjukkan pada Gambar 13 menunjukkan penurunan yang konsisten dan berkelanjutan, menandakan proses pembelajaran yang lebih stabil dan efektif. Nilai β yang rendah dalam konteks ini membuat *optimizer* lebih adaptif terhadap gradien terkini, yang justru menguntungkan dalam kondisi *learning rate* rendah.

Hasil ini diperkuat oleh analisis kuantitatif pada Gambar 14, di mana Percobaan 8 secara konsisten menunjukkan performa terbaik pada seluruh metrik evaluasi terhadap data latih, termasuk MAE sebesar 221.23. Hal ini menjadikan konfigurasi tersebut sebagai yang paling optimal untuk RMSProp dalam eksperimen ini.



Gambar 13 Grafik perbandingan hasil pelatihan optimizer RMSProp



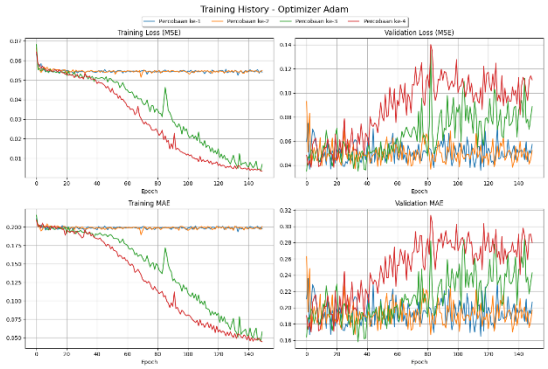
Gambar 14 Perbandingan metrik hasil pelatihan optimizer RMSProp

3. Hasil Pelatihan *Optimizer* Adam

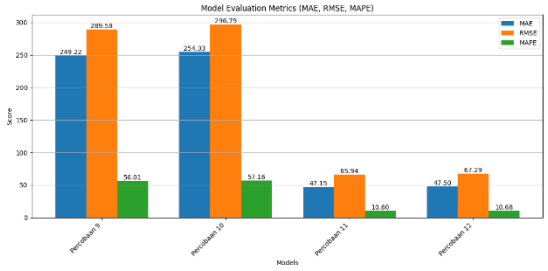
Optimizer Adam menunjukkan performa paling unggul pada data latih dibandingkan dua *optimizer* lainnya. Faktor utama yang memengaruhi keberhasilan ini adalah penggunaan *learning rate* rendah. Pada Percobaan 11 dan 12 (η=0.001), kurva *training loss* (Gambar 15) menunjukkan penurunan yang drastis dan stabil sejak awal pelatihan, yang mencerminkan kemampuan belajar yang sangat efisien dan cepat.

Namun demikian, pada kurva *validation loss* terlihat adanya tren kenaikan setelah sekitar epoch ke-80, yang merupakan indikasi terjadinya *overfitting*. Fenomena ini menunjukkan bahwa model mulai terlalu menyesuaikan diri dengan data pelatihan, sehingga kehilangan kemampuan generalisasi terhadap data baru.

Variasi pada parameter β_1 dan β_2 dalam Percobaan 11 dan 12 tidak menunjukkan pengaruh signifikan terhadap performa, yang mengindikasikan *robustness* Adam terhadap perubahan *hyperparameter* tersebut ketika *learning rate* berada pada tingkat rendah.



Gambar 15 Grafik perbandingan hasil pelatihan optimizer Adam



Gambar 16 Perbandingan metrik hasil pelatihan optimizer Adam

Dari sisi evaluasi kuantitatif (Gambar 16), kedua percobaan ini secara konsisten mencatatkan nilai kesalahan terendah dibandingkan seluruh percobaan lainnya, dengan nilai MAE sekitar 47 dan MAPE sekitar 10,6%. Hasil ini menegaskan dominasi Adam sebagai *optimizer* dengan akurasi prediksi tertinggi pada data pelatihan.

Dari seluruh eksperimen yang telah dilakukan, terdapat empat konfigurasi model yang menunjukkan performa paling menonjol dan dianggap sebagai kandidat terbaik untuk tahap selanjutnya. Keempat konfigurasi ini dirangkum dalam Tabel 5 dan akan digunakan dalam evaluasi akhir untuk menentukan model terbaik berdasarkan kemampuannya dalam melakukan generalisasi terhadap data uji.

