PENERAPAN RESNET-LSTM UNTUK MENGATASI *VANISHING GRADIENT* PADA PREDIKSI KONSENTRASI POLUTAN DALAM KUALITAS UDARA DKI JAKARTA

TUGAS AKHIR

Oleh :

Syafira Widiyanti
123200057

PROGRAM STUDI INFORMATIKA
JURUSAN INFORMATIKA
FAKULTAS TEKNIK INDUSTRI
UNIVERSITAS PEMBANGUNAN NASIONAL "VETERAN"
YOGYAKARTA

2024

PENERAPAN RESNET-LSTM UNTUK MENGATASI *VANISHING GRADIENT* PADA PREDIKSI KONSENTRASI POLUTAN DALAM KUALITAS UDARA DKI JAKARTA

TUGAS AKHIR

Tugas Akhir ini sebagai salah satu syarat untuk memperoleh gelar sarjana S-1 di Program Studi Informatika, Jurusan Informatika, Fakultas Teknik Industri, Universitas Pembangunan Nasional "Veteran" Yogyakarta

Oleh :

Syafira Widiyanti
123200057

PROGRAM STUDI INFORMATIKA
JURUSAN INFORMATIKA
FAKULTAS TEKNIK INDUSTRI
UNIVERSITAS PEMBANGUNAN NASIONAL "VETERAN"
YOGYAKARTA
2024

HALAMAN PENGESAHAN PEMBIMBING

PENERAPAN RESNET-LSTM UNTUK MENGATASI *VANISHING GRADIENT* PADA PREDIKSI KONSENTRASI POLUTAN DALAM KUALITAS UDARA DKI JAKARTA

Disusun oleh:
Syafira Widiyanti
123200057

Telah diuji dan dinyatakan lulus oleh pembimbing
pada tanggal: 16 Mei 2024

Menyetujui,
Pembimbing

Dr. Herlina Jayadianti, S.T., M.T.
NIDN. 0527087701

Mengetahui,
Koordinator Program Studi

Dessyanto Budi P., S.T., M.T.
NIDN. 0505127501

HALAMAN PENGESAHAN PENGUJI

PENERAPAN RESNET-LSTM UNTUK MENGATASI *VANISHING GRADIENT* PADA PREDIKSI KONSENTRASI POLUTAN DALAM KUALITAS UDARA DKI JAKARTA

Disusun oleh:
Syafira Widiyanti
123200057

Telah diuji dan dinyatakan lulus oleh penguji
Pada tanggal: 16 Mei 2024

Menyetujui,

Penguji I	Penguji II
Dr. Herlina Jayadianti, S.T., M.T.	Wilis Kaswidjanti, S.Si., M. Kom.
NIDN. 0527087701	NIDN. 0513047601

Penguji III	Penguji IV
Dessyanto Budi P., S.T., M.T.	Ahmad Taufiq Akbar, S.Si., M.Cs.
NIDN. 0505127501	NIDN. 0518118701

SURAT PERNYATAAN
KARYA ASLI TUGAS AKHIR

Sebagai mahasiswa Program Studi Informatika Fakultas Teknik Industri Universitas Pembangunan Nasional "Veteran" Yogyakarta, yang bertanda tangan di bawah ini, Saya:

Nama : Syafira Widiyanti

NIM : 123200057

Menyatakan bahwa karya ilmiah saya yang berjudul:

Penerapan ResNet-LSTM Untuk Mengatasi *Vanishing Gradient* pada Prediksi Konsentrasi Polutan dalam Kualitas Udara DKI Jakarta

Merupakan karya asli dan belum pernah dipublikasikan di mana pun. Apabila di kemudian hari, karya saya disinyalir bukan merupakan karya asli saya, maka saya bersedia menerima konsekuensi apa pun yang diberikan Program Studi Informatika Fakultas Teknik Industri Universitas Pembangunan Nasional "Veteran" Yogyakarta kepada saya.

Demikian surat pernyataan ini saya buat dengan sebenarnya.

Dibuat di : Yogyakarta
Pada tanggal : 3 Mei 2024

Yang menyatakan,

Syafira Widiyanti
NIM. 123200057

PERNYATAAN BEBAS PLAGIASI

Saya yang bertanda tangan di bawah ini:

Nama : Syafira Widiyanti

NIM : 123200057

Fakulitas/Prodi : Teknik Industri/Informatika

Dengan ini saya menyatakan bahwa judul Tugas Akhir

Penerapan ResNet-LSTM Untuk Mengatasi *Vanishing Gradient* pada Prediksi Konsentrasi Polutan dalam Kualitas Udara DKI Jakarta

adalah hasil kerja saya sendiri dan benar bebas dari plagiasi kecuali cuplikan serta ringkasan yang terdapat di dalamnya telah saya jelaskan sumbernya (Sitasi) dengan jelas. Apabila pernyataan ini terbukti tidak benar maka saya bersedia menerima sanksi sesuai peraturan Mendiknas RI No 17 Tahun 2010 dan Peraturan Perundang-undangan yang berlaku.

Demikian surat pernyataan ini saya buat dengan penuh tanggung jawab.

Yogyakarta, 3 Mei 2024
Yang membuat pernyataan,

Syafira Widiyanti
NIM. 123200057

ABSTRAK

Penurunan kualitas udara akibat polusi berdampak buruk terhadap kesehatan manusia. Oleh karena itu, prediksi konsentrasi polutan dalam kualitas udara perlu dilakukan guna memberikan peringatan dini kepada masyarakat. Berbagai studi menunjukkan prediksi konsentrasi polutan dapat dilakukan menggunakan *Recurrent Neural Network* (RNN) atau *Convolutional Neural Network* (CNN). Namun, metode ini rentan meningkatkan kompleksitas jaringan yang dapat menyebabkan *vanishing gradient*. Akibatnya hasil akurasi prediksi menjadi tidak optimal. Selain itu, masing-masing metode hanya dapat mengekstrak salah satu fitur, baik temporal atau spasial saja.

Dalam penelitian ini, metode *Long Short-Term Memory* (LSTM) digunakan untuk mengatasi masalah *vanishing gradient* karena arsitekturnya yang mampu mengontrol aliran informasi. Metode ini dikombinasikan dengan arsitektur *Residual Network* (ResNet) yang mampu mengatasi *vanishing gradient* dan degradrasi jaringan dengan teknik *skip connections,* sehingga akurasi prediksi menjadi optimal. Metode ResNet dan LSTM juga digunakan untuk mengesktrak fitur *spatiotemporal* pada data dan digunakan selama pembangunan model prediksi.

Evaluasi model pada penelitian ini menggunakan *Mean Absolute Percentage Error* (MAPE) dan diuji dengan menjalankan lima skenario yaitu prediksi konsentrasi polutan menggunakan LSTM, CNN, ResNet, CNN-LSTM, dan ResNet-LSTM. Hasil evaluasi MAPE model ResNet-LSTM untuk *training error* sebesar 7.5%, *validation error* sebesar 22.7%, dan *testing error* sebesar 19.1%. Berdasarkan hasil tersebut, metode ResNet-LSTM memiliki nilai *error* yang lebih rendah dibandingkan metode CNN-LSTM. Hal tersebut menunjukkan bahwa metode ResNet-LSTM masih lebih baik dalam mengatasi *vanishing gradient* karena memiliki nilai *error* yang lebih rendah selama proses pelatihan model.

Kata kunci: *long short-term memory*, *residual network*, prediksi, *vanishing gradient*, polutan, meteorologi

ABSTRACT

Deterioration in air quality due to pollution has negative impact on human health. Therefore, prediction of pollutant concentrations in air quality needs to be done to provide early warning to the public. Various studies show prediction of pollutant concentrations can be done using Recurrent Neural Network (RNN) or Convolutional Neural Network (CNN). However, this method is prone to increasing network complexity which can lead to gradient vanishing. As a result, the prediction accuracy results are not optimal. In addition, each method can extract only one of the features, either temporal or spatial.

In this study, the Long Short-Term Memory (LSTM) method was used to overcome vanishing gradient problem due to its architecture capable of controlling the flow of information. This method is combined with the Residual Network (ResNet) architecture that is able to overcome vanishing gradient and network degradation with skip connections techniques, so that prediction accuracy is optimal. ResNet and LSTM methods are also used to extract spatiotemporal features in data and are used during the construction of prediction models.

The evaluation of the model in this study used Mean Absolute Percentage Error (MAPE) and was tested by running five scenarios, namely prediction of pollutant concentrations using LSTM, CNN, ResNet, CNN-LSTM, and ResNet-LSTM. The results of the MAPE evaluation of the ResNet-LSTM model for training error of 7.5%, validation error of 22.7%, and testing error of 19.1%. Based on these results, the ResNet-LSTM method has a lower error value than the CNN-LSTM method. This shows that the ResNet-LSTM method is still better at overcoming the vanishing gradient because it has a lower error value during the model training process.

Keywords: long short-term memory, residual network, prediction, vanishing gradient, pollutant, meteorology

KATA PENGANTAR

Puji dan syukur kehadirat Allah SWT yang senantiasa mencurahkan rahmat dan karunia-Nya sehingga penulis dapat menyelesaikan tugas akhir yang berjudul "Penerapan ResNet-LSTM Untuk Mengatasi *Vanishing Gradient* pada Prediksi Konsentrasi Polutan dalam Kualitas Udara DKI Jakarta". Tugas akhir ini disusun untuk memenuhi salah satu syarat dalam menyelesaikan studi di Program Studi Informatika, Jurusan Informatika, Fakultas Teknik Industri, Universitas Pembangunan Nasional "Veteran" Yogyakarta.

Selama menyelesaikan tugas akhir ini, penulis menyadari bahwa penulisan tugas akhir ini dapat diselesaikan dengan baik tidak terlepas dari bimbingan, arahan, dan bantuan yang diberikan oleh berbagai pihak. Oleh karena itu, penulis ingin mengucapkan terima kasih kepada semua pihak yang telah terlibat, yaitu:

1. Bapak Dr. Awang Hendrianto Pratomo, S.T., M.T. selaku Ketua Jurusan Informatika UPN "Veteran" Yogyakara,
2. Bapak Dr. Heriyanto, A.Md., S.Kom., M.Cs., selaku Koordinator Program Studi Informatika UPN "Veteran" Yogyakarta,
3. Ibu Dr. Herlina Jayadianti, S.T., M.T. selaku dosen pembimbing tugas akhir atas bantuan, bimbingan, arahan, serta saran yang telah diberikan kepada penulis,
4. Bapak Budi Santosa, S.Si., M.T. selaku dosen wali,
5. Teman-teman kelas H Informatika 2020 dan UKM ISR UPN "Veteran" Yogyakarta, serta
6. Bapak, Ibu, adik, dan keluarga penulis yang selalu mendoakan dan memberikan semangat serta dukungan selama proses pengerjaan tugas akhir ini.

Akhir kata, semoga tugas akhir ini dapat memberikan manfaat kepada pembaca dan berkontribusi dalam pengembangan ilmu pengetahuan ke depannya. Penulis menyadari bahwa tugas akhir ini masih belum sempurna karena keterbatasan ilmu pengetahuan dan pengalaman penulis. Besar harapan penulis untuk mendapatkan kritik dan saran yang membangun sebagai upaya pembelajaran dan penyempurnaan lebih lanjut.

