

**LAPORAN PROJECT BASED LEARNING**  
**MATA KULIAH DEEP LEARNING**

**“Prediksi Kecepatan dan Arah Angin Menggunakan Long Short-Term  
Memory (LSTM) dan Bidirectional LSTM (BiLSTM): Studi Kasus BMKG  
Juanda”**



**DISUSUN OLEH:**

1. Nayya Ramadhani Putri Widjanarko (21083010075)
2. Firly Setya Wardani (21083010093)
3. Radya Ardi Ninang Pudyastuti (21083010097)
4. Novita Anggraini (21083010103)
5. Cesaria Deby Nurhalizah (21083010120)

**DOSEN PENGAMPU:**

Tresna Maulana Fahrudin S.ST., M.T.	(NIP. 199305012022031007)
Alfan Rizaldy Pratama, S.Tr.T., M.Tr.Kom.	(NIP. 199906062024061001)

**PROGRAM STUDI SAINS DATA**  
**FAKULTAS ILMU KOMPUTER**  
**UNIVERSITAS PEMBANGUNAN NASIONAL “VETERAN” JAWA TIMUR**  
**2024**

## DAFTAR ISI

<b>DAFTAR ISI.....</b>	<b>1</b>
<b>DAFTAR GAMBAR.....</b>	<b>3</b>
<b>DAFTAR TABEL.....</b>	<b>4</b>
<b>BAB 1</b>	
<b>PENDAHULUAN.....</b>	<b>1</b>
1.1. Latar Belakang.....	1
1.2. Rumusan Masalah.....	3
1.3. Tujuan.....	4
1.4. Manfaat.....	4
<b>BAB 2</b>	
<b>TINJAUAN PUSTAKA.....</b>	<b>5</b>
2.1 Review Penelitian Sebelumnya.....	5
2.2 Dasar Teori.....	8
2.2.1 Meteorologi.....	8
2.2.2 Prediksi.....	9
2.2.3 Long Short-Term Memory (LSTM).....	10
2.2.4 Bidirectional Long Short-Term Memory (Bi-LSTM).....	12
2.2.5 Metrik Evaluasi.....	14
<b>BAB 3</b>	
<b>METODOLOGI PENELITIAN.....</b>	<b>16</b>
3.1 Metodologi Penelitian.....	16
3.1.1 Akuisisi Data.....	16
3.1.2 Data Preprocessing.....	17
3.1.2.1 Import Data.....	18
3.1.2.2 Penanganan Missing Value.....	19
3.1.2.3 Mengubah Tipe Data Arah Angin.....	21
3.1.2.4 Normalisasi Data.....	22
3.1.2.5 Menentukan Jumlah Lag dan Mendefinisikan Variabel X dan y.....	23
3.1.2.6 Pembagian Data menjadi Set Training dan Testing.....	24
3.1.2.7 Transform Input Menjadi Format 3D.....	25
3.1.3 Modeling Bi-LSTM.....	26
3.1.3.1 Mendefinisikan Model LSTM.....	26
3.1.3.2 Mendefinisikan Model Bi-LSTM.....	27
3.1.4 Evaluasi.....	29
3.1.4.1 Pengumpulan Data Loss dan Validation Loss.....	29
3.1.4.2 Penghitungan Metrik Evaluasi.....	30
3.1.4.3 Penentuan Model Terbaik.....	30

<b>BAB 4</b>	
<b>HASIL DAN ANALISIS.....</b>	<b>32</b>
4.1 Hasil Pemodelan.....	32
4.1.1 Arsitektur LSTM.....	33
4.1.2 Arsitektur Bi-LSTM.....	35
4.2 Pemilihan Model Terbaik.....	38
4.2.1 Prediksi dengan Model Hyperparameter Terbaik.....	38
4.2.2 Evaluasi Metrik.....	40
<b>BAB 5</b>	
<b>KESIMPULAN DAN SARAN.....</b>	<b>42</b>
5.1 Kesimpulan.....	42
5.2 Saran.....	42
<b>DAFTAR PUSTAKA.....</b>	<b>43</b>

## DAFTAR GAMBAR

Gambar 2.1 Arsitektur LSTM.....	10
Gambar 2.2 Arsitektur LSTM Dua Arah.....	13
Gambar 3.1 Flowchart Metodologi Penelitian.....	16
Gambar 3.2 Flowchart Preprocessing Data.....	18
Gambar 3.3 Hasil import data pada Python.....	19
Gambar 3.4 Hasil Pengisian Missing Value.....	20
Gambar 3.5 Hasil Mengubah Tipe Data Angin.....	21
Gambar 3.6 Data Setelah Dilakukan Proses Normalisasi.....	23
Gambar 3.7 Alur Penghitungan Evaluasi Model.....	29
Gambar 4.1 Data train_X, train_y, test_X, test_y.....	32
Gambar 4.2 Grafik Training dan Validation Loss Model LSTM.....	34
Gambar 4.3 Training dan Validation Loss Model Bi-LSTM.....	37
Gambar 4.4 Grafik Perbandingan Prediksi dan Aktual Kecepatan Angin Model LSTM.....	38
Gambar 4.5 Grafik Perbandingan Prediksi dan Aktual Arah Angin Model LSTM.....	39
Gambar 4.6 Grafik Perbandingan Prediksi dan Aktual Kecepatan Angin Model Bi-LSTM.....	39
Gambar 4.7 Grafik Perbandingan Prediksi dan Aktual Arah Angin Model Bi-LSTM.....	40

## DAFTAR TABEL

Tabel 2.1 Studi Literatur.....	9
Tabel 3.1 Dataset Angin.....	21
Tabel 3.1 Kombinasi Konfigurasi Model LSTM.....	30
Tabel 3.2 Kombinasi Konfigurasi Model Bi-LSTM.....	32
Tabel 4.1 Arsitektur Model LSTM.....	36
Tabel 4.2 Tabel Hyperparameter Tuning Model LSTM.....	37
Tabel 4.3 Hasil Training dan Validation Loss Parameter Model LSTM.....	37
Tabel 4.4 Arsitektur Model Bidirectional LSTM.....	38
Tabel 4.5 Tabel Hyperparameter Tuning Model Bidirectional LSTM.....	39
Tabel 4.6 Hasil Training dan Validation Loss Parameter Model Bi-LSTM.....	40
Tabel 4.7 Evaluasi Metrik Prediksi Model.....	43

## **BAB 1**

### **PENDAHULUAN**

#### **1.1. Latar Belakang**

Dalam era perubahan iklim yang semakin nyata, karakteristik iklim dan cuaca mengalami variasi yang semakin sulit diprediksi. Berbagai fenomena ekstrem, seperti badai, angin kencang, dan cuaca panas yang berkepanjangan, semakin sering terjadi dan mempengaruhi kehidupan manusia. Di Indonesia, yang merupakan negara tropis, kecepatan dan arah angin sangat penting untuk berbagai sektor, seperti pertanian, perikanan, penerbangan, dan pelayaran. Prediksi yang akurat mengenai pola angin dan cuaca sangat dibutuhkan untuk mengantisipasi dampak buruk yang bisa timbul akibat perubahan cuaca. Tantangan ini menjadi semakin besar mengingat Indonesia berada di daerah ekuator dan dikelilingi perairan, sehingga dipengaruhi oleh fenomena angin muson dan perubahan suhu permukaan laut. Fenomena cuaca seperti La Nina dan El Nino juga memperparah tingkat ketidakpastian dalam pola iklim, yang menuntut kemampuan prediksi yang lebih baik. Kepala Badan Meteorologi, Klimatologi, dan Geofisika, Dwikorita Karnawati menyebutkan bahwa masyarakat diminta untuk waspada dan siap-siaga menghadapi cuaca ekstrem dan potensi bencana hidrometeorologi. Dimana saat ini sebagian besar wilayah Indonesia telah memasuki musim penghujan. Adanya fenomena La Nina mengakibatkan potensi penambahan curah hujan hingga 20% sampai awal 2025.

Angin memiliki peran yang cukup besar dalam mempengaruhi perubahan cuaca ekstrem, terutama pada wilayah maritim seperti Indonesia. Angin adalah pergerakan massa udara yang berlangsung secara mendatar, fenomena ini terjadi ketika terdapat perbedaan tekanan antara satu lokasi dengan lokasi lainnya. Pergerakan inilah yang dapat membawa perubahan suhu, kelembapan, dan tekanan udara yang drastis, sehingga memicu fenomena cuaca ekstrem seperti badai, angin kencang, dan hujan lebat. Untuk mengetahui pergerakan angin dapat dilihat dari polanya dengan menggunakan data arah dan kecepatan angin yang telah dilakukan selama 24 jam. Di Indonesia, yang terletak di antara dua samudra dan memiliki iklim tropis, pergerakan angin menjadi faktor penting dalam pola cuaca harian.

Di wilayah tropis seperti Indonesia, angin muson dan angin lokal berperan besar dalam perubahan cuaca secara musiman dan harian. Pola angin muson, misalnya, /membawa udara basah yang dapat memicu curah hujan tinggi selama musim hujan, serta mendukung pembentukan sistem tekanan rendah yang meningkatkan risiko cuaca ekstrem, seperti badai tropis dan puting beliung. Selain itu, variasi dalam kecepatan dan arah angin juga dapat memberikan indikasi awal mengenai potensi terjadinya cuaca ekstrem, seperti gelombang panas atau badai. Dengan analisis data kecepatan dan arah angin, kita bisa memahami dinamika atmosfer yang kompleks dan merancang model

prediktif yang lebih akurat untuk memitigasi dampak dari cuaca ekstrem, terutama di daerah dengan aktivitas ekonomi dan populasi yang tinggi.

Prediksi angin di kawasan perkotaan padat memiliki peran penting dalam berbagai aspek kehidupan perkotaan. Dalam konteks mitigasi bencana, informasi mengenai kecepatan dan arah angin dapat digunakan untuk mengantisipasi dampak buruk dari fenomena cuaca ekstrem seperti angin kencang, badai, atau penyebaran polusi udara. Angin juga mempengaruhi pola hujan dan distribusi kelembaban, yang berpotensi menimbulkan banjir di kawasan perkotaan padat. Ketika angin bergerak lambat atau mengalami konvergensi, uap air akan terkonsentrasi pada wilayah tertentu dan dapat menyebabkan hujan lebat dengan intensitas yang tinggi dalam waktu singkat. Hal ini meningkatkan risiko banjir di area yang infrastrukturnya belum siap menampung volume air besar dalam waktu singkat. Fenomena ini menunjukkan bahwa angin memiliki peran penting dalam berbagai aspek kondisi cuaca yang berdampak pada kehidupan masyarakat kota.

Selain itu, prediksi angin menjadi esensial bagi perencanaan tata ruang, terutama untuk mendukung desain bangunan yang ramah lingkungan dan pengelolaan kualitas udara. Kawasan perkotaan yang padat penduduk, seperti wilayah studi BMKG Juanda, menghadapi tantangan besar dalam meminimalkan risiko akibat pola aliran angin yang kompleks akibat efek gedung-gedung tinggi (*urban canyon effect*). Oleh karena itu, penggunaan model prediktif berbasis Long Short-Term Memory (LSTM) dapat memberikan solusi efektif dengan memanfaatkan data historis dan mendeteksi pola non-linear pada variabel angin.

Menurut penelitian berjudul *Weather Forecast from Time Series Data Using LSTM Algorithm* oleh Nugraha, dkk (2021), model LSTM menunjukkan kinerja yang bervariasi tergantung pada parameter yang digunakan dalam memprediksi cuaca di Kota Serang. Berdasarkan hasil analisis, akurasi model terlihat lebih tinggi ketika memprediksi data suhu dengan nilai RMSE 0,37, diikuti oleh RMSE kecepatan angin 0,72, RMSE sinar matahari 2,79, dan RMSE kelembaban sebesar 5,05. Analisis tersebut juga mengungkapkan adanya tren kenaikan yang tidak terlalu signifikan dalam prediksi suhu di Kota Serang, yang serupa dengan hasil prediksi kelembaban dan intensitas sinar matahari.

Selanjutnya, menurut penelitian berjudul *Analisis Efektivitas Teknik Imputasi pada LSTM untuk Meningkatkan Kualitas Data pada Peramalan Curah Hujan* oleh Nugroho, dkk (2024), metode ini menghasilkan model data *time series* yang lebih unggul dibandingkan RNN. Hasil penelitian menunjukkan bahwa teknik imputasi terbaik adalah metode KNN yang dikombinasikan dengan Bidirectional LSTM (BiLSTM). Model ini mencapai nilai *evaluation metrics* dengan Mean Absolute Error (MAE) sebesar 3,35, Mean Square Error (MSE) sebesar 78,43, Root Mean Squared Error (RMSE) sebesar 8,86, dan R-squared sebesar 0,54.

Penelitian oleh Agung Pramono et al. mengkaji pola angin di pesisir Kendari sepanjang tahun 2021 dengan memisahkan data berdasarkan bulan. Hasilnya menunjukkan bahwa arah angin didominasi oleh timur dan timur laut, dengan kecepatan tertinggi berkisar antara 2,1 hingga 3,6 m/s. Penelitian ini menyoroti variasi kondisi angin yang signifikan tergantung pada musim, di mana bulan Desember hingga Februari menunjukkan dominasi arah barat dan barat daya, sementara bulan lainnya didominasi oleh arah timur.

Penelitian selanjutnya mengusulkan sistem prediksi kecepatan dan arah angin menggunakan Bidirectional Long Short-Term Memory (BiLSTM) untuk periode satu jam ke depan, dengan data dari AMeDAS di Jepang. Sistem BiLSTM, yang memproses data dalam dua arah, menunjukkan peningkatan akurasi signifikan dibandingkan dengan Stacked Long Short-Term Memory (SLSTM) dan Fully Connected Neural Network (FCNN), berdasarkan evaluasi menggunakan Mean Absolute Error (MAE) dan Root Mean Square Error (RMSE). Hasilnya menegaskan bahwa BiLSTM adalah metode yang lebih efektif untuk memodelkan kecepatan dan arah angin, berkontribusi pada pengembangan teknologi prediksi cuaca yang lebih akurat.