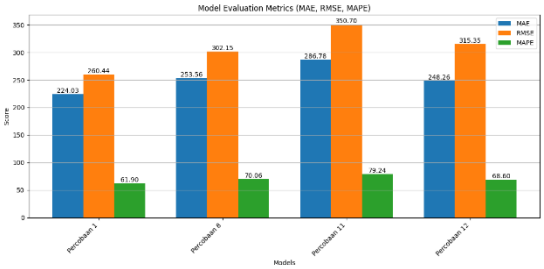
Tabel 5 Empat konfigurasi terbaik berdasarkan evaluasi pada data latih

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	$\eta = 0.001$, $\beta1 = 0.9$, $\beta2 = 0.99$	47.15	65.94	10.60
Percobaan 12	Adam	$\eta = 0.001$, $\beta1 = 0.5$, $\beta2 = 0.85$	47.50	67.29	10.68

B. Evaluasi Akhir dan Pemilihan Model Terbaik

Evaluasi akhir bertujuan untuk menentukan model yang paling mampu melakukan generalisasi terhadap data uji yang belum pernah dilihat sebelumnya. Evaluasi ini mencakup pendekatan kuantitatif, melalui metrik evaluasi, serta pendekatan kualitatif melalui visualisasi hasil prediksi terhadap data aktual.

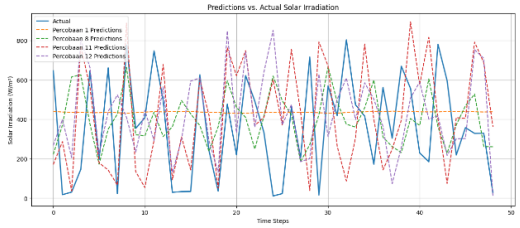
Berdasarkan hasil evaluasi kuantitatif (Gambar 17), ditemukan hasil yang cukup kontras dengan ekspektasi awal. Model dari Percobaan 1, yang menggunakan *optimizer* SGD, justru mencatatkan nilai *error* terendah pada data uji, yakni MAE sebesar 224,03 dan RMSE sebesar 260,44. Sebaliknya, model dari Percobaan 11 dan 12, yang menggunakan *optimizer* Adam dan sebelumnya unggul pada data latih, justru mengalami peningkatan *error* pada data uji.



Gambar 17 Grafik perbandingan metrik 4 model terbaik pada data uji

Namun demikian, analisis kualitatif melalui visualisasi hasil prediksi (Gambar 18)

mengungkapkan perbedaan mendasar dalam performa model-model tersebut.



Gambar 18 Grafik perbandingan hasil prediksi 4 model terbaik

Model dari Percobaan 1 (SGD), yang ditampilkan dengan garis oranye putus-putus, menghasilkan prediksi yang nyaris konstan, gagal menangkap dinamika iradiasi matahari harian. Karakteristik ini mengindikasikan *underfitting*, di mana model terlalu sederhana dan hanya merepresentasikan nilai rata-rata, sehingga kurang berguna dalam konteks aplikasi praktis.

Model dari Percobaan 8 (RMSProp, garis hijau) menunjukkan peningkatan dalam mengikuti pola data, namun masih kurang responsif terhadap fluktuasi ekstrem, seperti puncak dan lembah. Sementara itu, model dari Percobaan 11 dan 12 (Adam) yang digambarkan dengan garis merah dan ungu, memperlihatkan performa kualitatif yang lebih unggul. Meskipun nilai *error*-nya lebih tinggi secara absolut, model-model ini berhasil mengikuti pola fluktuasi data aktual dengan baik, termasuk dalam mendeteksi sebagian besar puncak dan lembah iradiasi.

Dengan mempertimbangkan kedua aspek tersebut, dapat disimpulkan bahwa keunggulan kuantitatif dari model Percobaan 1 diperoleh melalui strategi prediksi rata-rata, yang meskipun mencatatkan nilai MAE dan RMSE rendah, tidak mampu menangkap kompleksitas data. Sebaliknya, model dari Percobaan 12 memberikan keseimbangan optimal antara akurasi metrik dan kemampuan merepresentasikan pola data secara dinamis.

Secara numerik, Percobaan 12 menempati posisi kedua terbaik pada metrik MAE (248,26) dan MAPE (68,60%), namun secara visual, prediksi yang dihasilkan berhasil mengikuti tren aktual, termasuk perubahan cuaca harian yang berdampak pada iradiasi. Hal ini menunjukkan bahwa model ini memiliki kapasitas generalisasi yang baik serta relevansi praktis yang tinggi dalam konteks prediksi iradiasi matahari.

Berdasarkan pertimbangan komprehensif antara evaluasi kuantitatif dan kualitatif, model Percobaan 12 dengan konfigurasi *optimizer* Adam $(\eta = 0.001, \beta^1 = 0.5, \beta^2 = 0.85)$ dipilih sebagai model terbaik dalam penelitian ini. Model ini tidak hanya menunjukkan performa metrik yang sangat kompetitif, tetapi juga mampu merepresentasikan pola fluktuasi data secara konsisten, menjadikannya layak untuk digunakan dalam aplikasi prediktif jangka panjang.