Yogyakarta, 3 Mei 2024
Penulis

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR PERSAMAAN

DAFTAR MODUL

BAB I
PENDAHULUAN

1.1 Latar Belakang

Penurunan kualitas udara di Indonesia disebabkan karena adanya polusi udara oleh pencemaran emisi dari kendaraan bermotor, industri/pabrik, pembangkit listrik, rumah tangga, pembakaran lahan, dan sebagainya. Tingkat kandungan zat-zat polutan yang tinggi dapat mempengaruhi kesehatan manusia dan lingkungan, di mana akan berdampak sangat buruk pada seluruh aspek kehidupan di muka bumi (Amalia et al., 2022). Berdasarkan data IQAir dan AQI.in, pada bulan Desember 2023, Ibukota Negara Indonesia, DKI Jakarta menduduki peringkat ke-2 kota paling berpolusi di Indonesia dengan konsentrasi partikel polutan $PM_{2.5}$ sebesar 43 mikrogram per meter kubik dan PM_{10} sebesar 89 mikrogram per meter kubik. Konsentrasi partikel ini 2.9 kali lebih tinggi dari ambang batas yang ditetapkan oleh WHO. $PM_{2.5}$ merupakan salah satu dari tujuh zat polutan yang menjadi parameter pengukuran Indeks Standar Pencemar Udara (ISPU) oleh Kementerian Lingkungan Hidup dan Kehutanan (KLHK), yaitu PM_{10}, $PM_{2.5}$, NO_2, SO_2, CO, O_3, dan HC yang berdampak akut dan kronis terhadap kesehatan manusia (Chaniago et al., 2020a). Adanya prediksi konsentrasi polutan dalam kualitas udara dapat memberikan peringatan dini yang berguna bagi masyarakat untuk mengambil tindakan yang tepat. Namun, konsentrasi polutan ini sangat dipengaruhi oleh data meteorologi seperti kelembapan, suhu, kecepatan angin, arah angin, dan curah hujan (Hidayat & Anov, 2023; Liu et al., 2019; Millah et al., 2022; Serlina, 2020). Oleh karena itu, data polutan dan meteorologi akan untuk melakukan prediksi konsentrasi polutan dalam kualitas udara.

Penelitian mengenai prediksi konsentrasi polutan telah dilakukan sebelumnya dengan berbagai metode. Metode *deep learning* memiliki kemampuan pengembangan fitur yang lebih kuat dan unggul dibandingkan metode *machine learning* atau statistik (Dun et al., 2022). Salah satu metode yang paling representatif adalah *Recurrent Neural Network* (RNN) untuk memprediksi data *time series* (J. Wang et al., 2022). RNN memiliki kelebihan dalam menangani tugas pembelajaran sekuensial. Namun, RNN mencatat banyak informasi tidak berguna yang dapat meningkatkan kompleksitas jaringan secara signifikan. Selain itu, RNN menggunakan sigmoid sebagai fungsi aktivasi yang dapat menyebabkan *vanishing gradient* (Sheng et al., 2023). Hal ini dapat membuat informasi yang relevan tidak dapat dipelajari dengan baik dan akurasi prediksi menjadi tidak optimal (Wu et al., 2023).

Metode *Recurrent Neural Network* (RNN), seperti *Gated Recurrent Unit* (GRU) (Duan et al., 2021) dan *Long-Short Term Memory* (LSTM) (Li et al., 2023) digunakan untuk melakukan ekstraksi fitur temporal jangka panjang padmea data *time series*. LSTM terbukti cocok untuk memprediksi data *time series* dengan performa yang lebih baik dalam mengatasi masalah *vanishing gradient*

dibandingkan RNN karena arsitekturnya yang mencakup mekanisme gerbang yang mampu mengontrol aliran informasi (Qin et al., 2019; Q. Zhang et al., 2020). Namun, metode ini hanya berfokus pada korelasi temporal, sehingga tidak efisien dalam menangani fitur spasial yang mungkin dapat membatasi performa model dalam menangani data *spatiotemporal*.

Metode *Convolution Neural Network* (CNN) juga banyak digunakan oleh beberapa peneliti karena mampu melakukan ekstraksi fitur-fitur penting pada berbagai tingkatan abstraksi, terutama pola spasial (Kow et al., 2020), tetapi memiliki tingkat sensitivitas yang rendah pada perbedaan yang cukup kecil sehingga sulit untuk membedakan kelas yang sangat mirip pada data yang serupa (Z. Wang et al., 2019). Metode CNN juga dikombinasikan dengan LSTM (Portal-Porras et al., 2023; Qin et al., 2019), di mana LSTM mampu mengekstraksi fitur temporal dan CNN mengekstraksi fitur spasial.

Arsitektur CNN juga digunakan untuk meningkatkan akurasi prediksi kualitas udara, salah satunya adalah *Residual Network* (ResNet). Arsitektur ini juga dapat dikombinasikan dengan *deep learning* menjadi *Deep Residual Networks* (Q. Zhang et al., 2020), tetapi belum mampu melakukan prediksi secara akurat pada data yang tidak stabil dan terus berubah seiring waktu sehingga hanya menghasilkan prediksi sebesar 80%. Arsitektur ResNet untuk prediksi $PM_{2.5}$ (Cheng et al., 2022; Song et al., 2020) memiliki hasil yang cukup bagus, di mana hasil akurasi yang didapatkan lebih dari 80% pada data validasi dan pengujian. Selain itu, teknik *skip connections* atau *residual connections* yang dimiliki ResNet dapat mengatasi masalah *vanishing gradient* dan degradasi jaringan akibat peningkatan kompleksitas jaringan (Kalajdjieski et al., 2020; Szegedy et al., 2017). Kombinasi ResNet dan LSTM tidak hanya menyelesaikan masalah korelasi fitur spasial dan temporal tetapi juga mampu mengatasi *vanishing gradient* (L. Zhang et al., 2017; Q. Zhang et al., 2020). Oleh karena itu, ResNet dan LSTM dapat dijadikan solusi untuk mengoptimalkan akurasi akibat menyelesaikan permasalahan *vanishing gradient* pada prediksi konsentrasi polutan dalam kualitas udara.

Berdasarkan landasan pengetahuan yang didapatkan, penelitian ini akan memanfaatkan ResNet untuk ekstraksi fitur spasial dan LSTM untuk ekstraksi fitur *temporal* pada data polutan dan meteorologi. Diharapkan penggunaan arsitektur ResNet-LSTM dapat mengoptimalkan akurasi akibat mengatasi *vanishing gradient* saat melakukan prediksi konsentrasi polutan dalam kualitas udara.

1.2 Rumusan Masalah

Rumusan masalah pada penelitian ini, yaitu:

1. Hilangnya nilai gradien pada tiap layer selama proses pelatihan diakibatkan oleh meningkatnya kompleksitas jaringan pada prediksi konsentrasi polutan dalam kualitas udara, mengakibatkan akurasi menjadi tidak optimal.
2. Apakah ResNet-LSTM dapat mengatasi masalah *vanishing gradient* sehingga akurasi menjadi lebih optimal?

3. Apakah ResNet-LSTM dapat meningkatkan akurasi prediksi konsentrasi polutan dalam kualitas udara?

1.3 Batasan Masalah

Batasan masalah pada penelitian ini, yaitu:

1. Penelitian ini mengambil dua jenis data historis, yaitu data konsentrasi polutan dan data meteorologi dari lima stasiun pemantauan kualitas udara pada lima kota/kabupaten administrasi di DKI Jakarta, Indonesia.
2. Sumber data merupakan kombinasi data primer-sekunder yang diambil dari Satu Data Jakarta untuk data konsentrasi polutan dan Data Online Pusat Database BMKG untuk data meteorologi.
3. Data konsentrasi polutan meliputi data pengukuran zat PM_{10}, SO_2, CO, O_3, dan NO_2 harian dari tahun 2019 – 2021.
4. Data meteorologi meliputi data pengukuran suhu rata-rata, kelembaban rata-rata, kecepatan angin rata-rata, dan arah angin harian dari tahun 2019 – 2021.
5. Data pendukung seperti data lalu lintas dan penggunaan lahan tidak digunakan dalam penelitian ini.

1.4 Tujuan Penelitian

Tujuan dari penelitian ini, yaitu:

1. Menerapkan ResNet-LSTM untuk mengatasi *vanishing gradient* akibat meningkatnya kompleksitas jaringan pada prediksi.
2. Mendapatkan hasil prediksi konsentrasi polutan dalam kualitas udara yang optimal menggunakan ResNet-LSTM.

1.5 Manfaat Penelitian

Hasil penelitian ini dapat dijadikan acuan dan bahan pertimbangan dalam membuat model *machine learning* untuk memprediksi konsentrasi polutan dalam kualitas udara menggunakan metode ResNet dan LSTM.

1.6 Tahapan Penelitian

Tahapan-tahapan penelitian yang akan dilakukan dijabarkan sebagai berikut:

1.6.1 Metodologi Penelitian

Penelitian yang dilakukan menggunakan metode kuantitatif dengan menerapkan model ResNet-LSTM (Q. Zhang et al., 2020) untuk melakukan pengukuran evaluasi, sebagai berikut:

1. Identifikasi Masalah

   Tahapan pertama dari penelitian ini adalah mengidentifikasi masalah yang bertujuan untuk mendapatkan informasi sebagai langkah awal penelitian. Masalah yang diidentifikasi diperoleh melalui studi literatur dari jurnal, buku, dan penelitian terdahulu yang digunakan sebagai landasan dari penelitian yang dikerjakan.

2. *Data Understanding*

   Tahapan selanjutnya adalah *data understanding* yang terdiri dari pengumpulan data dan pemahaman data. Data yang digunakan pada penelitian adalah data primer-sekunder yang bersifat publik dan diambil dari beberapa sumber dengan total 27.400 data. Data tersebut merupakan data historis lima zat polutan dari lima stasiun pemantauan kualitas udara dan empat data meteorologi dari lima stasiun meteorologi di DKI Jakarta dari tahun 2019 – 2021.

3. *Data Preprocessing*

   Tahapan selanjutnya adalah melakukan *data preprocessing* pada data yang telah diperoleh, yaitu dengan menghapus data yang tidak digunakan dan mengisi data yang hilang menggunakan interpolasi linear. Kemudian, menerapkan *correlation coefficient Pearson* untuk menganalisis korelasi *spatiotemporal* antara data polutan dan meteorologi. Terakhir, data akan dibagi menjadi data *training* dan *testing*.

4. Pembangunan Model

   Setelah didapatkan data yang sesuai, proses selanjutnya adalah melakukan konfigurasi arsitektur model, pemilihan parameter, dan pembangunan struktur menggunakan metode ResNet yang mampu mengatasi *vanishing gradient* untuk mengekstrak fitur spasial. Kemudian, menambahkan metode LSTM untuk mengekstrak fitur *temporal*. Terakhir, menambahkan *fully-connected layer* untuk melakukan prediksi konsentrasi polutan dalam kualitas udara.