Penelitian ini bertujuan untuk memprediksi kecepatan dan arah angin di BMKG Juanda menggunakan dua model machine learning berbasis LSTM, yaitu LSTM dan Bi-LSTM. Dengan membagi data secara bulanan, penelitian ini berfokus pada analisis pola dan tren angin tahunan, yang diharapkan dapat meningkatkan akurasi prediksi harian. Pendekatan ini diambil karena faktor musiman dan perubahan iklim dapat secara signifikan mempengaruhi pola angin. Dengan memanfaatkan kemampuan LSTM dan Bi-LSTM dalam menangkap ketergantungan jangka panjang dan pendek dalam data deret waktu, penelitian ini bertujuan untuk memberikan wawasan yang lebih baik mengenai dinamika angin di wilayah tersebut. Selain itu, penelitian ini juga akan membandingkan kinerja kedua model dalam hal akurasi prediksi, dengan menggunakan matrik evaluasi seperti MAE dan RMSE. Hasil dari penelitian ini diharapkan dapat memberikan kontribusi yang berarti bagi pengembangan sistem prediksi cuaca yang lebih efisien dan akurat, serta mendukung upaya mitigasi dampak perubahan iklim di Indonesia.

## **1.2. Rumusan Masalah**

1. Bagaimana tahapan preprocessing data kecepatan dan arah angin dari BMKG Juanda?
2. Bagaimana melakukan prediksi kecepatan dan arah angin menggunakan model LSTM dan BiLSTM?
3. Bagaimana hasil evaluasi prediksi kecepatan dan arah angin menggunakan model LSTM dan BiLSTM?



### **1.3. Tujuan**

1. Mengetahui tahapan preprocessing data kecepatan dan arah angin dari BMKG Juanda.
2. Melakukan prediksi kecepatan dan arah angin menggunakan model LSTM dan BiLSTM
3. Mengetahui hasil evaluasi prediksi dari data kecepatan dan arah angin menggunakan LSTM dan BiLSTM

### **1.4. Manfaat**

Penelitian terkait Long Short-Term Memory (LSTM) dengan studi kasus ini dapat memberikan manfaat yang dapat dijelaskan dalam dua aspek, yaitu teoritis dan praktis:

#### **a. Manfaat Teoritis**

Secara teoritis, penelitian ini berkontribusi pada pengembangan ilmu pengetahuan di bidang meteorologi dan pemodelan prediktif. Dengan memanfaatkan metode Long Short-Term Memory (LSTM), penelitian ini memperkaya literatur tentang penggunaan model deep learning untuk memprediksi kecepatan dan arah angin. Penggunaan LSTM dalam studi kasus ini dapat menjadi referensi bagi penelitian selanjutnya yang ingin mengembangkan atau meningkatkan akurasi model prediktif di bidang meteorologi.

#### **b. Manfaat Praktis**

Secara praktis, hasil penelitian ini dapat dimanfaatkan oleh lembaga seperti BMKG (Badan Meteorologi, Klimatologi, dan Geofisika) dalam memperbaiki ketepatan prediksi angin. Prediksi kecepatan dan arah angin yang lebih akurat dapat membantu dalam berbagai aplikasi, seperti peringatan dini cuaca ekstrem, mitigasi risiko bencana, dan perencanaan operasional transportasi udara di Bandara Juanda, sehingga masyarakat dan sektor industri memiliki waktu lebih untuk mempersiapkan tindakan pencegahan. Dengan demikian, penelitian ini tidak hanya bermanfaat untuk kepentingan akademis tetapi juga memiliki dampak langsung pada keselamatan publik dan pengambilan keputusan dalam sektor pemerintahan dan industri.

## BAB 2

### TINJAUAN PUSTAKA

#### 2.1 Review Penelitian Sebelumnya

Tabel 2.1 Studi Literatur

No	Profil Pustaka	Metode dan Temuan
1	<p><b>Judul:</b>  <i>Weather Forecast from Time Series Data Using LSTM Algorithm</i></p> <p><b>Penulis:</b>  Yoga Estu Nugraha, Ishak Ariawan, Wildan Aprizal Arifin</p> <p><b>Jurnal/Prosiding:</b>  Jurnal Ilmiah Teknologi Informasi dan Komunikasi (JTIK)</p>	<p><b>Metode:</b>  Penelitian ini menggunakan data yang berasal dari BMKG Stasiun Marinir 1 Serang. Variabel data yang digunakan yaitu temperatur, kelembaban, penyinaran matahari, dan kecepatan angin. Penelitian ini melakukan pengujian menggunakan model LSTM sequential dengan optimizer ADAM dan activation function RELu.</p> <p><b>Temuan:</b>  Model LSTM menunjukkan kinerja yang bervariasi tergantung pada parameter yang digunakan dalam memprediksi cuaca di Kota Serang. Berdasarkan hasil analisis, akurasi model terlihat lebih tinggi ketika memprediksi data suhu dengan nilai RMSE 0,37, diikuti oleh RMSE kecepatan angin 0,72, RMSE sinar matahari 2,79, dan RMSE kelembaban sebesar 5,05. Sehingga dapat disimpulkan bahwa model LSTM cocok digunakan untuk memprediksi data suhu, namun kurang cocok untuk digunakan dalam memprediksi data kelembaban.</p>

No	Profil Pustaka	Metode dan Temuan
2.	<p><b>Judul:</b>  <i>Analisis Efektivitas Teknik Imputasi pada LSTM untuk Meningkatkan Kualitas Data pada Peramalan Curah Hujan</i></p> <p><b>Penulis:</b>  Ariyanto Adi Nugroho,  Muhammad Haris</p> <p><b>Jurnal/Prosiding:</b>  Jurnal Informatika &amp; Rekayasa Elektronika (JIRE)</p>	<p><b>Metode:</b>  Penelitian ini menggunakan data iklim dari Stasiun Meteorologi kelas III Kemayoran Jakarta Pusat selama 3530 hari dari Januari 2011 hingga Agustus 2020. Model yang digunakan adalah LSTM dan BiLSTM dengan parameter yang digunakan <i>Window Size 240, Dense Units 25, Dropout Rate 0,5, Optimizer ADAM, Number of Neurons in Dense Layer 25, Epoch 50, Batch Size 500, dan Loss Function Mean Squared Error.</i></p> <p><b>Temuan:</b>  Metode ini menghasilkan model data <i>time series</i> yang lebih unggul dibandingkan RNN. Hasil penelitian menunjukkan bahwa teknik imputasi terbaik adalah metode KNN yang dikombinasikan dengan Bidirectional LSTM (BiLSTM). Model ini mencapai nilai <i>evaluation metrics</i> dengan Mean Absolute Error (MAE) sebesar 3,35, Mean Square Error (MSE) sebesar 78,43, Root Mean Squared Error (RMSE) sebesar 8,86, dan R-squared sebesar 0,54.</p>

No	Profil Pustaka	Metode dan Temuan
3	<p><b>Judul:</b>  <i>Analisis Kondisi Angin Wilayah Pesisir dengan Diagram Windrose di Kota Kendari Tahun 2021</i></p> <p><b>Penulis:</b>            Agung Pramono, Apit Sutaryani, Dewi Tamara Qothrunada, Hendri Satria WD</p> <p><b>Jurnal/Prosiding:</b>            Seminar Nasional TREnD</p>	<p><b>Metode:</b>            Penelitian ini menggunakan data yang bersumber dari Stasiun Meteorologi Maritim Kendari. Variabel data yang digunakan adalah arah dan kecepatan angin di wilayah pesisir Kendari dengan spesifikasi 24 jam dalam rentang waktu 1 tahun dari bulan Januari - Desember tahun 2021. Metode yang digunakan untuk analisis adalah metode <i>windrose</i>.</p> <p><b>Temuan:</b>            Hasilnya menunjukkan bahwa arah angin didominasi oleh timur dan timur laut, dengan kecepatan tertinggi berkisar antara 2,1 hingga 3,6 m/s. Penelitian ini menyoroti variasi kondisi angin yang signifikan tergantung pada musim, di mana bulan Desember hingga Februari menunjukkan dominasi arah barat dan barat daya, sementara bulan lainnya didominasi oleh arah timur.</p>

No	Profil Pustaka	Metode dan Temuan
4	<p><b>Judul:</b>  <i>Sistem Prediksi Kecepatan dan Arah Angin Menggunakan Bidirectional Long Short-Term Memory</i></p> <p><b>Penulis:</b>            Anggraini Puspita Sari, Ermanu Azizul Hakim, Dwi Arman Prasetya, Rahman Arifuddin, Puput Dani Prasetyo Adi</p> <p><b>Jurnal/Prosiding:</b>            Seminar keinsinyuran 2021</p>	<p><b>Metode:</b>            Penelitian ini menggunakan data yang bersumber dari AMeDAS (<i>Automated Meteorological Data Acquisition System</i>) Jepang dengan interval waktu 10 menit pada kota Takamatsu dengan variabel yang digunakan adalah kecepatan dan arah angin. Model yang digunakan pada analisis ini adalah BiLSTM (<i>Bidirectional LSTM</i>) yang dibandingkan dengan FCNN (<i>Fully Connected Neural Network</i>) dan SLSTM (<i>Stacked LSTM</i>). Evaluasi performa yang digunakan adalah MAE dan RMSE.</p> <p><b>Temuan:</b>            Hasil dari penelitian ini adalah sistem SLSTM dan BiLSTM dapat meningkatkan akurasi dari sistem FCNN. Hasil prediksi dari sistem BiLSTM lebih baik dari sistem FCNN dan SLSTM. Dalam perbandingan kenaikan akurasi menggunakan sistem FCNN, sistem BiLSTM dapat meningkatkan akurasi secara signifikan dibandingkan dengan sistem SLSTM. Secara keseluruhan, sistem BiLSTM merupakan sistem prediksi benchmark dan terbaik dibandingkan sistem lainnya.</p>

## 2.2 Dasar Teori

Dasar teori berisi teori-teori yang berkaitan dan digunakan dalam melakukan pengolahan data maupun melakukan analisis.

### 2.2.1 Meteorologi

Meteorologi adalah cabang ilmu yang mempelajari atmosfer, termasuk fenomena cuaca dan iklim. Studi ini mencakup analisis parameter seperti suhu, kelembapan, tekanan udara, serta kecepatan dan arah angin. Ilmu meteorologi memiliki peran penting dalam memahami perubahan cuaca harian, memprediksi fenomena ekstrem, dan memantau dampaknya terhadap aktivitas manusia. Di Indonesia, yang berada di wilayah tropis, meteorologi sangat relevan karena

pengaruh besar dari siklus musiman seperti monsun dan fenomena global seperti El Niño dan La Niña. Selain itu, meteorologi juga berperan dalam manajemen sumber daya alam dan mitigasi bencana. Pemahaman terhadap pola curah hujan, misalnya, membantu perencanaan sektor pertanian dan pengelolaan sumber daya air. Dengan alat dan teknologi seperti satelit dan model prediksi cuaca, ilmuwan meteorologi dapat memberikan informasi akurat untuk membantu masyarakat mempersiapkan diri terhadap perubahan cuaca yang dinamis (Mujiasih, 2013). Aplikasi teknologi ini mencakup pengembangan sistem prakiraan cuaca maritim yang mengintegrasikan data arah angin, tinggi gelombang, dan kondisi atmosfer

### **2.2.2 Prediksi**

Prediksi adalah proses untuk memperkirakan nilai atau kejadian di masa depan berdasarkan pola dari data historis. Menurut kamus besar Bahasa Indonesia, prediksi merupakan hasil dari kegiatan memprediksi atau meramal atau memperkirakan nilai pada masa yang akan datang dengan menggunakan data di masa lalu. Teknik ini memiliki peran penting dalam berbagai bidang seperti keuangan, kesehatan, pemasaran, hingga pengelolaan sumber daya alam. Metode yang digunakan dalam prediksi dapat berupa pendekatan statistik seperti regresi linier dan analisis deret waktu, maupun pendekatan berbasis pembelajaran mesin yang lebih kompleks seperti neural network. Tujuan utama dari prediksi adalah memberikan gambaran yang lebih jelas dan terukur untuk mendukung pengambilan keputusan yang lebih baik.

Prediksi bisa berdasarkan metode ilmiah ataupun subjektif belaka. Sebagai contoh, prediksi cuaca selalu berdasarkan data dan informasi terbaru yang didasarkan pengamatan termasuk oleh satelit. Begitupun prediksi gempa, gunung meletus ataupun bencana secara umum. Namun, prediksi seperti pertandingan sepakbola, olahraga, dan lain-lain umumnya berdasarkan pandangan subjektif dengan sudut pandang sendiri yang memprediksinya. Pada awalnya, pengkajian mengenai alternatif masa depan adalah suatu disiplin baru. Perhatian tentang masa depan ini terus berkembang hingga bidang astrologi pada tahun 1973. Peramalan (*forecasting*) merupakan suatu prosedur untuk membuat informasi faktual mengenai situasi sosial masa depan atas dasar informasi yang telah ada tentang masalah kebijakan. Ramalan mempunyai tiga bentuk utama: proyeksi, prediksi, dan perkiraan.

1. Suatu proyeksi adalah ramalan yang didasarkan pada ekstrapolasi atas kecenderungan masa lalu maupun masa kini ke masa depan. Proyeksi membuat pertanyaan yang tegas berdasarkan argumen yang diperoleh dari metode tertentu dan kasus yang paralel.
2. Sebuah prediksi adalah ramalan yang didasarkan pada asumsi teoritik yang tegas. Asumsi ini dapat berbentuk hukum teoretis (misalnya hukum berkurangnya nilai uang), proposisi teoritis (misalnya proposisi bahwa

pecahnya masyarakat sipil diakibatkan oleh kesenjangan antara harapan dan kemampuan), atau analogi (misalnya analogi antara pertumbuhan organisasi pemerintah dengan pertumbuhan organisme biologis).