V. KESIMPULAN DAN SARAN

Berdasarkan analisis dan pembahasan yang telah dilakukan, dapat disimpulkan bahwa penerapan algoritma *Machine Learning* untuk prediksi iradiasi matahari telah berhasil dilakukan. Terdapat perbedaan performa yang signifikan antara ketiga *optimizer* yang diuji, yaitu SGD, RMSProp, dan Adam. Di antara ketiganya, *optimizer* Adam menunjukkan kecepatan konvergensi dan stabilitas terbaik selama pelatihan.

Pada tahap pengujian, model dengan *optimizer* SGD dan RMSProp cenderung mengalami *underfitting* dan gagal menangkap fluktuasi data aktual. Sebaliknya, model dengan *optimizer* Adam terbukti paling mampu mengikuti pola dinamis dari data aktual. Adam dipilih sebagai *optimizer* paling optimal untuk kasus prediksi iradiasi matahari dalam penelitian ini.

Konfigurasi terbaik diperoleh pada Percobaan 12. Model ini dipilih bukan hanya karena metrik *error*-nya yang kompetitif, melainkan juga karena kemampuannya yang unggul secara kualitatif dalam menangkap volatilitas pada data uji. Model terbaik ini menghasilkan performa akhir dengan nilai MAE 248.26, RMSE 315.35, dan MAPE 68.60%.

Model terbaik ini memiliki potensi untuk dikembangkan menjadi sistem prediksi praktis yang dapat diintegrasikan dengan *Energy Management System* (EMS) pada instalasi panel surya, guna mengoptimalkan penjadwalan penggunaan dan penyimpanan energi.

Untuk pengembangan penelitian selanjutnya, mengingat kecenderungan *underfitting* pada model SGD dan RMSProp, disarankan untuk mengeksplorasi arsitektur *deep dearning* yang lebih kompleks, seperti menambah jumlah lapisan/unit pada LSTM,

atau mempertimbangkan penggunaan arsitektur sekuensial lain seperti *Gated Recurrent Unit* (GRU) atau model sekuensial lainnya.

Kemudian untuk meningkatkan akurasi prediksi, dapat dipertimbangkan untuk mengembangkan model multivariat dengan menyertakan kembali fitur cuaca (setelah melalui proses *encoding*) atau menambahkan data meteorologi lain yang relevan seperti suhu udara, kelembapan, dan kecepatan angin.

Untuk menemukan konfigurasi *hyperparameter* yang lebih optimal, disarankan menggunakan teknik optimasi *hyperparameter* otomatis seperti *Grid Search, Random Search*, atau *Bayesian Optimization*.

Penelitian selanjutnya juga dapat menjelajahi pengaruh penggunaan ukuran *batch* pelatihan yang lebih besar (misalnya 8, 16, atau 32) yang berpotensi meningkatkan kecepatan konvergensi, stabilitas pelatihan, serta efisiensi komputasi.

DAFTAR PUSTAKA

[1] A. O. M. 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] 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.

[8] 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.

[9] 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.

[10] 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.

[11] 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.

[12] 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

[13] 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.

[14] 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.
[15] Y. Yu, X. Si, C. Hu, dan J. Zhang, "A Review of *Recurrent Neural Network*s: LSTM Cells and Network Architectures," *Neural Computation*, vol. 31, no. 7, hlm. 1235–1270, Jul 2019, doi: 10.1162/neco_a_01199.
[16] L. Wiranda dan M. Sadikin, "PENERAPAN LONG SHORT TERM MEMORY PADA DATA TIME SERIES UNTUK MEMPREDIKSI PENJUALAN PRODUK PT. METISKA FARMA," vol. 8, 2019.
[17] 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
[18] "*Optimizer*s 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-*optimizer*s/
[19] V. Efimov, "Understanding *Deep* Learning *Optimizer*s: Momentum, AdaGrad, RMSProp & Adam," Medium. Diakses: 25 Juni 2024. [Daring]. Tersedia pada: https://towardsdatascience.com/understanding-*deep*-learning-*optimizer*s-momentum-adagrad-rmsprop-adam-e311e377e9c2
[20] 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.
[21] "Evaluation Metrics in *Machine Learning*," GeeksforGeeks. Diakses: 25 Juni 2024. [Daring]. Tersedia pada: https://www.geeksforgeeks.org/metrics-for-machine-learning-model/
[22] 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
[23] 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/
[24] apoorvakumar169, "*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/