5. Pelatihan Model

   Model yang telah dibangun, kemudian di-*compile* menggunakan Adam *optimizer* dan dilatih dengan *dataset* yang telah disiapkan untuk validasi dan tes model.

6. Analisis dan Evaluasi Hasil

   Setelah seluruh tahapan telah dilakukan, langkah terakhir adalah mengevaluasi model menggunakan MAPE dan memvisualisasikan performa model untuk dianalisis apakah hasil yang didapat mampu menjawab masalah yang telah dirumuskan sebelumnya. Kemudian, akan dilakukan pula pengujian secara *real-time* menggunakan data baru. Hasil analisis dan evaluasi yang didapatkan pada tahapan ini akan digunakan sebagai acuan dalam penarikan kesimpulan mengenai kemampuan model melakukan prediksi konsentrasi polutan pada kualitas udara menggunakan metode ResNet-LSTM.

1.6.2 Metodologi Pengembangan Sistem

Pengembangan sistem menggunakan metode *Waterfall* (Pressman, 2015) dengan detail tahapan sebagai berikut

a. Requirements

Tahapan pertama adalah mengumpulkan informasi mengenai kebutuhan sistem perangkat lunak melalui observasi. Informasi kemudian diolah dan

dianalisis untuk mendapatkan data yang lengkap mengenai spesifikasi sistem perangkat lunak yang akan dikembangkan.

b. *Design*

Tahapan selanjutnya adalah melakukan perancangan arsitektur, proses, dan desain sistem berdasarkan kebutuhan yang telah diidentifikasi. Rancangan ini dilakukan untuk memberikan gambaran lengkap mengenai apa yang harus dikerjakan, menyiapkan kebutuhan perangkat keras, dan mendefinisikan arsitektur perangkat lunak secara keseluruhan.

c. *Development*

Pada tahap ini, sistem dikembangkan melalui pembagian menjadi unit-unit kecil dengan menggunakan bahasa pemrograman. Setiap unit akan dilakukan tahap pengujian dan pemeriksaan fungsionalitas.

d. *Testing*

Setiap unit yang dikembangkan akan diintegrasikan menjadi satu kesatuan dan dilakukan pemeriksaan serta pengujian sistem secara keseluruhan untuk memastikan sistem telah memenuhi kriteria dan mengidentifikasi adanya kegagalan. Sistem diuji menggunakan metode pengujian *black box*.

e. *Maintenance*

Pada tahap ini, sistem sudah dapat dioperasikan oleh pengguna dan dilakukan pemeliharaan untuk memperbaiki kesalahan yang tidak terdeteksi dalam tahap pembuatan. Selain itu, juga dilakukan pengembangan unit sistem, serta peningkatan dan penyesuaian sistem dengan kebutuhan pengguna berdasarkan *feedback* yang diberikan.

1.7 Sistematika Penulisan

Sistematika penulisan dalam penyusunan laporan penelitian ini terbagi menjadi lima bab, yaitu:

BAB I PENDAHULUAN

Menjelaskan latar belakang penelitian, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, tahapan penelitian, dan sistematika penulisan.

BAB II TINJAUAN PUSTAKA

Menjelaskan studi pustaka mengenai data polutan dan meteorologi, metode ResNet dan LSTM, interpolasi linear untuk mengisi data yang hilang, *coefficient correlation Pearson* untuk menentukan korelasi data spatiotemporal, dan metode evaluasi model MAPE yang mendasari penelitian secara terperinci serta memuat landasan teori yang akan dibahas pada penelitian, hasil penelitian sebelumnya, dan *gap research* yang akan dilakukan.

BAB III METODOLOGI PENELITIAN DAN PENGEMBANGAN SISTEM

Menjelaskan metode ResNet-LSTM yang digunakan dalam pembangunan model untuk menyelesaikan permasalahan *vanishing gradient* yang diangkat

dalam penelitian ini. Proses utama dalam sistem yang dibentuk adalah *preprocessing* yang terdiri dari penghapusan data tidak digunakan, interpolasi linear, analisis korelasi *spatiotemporal*, *splitting* data, normalisasi, dan *reshaping* array, ekstraksi fitur spasial menggunakan ResNet, ekstraksi fitur temporal menggunakan LSTM, prediksi konsentrasi polutan, dan evaluasi model.

BAB IV HASIL DAN PEMBAHASAN

Menjelaskan analisis dan pembahasan dari hasil evaluasi dari sistem prediksi konsentrasi polutan ResNet-LSTM yang telah dibangun. Hasil penelitian mencakup tampilan program dan hasil prediksi konsentrasi polutan pada kualitas udara yang dibuat berdasarkan rancangan pada metodologi penelitian.

BAB V KESIMPULAN DAN SARAN

Menjelaskan kesimpulan yang didapat dari hasil penelitian ini dan memberikan saran yang dapat digunakan sebagai acuan untuk pengembangan penelitian selanjutnya.

BAB II
TINJAUAN LITERATUR

2.1 Landasan Teori

2.1.1 Zat Polutan

Polutan adalah gas yang berkontribusi pada pencemaran atau polusi udara, yang dapat menyebabkan berbagai jenis penyakit mulai dari gangguan pernapasan hingga kerusakan organ dalam (Ashshiddiqi et al., 2013). Zat polutan berperan dalam menentukan kualitas udara dan membentuk indeks kualitas udara. Indeks kualitas udara merupakan suatu parameter tanpa satuan yang memberikan informasi mengenai gambaran kondisi kualitas udara ambien di lokasi dan waktu tertentu kepada masyarakat. Indeks kualitas udara yang resmi digunakan di Indonesia berdasarkan Keputusan Menteri Negara Lingkungan Hidup Nomor 45 Tahun 1997 adalah Indeks Standar Pencemar Udara (ISPU) (Eko H., 2017). Penilaian ISPU didasarkan pada dampak yang dapat ditimbulkan terhadap kesehatan manusia, nilai estetika lingkungan, dan makhluk hidup lainnya. ISPU dapat dijadikan sebagai sistem peringatan dini (*early warning system*) bagi masyarakat yang tinggal di daerah rawan terdampak kebakaran hutan dan lahan. ISPU juga digunakan sebagai bahan pertimbangan untuk merumuskan upaya-upaya pengendalian pencemaran udara, baik oleh pemerintah pusat maupun daerah.

Terdapat tujuh parameter polutan pencemar udara yang digunakan dalam perhitungan ISPU dan didasari pada besarnya risiko terhadap kesehatan manusia, yaitu PM_{10}, $PM_{2.5}$, NO_2, SO_2, CO, O_3, dan HC (Chaniago et al., 2020b). PM_{10} merupakan partikel udara berukuran lebih kecil dari 10 mikron (mikrometer). Nilai Ambang Batas (NAB) adalah tingkat konsentrasi polusi udara yang diperbolehkan dalam udara ambien. NAB konsentrasi PM_{10} adalah 150 $\mu g/m^3$ (Hernawati et al., 2020). $PM_{2.5}$ merupakan partikel udara berukuran lebih kecil dari 2.5 mikron (mikrometer). NAB konsentrasi $PM_{2.5}$ adalah 65 $\mu gram/m^3$ (Mufadhol, 2022). Nitrogen dioksida (NO_2) merupakan gas reaktif yang dihasilkan dari pembakaran bahan bakar fosil, seperti kendaraan dan industri. NAB NO_2 adalah 40 $\mu g/m^3$.

Sulfur dioksida (SO_2) merupakan gas polutan yang dihasilkan dari pembakaran bahan bakar fosil berisi sulfur, seperti batu bara dan minyak bumi. NAB SO_2 adalah 40 $\mu g/m^3$ (Wiharja, 2002). Karbon Monoksida merupakan gas polutan yang dihasilkan dari pembakaran tidak sempurna bahan bakar fosil, seperti minyak, gas dan batu bara. Gas ini dapat terkumpul di daerah dengan polusi udara tinggi. Di Eropa, NAB CO adalah 1 mg/m^3. Ozon (O_3) terbentuk oleh reaksi kimia antara nitrogen dioksida (NOx) dan senyawa organik yang terbawa oleh sinar matahari. Seperti CO, zat ini memiliki batas yang berbeda berdasarkan negara. Di Eropa, NAB O_3 adalah 120 O_3 $\mu g/m^3$ (Higienis, 2023). Terakhir, Hidrokarbon (HC) merupakan polutan yang terdiri dari senyawa organik dari

pembakaran bahan bakar fosil, seperti minyak, gas, dan batu bara. NAB HC di Eropa adalah 50 mg/m^3(Arif, 2019). Semua zat polutan ini berdampak negatif pada kesehatan, terutama masalah pernapasan seperti radang paru-paru, ISPA (Infeksi Saluran Pernapasan Atas), dan gangguan pada sistem kardiovaskuler, keracunan, dan bahkan kematian.

ISPU dihitung berdasarkan pada nilai-nilai ISPU batas atas dan bawah, konsentrasi ambien batas atas dan bawah, serta konsentrasi ambien dari hasil pengukuran. Persamaan perhitungan ISPU sebagai berikut:

$$I = \frac{I_a - I_b}{X_a - X_b}(X_x - X_b) + I_b \quad \text{.....(2.1)}$$

Keterangan:
I = ISPU terhitung
I_a = ISPU batas atas
I_b = ISPU batas bawah
X_a = Konsentrasi ambien batas atas ($\mu g/m^3$)
X_b = Konsentrasi ambien batas bawah ($\mu g/m^3$)
X_x = Konsentrasi ambien nyata hasil pengukuran ($\mu g/m^3$)

Nilai batas atas ISPU paling tinggi adalah lebih dari 300, sedangkan nilai batas bawah ISPU paling rendah adalah 50. Konsentrasi ambien batas atas dan bawah berbeda pada tiap parameter dan diperoleh dari tabel konversi nilai konsentrasi parameter ISPU. Nilai konsentrasi ambien yang sebenarnya diperoleh dari rata-rata konsentrasi ambien selama 24 jam pengukuran (Firman et al., 2023). Tabel konversi nilai konsentrasi parameter ISPU dapat dilihat pada **Gambar 2.1** sebagai berikut:

ISPU	24 Jam PM10 (µg/m^3)	24 Jam PM2.5 (µg/m^3)	24 Jam SO_2 (µg/m^3)	24 Jam CO (µg/m^3)	24 Jam O_3 (µg/m^3)	24 Jam NO_2 (µg/m^3)	24 Jam HC (µg/m^3)
0 - 50	50	15,5	52	4000	120	80	45
51 - 100	150	55,4	180	8000	235	200	100
101 - 200	350	150,4	400	15000	400	1130	215
201 - 300	420	250,4	800	30000	800	2260	432
>300	500	500	1200	45000	1000	3000	648

Keterangan:
- Data pengukuran selama 24 jam secara terus-menerus.
- Hasil perhitungan ISPU parameter partikulat (PM2.5) disampaikan tiap jam selama 24 jam.
- Hasil perhitungan ISPU parameter partikulat (PM10), sulfur dioksida (SO_2), karbon monoksida (CO), ozon (O_3), nitrogen dioksida (NO_2) dan hidrokarbon (HC), diambil nilai ISPU parameter tertinggi dan paling sedikit disampaikan setiap jam 09.00 dan jam 15.00.