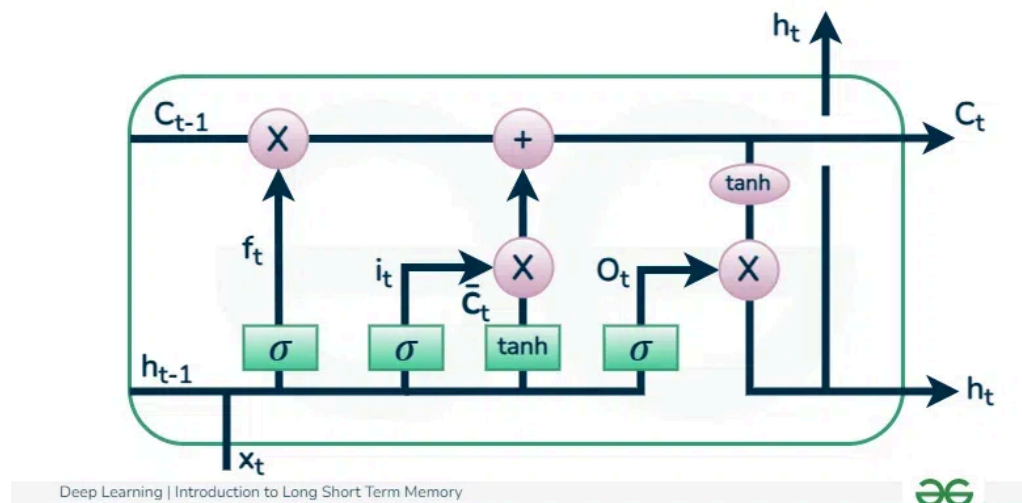
3. Suatu perkiraan (conjecture) adalah ramalan yang didasarkan pada penilaian yang informative atau penilaian pakar tentang situasi masyarakat masa depan.

Tujuan dari pada diadakannya peramalan kebijakan adalah untuk memperoleh informasi mengenai perubahan di masa yang akan datang yang akan mempengaruhi terhadap implementasi kebijakan serta konsekuensinya.

### 2.2.3 Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM) adalah varian dari Recurrent Neural Network (RNN) yang dirancang khusus untuk mengatasi kelemahan utama yang dimiliki RNN standar, yaitu kesulitan dalam mempelajari ketergantungan jangka panjang. Pada RNN konvensional, masalah seperti vanishing gradient dapat terjadi selama pelatihan, yang menyebabkan model kesulitan untuk "mengingat" informasi dari urutan yang jauh di masa lalu. LSTM diperkenalkan untuk mengatasi masalah ini dan berhasil menjaga serta memanipulasi informasi secara lebih efektif, bahkan untuk jangka panjang dalam data urutan.

LSTM mampu bekerja dengan baik dalam aplikasi-aplikasi yang memerlukan pemahaman konteks sekuensial yang kompleks, seperti pemrosesan bahasa alami (NLP), pengenalan suara, analisis sentimen, serta time-series forecasting. Keunggulan utamanya adalah pada kemampuannya untuk menyimpan informasi dalam jangka waktu yang panjang sambil secara dinamis menentukan kapan harus melupakan atau mengingat informasi tertentu.



Gambar 2.1 Arsitektur LSTM

Arsitektur Long Short-Term Memory (LSTM) dirancang untuk menangani masalah utama yang ada pada jaringan saraf berulang standar (Recurrent Neural

Network, RNN), yaitu kesulitan dalam mengingat hubungan jangka panjang pada data sekuensial. LSTM memiliki struktur yang lebih kompleks dibandingkan RNN biasa, dengan elemen-elemen yang memungkinkan model untuk menyimpan, melupakan, dan memproses informasi dari urutan data dengan lebih efisien.

- Cell State

Jalur utama untuk aliran informasi melalui LSTM. Cell state berfungsi sebagai memori jangka panjang yang mengalir melalui jaringan dari waktu ke waktu. Informasi di cell state dapat tetap utuh atau diubah melalui gerbang tertentu, memungkinkan jaringan untuk mengingat informasi penting yang diperlukan untuk tugas tertentu.

- Hidden State

Hidden state menyimpan informasi jangka pendek yang dibawa dari langkah waktu ke langkah waktu dalam sequence data. Pada setiap langkah waktu, hidden state diperbarui berdasarkan cell state dan input dari langkah sebelumnya.

- Gerbang LSTM

Salah satu ciri khas LSTM adalah adanya tiga gerbang utama yang berfungsi untuk mengontrol aliran informasi di dalam dan luar cell state. Ketiga gerbang ini adalah kunci dari arsitektur LSTM yang memungkinkan model untuk memutuskan apakah informasi tertentu harus disimpan, dibuang, atau digunakan pada setiap langkah waktu. Forget gate memutuskan informasi mana yang harus dibuang dari cell state. Ini dihitung menggunakan fungsi sigmoid yang mengambil hidden state dari langkah sebelumnya dan input saat ini, dengan output nilai antara 0 dan 1. Nilai 0 berarti informasi dihapus sepenuhnya, sedangkan nilai 1 berarti informasi dipertahankan.

Formula forget gate:

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

Gerbang input gate mengontrol seberapa banyak informasi baru yang akan disimpan di cell state. Sama seperti forget gate, input gate juga menggunakan fungsi sigmoid untuk menghasilkan nilai antara 0 dan 1. Fungsi ini menentukan seberapa besar bagian dari input baru yang akan ditambahkan ke cell state.

Formula input gate:

$$i_t = \sigma(w_i \cdot [h_{t-1}, x_t] + b_i)$$

Selain itu, ada lapisan tanh yang menghasilkan kandidat nilai  $\tilde{C}_t$  baru untuk ditambahkan ke cell state.

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



Output gate menentukan informasi apa yang akan diambil dari cell state dan diteruskan ke hidden state. Nilai dari output gate dikalikan dengan cell state yang telah diubah menggunakan fungsi tanh, menghasilkan hidden state baru.

Formula output gate:

$$o_t = \sigma(w_o \cdot [h_{t-1}, x_t] + b_o)$$

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

- **Pembaruan Cell State**

Cell state diperbarui pada setiap langkah waktu dengan mengkombinasikan informasi yang dipertahankan oleh forget gate dan informasi baru yang dihasilkan oleh input gate.

Formula pembaruan cell state:

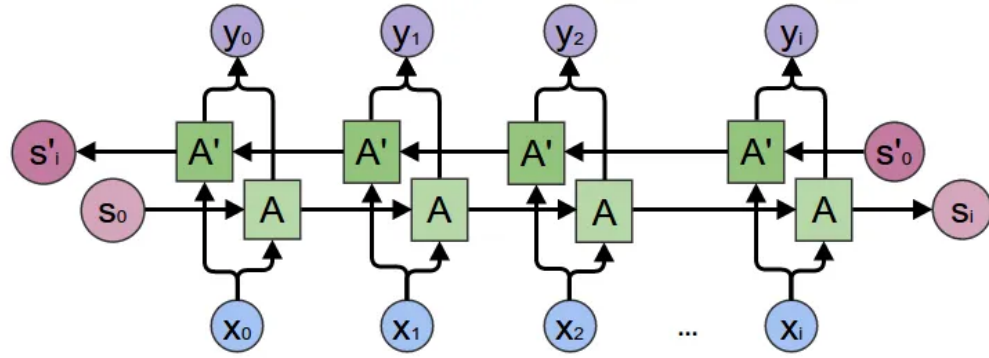
$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t$$

Di sini,  $C_{t-1}$  adalah cell state sebelumnya, dan  $\tilde{C}_t$  adalah kandidat informasi baru yang dihasilkan oleh input gate dan lapisan tanh.

#### 2.2.4 Bidirectional Long Short-Term Memory (Bi-LSTM)

BILSTM adalah salah satu jenis dari LSTM yang versi lanjutan dengan mempertimbangkan data forward dan backward untuk meningkatkan akurasi dari hasil prediksi. Pada BILSTM terdapat dua jaringan LSTM dengan jaringan LSTM pertama memproses urutan masukan data ke arah depan (forward) dan jaringan LSTM kedua memproses urutan data dalam arah sebaliknya (backward). Output dari jaringan LSTM pertama dan kedua digabungkan pada setiap urutan waktu.

Konfigurasi dari sistem BILSTM terdiri dari sebuah layer input, layer BILSTM, sebuah layer FC, dan sebuah layer output yang ditunjukkan dalam gambar. Panjang urutan waktu pada sistem BILSTM adalah 144 (satu hari data) dan sama dengan dimensi input pada sistem FCNN dan LSTM. Panjang urutan waktu yang terlalu panjang (144) pada sistem BILSTM mengakibatkan adanya kenaikan error. Setiap layer BILSTM menggunakan 128 neuron. Pada akhir proses BILSTM terdapat FC layer yang berfungsi untuk memudahkan dalam memperoleh output.



Gambar 2.2 Arsitektur LSTM Dua Arah

Arsitektur LSTM dua arah terdiri dari dua LSTM searah yang memproses urutan dalam arah maju dan mundur. Arsitektur ini dapat diartikan memiliki dua jaringan LSTM terpisah, satu mendapatkan urutan token sebagaimana adanya sementara yang lain mendapatkan urutan terbalik. Kedua jaringan LSTM ini mengembalikan vektor probabilitas sebagai output dan output akhir adalah kombinasi dari kedua probabilitas ini. Output ini dapat direpresentasikan sebagai:

$$p_t = p_t^f + p_t^b$$

Dimana,

$p_t$ : Vektor probabilitas akhir jaringan

$p_t^f$ : Vektor probabilitas dari jaringan LSTM maju

$p_t^b$ : Vektor probabilitas dari jaringan LSTM mundur.

Arsitektur BiLSTM terdiri dari dua bagian utama: LSTM maju dan LSTM mundur. Setiap LSTM memiliki unit memori yang disebut "cell", yang memungkinkan jaringan untuk mengingat informasi dalam jangka panjang. Berikut adalah langkah-langkah umum dalam arsitektur BiLSTM :

- Input Layer, Langkah pertama dalam arsitektur BiLSTM adalah mengambil urutan data sebagai input, seperti teks atau suara. Setiap elemen dalam urutan data direpresentasikan sebagai vektor fitur.
- LSTM Maju, LSTM maju digunakan untuk memproses urutan data dari awal hingga akhir. Setiap elemen urutan data dimasukkan ke dalam LSTM maju, dan informasi di dalamnya diproses dan disimpan dalam unit memori LSTM.
- LSTM Mundur, LSTM mundur memiliki struktur yang sama dengan LSTM maju, tetapi memproses urutan data dari akhir hingga awal. Hal ini memungkinkan jaringan untuk mengakses informasi masa depan dalam urutan data.

- d. Output Layer, Setelah melalui LSTM maju dan LSTM mundur, output dari kedua bagian tersebut digabungkan untuk menghasilkan representasi fitur yang kaya dari urutan data. Representasi ini dapat digunakan untuk tugas seperti klasifikasi, pemodelan bahasa, atau pemrosesan teks.

### 2.2.5 Metrik Evaluasi

Metrik evaluasi merupakan alat ukur yang digunakan untuk menilai seberapa baik sebuah model dalam membuat prediksi. Salah satu metode evaluasi yang digunakan dalam analisis ini melibatkan penghitungan *Mean Absolute Error* (MAE), *Mean Square Error* (MSE), dan *Root Mean Square Error* (RMSE). Mean Absolute Error (MAE) menghitung rata-rata nilai absolut dari perbedaan antara data observasi aktual dengan hasil prediksi. MAE memberikan informasi mengenai seberapa besar rata-rata kesalahan prediksi model dibandingkan dengan nilai aktual tanpa memperhatikan arah kesalahan tersebut. Persamaan MAE dapat dituliskan sebagai berikut:

$$MAE = \frac{1}{n} \sum_{i=1}^n \left| y_i - \hat{y}_i \right|$$

Keterangan:

MAE = nilai *mean absolute error*

$y_i$  = nilai observasi

$\hat{y}_i$  = nilai hasil prediksi

i = urutan data pada data

n = jumlah data

Mean Squared Error (MSE) menghitung rata-rata kuadrat dari selisih antara nilai observasi aktual dan hasil prediksi. MSE memberikan bobot lebih pada kesalahan yang besar karena penggunaan kuadrat dari perbedaan tersebut. Persamaan MSE dapat dituliskan sebagai berikut:

$$MSE = \frac{1}{n} \sum_{i=1}^n \left( y_i - \hat{y}_i \right)^2$$

Keterangan:

MSE = nilai *root mean square error*

$y_i$  = nilai observasi

$\hat{y}_i$  = nilai hasil prediksi

i = urutan data pada data  
n = jumlah data

Root Mean Squared Error (RMSE) adalah metode alternatif yang digunakan untuk menilai tingkat akurasi teknik peramalan yang diterapkan. RMSE mengukur rata-rata diferensiasi absolut antara nilai prediksi dan nilai aktual, yang dinyatakan dalam bentuk persentase. Dengan menggunakan RMSE, dapat dievaluasi seberapa efektif model yang telah dioptimalkan dalam memprediksi suhu berdasarkan data meteorologi yang tersedia. Nilai RMSE dapat dihitung menggunakan persamaan berikut:

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

Keterangan:

RMSE = nilai *root mean square error*

$y_i$  = nilai observasi

$\hat{y}_i$  = nilai hasil prediksi

i = urutan data pada data

n = jumlah data

## BAB 3

### METODOLOGI PENELITIAN

#### 3.1 Metodologi Penelitian

Dalam penelitian ini, alur penelitian yang dilakukan di antaranya adalah akuisisi data, *preprocessing data*, pemodelan, dan evaluasi. Tahapan pertama adalah *data acquisition*, yaitu pengumpulan data suhu yang dilakukan melalui website Data Online - Pusat Database BMKG. Setelah data diperoleh, dilakukan proses *data preprocessing*, yang dimulai dari *data cleaning* hingga normalisasi data untuk menyamakan skala nilai pada dataset, serta menentukan jumlah lag untuk membangun hubungan antar *time series* data. Tahapan ini juga memisahkan dataset menjadi data training dan data testing. Pada tahap *modelling*, data yang telah diproses disiapkan untuk dimasukkan ke dalam model arsitektur Long Short-Term Memory (LSTM) dan Bidirectional LSTM (Bi-LSTM) di mana desain dan pelatihan model dilakukan. Selanjutnya, prediksi dilakukan pada data testing menggunakan model yang telah dilatih, dan hasil prediksi yang telah dinormalisasi diubah kembali ke skala aslinya melalui inverse transform agar dapat diinterpretasikan sesuai dengan data aktual. Pada tahap *evaluation*, dilakukan penghitungan metrik evaluasi seperti *Root Mean Square Error* (RMSE), *Mean Square Error* (MSE), dan *Mean Absolute Error* (MAE) untuk menilai performa model. Diagram alir penelitian ditampilkan pada Gambar 3.1.



Gambar 3.1 *Flowchart* Metodologi Penelitian

##### 3.1.1 Akuisisi Data

Pada tahap akuisisi data dalam penelitian ini, data yang digunakan berasal dari Stasiun Meteorologi BMKG Juanda yang berlokasi di Surabaya, Jawa Timur. Data yang dikumpulkan berupa data angin harian selama hampir 25 tahun, yaitu dari Januari 2000 hingga November 2024. Adapun parameter angin harian yang dicatat dan direkam oleh stasiun meteorologi tersebut meliputi kecepatan angin maksimum (*ff\_x*) dan arah angin terbanyak (*ddd\_car*).

Kecepatan angin maksimum (*ff\_x*) adalah nilai tertinggi dari kecepatan angin yang tercatat selama satu hari penuh, mulai dari pukul 00.00 hingga 23.59 WIB. Parameter ini merepresentasikan intensitas maksimum pergerakan angin yang terjadi pada hari tersebut dan memberikan informasi penting mengenai potensi dampak angin terhadap lingkungan maupun aktivitas manusia, seperti transportasi udara atau kelautan. Data ini diukur dalam satuan meter per detik (m/s) sesuai standar pengukuran meteorologi.

Sementara itu, arah angin terbanyak (*ddd\_car*) menunjukkan arah dominan angin yang tercatat berdasarkan frekuensi kejadian selama periode pengamatan

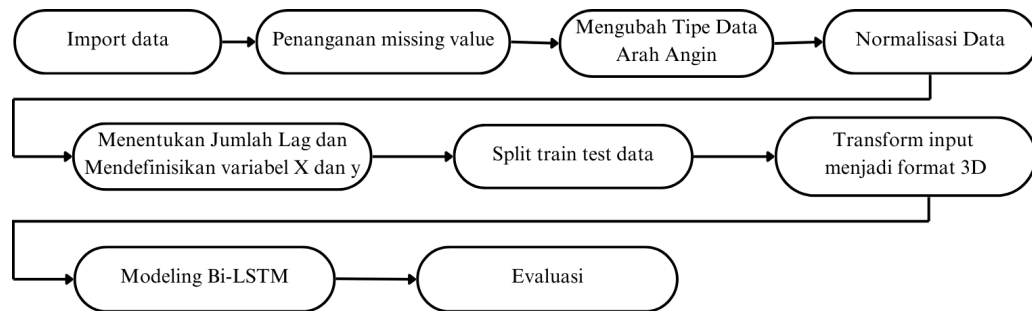
harian. Arah ini dinyatakan dalam format mata angin Utara (N), Selatan (S), Timur (E), Barat (W) atau kombinasi Timur Laut (NE) dan Barat Daya (SW). Parameter ini memberikan gambaran pola utama pergerakan angin di wilayah tertentu, yang dapat digunakan untuk analisis tren angin, potensi angin muson, atau faktor lingkungan lainnya. Data yang diakuisisi berformat time series sebagaimana yang ditampilkan pada Tabel 3.1.

Tabel 3.1 Dataset Angin

<b>Tanggal</b>	<b>ff_x</b>	<b>ddd_car</b>
<b>01/01/2000</b>	8	N
<b>02/01/2000</b>	8	W
<b>03/01/2000</b>	5	W
<b>04/01/2000</b>	4	E
.....	.....	.....
.....	.....	.....
<b>13/11/2024</b>	4	E
<b>14/11/2024</b>	5	E
<b>15/11/2024</b>	7	E

### 3.1.2 Data Preprocessing

Data preprocessing merupakan tahapan awal yang penting dalam proses pengolahan data. Setelah data diperoleh, tahap selanjutnya adalah melakukan preprocessing terhadap data tersebut sebelum dimasukkan ke dalam model algoritma pemodelan. Preprocessing data bertujuan untuk memastikan bahwa data yang akan digunakan dalam analisis selanjutnya telah memenuhi syarat dan siap untuk diproses. Beberapa hal utama yang dilakukan pada tahap preprocessing antara lain: pemeriksaan kelengkapan data, penanganan data yang hilang (missing values), pembersihan dan transformasi data, serta normalisasi data. Alur dari tahap preprocessing data ini dapat digambarkan melalui bagan seperti yang ditunjukkan pada Gambar 3.2. Bagan tersebut memberikan gambaran visual tentang langkah-langkah yang dilakukan dalam mempersiapkan data sebelum masuk ke tahap pemodelan.



Gambar 3.2 Flowchart Preprocessing Data

### 3.1.2.1 Import Data

Pada tahap awal pengolahan data, langkah pertama adalah mengimpor library yang diperlukan untuk mempermudah manipulasi, analisis, dan visualisasi data. Library yang digunakan mencakup pandas untuk manipulasi data berbasis DataFrame, numpy untuk komputasi numerik, dan matplotlib.pyplot untuk visualisasi data. Library ini merupakan komponen kunci dalam proses analisis data di Python, memberikan fleksibilitas dan efisiensi dalam menangani berbagai jenis dataset.

Setelah mengimpor library, langkah selanjutnya adalah menetapkan sumber dataset dalam sebuah variabel yang diberi nama link. Variabel ini menyimpan URL dari dataset yang akan digunakan yang berada pada Google Drive. Dataset ini kemudian dibaca menggunakan fungsi `pd.read_csv()` dari library pandas, dengan menentukan separator data berupa titik koma (;) agar sesuai dengan format file. Dataset yang berhasil dibaca disimpan dalam sebuah DataFrame bernama df. DataFrame ini adalah struktur data dua dimensi yang mempermudah analisis data lebih lanjut. Langkah terakhir adalah menampilkan isi DataFrame menggunakan fungsi `print(df)` untuk memastikan data telah berhasil diimpor dan siap diolah. Dengan demikian, data siap untuk digunakan dalam proses analisis lebih lanjut.

1. Impor library: pandas, numpy, matplotlib.pyplot.
2. Tentukan URL dataset di variabel 'link'.
3. Baca dataset dari URL menggunakan pandas dengan separator ';', simpan ke 'df'.
4. Tampilkan isi 'df'.

Hasil dari langkah impor ditampilkan pada Gambar 3.3 di bawah ini. Gambar tersebut menunjukkan sejumlah 9086 baris data yang merupakan hasil rekapitulasi data kecepatan dan arah angin harian dari tahun 2000 hingga 2024.

	Tanggal	ff_x	ddd_car
0	2000-01-01	8.0	N
1	2000-01-02	8.0	W
2	2000-01-03	5.0	W
3	2000-01-04	4.0	E
4	2000-01-05	4.0	S
...	...	...	...
9081	2024-11-11	5.0	E
9082	2024-11-12	5.0	NE
9083	2024-11-13	4.0	E
9084	2024-11-14	5.0	E
9085	2024-11-15	7.0	E

9086 rows x 3 columns

Gambar 3.3 Hasil import data pada Python

### 3.1.2.2 Penanganan Missing Value

Langkah awal dalam penanganan data adalah melakukan pengecekan keberadaan missing value untuk setiap kolom pada dataset. Proses ini dilakukan dengan melibatkan iterasi atau loop pada setiap kolom di DataFrame. Dalam setiap iterasi, jumlah nilai yang hilang dihitung dan disimpan dalam variabel `missing_count`. Jika jumlah nilai yang hilang pada suatu kolom lebih besar dari nol, maka nama kolom tersebut beserta jumlah nilai yang hilang akan dicetak. Tahap ini bertujuan untuk mengidentifikasi secara spesifik kolom mana saja yang memiliki data yang tidak lengkap sehingga dapat dilakukan penanganan yang sesuai di langkah selanjutnya. Pengecekan ini menjadi dasar penting untuk menjaga kualitas dataset sebelum analisis atau pemodelan dilakukan.

1. Loop tiap kolom di `df`.
2. Hitung nilai hilang, simpan di `missing_count`.
3. Jika `missing_count > 0`, cetak nama kolom dan jumlahnya.

Missing value yang dihasilkan dari pengecekan awal:

```
Column 'ff_x' has 8 missing values.
Column 'ddd_car' has 30 missing values.
```

Langkah kedua adalah menambahkan kolom baru bernama 'Bulan' yang dihasilkan dari ekstraksi informasi bulan pada kolom 'Tanggal'. Hal



ini bertujuan untuk mempermudah analisis musiman. Selanjutnya, membuat kolom 'Musim' dengan menentukan musim berdasarkan nilai di kolom 'Bulan'. Penentuan ini dilakukan menggunakan fungsi khusus yang memetakan setiap bulan ke musim yang relevan. Data kemudian dikelompokkan berdasarkan 'Musim' untuk menghitung statistik deskriptif seperti rata-rata pada kolom 'ff\_x' dan jumlah data pada kolom 'ddd\_car'. Proses ini bertujuan untuk menggambarkan karakteristik data per musim. Selanjutnya, baris yang memiliki missing value diidentifikasi dan dikelompokkan berdasarkan musim, lalu dicetak untuk memahami pola atau sebarannya. Untuk menangani missing value, digunakan metode pengisian berdasarkan karakteristik data: nilai rata-rata digunakan untuk mengisi kolom 'ff\_x', sementara nilai modus digunakan untuk kolom 'ddd\_car'. Setelah semua missing value diisi, indeks data direset untuk memastikan struktur data tetap teratur dan rapi.

Mengisi missing value dengan statistik berdasarkan kelompok bulan DJF, MAM, JJA, SON:

1. Tambah kolom 'Bulan' dari 'Tanggal'.
2. Tentukan 'Musim' dari 'Bulan' menggunakan fungsi.
3. Kelompokkan data per 'Musim', hitung statistik ('ff\_x') dan jumlah ('ddd\_car').
4. Cari baris hilang, kelompokkan, dan cetak per 'Musim'.
5. Hitung rata-rata ('ff\_x') dan modus ('ddd\_car') per 'Musim'.
6. Isi nilai hilang: rata-rata untuk 'ff\_x', modus untuk 'ddd\_car', reset indeks.

Hasil pengisian missing value:

	Tanggal	ff_x	ddd_car	Bulan	Musim
0	2000-01-01	8.0	N	1	DJF
1	2000-01-02	8.0	W	1	DJF
2	2000-01-03	5.0	W	1	DJF
3	2000-01-04	4.0	E	1	DJF
4	2000-01-05	4.0	S	1	DJF
...	...	...	...	...	...
9081	2024-11-11	5.0	E	11	SON
9082	2024-11-12	5.0	NE	11	SON
9083	2024-11-13	4.0	E	11	SON
9084	2024-11-14	5.0	E	11	SON
9085	2024-11-15	7.0	E	11	SON

9086 rows x 6 columns

Gambar 3.4 Hasil Pengisian Missing Value

### 3.1.2.3 Mengubah Tipe Data Arah Angin

Langkah ini bertujuan untuk mengonversi data arah angin berbentuk kategori menjadi nilai numerik berupa derajat. Pertama, dibuat sebuah dictionary dengan nama 'arah\_ke\_derajat' yang memetakan setiap nilai kategori pada kolom 'ddd\_car' (misalnya, N, NE, E, dll.) ke nilai derajat tertentu sesuai dengan konvensi arah mata angin. Selanjutnya, ditambahkan kolom baru bernama 'ddd\_deg' ke DataFrame `df_filled`, di mana setiap nilai dalam kolom 'ddd\_car' dipetakan menggunakan dictionary tersebut untuk menghasilkan nilai derajat yang sesuai. Proses ini memungkinkan analisis yang lebih mendalam dan memungkinkan data arah angin digunakan dalam perhitungan berbasis numerik. Setelah proses konversi selesai, DataFrame `df_filled` yang telah diperbarui ditampilkan untuk memverifikasi hasil transformasi. Langkah ini memastikan data lebih terstruktur dan siap untuk keperluan analisis lebih lanjut.

Mengubah tipe data angin string menjadi angka numerik:

1. Buat dictionary 'arah\_ke\_derajat' untuk konversi arah angin ke derajat.
2. Tambah kolom 'ddd\_deg' di 'df\_filled' dengan memetakan 'ddd\_car' menggunakan dictionary.
3. Tampilkan hasil 'df\_filled'..

Hasil:

	Tanggal	ff_x	ddd_car	Bulan	Musim	ddd_deg
0	2000-01-01	8.0	N	1	DJF	360
1	2000-01-02	8.0	W	1	DJF	270
2	2000-01-03	5.0	W	1	DJF	270
3	2000-01-04	4.0	E	1	DJF	90
4	2000-01-05	4.0	S	1	DJF	180
...	...	...	...	...	...	...
9081	2024-11-11	5.0	E	11	SON	90
9082	2024-11-12	5.0	NE	11	SON	45
9083	2024-11-13	4.0	E	11	SON	90
9084	2024-11-14	5.0	E	11	SON	90
9085	2024-11-15	7.0	E	11	SON	90

9086 rows × 6 columns

Gambar 3.5 Hasil Mengubah Tipe Data Angin

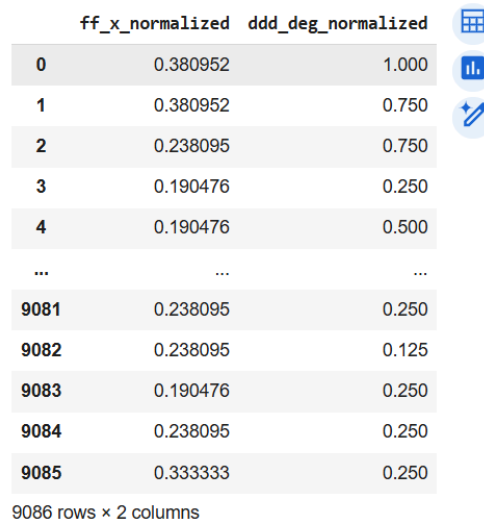
### 3.1.2.4 Normalisasi Data

Proses ini bertujuan untuk menormalisasi data numerik dalam dataset agar berada dalam rentang tertentu, yaitu 0 hingga 1, menggunakan metode *Min-Max Scaling*. Langkah pertama adalah mengimpor modul `MinMaxScaler` dari pustaka *scikit-learn*, yang menyediakan alat untuk melakukan transformasi ini. Setelah itu, scaler diinisialisasi dengan rentang nilai minimal 0 dan maksimal 1, memastikan semua nilai dalam kolom terpilih akan ditransformasikan sesuai dengan rentang tersebut.