Gambar 2.1. Tabel Konversi Nilai Konsentrasi Parameter ISPU

Berdasarkan Kementerian Lingkungan Hidup dan Kehutanan, ISPU terbagi menjadi lima kategori sesuai rentang nilainya (Wibawana, 2023), yaitu:

1. Kategori Baik (0 – 50)

   ISPU dengan rentang nilai 0 – 50 memiliki kategori Baik. Tingkat kualitas udara masih sangat baik dan tidak memberikan efek negatif terhadap kesehatan manusia dan hewan, serta tidak berpengaruh terhadap tumbuhan, bangunan, dan nilai estetika.
2. Kategori Sedang (51 – 100)

   ISPU dengan rentang nilai 51 – 100 memiliki kategori Sedang. Tingkat kualitas udara masih dapat diterima oleh kesehatan manusia dan hewan, tetapi sedikit berpengaruh terhadap tumbuhan yang sensitif dan nilai estetika.
3. Kategori Tidak Sehat (101 – 200)

   ISPU dengan rentang nilai 101 – 200 memiliki kategori Tidak Sehat. Tingkat kualitas udara bersifat merugikan pada manusia maupun hewan atau dapat menimbulkan kerusakan pada tumbuhan dan nilai estetika.
4. Kategori Sangat Tidak Sehat (201 – 300)

   ISPU dengan rentang nilai 201 – 300 memiliki kategori Sangat Tidak Sehat. Tingkat kualitas udara dapat menyebabkan kerugian dan meningkatkan risiko kesehatan pada beberapa kelompok populasi yang terpapar.
5. Kategori Berbahaya (300+)

   ISPU dengan rentang nilai lebih dari 300 memiliki kategori Berbahaya. Tingkat kualitas udara berbahaya dan secara umum dapat menyebabkan kerugian kesehatan serius pada populasi dan memerlukan penanganan dengan cepat.

Data ISPU diperoleh dari pengoperasian Stasiun Pemantauan Kualitas Udara Ambien dan disampaikan kepada masyarakat setiap 24 jam dari data sebelumnya dan berlaku 24 jam ke depan. Waktu pengambilan data terakhir dilakukan pada pukul 15.00 WIB. ISPU dapat diakses melalui aplikasi ISPU Net yang memungkinkan masyarakat mengetahui kondisi kualitas udara secara *real-time* di seluruh wilayah (Ramadhan P, 2021).

2.1.2 Meteorologi

Meteorologi menurut Bayong Tjasjono (1999) adalah ilmu yang mengkaji fenomena cuaca dan iklim, serta proses fisik yang terjadi dalam atmosfer (Tjasjono, 1999). Ilmu meteorologi sangat bergantung pada observasi atau pengamatan untuk memperoleh data dari faktor-faktor yang berpengaruh terhadap perubahan cuaca. Beberapa faktor meteorologi yang diamati, yaitu suhu, tekanan, angin, kelembaban, hujan dan sebagainya. Faktor meteorologi ini berpengaruh terhadap sebaran polutan di udara (B. Zhang et al., 2022).

Suhu atau temperatur udara merupakan hasil perhitungan energi kinetik rata-rata dari gerakan molekul-molekul yang diukur berdasarkan skala tertentu

(Ackerman & Knox, 2011; Kartasapoetra, 2004). Suhu udara diukur menggunakan alat termometer dengan satuan derajat celcius (ºC). Beberapa negara lain menetapkan suhu dengan satuan derajat Fahrenheit (ºF) (Suryanto & Luthfian, 2019). Suhu dekat permukaan dipengaruhi oleh penerimaan dan kehilangan energi di permukaan tanah. Tanah memperoleh energi dari matahari selama siang hari dan memancarkan radiasi gelombang panjang baik siang maupun malam. Ketidakseimbangan antara penerimaan dan kehilangan menyebabkan inversi suhu udara. Umumnya, suhu udara tertinggi terjadi pada pukul 13.00–14.00, sedangkan suhu udara terendah terjadi pada pukul 04.00–05.00 (Ackerman & Knox, 2011). Inversi suhu dapat mempengaruhi polusi udara, di mana polutan tersebar di atmosfer. Ketika terjadi inversi, polutan akan bergumpal dan terjebak di lapisan udara yang lebih dingin, sehingga konsentrasi polutan semakin tinggi di permukaan bumi (Sari, 2015).

Tekanan merupakan merupakan hasil dari berat udara yang menekan suatu area tertentu di atas suatu titik yang merepresentasikan tekanan atmosfer di titik tersebut (Kartasapoetra, 2004). Tekanan udara diukur berdasarkan gaya tekan pada permukaan dengan luas tertentu dengan satuan atmosfer (atm) atau mmHg atau mbar. Tekanan udara diukur dengan alat barometer dengan ukuran 1atm = 760mmHG = 1.013mbar. Tekanan udara cenderung berkurang seiring meningkatnya ketinggian suatu tempat (elevasi atau *altitude*). Semakin tinggi suatu tempat dari permukaan laut, maka tekanan udara semakin rendah, begitu pun sebaliknya. Umumnya, tekanan udara berkurang sebesar 11mbar setiap kenaikan ketinggian tempat sebesar 100m (Lakitan, 2002). Perbedaan tekanan udara mengakibatkan terjadinya gerakan udara dari daerah bertekanan udara tinggi (maksimum) menuju daerah bertekanan udara rendah (minimum), yang disebut angin.

Angin merupakan perpindahan atau pergerakan massa udara dari satu tempat ke tempat lain secara horizontal (Kartasapoetra, 2004). Arah angin mengacu pada kompas dan memiliki 16 titik berasal dari di mana angin bertiup yang dinyatakan dengan huruf atau angka (U, UTL, TL, TTL, dan sebagainya) untuk angin permukaan. Angin terjadi akibat pergerakan udara yang disebabkan oleh perbedaan suhu, yang kemudian mengakibatkan perubahan tekanan. Kecepatan angin dinyatakan dengan satuan km/jam, mil/jam, m/det, dan knot, di mana 1km/jam = 0.621 mil/jam = 0.278 knot, 1knot = 1.852km/jam = 1.151 mil/jam – 0.514m/det (Linsley et al., 1996). Kecepatan angin bervariasi menurut jarak di atas permukaan tanah dan variasinya sangat cepat di dekat permukaan tanah.

Kelembaban merupakan banyaknya kadar uap air yang terkandung dalam udara (Kartasapoetra, 2004). Kelembaban dikenal dalam beberapa istilah, yaitu Kelembaban mutlak, Kelembaban spesifik, dan Kelembaban relatif. Kelembaban mutlak adalah massa uap air yang terdapat dalam satu unit volume udara, yang

diukur dalam gram/m^3. Kelembaban mutlak *a* dihitung dengan rumus sebagai berikut:

$$a = 217\frac{E}{T} gram/m^3 \quad (2.2)$$

Kelembaban spesifik didefinisikan sebagai jumlah gram uap air yang terkandung dalam 1kg udara alami. Kelembaban ini merupakan perbandingan massa uap air dengan satuan massa udara, yang dinyatakan dalam gram/kg. Kelembaban spesifik *q* dapat dihitung dari tekanan atmosfer *p* dan tekanan uap air *e* dengan rumus sebagai berikut:

$$q = \frac{623e}{p-0.377e} gram/kg \quad (2.3)$$

Kelembaban relatif merupakan perbandingan antara jumlah uap air yang sebenarnya terkandung dalam udara *e* dengan jumlah maksimum uap air yang dapat ditampung *E* pada suhu tertentu, dinyatakan dalam persen (%). Kelembaban relatif *R* dihitung dengan rumus sebagai berikut:

$$R = 100\frac{e}{E}\% \quad (2.4)$$

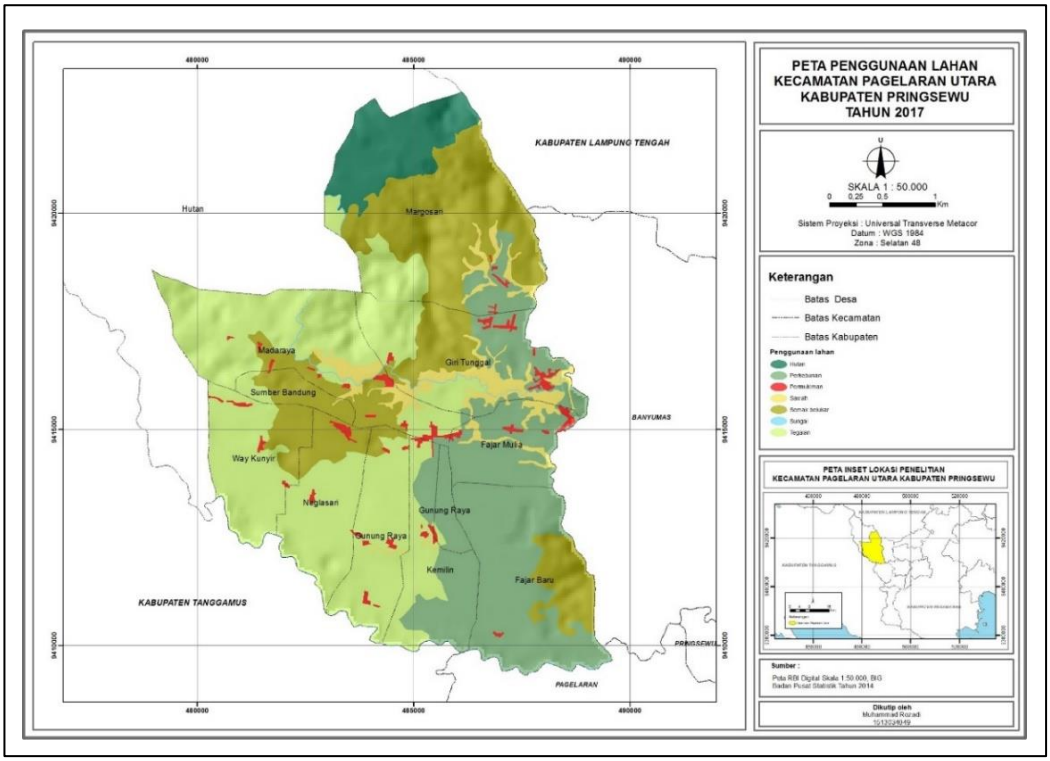
Kelembaban relatif berkisar dari 0 hingga 100%, di mana 0% mengindikasikan udara yang sangat kering, sementara 100% menunjukkan udara yang sudah jenuh dengan uap air, di mana kondensasi dapat terjadi. Kelembaban udara sering diukur dengan kelembaban nisbi, yang bisa berubah sesuai dengan lokasi dan waktu. Umumnya, kelembaban nisbi cenderung turun menjelang tengah hari, dan kemudian meningkat pada sore hari hingga menjelang pagi (Tjasjono, 2004).