Scaler kemudian diterapkan pada dua kolom numerik, yaitu `'ff_x'` dan `'ddd_deg'`, yang ada dalam DataFrame `df_filled`. Hasil normalisasi disimpan dalam array bernama `data_scaled`, yang memuat nilai yang telah diubah ke skala yang seragam. Data ini kemudian digunakan untuk membuat DataFrame baru bernama `df_scaled`, dengan dua kolom hasil normalisasi, yaitu `'ff_x_normalized'` dan `'ddd_deg_normalized'`. Langkah terakhir adalah menampilkan DataFrame `df_scaled` untuk memastikan bahwa proses normalisasi telah berjalan dengan benar dan menghasilkan data yang siap digunakan untuk analisis atau model prediktif yang memerlukan data berskala seragam. Berikut ini adalah algoritma dari proses normalisasi data menggunakan `MinMaxScaler`:

1. Import `MinMaxScaler` dari `sklearn`.
2. Inisialisasi scaler dengan rentang (0, 1).
3. Terapkan normalisasi pada kolom `'ff_x'` dan `'ddd_deg'` di `'df_filled'`.
4. Simpan hasil normalisasi ke `'data_scaled'`.
5. Buat DataFrame baru `'df_scaled'` dengan kolom `'ff_x_normalized'` dan `'ddd_deg_normalized'`.
6. Tampilkan `'df_scaled'`.

Hasil dari proses normalisasi adalah:



	ff_x_normalized	ddd_deg_normalized
0	0.380952	1.000
1	0.380952	0.750
2	0.238095	0.750
3	0.190476	0.250
4	0.190476	0.500
...	...	...
9081	0.238095	0.250
9082	0.238095	0.125
9083	0.190476	0.250
9084	0.238095	0.250
9085	0.333333	0.250

9086 rows x 2 columns

Gambar 3.6 Data Setelah Dilakukan Proses Normalisasi

### 3.1.2.5 Menentukan Jumlah Lag dan Mendefinisikan Variabel X dan y

Salah satu tahapan yang sangat penting di penelitian ini yaitu menentukan jumlah lag. Pada model Long Short-Term Memory (LSTM), lag merujuk pada nilai-nilai data sebelumnya dalam sebuah urutan (time series) yang digunakan sebagai input untuk memprediksi nilai saat ini atau masa depan. Lag berguna untuk menangkap pola atau korelasi temporal dalam data, terutama dalam prediksi yang berbasis waktu. Pada LSTM, model tidak secara otomatis mengetahui nilai lag yang relevan sehingga diperlukan untuk memasukan data lag sebagai fitur tambahan dalam input. Pemilihan lag yang tepat sangat penting karena lag yang terlalu sedikit dapat menyebabkan model kehilangan informasi historis yang relevan, sedangkan lag yang terlalu banyak dapat memperkenalkan noise atau informasi yang tidak signifikan. Dengan menentukan jumlah lag yang optimal, model dapat mempelajari pola jangka pendek dan jangka panjang dengan lebih efektif, sehingga meningkatkan akurasi prediksi.

Dalam kode, jumlah lag ditentukan oleh variabel `sequence_length`. Variabel `sequence_length` diatur ke 64, yang berarti 64 nilai sebelumnya akan digunakan untuk memprediksi nilai berikutnya. Setelah jumlah lag ditentukan, data diolah menjadi dua variabel utama yaitu Variabel X dan variabel Y. Variabel X berisi nilai-nilai data sebelumnya dalam bentuk sekuensial, sesuai dengan jumlah lag yang ditentukan. Variabel ini menjadi input bagi model LSTM untuk belajar pola temporal. Variabel Y berisi nilai target yang akan diprediksi, yaitu nilai setelah `sequence_length` data. Proses pembentukan X dan y dilakukan melalui fungsi `create_dataset`, yang memetakan data mentah ke dalam format sekuensial.

Dengan langkah ini, model dapat memahami hubungan temporal antara nilai-nilai sebelumnya (X) dan nilai target (y).

1. Definisikan fungsi untuk membuat dataset sequence:
  - a. Loop melalui data untuk membentuk urutan (X) dan target (y) dengan panjang urutan `'sequence_length'`.
2. Tentukan value `'sequence_length' = 64`.
3. Panggil fungsi untuk menghasilkan X dan y dari `'data_scaled'`.

Proses pembentukan X dan y dilakukan melalui fungsi `create_dataset`, yang memetakan data mentah ke dalam format sekuensial. Dengan langkah ini, model dapat memahami hubungan temporal antara nilai-nilai sebelumnya (X) dan nilai target (y).

Output `X.shape` dan `y.shape`

```
(9056, 30, 2) (9056, 2)
```

### 3.1.2.6 Pembagian Data menjadi Set Training dan Testing

Salah satu tahapan penting dalam pengembangan model prediktif, seperti model Long Short-Term Memory (LSTM) yaitu pembagian data menjadi set training dan testing. Tahapan ini bertujuan untuk memastikan bahwa model yang di buat dapat dievaluasi secara objektif pada data yang belum pernah dilihat sebelumnya, sehingga dapat menghindari masalah overfitting (model yang terlalu menyesuaikan pada data training) atau underfitting (model tidak cukup belajar dari data).

Pada data time series seperti pada model LSTM, pembagian data menjadi set training dan testing tidak dapat dilakukan secara acak bertujuan untuk menjaga urutan temporal. Data terbaru sering digunakan untuk testing, sedangkan data sebelumnya digunakan untuk training. Tahapan pembagian data menjadi training dan testing dilakukan dengan menggunakan fungsi `train_test_split` dari library `scikit-learn`.

Proses pembagian dilakukan pada variabel input X (data sequence) dan output y (target prediksi). Proporsi pembagian data ditentukan oleh parameter `test_size=0.2`, yang menunjukkan bahwa 80% data digunakan untuk training dan 20% data digunakan untuk testing. Parameter `shuffle=False` memastikan data tidak diacak, karena mempertahankan urutan temporal sangat penting dalam pemrosesan data time series seperti pada model LSTM.

1. Import `'train_test_split'`.

2. Bagi data menjadi:
  - a. X\_train dan X\_test (fitur)
  - b. y\_train dan y\_test (target)
3. Gunakan parameter `test\_size=0.2` (80% data untuk train, 20% untuk test) dan `shuffle=False`.

Setelah pembagian, data training (X\_train, y\_train) digunakan untuk melatih model, sementara data testing (X\_test, y\_test) digunakan untuk mengevaluasi performa model. Dimensi data kemudian diubah menggunakan metode reshape agar sesuai dengan format input model LSTM, yaitu [samples, timesteps, features]. Tahapan ini memastikan bahwa data tersusun dalam bentuk sekuensial yang sesuai untuk mengidentifikasi pola temporal pada time series.

X\_train.shape, X\_test.shape, y\_train.shape, y\_test.shape

(7244, 30, 2) (1812, 30, 2) (7244, 2) (1812, 2)

### 3.1.2.7 Transform Input Menjadi Format 3D

Transformasi data input menjadi format 3D adalah langkah penting dalam mempersiapkan data untuk digunakan pada model deep learning, khususnya yang dirancang untuk menangani data sequential atau time series, seperti Long Short-Term Memory (LSTM) atau model Recurrent Neural Network (RNN). Proses ini melibatkan pengubahan struktur data agar mencerminkan dimensi (samples, time\_steps, features), yang diperlukan untuk menangkap pola atau hubungan temporal antar waktu.

Dalam proses pembagian data untuk keperluan pelatihan dan pengujian model, langkah pertama yang perlu dilakukan adalah mengimpor fungsi train\_test\_split dari pustaka scikit-learn. Fungsi ini digunakan untuk membagi dataset menjadi dua subset utama: data pelatihan (training data) dan data pengujian (testing data). Selanjutnya, data yang akan digunakan dipisahkan menjadi dua komponen utama, yaitu data fitur dan data target. Data fitur, yang merepresentasikan variabel independen atau input, dibagi menjadi X\_train untuk data pelatihan dan X\_test untuk data pengujian. Sementara itu, data target, yang merepresentasikan variabel dependen atau output yang akan diprediksi, dibagi menjadi y\_train untuk pelatihan dan y\_test untuk pengujian. Dalam proses pembagian ini, parameter test\_size diatur sebesar 0.2, yang berarti

20% dari total data akan digunakan sebagai data pengujian, sedangkan 80% sisanya digunakan untuk pelatihan model.

#### Menentukan train test

1. Import `'train_test_split'`.
2. Bagi data menjadi:
  - a. `X_train` dan `X_test` (fitur)
  - b. `y_train` dan `y_test` (target)
3. Gunakan parameter `'test_size=0.2'` (80% data untuk train, 20% untuk test) dan `'shuffle=False'`.

Setelah proses pembagian, data pelatihan (`X_train`, `y_train`) digunakan untuk melatih model, sedangkan data pengujian (`X_test`, `y_test`) digunakan untuk mengevaluasi performa model. Data fitur kemudian diubah ke dalam format tiga dimensi menggunakan metode transformasi agar sesuai dengan kebutuhan input model LSTM, yaitu dengan struktur `[samples, time_steps, features]`. Pada kasus ini, data pelatihan (`X_train`) memiliki dimensi (7244, 30, 2) dan data pengujian (`X_test`) memiliki dimensi (1812, 30, 2), mencerminkan jumlah sampel, langkah waktu, dan fitur yang relevan. Proses ini memastikan data tersusun dalam bentuk sekuensial yang memungkinkan model untuk mempelajari pola temporal yang ada pada data time series secara efektif. Target (`y_train` dan `y_test`) tetap dalam format dua dimensi, sehingga memudahkan model untuk mencocokkan hasil prediksi dengan nilai aktual selama pelatihan dan evaluasi.

`X_train.shape, X_test.shape, y_train.shape, y_test.shape`

`(7244, 30, 2) (1812, 30, 2) (7244, 2) (1812, 2)`

### 3.1.3 Modeling Bi-LSTM

#### 3.1.3.1 Mendefinisikan Model LSTM

Pada tahap ini, dirancang sebuah model LSTM dasar yang menggunakan satu lapisan LSTM dan satu lapisan Dense untuk menangani tugas prediksi deret waktu. Lapisan LSTM bertanggung jawab memproses data deret waktu sebagai input dan mengekstrak fitur temporal yang relevan. Jumlah unit dalam lapisan LSTM divariasikan menjadi 50, 100, dan 150 unit melalui grid search untuk menemukan konfigurasi terbaik. Model ini menggunakan dropout rate yang bervariasi dari 0.0

hingga 0.4 untuk mencegah overfitting. Lapisan Dense digunakan sebagai lapisan output dengan satu unit untuk menghasilkan prediksi akhir berdasarkan fitur yang telah diekstraksi oleh lapisan LSTM.

Fungsi aktivasi yang digunakan pada lapisan LSTM mengikuti default TensorFlow (tanh), yang cocok untuk menangani pola non-linear pada data. Model menerima input berdimensi (time steps, features) sesuai dengan data yang diolah menggunakan lag 2. Optimizer Adam dipilih karena efisiensinya dalam pembaruan bobot secara adaptif, sementara fungsi loss mean\_squared\_error digunakan untuk meminimalkan selisih antara prediksi model dan nilai aktual. Hyperparameter tambahan seperti ukuran batch (16, 32, dan 64), jumlah epoch (50 dan 100), dan learning rate (0.001 dan 0.01) diuji untuk menentukan konfigurasi yang menghasilkan performa terbaik. Melalui evaluasi dengan data uji, model terbaik dipilih berdasarkan nilai MSE terendah.

Tabel 3.1 Kombinasi Konfigurasi Model LSTM

Layer Type	Hyperparameter	Range
LSTM	Units	50, 100, 150
	Dropout Rate	0.01
Dense	Units	Number of outputs (2)
Training	Batch Size	16, 32, 64
	Epochs	100
Optimizer	Learning Rate	0.01

### 3.1.3.2 Mendefinisikan Model Bi-LSTM

Proses yang menghasilkan pengaturan hyperparameter seperti di atas menggunakan arsitektur *Bidirectional Long Short-Term Memory* (BiLSTM) untuk pemodelan data sekuensial. Proses yang menghasilkan pengaturan hyperparameter di atas menggunakan arsitektur *Bidirectional Long Short-Term Memory* (BiLSTM), yang dirancang untuk memodelkan data sekuensial dengan mempertimbangkan konteks dua arah, baik maju (forward) maupun mundur (backward). Pada lapisan BiLSTM, jumlah unit disetel dalam rentang 50, 100, atau 150, yang menentukan dimensi ruang tersembunyi (*hidden state*) dan kemampuan model dalam menangkap pola kompleks. Setelah itu, output dari BiLSTM melewati lapisan *dropout* dengan dropout rate 0.01, yang berfungsi untuk mencegah *overfitting* dengan secara acak menghilangkan 1% koneksi antar neuron selama pelatihan. Selanjutnya, data



diteruskan ke lapisan *dense* sebagai lapisan output dengan 2 unit, sesuai jumlah keluaran model, misalnya untuk kasus klasifikasi biner.

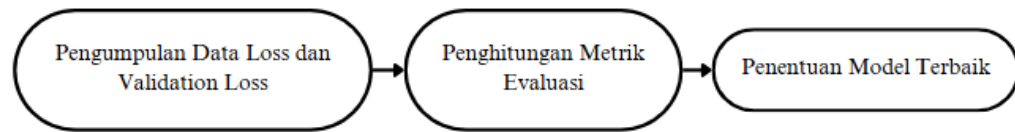
Proses pelatihan model dilakukan dengan beberapa pengaturan hyperparameter. Ukuran batch (*batch size*), yang menentukan jumlah sampel data yang diproses sebelum pembaruan bobot, diatur dalam rentang 16, 32, atau 64. Model dilatih selama 100 *epochs*, yang berarti seluruh dataset dilalui sebanyak 100 kali. Pembaruan bobot dilakukan menggunakan sebuah *optimizer* dengan learning rate 0.01, yang mengatur langkah pembaruan bobot berdasarkan gradien error. Kombinasi berbagai pengaturan hyperparameter ini diuji untuk mendapatkan performa model terbaik sesuai kebutuhan.