Hujan merupakan peristiwa presipitasi di mana uap air yang terkandung dalam awan di atmosfer berubah menjadi cair dan jatuh ke bumi (Kartasapoetra, 2004). Diperlukan titik-titik kondensasi, amoniak, debu, dan asam belerang untuk dapat terjadi hujan. Partikel-partikel ini memiliki sifat menarik uap air, sehingga menyebabkan uap air dalam udara mengembun dan membentuk tetesan air yang kemudian turun sebagai hujan. Curah hujan diukur dalam inci atau milimeter (1inci = 25.4mm) (Tjasjono, 2004).

2.1.3 Data *Spatiotemporal*

Data *spatiotemporal* terdiri dari representasi spasial dan temporal yang mengacu pada data yang dikumpulkan dan dianalisis melintasi kedua dimensi ruang dan waktu (Hamdi et al., 2022). Data spasial merepresentasikan objek spasial yang terdiri dari titik, garis, daerah, poligon sederhana, volume, bahkan data berdimensi tinggi yang mencakup waktu. Atribut data spasial mencakup kota,

sungai, jalan raya, negara bagian, pegunungan, dan sebagainya. Contoh properti spasial mencakup luas sungai tertentu, batas wilayah, peta penggunaan lahan (Rozadi, 2019) yang ditunjukkan pada **Gambar 2.2**, atau informasi atribut non-spasial seperti ketinggian elevasi, nama kota, dan lainnya. Data spasial banyak digunakan dalam aplikasi pemantauan lingkungan, ruang, perencanaan kota, pengelolaan sumber daya, dan sistem informasi geografis (GIS) (Samet, 1995).

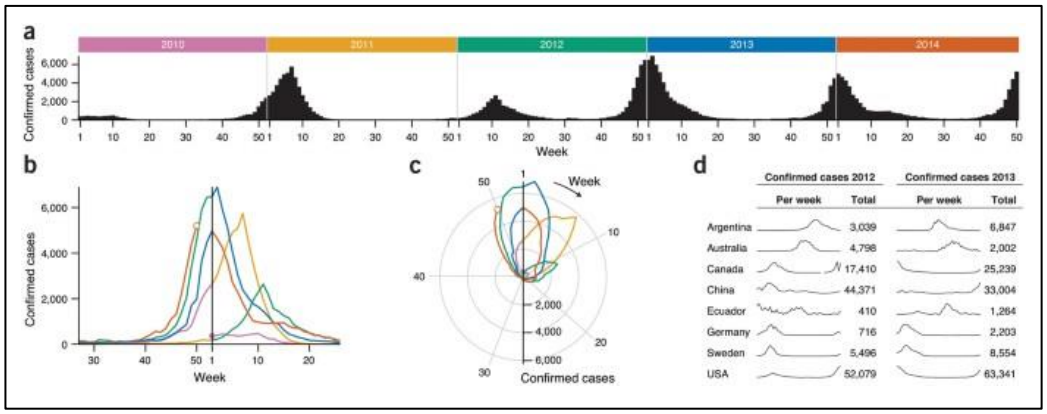


Gambar 2.2. Data Spasial: Peta Penggunaan Lahan Kecamatan Pagelaran Utara, Kabupaten Pringsewu Tahun 2017

Data temporal berkaitan dengan peristiwa yang diurutkan berdasarkan satu atau lebih dimensi waktu baik masa lalu, masa kini, maupun masa depan, seperti data Kasus Influenza yang Terkonfirmasi oleh WHO di Amerika Serikat Tahun 2010 – 2014 (World Health Organization, 2010) yang ditunjukkan pada **Gambar 2.3**. Data temporal dibedakan menjadi dua area secara luas, yaitu berkaitan dengan hubungan sebab akibat di antara peristiwa-peristiwa yang berorientasi pada waktu dan berkaitan dengan penemuan pola serupa dalam rangkaian waktu yang sama atau di antara rangkaian waktu yang berbeda, disebut analisis deret waktu (*time series*) (Roddick & Spiliopoulou, 1999). Aspek penelitian analisis *time series* meliputi pendekatan kurva dengan metode matematika, pengurangan *noise*, perbandingan deret waktu menggunakan teknik pencocokan pola dan prediksi menggunakan metode matematika atau *neural network* (Weigend, 2018).

Data *spatiotemporal* memiliki kompleksitas tinggi dan digunakan dalam berbagai bidang seperti meteorologi, ilmu lingkungan, kedokteran, transportasi,

dan lainnya (Amran et al., 2020). Metode *data mining* dapat digunakan untuk mengolah dan menganalisis data *spatiotemporal* untuk berbagai tujuan, seperti visualisasi pemetaan sebaran penyakit, analisis geografis, pengukuran nilai tanah, hingga prediksi konsentrasi polutan dan kualitas udara (Alizanovic, 2023; Asgari et al., 2022; Heldayani et al., 2021).



Gambar 2.3. Data Temporal: Kasus Influenza Terkonfirmasi di Amerika Serikat 2010 – 2014

2.1.4 *Machine Learning*

Machine Learning merupakan sebuah teknologi yang memungkinkan mesin untuk belajar dan membuat keputusan secara otomatis tanpa instruksi pengguna. *Machine Learning* melalui tahap pembelajaran menggunakan data pelatihan khusus untuk mengotomatiskan proses pembuatan model analitis, yang memungkinkan model untuk membuat keputusan yang tepat dan menyelesaikan tugas-tugas terkait (Janiesch et al., 2021). Terutama pada tugas yang berkaitan dengan data *high-dimensional*, seperti klasifikasi, *clustering*, dan regresi, *Machine Learning* menunjukkan penerapan model yang baik. Teknologi ini digunakan dalam berbagai bidang, seperti pendidikan, ekonomi, teknologi, sosial, dan lainnya. Sudah banyak penerapan algoritma *machine learning* yang telah dilakukan seperti deteksi penipuan (Gupta et al., 2022), pengenalan ucapan dan gambar (Jimenez-Mesa et al., 2023), pemrosesan bahasa alami (NLP), dan lainnya (Janiesch et al., 2021).

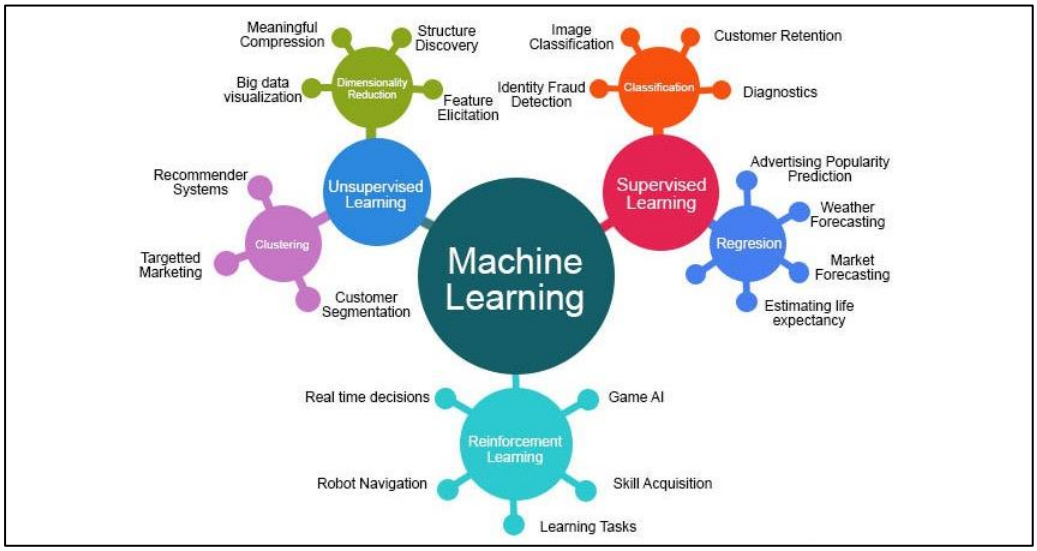
Terdapat tiga tipe utama *Machine Learning* yang umum digunakan seperti pada **Gambar 2.4**, yaitu *Supervised Learning*, *Unsupervised Learning*, dan *Reinforcement Learning*. *Supervised Learning* adalah tipe *Machine Learning* di mana algoritma dilatih menggunakan kumpulan data yang diberi label, dengan setiap titik data memiliki label yang sesuai (Mobarak et al., 2023). Hal ini menyiratkan bahwa beberapa data telah ditandai dengan jawaban yang benar. Tipe ini diterapkan untuk model klasifikasi dan regresi, seperti prediksi.

Prediksi merupakan salah satu metode untuk memperkirakan *output* di masa yang akan datang melalui proses pembelajaran atau pelatihan. Pembelajaran ini

dilakukan berdasarkan data *input* dari masa lalu dan masa kini yang telah diberikan dengan tujuan untuk meminimalkan kesalahan yang terjadi (Dewi et al., 2022). Data prediksi dapat berasal dari berbagai sumber, seperti data historis, data *spatio-temporal*, atau data yang terus-menerus diperbarui (Hartatik et al., 2023). Data ini kemudian digunakan untuk melatih model agar dapat mempelajari pola antara *input* dan *output*, sehingga dapat melakukan prediksi terhadap data baru. Beberapa algoritma yang umum digunakan dalam *Supervised Learning*, yaitu *Support Vector Machine* (SVM), Naïve Bayes, K-NN, *Random Forest*, *Linear Regression*, De*cision Tree*, dan lainnya (Allenbrand, 2023).

Berbeda dengan *Supervised Learning*, *Unsupervised Learning* tidak memerlukan data pelatihan yang diberi label. Tipe ini menganalisis serta mempelajari struktur dan pola secara ekslusif dari data yang tidak berlabel dan sudah ada sebelumnya, kemudian mengungkap informasi tersembunyi dari data yang diberikan (Mobarak et al., 2023). Tipe ini diterapkan untuk model *clustering*, deteksi anomali, dan reduksi dimensi (Zipfel et al., 2023). Berapa algoritma yang digunakan dalam *Unsupervised Learning*, yaitu K-Means, Fuzzy, C-Means, dan lainnya.