Proses optimasi model dilakukan menggunakan sebuah *optimizer*, yang bertugas memperbarui bobot berdasarkan gradien error. Parameter *learning rate* ditetapkan pada 0.01, yang menentukan seberapa besar langkah pembaruan bobot pada setiap iterasi. Nilai *learning rate* ini merupakan kompromi antara kecepatan konvergensi dan stabilitas pelatihan; nilai yang terlalu tinggi dapat menyebabkan model melompat-lompat di sekitar solusi optimal, sementara nilai yang terlalu rendah dapat membuat pelatihan berjalan sangat lambat. Seluruh kombinasi hyperparameter, termasuk jumlah unit di BiLSTM, ukuran batch, dan *learning rate*, diuji secara sistematis untuk menemukan konfigurasi yang menghasilkan performa model terbaik. Pendekatan ini memastikan bahwa model tidak hanya akurat dalam prediksi tetapi juga cukup generalisasi untuk data baru.

Tabel 3.2 Kombinasi Konfigurasi Model Bi-LSTM

Layer Type	Hyperparameter	Range
Bidirectional LSTM	Units	50, 100, 150
	Dropout Rate	0.01
Dense	Units	Number of outputs (2)
Training	Batch Size	16, 32, 64
	Epochs	100
Optimizer	Learning Rate	0.01

### 3.1.4 Evaluasi



Gambar 3.7 Alur Penghitungan Evaluasi Model

#### 3.1.4.1 Pengumpulan Data Loss dan Validation Loss

Langkah awal dalam penelitian ini adalah melakukan inisialisasi dua daftar kosong, yaitu `losses` dan `val_losses`, yang akan digunakan untuk menyimpan nilai *loss* dan *validation loss* dari setiap percobaan yang dilakukan. Pendekatan ini bertujuan untuk memastikan bahwa seluruh data evaluasi dari setiap percobaan dapat terkumpul secara sistematis. Setelah daftar diinisialisasi, proses berlanjut dengan iterasi melalui 10 percobaan yang tersimpan dalam direktori tertentu. Pada setiap iterasi, sistem membuat direktori percobaan dengan format `'trial_{:02d}'`, yang secara otomatis menghasilkan direktori untuk setiap percobaan dengan penomoran unik. Kemudian, jalur file untuk menyimpan data percobaan diatur dengan format `{}/trial.json`, yang menunjuk pada file data percobaan di dalam direktori tersebut.

Selanjutnya, sistem memeriksa apakah file `trial.json` sudah ada di direktori yang dibuat. Jika file tersebut ditemukan, data percobaan yang tersimpan di dalamnya dibuka dan dibaca. Proses pembacaan ini bertujuan untuk mengekstraksi nilai *loss* dan *validation loss* yang dihasilkan oleh model dalam percobaan tersebut. Setelah diekstraksi, nilai *loss* ditambahkan ke dalam daftar `losses`, sedangkan nilai *validation loss* dimasukkan ke dalam daftar `val_losses`. Kedua daftar ini akan terus diperbarui pada setiap iterasi hingga seluruh 10 percobaan selesai diproses.

Setelah proses iterasi selesai, daftar `losses` dan `val_losses` telah terisi dengan data yang diperlukan dari seluruh percobaan. Data ini menjadi dasar untuk analisis lebih lanjut, seperti perhitungan metrik evaluasi, perbandingan kinerja model, atau proses optimasi model. Dengan pendekatan ini, peneliti dapat memastikan bahwa setiap nilai *loss* dan *validation loss* dari masing-masing percobaan terdokumentasi secara terstruktur, sehingga mendukung validitas dan keandalan hasil penelitian.

#### 3.1.4.2 Penghitungan Metrik Evaluasi

Langkah pertama dimulai dengan menginisialisasi dua daftar kosong, yaitu `losses` dan `val_losses`, untuk menyimpan nilai loss dan validation loss dari setiap percobaan. Selanjutnya, dilakukan iterasi melalui 10 percobaan yang terdapat dalam direktori tertentu. Pada setiap iterasi, nama direktori dibuat menggunakan format `'trial_{:02d}'`. format(trial), dan jalur file untuk data percobaan dihasilkan dengan format `'trial_dir/trial.json'`. Sistem kemudian memeriksa keberadaan file `trial.json`. Jika file ditemukan, data di dalamnya dibuka dan dibaca untuk mengekstrak nilai loss dan validation loss. Nilai-nilai tersebut kemudian ditambahkan ke dalam daftar `losses` dan `val_losses`. Setelah iterasi selesai, kedua daftar ini akan berisi data loss dan validation loss dari semua percobaan yang tersedia.

#### 3.1.4.3 Penentuan Model Terbaik

Penentuan model terbaik dalam kasus ini didasarkan pada perbandingan nilai RMSE (Root Mean Squared Error) dari berbagai percobaan. RMSE adalah salah satu metrik evaluasi utama dalam regresi, di mana nilai yang lebih rendah mengindikasikan model yang lebih akurat. Model digunakan untuk memprediksi suhu harian di Kota Surabaya berdasarkan data yang tersedia.

Mean Absolute Error (MAE) merupakan metrik evaluasi yang menghitung rata-rata selisih absolut antara nilai prediksi dan nilai aktual. Metrik ini memberikan gambaran mengenai tingkat kesalahan prediksi model secara keseluruhan tanpa memperhatikan arah kesalahan (positif atau negatif). Nilai MAE yang lebih kecil menunjukkan prediksi yang lebih baik. Perhitungan MAE dilakukan menggunakan fungsi `calculate_mae`, yang menerima dua parameter: `y_true` (nilai aktual) dan `y_pred` (nilai prediksi), lalu menghitung rata-rata selisih absolut di antara keduanya.

Mean Squared Error (MSE) adalah metrik evaluasi yang menghitung rata-rata kuadrat dari selisih antara nilai prediksi dan nilai aktual. Karena kuadrat kesalahan dihitung, MSE memberikan bobot lebih besar pada kesalahan yang signifikan. Nilai MSE yang lebih kecil menunjukkan model dengan kesalahan prediksi yang lebih kecil dan konsisten. Penghitungan MSE dilakukan melalui fungsi `calculate_mse`, yang menerima parameter `y_true` dan `y_pred`, kemudian mengembalikan rata-rata kuadrat dari selisih nilai tersebut.

Root Mean Squared Error (RMSE) dihitung menggunakan fungsi `calculate_rmse`, yang pertama-tama menghitung nilai MSE, lalu

mengambil akar kuadratnya untuk menghasilkan RMSE. Fungsi ini menerima dua parameter, yaitu `y_true` dan `y_pred`. RMSE digunakan untuk mengevaluasi seberapa baik model memprediksi suhu harian. Setelah model dilatih, prediksi dilakukan pada data uji (`test_X`), dan hasil prediksi disimpan dalam variabel `predictions`. Untuk mengevaluasi kinerja model, prediksi dibandingkan dengan nilai aktual pada data uji (`test_y`). Evaluasi dilakukan secara terpisah untuk masing-masing fitur target: suhu maksimum (`max_temp`), suhu minimum (`min_temp`), dan suhu rata-rata (`avg_temp`). Dengan menghitung MAE, MSE, dan RMSE untuk setiap fitur target, kita dapat menilai seberapa akurat model dalam memprediksi berbagai aspek suhu harian. Semakin rendah nilai MAE, MSE, dan RMSE, semakin baik performa model dalam memprediksi fitur terkait

## BAB 4

### HASIL DAN ANALISIS

#### 4.1 Hasil Pemodelan

Pada penelitian ini, pengujian dilakukan menggunakan pendekatan eksperimental yang difokuskan pada evaluasi model LSTM (*Long Short-Term Memory*) dengan variasi konfigurasi parameter sebagaimana yang telah dirancang pada bab sebelumnya. Data yang digunakan adalah data yang telah dilakukan preprocessing sebelumnya sebagaimana pada Gambar 4.1.

Train X:			Train Y:		
	ff_x (t-1)	ddd_deg (t-1)		ff_x (t)	ddd_deg (t)
0	0.380952	1.000	0	0.380952	0.750
1	0.380952	0.750	1	0.238095	0.750
2	0.238095	0.750	2	0.190476	0.250
3	0.190476	0.250	3	0.190476	0.500
4	0.190476	0.500	4	0.285714	0.750
...	...	...	...	...	...
7263	0.380952	0.250	7263	0.333333	0.375
7264	0.333333	0.375	7264	0.380952	0.250
7265	0.380952	0.250	7265	0.333333	0.250
7266	0.333333	0.250	7266	0.380952	0.250
7267	0.380952	0.250	7267	0.380952	0.250
[7268 rows x 2 columns]			[7268 rows x 2 columns]		

Test X:			Test Y:		
	ff_x (t-1)	ddd_deg (t-1)		ff_x (t)	ddd_deg (t)
0	0.380952	0.250	0	0.380952	0.250
1	0.380952	0.250	1	0.380952	0.250
2	0.380952	0.250	2	0.428571	0.250
3	0.428571	0.250	3	0.380952	1.000
4	0.380952	1.000	4	0.619048	0.875
...	...	...	...	...	...
1812	0.238095	0.250	1812	0.238095	0.250
1813	0.238095	0.250	1813	0.238095	0.125
1814	0.238095	0.125	1814	0.190476	0.250
1815	0.190476	0.250	1815	0.238095	0.250
1816	0.238095	0.250	1816	0.333333	0.250
[1817 rows x 2 columns]			[1817 rows x 2 columns]		

Gambar 4.1 Data train\_X, train\_y, test\_X, test\_y

#### 4.1.1 Arsitektur LSTM

Long Short-Term Memory (LSTM) adalah varian dari Recurrent Neural Network (RNN) yang dirancang khusus untuk mengatasi kelemahan utama yang dimiliki RNN standar, yaitu kesulitan dalam mempelajari ketergantungan jangka panjang. Pada RNN konvensional, masalah seperti vanishing gradient dapat terjadi selama pelatihan, yang menyebabkan model kesulitan untuk “mengingat” informasi dari urutan yang jauh di masa lalu. LSTM diperkenalkan untuk mengatasi masalah ini dan berhasil menjaga serta memanipulasi informasi secara lebih efektif, bahkan untuk jangka panjang dalam data sequence.

Arsitektur LSTM yang digunakan dalam penelitian ini dirancang untuk memprediksi dua fitur utama, yaitu `ff_x` dan `ddd_deg`, dengan memanfaatkan kemampuan Long Short-Term Memory (LSTM) dalam menangani data sekuensial. Model dibangun menggunakan kerangka Sequential, di mana data input memiliki dimensi (1, 2), yang menunjukkan bahwa setiap sampel terdiri dari satu time step dengan dua fitur input. Arsitektur ini dimulai dengan lapisan LSTM sebagai komponen utama, yang diuji dengan berbagai konfigurasi jumlah unit, yaitu 50, 100, dan 150. Tujuan dari variasi ini adalah untuk menemukan konfigurasi jumlah unit yang paling optimal dalam mempelajari pola temporal pada fitur input. Setelah lapisan LSTM, diterapkan lapisan Dropout dengan tingkat dropout sebesar 0.01 untuk mengurangi risiko overfitting selama proses pelatihan. Pada bagian output, model menggunakan lapisan Dense dengan dua unit yang dirancang untuk secara langsung memprediksi kedua fitur target, `ff_x` dan `ddd_deg`.

Proses pelatihan model menggunakan optimizer Adam dengan learning rate sebesar 0.01, yang dikenal andal dalam menangani pelatihan model dengan kompleksitas tinggi. Fungsi kerugian yang digunakan adalah Mean Squared Error (MSE), yang sesuai untuk tugas regresi dalam memprediksi nilai numerik. Selain itu, model diuji dengan kombinasi berbagai hyperparameter, meliputi jumlah unit LSTM (50, 100, dan 150), ukuran batch (16, 32, dan 64), serta jumlah epoch sebanyak 100. Dengan pendekatan ini, model diharapkan mampu menangkap pola kompleks pada data sekuensial untuk menghasilkan prediksi yang akurat terhadap fitur `ff_x` dan `ddd_deg`.

Tabel 4.1 Arsitektur Model LSTM

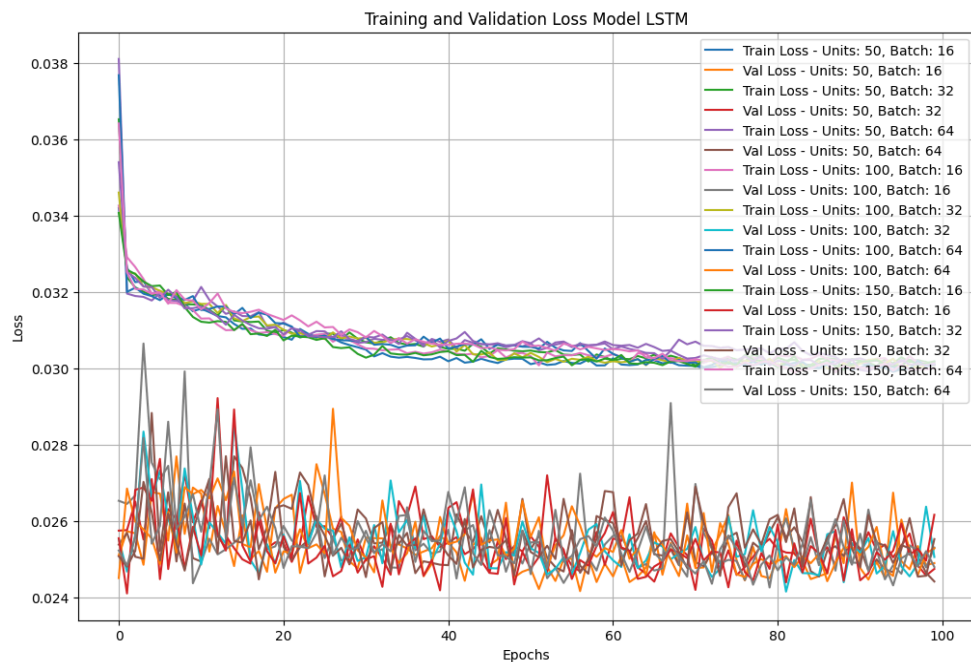
Lapisan	Tipe	Jumlah Unit/Parameter	Keterangan
1	LSTM	50, 100, 150	Menerima input sekuensial, units disesuaikan saat tuning.
2	Dropout	0.01	Mengurangi overfitting dengan menjatuhkan neuron.