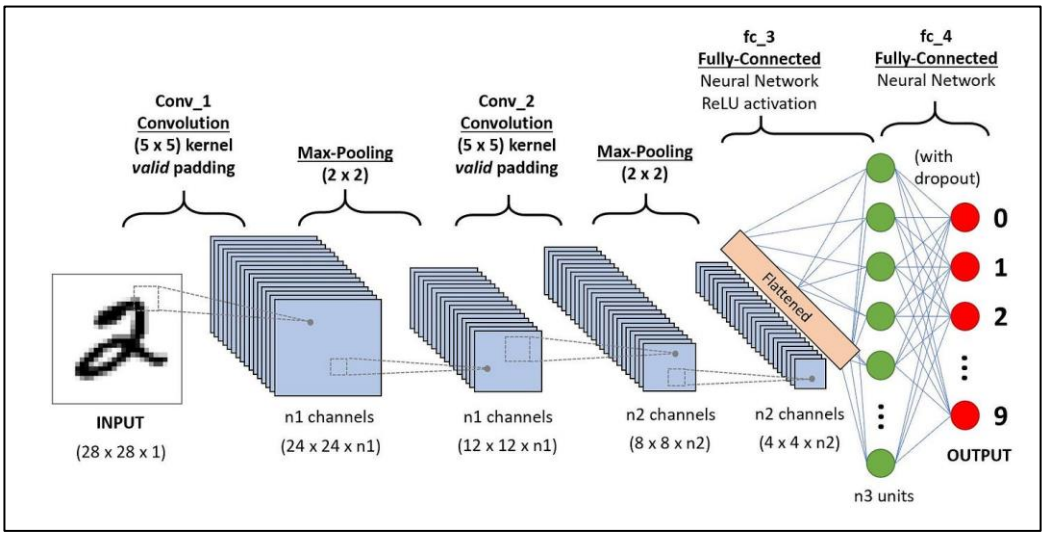
Reinforcement Learning adalah tipe *Machine Learning* di mana agen belajar berinteraksi dengan lingkungannya melalui *trial and error*, lalu menerima umpan balik dalam bentuk *reward* atau *punishment*. Umpan balik digunakan untuk menyesuaikan perilaku agen dan meningkatkan kinerjanya seiring waktu, Tujuan dari pembelajaran ini adalah mempelajari bagaimana mengambil tindakan untuk memaksimalkan *reward* yang didapatkan (Pichka, 2023). *Reinforcement Learning* telah diterapkan dalam berbagai bidang, seperti robotik, NLP, *game*, dan lainnya (Mandlekar et al., 2019; H. Wang, 2021).



Gambar 2.4. Tipe dan Algoritma *Machine Learning*

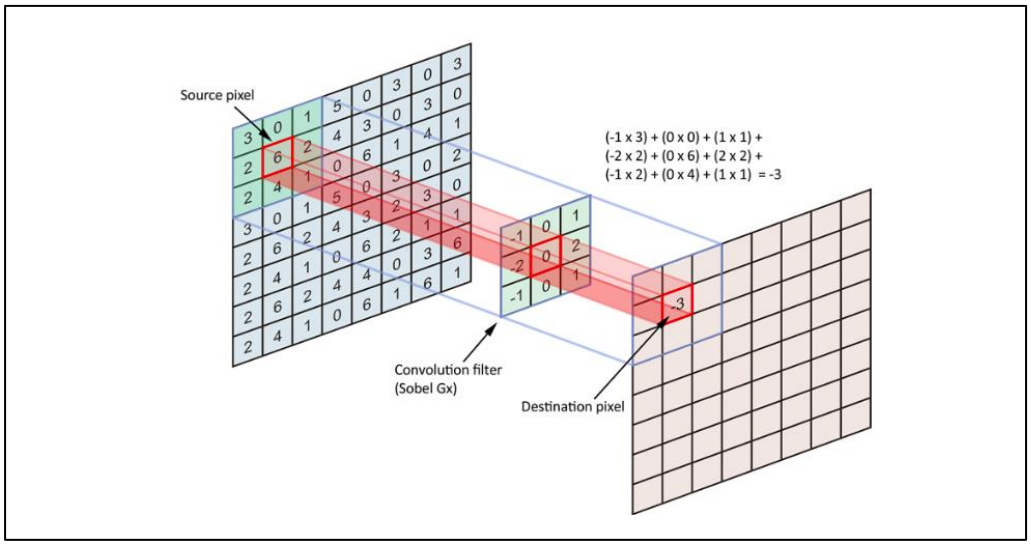
2.1.5 *Convolutional Neural Network* (CNN)

Convolutional Neural Network (CNN) merupakan salah satu algoritma *deep learning* yang termasuk dalam jaringan *feed-forward* di mana informasi mengalir searah dari *input* ke *output*. CNN dirancang untuk tugas pengolahan citra baik gambar, ucapan, atau sinyal audio, seperti klasifikasi, deteksi, segmentasi, dan prediksi (Keita, 2023). Pada beberapa penelitian, CNN juga digunakan pada data *time series* (Kim et al., 2023) dan *spatio-temporal* (L. Zhang et al., 2020). CNN terdiri dari lapisan-lapisan utama, yaitu *convolutional layer*, *ReLU layer*, *pooling layer*, dan *fully connected layer.* Arsitektur CNN secara umum (Keita, 2023) dapat dilihat pada **Gambar 2.5**.



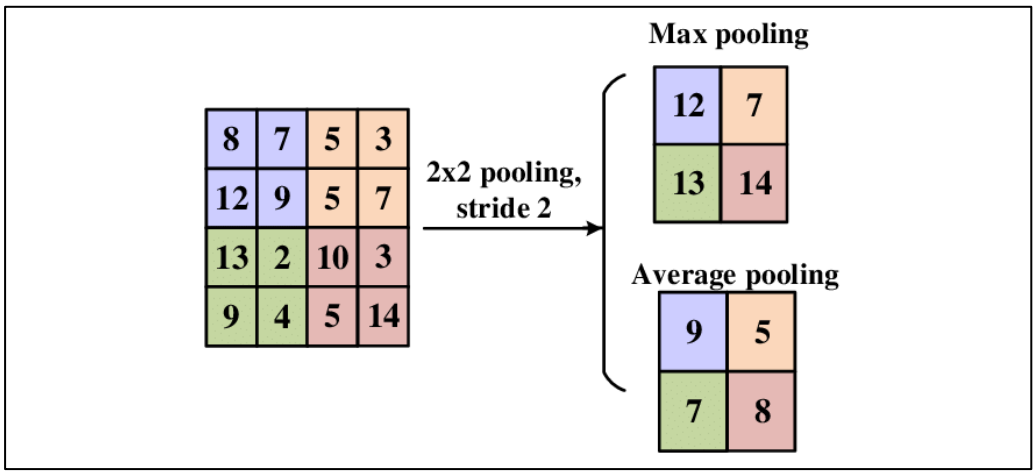
Gambar 2.5. Arsitektur *Convolutional Neural Network* (CNN)

Input pada *convolutional layer* adalah sebuah gambar dengan ukuran $m \; x \; m \; x \; r$ di mana m adalah tinggi dan lebar pada gambar dan r adalah jumlah saluran atau kedalaman (sesuai dengan RGB dalam gambar, jika hitam putih = 1 dan berwarna = 3). Pada *convolutional layer*, juga terdapat pendeteksi fitur yang dikenal sebagai kernel atau filter dengan ukuran $n \; x \; n \; x \; q$ di mana n lebih kecil dibandingkan dimensi gambar dan q dapat sama dengan jumlah saluran r atau lebih kecil dan mungkin berbeda pada tiap kernel. Kernel akan memeriksa apakah fitur tersebut ada pada gambar dengan melintasi bidang reseptif gambar, yang disebut dengan konvolusi. Operasi konvolusi dilakukan berulang kali hingga kernel mencakup seluruh bagian gambar dengan ukuran filter berupa matriks 3x3. Hasil proses konvolusi disebut *feature map* yang dapat dilihat pada **Gambar 2.6**.



Gambar 2.6. Proses Konvolusi pada *Convolutional Layer*

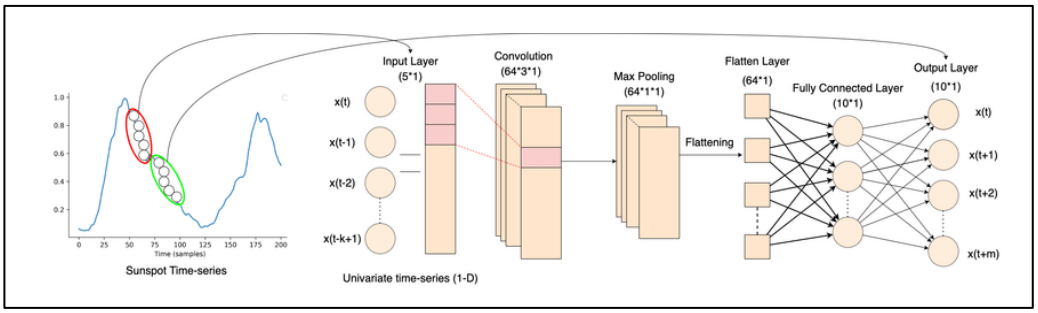
Pooling layer bertujuan untuk mengurangi jumlah parameter (*spatial resolution*) pada *feature map* yang disebut operasi *spatial pooling*. Pada *layer* ini, kernel mengisi *output array* dengan menerapkan fungsi agregasi dalam bidang reseptif. Terdapat dua jenis *spatial pooling*, yaitu *average pooling* dan *max pooling*. Perbedaan kedua jenis *pooling* ini ditunjukan pada **Gambar 2.7**. Matriks dari *feature map* dibagi-bagi dengan ukuran $m \; x \; m$. Pada *average pooling*, filter menghitung nilai rata-rata untuk dikirim ke *output array*. Sedangkan pada *max pooling*, filter memilih nilai maksimum dari tiap bagian untuk dikirim ke *output array*. Ekstraksi fitur ini mengurangi risiko *overfitting* dan kompleksitas, serta meningkatkan efisiensi model.



Gambar 2.7. Proses *Spatial Pooling* pada *Pooling Layer*

Pada *Fully Connected Layer*, setiap node pada *output layer* terhubung langsung ke node sebelumnya dengan *input* matriks satu dimensi. CNN menerapkan fungsi aktivasi ReLU (*Rectified Linear Unit)* untuk non-linieritas

pada model. Lapisan ini melakukan tugas seperti klasifikasi dan regresi berdasarkan fitur yang telah diekstrak sebelumnya menggunakan *softmax*. Selain data citra, CNN juga digunakan untuk melakukan klasifikasi pada data *time series* (Lara-Benítez et al., 2020; Zheng et al., 2014) dengan arsitektur yang ditunjukkan pada **Gambar 2.8**.



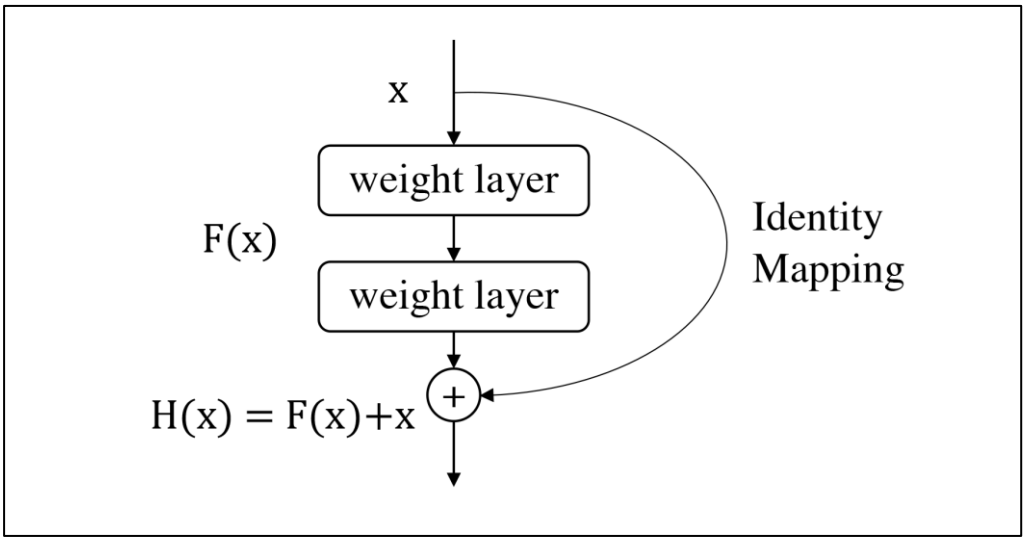
Gambar 2.8. Arsitektur *Convolutional Neural Network* untuk *Time Series*

Input layer berisi masukan data *time series* berukuran $N\ x\ k$, di mana k adalah jumlah data yang menjadi masukan dan N adalah panjang data. *Convolutional layer* berisi filter berukuran $m\ x\ l\ x\ k$, di mana m adalah jumlah filter dan l, k adalah ukuran filter dengan l merupakan panjang filter dan k merupakan jumlah data dari *input layer*. Nilai m dan l adalah parameter yang dinamis.

Pooling layer sama seperti arsitektur CNN secara umum, yaitu *max pooling* dan *average pooling*. *Flatten layer* merupakan lapisan di mana hasil operasi konvolusi dan *spatial pooling* disambung menjadi sebuah *time-series* baru dan menjadi input untuk *fully connected layer*. *Output layer* merupakan lapisan untuk melakukan tugas klasifikasi *time series* yang memiliki n buah neuron, di mana n adalah jumlah kelas untuk klasifikasi terkait.

2.1.6 *Residual Network* (ResNet)

Metode CNN mampu mengekstrak fitur spasial dari gambar lebih baik dengan menumpuk lebih banyak lapisan jaringan. Namun, penelitian sebelumnya menemukan bahwa sulit untuk melatih model CNN yang mendalam, karena semakin dalam tingkat jaringan, maka akurasi pelatihan menjadi semakin jenuh dan mulai menurun dengan cepat yang dikenal dengan masalah degradasi jaringan dan *vanishing gradient*. Untuk mengatasi hal ini, ResNet diusulkan untuk mengoptimalkan pelatihan model CNN yang dalam (Basodi et al., 2020; He et al., 2016). Metode ResNet mengatasi masalah degradasi dan gradien hilang dengan menambahkan pemetaan identitas, seperti yang ditunjukkan oleh garis lengkung pada **Gambar 2.9**.

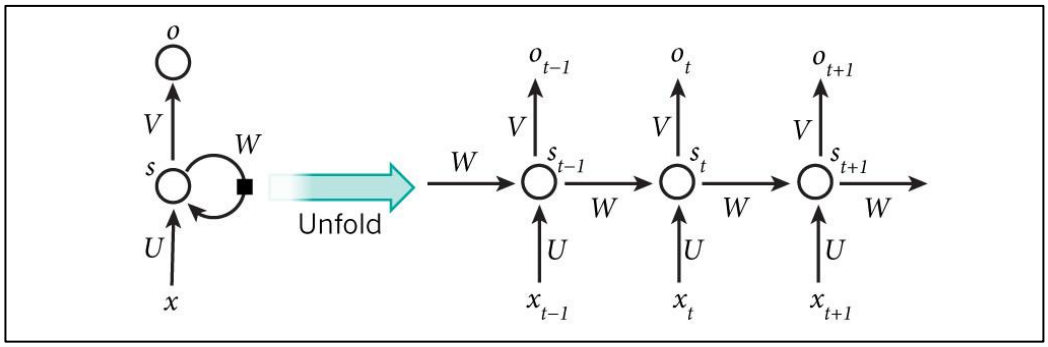


Gambar 2.9. Blok Residual

Di mana x mewakili *input* ke lapisan pertama dan $H(x)$ mewakili fungsi pemetaan yang diinginkan dari x ke *output* dari *stacked layers*. Selama pelatihan model, masalah mempelajari fungsi pemetaan yang diinginkan $H(x)$ dapat dirumuskan ulang menjadi mempelajari fungsi pemetaan sisa $F(x)$, di mana $F(x) = H(x) - x$. Mengingat bahwa fungsi pemetaan asli menjadi $F(x) + x$, bobot *stacked layers* yang sesuai dengan $F(x)$ bisa menjadi nol, untuk meniru *shortcut connection* antar lapisan. Seperti yang ditunjukkan pada (He et al., 2016; Ridhovan & Suharso, 2022), dengan memanfaatkan ResNet, model CNN dapat dioptimalkan bahkan ketika jumlah lapisan ditingkatkan dan mampu mengurangi risiko *overfitting*.

2.1.7 *Long Short-Term Memory* (LSTM)

Long Short-Term Memory (LSTM) merupakan turunan dari salah satu metode *deep learning*, yaitu *Recurrent Neural Network* (RNN). RNN adalah model jaringan yang berevolusi dari persepsi multi-layer dan memiliki struktur komposisi yang lebih kompleks dibandingkan *Artificial Neural Network* (ANN). RNN mampu "menghafal" informasi data historis yang dipelajari melalui *hidden layer* dan menerapkannya pada perhitungan keluaran saat ini untuk mencapai efek pemrosesan dan pemodelan sekuensial. Model RNN standar ditunjukkan pada **Gambar 2.10**.



Gambar 2.10. Struktur *Recurrent Neural Network* (RNN)

Keadaan S_t dari *hidden layer* pada waktu t dan keluaran y_t dari *output layer* pada waktu t dengan $W = W_x$, $U = W_h$ dan $V = W_C$ (LeCun et al., 2015) dapat dituliskan sebagai berikut:

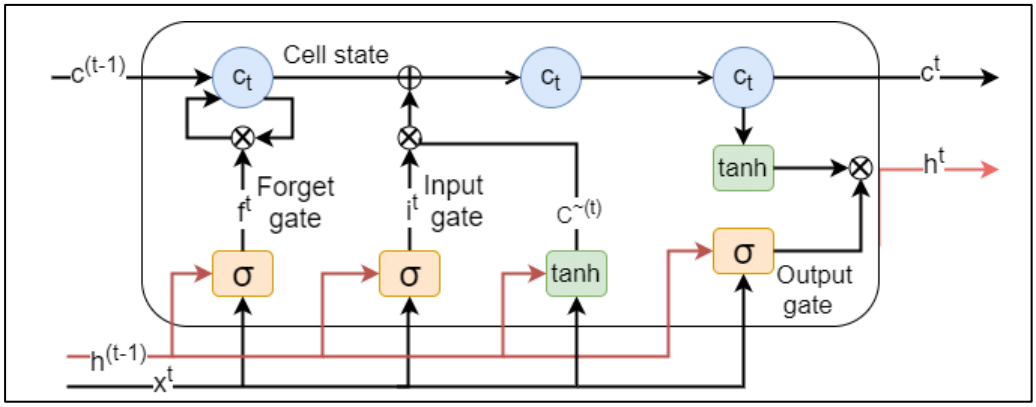
$$S_t = \tanh(Ux_t + Ws_{t-1}), y_t = \sigma(Vs_t) \quad (2.5)$$

Di mana, fungsi aktivasi s mewakili fungsi sigmoid dan U, V, dan W adalah matriks bobot jaringan. Parameter berbobot U, V, dan W diinisialisasi secara acak dan dipelajari menggunakan algortima *backpropagation* dengan penurunan gradien stokastik *(stochastic gradient descent)*. Namun, masalah *vanishing gradient* pada saat perhitungan *backpropagation* membuat pengaruh input tidak dapat ditransfer lebih jauh (Bengio et al., 1994).

Oleh karena itu, LSTM dirancang untuk meningkatkan kemampuan RNN dengan struktur *hidden layer* yang lebih kompleks (Hochreiter & Schmidhuber, 1997). Metode ini memiliki peningkatan besar dalam kemampuan pemodelan deret waktu. LSTM tidak hanya diterapkan untuk *speech recognition* (J. Wang et al., 2020), *language processing* (Onan & Toçoğlu, 2021), dan *machine translation* (Ren, 2020), tetapi juga memproses dan memprediksi peristiwa deret waktu dengan interval waktu yang panjang (Cheng et al., 2022; Portal-Porras et al., 2023; Qin et al., 2019). Metode RNN menghitung jumlah bobot sinyal input hanya dengan menerima *hidden state* dari langkah waktu sebelumnya, sedangkan, LSTM mengirimkan sel memori dan *hidden state* sesuai dengan langkah waktu, seperti yang ditunjukkan pada **Gambar 2.11**.

LSTM menggunakan metode pembelajaran *gradient descent*, di mana informasi penting di masa lalu dipertahankan melalui *feed-forward (forward propagation)*. Setelah nilai keluaran dibandingkan dengan nilai yang diharapkan dan diperoleh kesalahan entropi silang (*cross entropy*), maka digunakan algoritma *backpropagation* untuk memberikan umpan balik dan bergerak lapisan demi lapisan, menghitung bobot gradien tiap lapisan dan memodifikasi bobot neuron siklik. Melalui tahapan ini, bobot koneksi antar neuron yang berbeda di jaringan saraf akhir disesuaikan hingga optimal (Lu et al., 2021). Oleh karena itu, LSTM

memiliki keuntungan dalam memecahkan masalah ketergantungan jangka panjang.



Gambar 2.11. Struktur *Long Short-Term Memory* (LSTM)

Forget gate f_t, *input gate* i_t, kandidat *cell state* $\tilde{c}_t$, *memory cell state* c_t, *output gate* o_t, dan *hidden state* h_t pada **Gambar 2.11** (Chung et al., 2014) dituliskan sebagai berikut:

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf} \circ c_{t-1} + b_f) \quad (2.6)$$
$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci} \circ c_{t-1} + b_i) \quad (2.7)$$
$$\tilde{c}_t = \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_f) \quad (2.8)$$
$$c_t = f_t \circ c_{t-1} + i_t \circ \tilde{c}_t \quad (2.9)$$
$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co} \circ c_t + b_o) \quad (2.10)$$
$$h_t = o_t \circ \tanh(c_t) \quad (2.11)$$

2.1.8 Interpolasi Linear

Data *time series* dicatat dan dianalisis untuk memahami fenomena dan/atau perilaku variabel, yang kemudian dilakukan prediksi nilai masa depan, dan sebagainya. Sayangnya, pada beberapa data terdapat kesenjangan atau nilai hilang karena tahapan waktu pencatatan yang tidak teratur atau penghapusan titik data yang perlu diisi untuk analisis data, kalibrasi model, atau data dengan langkah waktu reguler (Lepot et al., 2017). Metode ini dapat digunakan untuk mengisi nilai yang hilang dalam data *time series* dengan garis lurus antara dua titik terdekat x_A dan x_B. Interpolasi linear memperkirakan nilai pada lokasi yang tidak diukur berdasarkan nilai yang diukur di sekitarnya. Berbagai persamaan ekuivalen untuk metode ini seperti (2.12) diberikan dalam (Gnauck, 2004) dan (2.13) diberikan dalam (Schlegel et al., 2012).

$$\tilde{x}(t) = \frac{x_{k+1}-x_k}{t_{k+1}-t_k}(t - t_k) + x_k \quad (2.12)$$

$$\tilde{x}(t) = (1 - \alpha)x_{k+1} + \alpha x_k \quad (2.13)$$

Di mana, α adalah faktor interpolasi yang bervariasi dari 0 hingga 1. Metode ini dianggap mudah digunakan dan efisien untuk memprediksi nilai yang hilang dengan laju konstan daripada metode interpolasi non-linear (Gnauck, 2004).