3	Dense	2	Lapisan output untuk memprediksi 2 fitur.
---	-------	---	---

Tabel 4.2 Tabel Hyperparameter Tuning Model LSTM

Lapisan	Jumlah Unit/Parameter	Keterangan
LSTM Units	50, 100, 150	Mengontrol kapasitas representasi model.
Dropout Rate	0.01	Probabilitas untuk menjatuhkan neuron selama pelatihan.
Batch Size	16, 32, 64	Ukuran batch data selama pelatihan.
Epochs	100	Jumlah iterasi pelatihan penuh pada data.
Learning Rate	0.01	Kecepatan pembaruan parameter dalam optimizer Adam.

Berdasarkan hasil pengujian Arsitektur LSTM yang menggunakan tuner pada Tabel 4.2, dapat disimpulkan bahwa konfigurasi model dengan parameter 'units' sebanyak 50, 'dropout' sebesar 0.01, dan 'batch\_size' 64 menunjukkan performa terbaik. Hal ini terbukti dari berbagai metrik performa yang dihasilkan oleh model tersebut, di mana kombinasi parameter ini mampu memberikan hasil yang optimal dalam memprediksi fitur yang diuji. Visualisasi pengujian ditampilkan dalam bentuk grafik Gambar 4.2 sebagai berikut.



Gambar 4.2 Grafik Training dan Validation Loss Model LSTM

Tabel 4.3 Hasil Training dan Validation Loss Parameter Model LSTM

index	Units	Dropout	Batch Size	Train Loss	Validation Loss
0	50	0.01	16	0.030085	0.024898
1	50	0.01	32	0.030105	0.024747
2	50	0.01	64	0.030173	0.024418
3	100	0.01	16	0.030115	0.025306
4	100	0.01	32	0.030079	0.025059
5	100	0.01	64	0.03009	0.025285
6	150	0.01	16	0.030179	0.026164
7	150	0.01	32	0.030102	0.025527
8	150	0.01	64	0.029998	0.024798

#### 4.1.2 Arsitektur Bi-LSTM

Arsitektur Bi-LSTM merupakan model kedua yang digunakan dalam penelitian ini, dirancang untuk memanfaatkan keunggulan dari Bidirectional Long Short-Term Memory (Bi-LSTM) dalam memahami hubungan temporal pada data sekuensial, khususnya untuk memprediksi dua fitur utama, yaitu `ff_x` dan `ddd_deg`. Bi-LSTM memiliki kemampuan untuk memproses data secara bersamaan dalam dua arah, yaitu maju dan mundur, sehingga model dapat menangkap informasi dari kedua sisi waktu pada data input. Model ini dibangun dengan menggunakan kerangka Sequential, di mana input data memiliki dimensi (1, 2), yang berarti satu time step dengan dua fitur yang digunakan.

Model ini dimulai dengan lapisan Bi-LSTM, dengan jumlah unit yang diuji bervariasi antara 50, 100, dan 150 untuk mencari konfigurasi yang paling optimal dalam menangkap pola pada data input. Setelah lapisan Bi-LSTM, diterapkan lapisan Dropout dengan tingkat dropout sebesar 0.01 untuk mencegah model terlalu mengandalkan beberapa fitur dan mengurangi risiko overfitting. Pada lapisan output, model menggunakan lapisan Dense dengan dua unit untuk memprediksi fitur target, yaitu `ff_x` dan `ddd_deg`.

Pelatihan model menggunakan optimizer Adam dengan learning rate 0.01, yang efektif dalam proses konvergensi. Fungsi kerugian yang digunakan adalah Mean Squared Error (MSE), yang umum digunakan dalam tugas regresi. Model ini diuji dengan kombinasi hyperparameter, termasuk jumlah unit Bi-LSTM (50, 100, dan 150), ukuran batch (16, 32, dan 64), serta jumlah epoch yang digunakan sebanyak 100. Dengan pendekatan ini, arsitektur Bi-LSTM diharapkan mampu menangkap pola yang lebih kompleks pada data sekuensial, baik dari hubungan



waktu ke depan maupun ke belakang, yang akan menghasilkan prediksi yang lebih akurat untuk fitur `ff_x` dan `ddd_deg`.

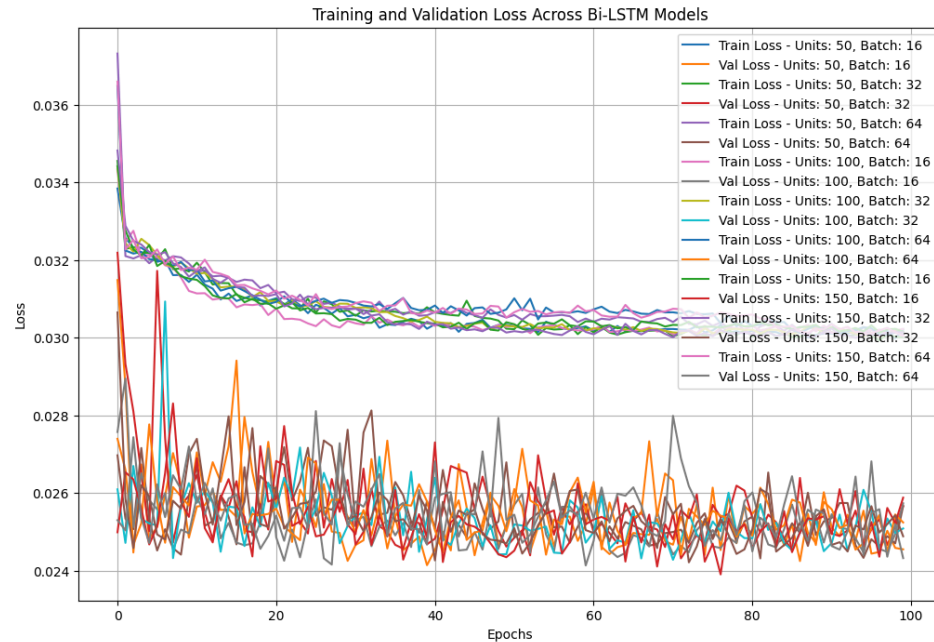
Tabel 4.4 Arsitektur Model Bidirectional LSTM

Lapisan	Tipe	Jumlah Unit/Parameter	Keterangan
1	Bidirectional LSTM	50, 100, 150	Memproses data sekuensial dua arah, units disesuaikan saat tuning.
2	Dropout	0.01	Mengurangi overfitting dengan men-drop neuron.
3	Dense	2	Lapisan output untuk memprediksi 2 fitur.

Tabel 4.5 Tabel Hyperparameter Tuning Model Bidirectional LSTM

Lapisan	Jumlah Unit/Parameter	Keterangan
Bi-LSTM Units	50, 100, 150	Mengontrol kapasitas representasi model.
Dropout Rate	0.01	Probabilitas untuk menjatuhkan neuron selama pelatihan.
Batch Size	16, 32, 64	Ukuran batch data selama pelatihan.
Epochs	100	Jumlah iterasi pelatihan penuh pada data.
Learning Rate	0.01	Kecepatan pembaruan parameter dalam optimizer Adam.

Berdasarkan hasil pengujian Arsitektur Bi-LSTM yang menggunakan tuner pada Tabel 4.4, dapat disimpulkan bahwa konfigurasi model dengan parameter 'units' sebanyak 150, 'dropout' sebesar 0.01, dan 'batch\_size' 64 memberikan performa terbaik. Grafik loss yang dihasilkan dari kombinasi parameter yang diuji ditampilkan dalam grafik Gambar 4.3 di bawah ini.



Gambar 4.3 Training dan Validation Loss Model Bi-LSTM

Tabel 4.6 Hasil Training dan Validation Loss Parameter Model Bi-LSTM

index	Units	Dropout	Batch Size	Train Loss	Validation Loss
0	50	0.01	16	0.030207	0.024551
1	50	0.01	32	0.030012	0.02566
2	50	0.01	64	0.030081	0.025666
3	100	0.01	16	0.030222	0.025735
4	100	0.01	32	0.030138	0.025087
5	100	0.01	64	0.030153	0.025242
6	150	0.01	16	0.030154	0.025881
7	150	0.01	32	0.030135	0.024887
8	150	0.01	64	0.03004	0.024325

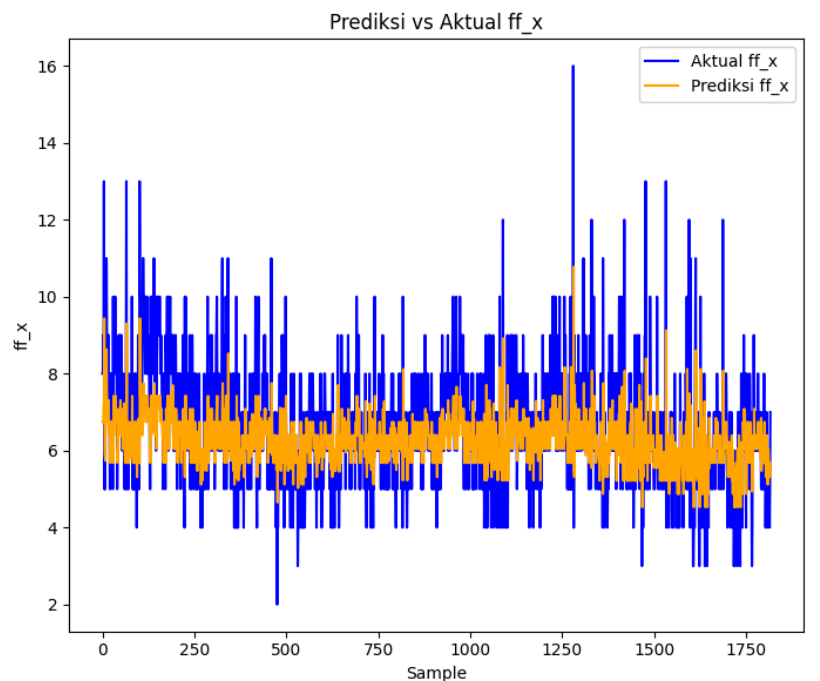
Data angin yang berpola setiap tiga bulan menunjukkan adanya siklus musiman yang khas, seperti yang sering terjadi dalam fenomena cuaca akibat perubahan muson atau kondisi atmosfer lainnya. Dalam analisis data cuaca, pola musiman ini umumnya dibagi ke dalam empat kelompok: DJF (Desember-Januari-Februari) yang mewakili musim dingin di belahan bumi utara, MAM (Maret-April-Mei) untuk musim semi, JJA (Juni-Juli-Agustus) untuk musim panas, dan SON (September-Oktober-November) untuk musim gugur. Pendekatan ini penting karena kondisi atmosfer, seperti arah dan kecepatan angin, sering kali berubah secara signifikan dari satu musim ke musim lainnya.

Dalam proses pelatihan model, batch size 64 kemungkinan besar membantu menangkap pola musiman ini dengan lebih baik karena mencakup lebih banyak sampel dalam satu iterasi, sehingga dapat merepresentasikan siklus musiman tiga bulanan tersebut secara lebih akurat dibandingkan batch yang lebih kecil. Selain itu, preprocessing data dengan menambahkan informasi waktu, seperti bulan atau musim, dapat membantu model lebih memahami konteks periodik ini. Dengan pelatihan yang cukup panjang dan struktur model yang dirancang untuk menangani data sekuensial seperti LSTM, model dapat mengenali pola musiman ini dan meningkatkan akurasi prediksi terhadap arah dan kecepatan angin. Pendekatan ini memastikan model lebih efektif dalam memanfaatkan siklus musiman untuk memberikan hasil yang lebih baik.

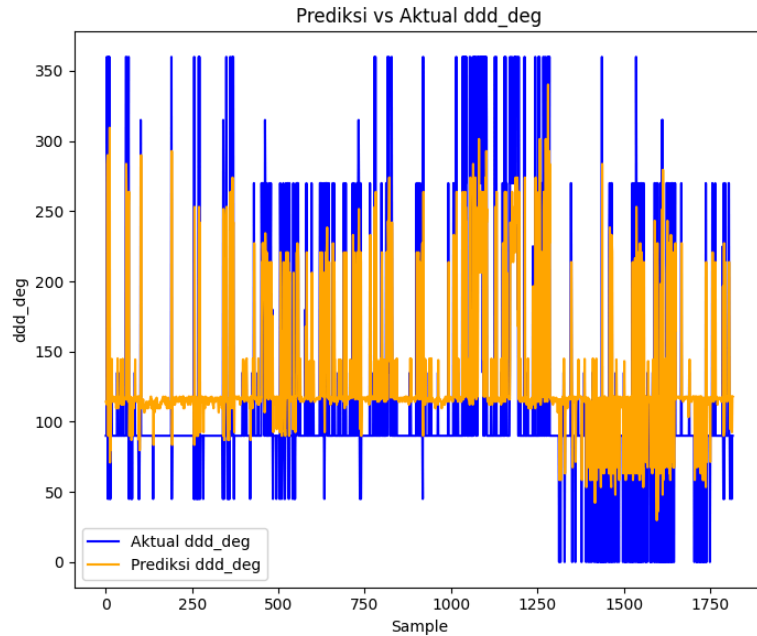
## 4.2 Pemilihan Model Terbaik

### 4.2.1 Prediksi dengan Model Hyperparameter Terbaik

Gambar di bawah ini menunjukkan hasil prediksi kecepatan angin (Gambar 4.4) dan arah angin (Gambar 4.5) menggunakan model LSTM dari data testing sebanyak 20% dari total data yang dimiliki.

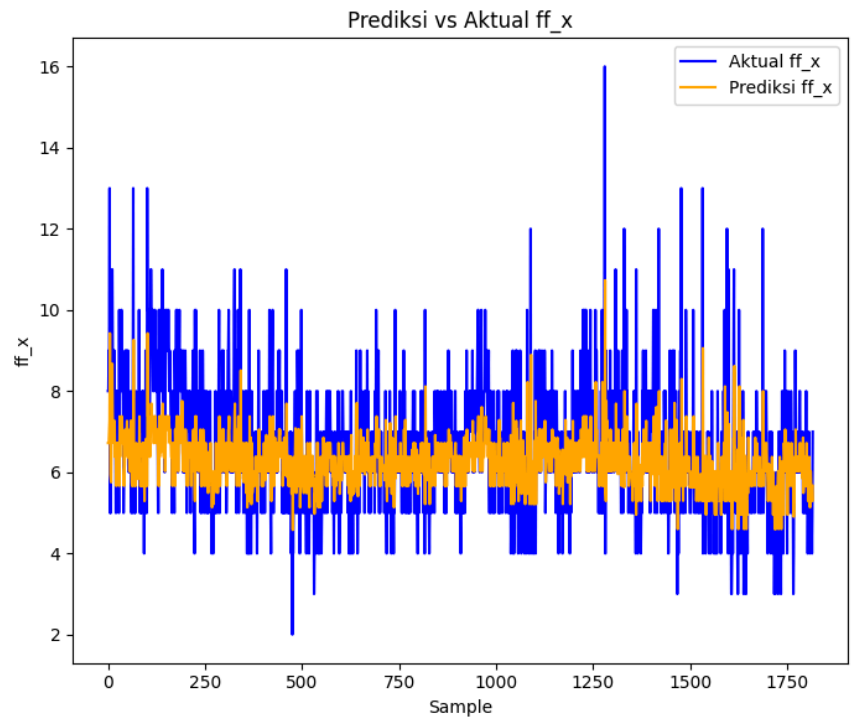


Gambar 4.4 Grafik Perbandingan Prediksi dan Aktual Kecepatan Angin Model LSTM

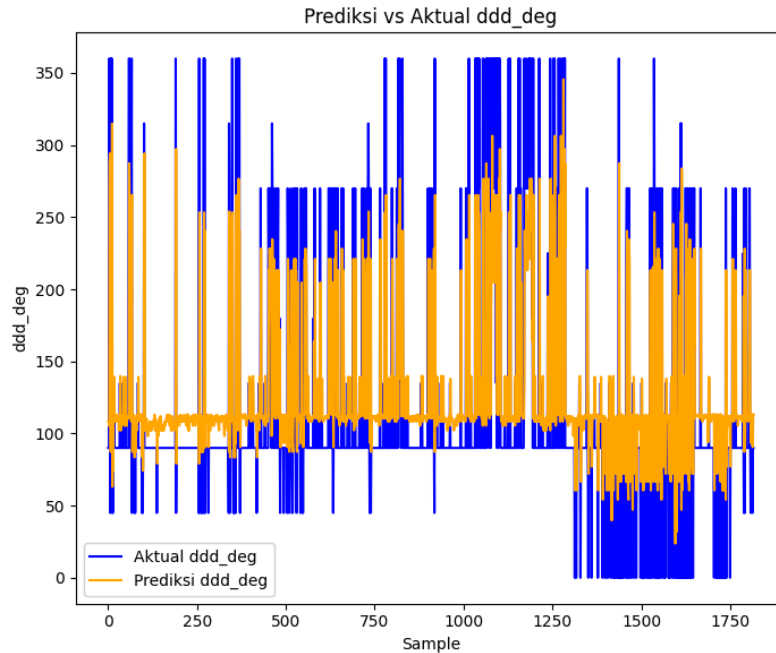


Gambar 4.5 Grafik Perbandingan Prediksi dan Aktual Arah Angin Model LSTM

Gambar di bawah ini menunjukkan hasil prediksi kecepatan angin (Gambar 4.6) dan arah angin (Gambar 4.7) menggunakan model Bidirecional LSTM dari data testing sebanyak 20% dari total data yang dimiliki.



Gambar 4.6 Grafik Perbandingan Prediksi dan Aktual Kecepatan Angin Model Bi-LSTM



Gambar 4.7 Grafik Perbandingan Prediksi dan Aktual Arah Angin Model Bi-LSTM

#### 4.2.2 Evaluasi Metrik

Pada tahap evaluasi metrik, dilakukan analisis performa model menggunakan beberapa metrik evaluasi seperti Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), dan R-squared ( $R^2$ ). Metrik ini digunakan untuk mengukur seberapa baik model dapat memprediksi dua variabel target, yaitu `ff_x` dan `ddd_deg`.

Tabel 4.7 Evaluasi Metrik Prediksi Model

Model	Target	MSE	RMSE	MAE	R-squared ( $R^2$ )
LSTM	<code>ff_x</code>	2.3765	1.5416	1.1621	0.0332
LSTM	<code>ddd_deg</code>	6015.655	77.561	54.198	0.0847
Bi-LSTM	<code>ff_x</code>	2.3832	1.5438	1.1636	0.0305
Bi-LSTM	<code>ddd_deg</code>	5938.195	77.06	51.024	0.0965

Pada model LSTM, hasil evaluasi menunjukkan bahwa untuk variabel `ff_x`, nilai MSE sebesar 2.3765 dan RMSE sebesar 1.5416. Nilai MAE mencapai 1.1621, sedangkan nilai  $R^2$  hanya sebesar 0.0332, yang mengindikasikan kemampuan model dalam menjelaskan variasi data target masih sangat rendah.

Untuk variabel `ddd_deg`, nilai MSE jauh lebih besar, yaitu 6015.655, dengan RMSE sebesar 77.561. Nilai MAE untuk variabel ini adalah 54.198, dan  $R^2$  sebesar 0.0847 menunjukkan bahwa model belum dapat sepenuhnya menjelaskan hubungan antara input dan target.

Pada model Bi-LSTM, untuk variabel `ff_x`, nilai MSE adalah 2.3832 dengan RMSE sebesar 1.5438, sedangkan MAE mencapai 1.1636. Nilai  $R^2$  sedikit lebih rendah dibandingkan LSTM, yaitu sebesar 0.0305. Untuk variabel `ddd_deg`, model ini menunjukkan performa yang lebih baik dibandingkan LSTM, dengan MSE sebesar 5938.195, RMSE sebesar 77.060, dan MAE sebesar 51.024. Nilai  $R^2$  meningkat menjadi 0.0965, yang menunjukkan adanya sedikit peningkatan kemampuan model dalam memprediksi variabel ini dibandingkan model LSTM.

Dari hasil evaluasi ini, dapat disimpulkan bahwa baik LSTM maupun Bi-LSTM masih memiliki keterbatasan dalam menjelaskan hubungan antara input dan target, terutama terlihat dari nilai  $R^2$  yang rendah. Namun, model Bi-LSTM menunjukkan performa yang sedikit lebih baik untuk variabel `ddd_deg` dibandingkan LSTM.

Berdasarkan evaluasi terhadap model yang dikembangkan, performa model untuk memprediksi target kecepatan angin (`ff_x`) masih belum optimal. Walaupun nilai Mean Squared Error (MSE) sebesar 2.3765 terlihat rendah dalam konteks numerik, namun dalam aplikasi kecepatan angin, nilai ini sebenarnya cukup signifikan. Hal ini dikarenakan perbedaan 1 m/s dalam kecepatan angin sudah dianggap besar dalam pengukuran meteorologi, mengingat perubahan sekecil itu dapat memiliki dampak besar pada kondisi cuaca atau aplikasi lain yang sensitif terhadap angin.

Ketidaksempurnaan model ini tidak hanya berasal dari arsitektur atau parameter yang digunakan, tetapi juga sangat dipengaruhi oleh karakteristik data yang digunakan dalam proses pelatihan. Salah satu masalah utama adalah kurangnya variasi dalam data kecepatan angin. Data yang tersedia hanya memiliki 30 nilai unik, yang berarti nilai-nilai tersebut lebih menyerupai kategori daripada distribusi data numerik kontinu. Akibatnya, model mungkin kesulitan untuk menangkap hubungan kontinu yang lebih kompleks, karena data lebih mendekati sifat kategoris.

Kurangnya variasi ini menyebabkan model tidak memiliki cukup informasi untuk belajar pola yang benar-benar mewakili fenomena kecepatan angin yang sebenarnya. Akibatnya, kemampuan generalisasi model terhadap data baru menjadi terbatas, sehingga kesalahan prediksi yang dihasilkan tetap signifikan.

## BAB 5

### KESIMPULAN DAN SARAN

#### 5.1 Kesimpulan

Berdasarkan hasil penelitian ini, dapat disimpulkan bahwa meskipun model LSTM dan Bi-LSTM memberikan hasil yang baik, keduanya masih memerlukan perbaikan untuk mencapai performa yang lebih optimal. Model LSTM dengan konfigurasi units = 50, dropout = 0.01, dan batch size = 64 menghasilkan nilai train loss sebesar 0.030173 dan val loss 0.024418. Untuk prediksi variabel *ff\_x*, model LSTM memiliki MSE 2.3765, RMSE 1.5416, MAE 1.1621, dan nilai R-squared ( $R^2$ ) 0.0332. Sementara itu, untuk prediksi variabel *ddd\_deg*, LSTM menghasilkan MSE 6015.655, RMSE 77.561, MAE 54.198, dan R-squared ( $R^2$ ) 0.0847.

Model Bi-LSTM dengan konfigurasi units = 150, dropout = 0.01, dan batch size = 64 menghasilkan train loss sebesar 0.03004 dan val loss 0.024325. Untuk prediksi variabel *ff\_x*, model Bi-LSTM memiliki MSE 2.3832, RMSE 1.5438, MAE 1.1636, dan R-squared ( $R^2$ ) 0.0305. Sedangkan untuk prediksi variabel *ddd\_deg*, Bi-LSTM menghasilkan MSE 5938.195, RMSE 77.06, MAE 51.024, dan R-squared ( $R^2$ ) 0.0965.

Secara keseluruhan, meskipun perbedaan antara LSTM dan Bi-LSTM tidak terlalu besar, Bi-LSTM sedikit lebih unggul dalam memprediksi variabel *ddd\_deg*, dengan nilai R-squared ( $R^2$ ) yang lebih tinggi, yaitu 0.0965, yang menunjukkan bahwa Bi-LSTM dapat menangkap pola temporal yang lebih kompleks. Namun, kedua model masih memiliki performa yang terbatas dalam menjelaskan hubungan antara input dan target, yang direpresentasikan dari nilai R-squared ( $R^2$ ) yang rendah.

#### 5.2 Saran

Untuk meningkatkan kinerja model LSTM dan Bi-LSTM dalam memprediksi variabel *ff\_x* dan *ddd\_deg*, penting untuk mengeksplorasi arsitektur lain seperti GRU atau Transformer. Pencarian hyperparameter (jumlah unit, *dropout*, ukuran *batch*, dan *learning rate*) dapat dilakukan dengan teknik Grid Search atau Random Search. Penyempurnaan dalam *preprocessing* data dan *feature engineering* juga perlu dilakukan untuk meningkatkan relevansi informasi yang digunakan oleh model. Terakhir, evaluasi menggunakan metrik seperti MAPE (Mean Absolute Percentage Error) akan memberikan gambaran yang lebih lengkap mengenai kinerja model, sehingga diharapkan hasil prediksi menjadi lebih akurat.

## DAFTAR PUSTAKA

- Dwi, C., Simbolon, L., Ruhiat, Y., & Saefullah, A. (2022). Analisis Arah dan Kecepatan Angin Terhadap Sebaran Curah Hujan Di Wilayah Kabupaten Tangerang. In *Jurnal Teori dan Aplikasi Fisika* (Vol. 10, Issue 01).
- Gede, I., & Suryana, P. E. (2022). Analisis Windrose untuk Prediksi Arah & Jangkauan Pencemaran Udara. *Jurnal Sistem Informasi Dan Komputer Terapan Indonesia (JSIKTI)*, 4(3), 132–141. <https://doi.org/10.22146/jsikti.xxxx>
- Madjid, F. M., & Kurniawan, T. B. (2022). PREDIKSI VISIBILITY MENGGUNAKAN LSTM DAN MLP DI BANDARA SULTAN MAHMUD BADARUDDIN II PALEMBANG. *Jurnal Sistem Informasi (JSI)*, 14(1). <http://ejournal.unsri.ac.id/index.php/jsi/index>
- Nikentari, N., Bettiza, M., Sasty, H., & #3, P. (n.d.). *JEPIN (Jurnal Edukasi dan Penelitian Informatika) Prediksi Kecepatan Angin Menggunakan Adaptive Neuro Fuzzy (ANFIS) dan Radial Basis Function Neural Network (RBFNN)*.
- Nugraha, Y. E. N., Ariawan, I., & Arifin, W. A. (2023). WEATHER FORECAST FROM TIME SERIES DATA USING LSTM ALGORITHM. *JURNAL TEKNOLOGI INFORMASI DAN KOMUNIKASI*, 14(1), 144–152. <https://doi.org/10.51903/jtikp.v14i1.531>
- Nugroho, A. A., & Haris, M. (2024). ANALISIS EFEKTIVITAS TEKNIK IMPUTASI PADA LSTM UNTUK MENINGKATKAN KUALITAS DATA PADA PERAMALAN CURAH HUJAN. In *Jakarta Nusa Mandiri Tower Jl. Jatiwaringin Raya* (Vol. 7, Issue 2). <http://e-journal.stmiklombok.ac.id/index.php/jireISSN.2620-6900>
- Pramono, A., Sutaryani, A., Tamara Qothrunada, D., Satria, H. W., Kelautan dan Perikanan Republik Indonesia, K., & Meteorologi Klimatologi dan Geofisika, B. (2022). *Seminar Nasional TREN D Technology of Renewable Energy and Development FTI Universitas*.
- Siti Nurjanah, Yoan Purbolingga, Dila Marta Putri, Asde Rahmawati, Fahrizal Fahrizal, & Bastul Wajhi Akramunnas. (2024). Prediksi Kecepatan Angin untuk Mengetahui Potensi Sumber Energi Alternatif menggunakan Model Regresi Lasso: Studi Kasus Kota Makassar pada Tahun 2024. *Jurnal Penelitian Rumpun Ilmu Teknik*, 3(1), 278–288. <https://doi.org/10.55606/juprit.v3i1.3501>