2.1.9 *Pearson Coefficient Correlation*

Penentuan korelasi *spatiotemporal* antara data zat polutan dan meteorologi menggunakan analisa korelasi sederhana, yaitu *Pearson coefficient correlation.* Nilai korelasi r memerlukan besaran dan arah positif atau negatif yang memiliki rentang nilai dari -1 hingga 0 hingga +1 yang nilainya absolut dan non-dimensi tanpa satuan. Koefisien korelasi 0 menunjukkan bahwa tidak ada hubungan antara variabel yang diukur. Semakin dekat nilai koefisien dengan ±1, semakin kuat hubungan linear antara kedua variabel. Koefisien korelasi positif menunjukkan bahwa peningkatan variabel pertama akan sebanding dengan peningkatan variabel kedua, sehingga menunjukkan hubungan searah antar variabel. Korelasi negatif menunjukkan hubungan terbalik, di mana satu variabel meningkat, sedangkan variabel kedua menurun (Taylor, 1990). *Pearson coefficient correlation* r_{xy} dihitung dengan uji parametrik yang memerlukan distribusi normal kontinu (Li et al., 2023) yang dituliskan sebagai berikut:

$$r_{xy} = \frac{\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i-\bar{y})^2}} \quad (2.14)$$

Di mana, x_i dan y_i adalah nilai dari variabel pertama dan kedua pada observasi ke-i, $\bar{x}$ dan $\bar{y}$ adalah rata-rata dari variabel pertama dan kedua, serta n adalah jumlah observasi.

2.1.10 *Z-Score Normalization*

Normalisasi Z-score adalah satu metode normalisasi atau *feature scaling* berdasarkan mean (rata-rata) dan standard deviasi (Abdi & Lynne J., 2010; Patro & sahu, 2015). Metode ini dituliskan sebagai berikut:

$$z = \frac{x_i-\mu}{\sigma} \quad (2.15)$$

Keterangan:
z = Nilai baru dari hasil normalisasi
x_i = Nilai aktual ke-i
μ = Nilai rata-rata populasi
σ = Nilai standard deviation

2.1.11 *Vanishing Gradient*

Vanishing gradient terjadi selama proses pelatihan *deep neural network*, di mana gradien yang digunakan untuk memperbarui jaringan menjadi semakin kecil dan bahkan mendekati nilai 0 atau "hilang" karena mengalami *backpropagation* dari lapisan keluaran ke lapisan masukan (Nurmaini et al., 2021). Misalnya suatu data memiliki total seluruh langkah waktu (*timestep*) *T*:

$$\frac{\partial E}{\partial W} = \sum_{t=1}^{T} \frac{\partial E_t}{\partial W} \quad (2.16)$$

Dengan *chain rule*, persamaan (2.16) dapat dijabarkan menjadi:

$$\frac{\partial E}{\partial W} = \sum_{t=1}^{T} \frac{\partial E_t}{\partial y_t} \frac{\partial y_t}{\partial h_t} \frac{\partial h_t}{\partial h_k} \frac{\partial h_k}{\partial W} \quad (2.17)$$

Anotasi $\frac{\partial h_t}{\partial h_k}$ pada persamaan (2.17) adalah turunan *hidden state* yang menyimpan memori pada waktu *t* yang berkaitan dengan *hidden state* pada waktu sebelumnya *k*. Proses ini melibatkan perkalian Jacobians $\frac{\partial h_i}{\partial h_{i-1}}$ untuk seluruh waktu *t* dan satu waktu *k*:

$$\frac{\partial E}{\partial W} = \frac{\partial h_t}{\partial h_{t-1}} \frac{\partial h_{t-1}}{\partial h_{t-2}} \dots \frac{\partial h_{k+1}}{\partial h_k} \quad (2.18)$$

$$= \prod_{i=k+1}^{t} \frac{\partial h_i}{\partial h_{i-1}} \quad (2.19)$$

Perkalian Jacobians pada persamaan (2.19) mewakili turunan dari h_t terhadap h_{t-1} (misal: $\frac{\partial h_t}{\partial h_{t-1}}$), di mana ketika dievaluasi $W^T[f'(h_{t-1})]$ dan diag mewakili proses konversi dari vektor ke matriks diagonal:

$$\prod_{i=k+1}^{t} \frac{\partial h_i}{\partial h_{i-1}} = \prod_{i=k+1}^{t} W^T diag[f'(h_{t-1})] \quad (2.20)$$

Matriks Jacobian $\frac{\partial h_t}{\partial h_{t-1}}$ menampilkan pemisahan nilai eigen (*eigendecomposition*) yang berasal dari $W^T diag[f'(h_{t-1})]$. Hal ini menghasilkan nilai eigen $\lambda_1, \lambda_2, \dots, \lambda_n$ di mana $|\lambda_1| > |\lambda_2| \dots |\lambda_n|$ serta vektor eigen yang sesuai $v_1, v_2, \dots, v_n$. Setiap perubahan pada *hidden state* Δh_t pada vektor v_i mempengaruhi perkalian antara nilai eigen terkait dengan vektor eigen ($\lambda_i \Delta h_t$).

Perkalian Jacobian pada persamaan (2.20) mencerminkan langkah-langkah waktu berikutnya, yang menghasilkan penskalaan perubahan faktor yang setara

dengan λ_i. λ_i mewakili nilai *eigen* i^{th}. Urutan $\lambda_i^1 \Delta h_1, \lambda_i^2 \Delta h_2, \dots, \lambda_i^n \Delta h_n$ memperlihatkan bahwa faktor λ_i^t akan didominasi Δh_t karena fase ini akan berkembang secara eksponensial seiring dengan $t \to \infty$. Jika nilai *eigen* terbesar dihasilkan $\lambda_i < 1$ maka akan terjadi *vanishing gradient*.

2.1.12 Mean Absolute Percentage Error (MAPE)

Mean Absolute Percentage Error (MAPE) merupakan salah satu metode evaluasi model yang menghitung rata-rata persentase kesalahan absolut antara nilai prediksi dan nilai sebenarnya. Pendekatan ini berguna ketika ukuran suatu variabel signifikan dalam mengevaluasi keakuratan suatu prediksi. MAPE menunjukkan seberapa besar kesalahan prediksi dibandingkan nilai sebenarnya (de Myttenaere et al., 2016; Khair et al., 2017). Rumus perhitungan MAPE adalah sebagai berikut:

$$MAPE = \frac{1}{N} \sum \left| \frac{y_i - p_i}{y_i} \right| x100\% \quad (2.21)$$

Keterangan:
y_i = Nilai aktual ke-i
p_i = Nilai hasil peramalan/prediksi ke-i
N = Banyaknya data / ukuran sampel

2.2 Studi Literatur

Studi-studi terdahulu menjadi elemen kunci dalam penelitian ini. Selain sebagai bahan referensi, penulis memastikan bahwa penelitian yang dilakukan memberikan kontribusi baru dalam pengembangan ilmu dan teknologi pada topik penelitian yang diambil. Oleh karena itu, berikut adalah hasil analisis dari beberapa penelitian terdahulu mengenai prediksi konsentrasi polutan dan kualitas udara dengan berbagai metode yang telah diimplementasikan serta prediksi *time series* menggunakan metode ResNet-LSTM, seperti yang ditunjukkan pada **Tabel 1.1**.

Penelitian yang dilakukan oleh Q. Zhang (2020) menggunakan data kualitas udara, meteorologi, dan lalu lintas dengan periode waktu 19 bulan (Jan, 2017 – Jul, 2018) di Beijing. Data kualitas udara diperoleh dari 35 stasiun pengamatan kualitas udara dan data meteorologi diperoleh dari 18 stasiun pengamatan meteorologi di Beijing. Data lalu lintas diperoleh dari web API Gaode Map dengan 227 jalan utama di Beijing sebagai masukan. Nilai yang hilang dari data *time series* diisi dengan interpolasi linear dalam dimensi temporal. Sedangkan, jaringan tanpa data dinamika perkotaan diisi dengan interpolasi Krigging dalam dimensi spasial. Akibat perbedaan saluran pada masukan, arsitektur AirRes digunakan untuk memfasilitas pertukaran infomasi dari berbagai saluran masukan data dan dilakukan ekstraksi fitur spasial dari setiap *timesteps*. Kemudian, LSTM mengambil matriks fitur yang telah dihasilkan oleh *deep residual component*, dan menghasilkan vektor untuk setiap jaringan

sebagai prediksi polusi udara. Hasil prediksi polusi udara pada penelitian ini memperoleh akurasi sebesar 80%.

Penelitian yang dilakukan Song (2020) memprediksi $PM_{2.5}$ dan PM_{10} dengan inputan berupa data gambar. Penelitian ini menggunakan metode ResNet18 yang dikombinasikan dengan LSTM. Dataset yang digunakan yaitu data zat polutan $PM_{2.5}$ dan PM_{10} dan meteorologi, yaitu suhu dan kelembapan. Kedua data ini dikalibrasi untuk mengurangi bias pada sensor menggunakan Polynomial Regression (PR) dan Support Vector Machine (SVR). Model dengan MAE terendah dipilih untuk kalibrasi. Penelitian ini mengambil gambar untuk dataset dari 1 gedung yang sama dengan jarak 500 m setiap menit dari pukul 14:30 – 19:30 selama 10 hari. Metode ResNet18 digunakan untuk mengekstrak fitur spasial pada gambar, sedangkan LSTM digunakan untuk mengekstrak fitur temporal pada data sekuensial. Dua *final hidden state* dari ResNet-LSTM dan paralel LSTM digabung dan digunakan untuk memprediksi $PM_{2.5}$ dan PM_{10} melalui *fully connected layer*. Prediksi pada penelitian ini memperoleh hasil akurasi sebesar 93% untuk $PM_{2.5}$ dan 89% untuk PM_{10}.

Penelitian yang dilakukan Cheng (2022) memprediksi konsentrasi $PM_{2.5}$ menggunakan 120 jam data polutan $PM_{2.5}$ dan meteorologi, seperti suhu, tekanan udara, suhu titik embun, arah angin, dan kecepatan angin sebagai inputan. Pada tahap *data preprocessing,* metode CORAL digunakan untuk mengisi data yang hilang pada dataset. Kemudian, dataset dipisah menggunakan *sliding window*. Penelitian ini menggunakan metode ResNet dan LSTM untuk melakukan prediksi. ResNet34 digunakan untuk mengesktrak fitur lokal pada data $PM_{2.5}$ dan meteorologi, sedangkan LSTM digunakan untuk mengekstrak fitur temporal. *Fully Connected Layer* digunakan untuk melakukan prediksi $PM_{2.5}$ selama 6 jam ke depan dan diperoleh hasil akurasi sebesar 80%.

Penelitian yang dilakukan oleh Wu (2023) memprediksi konsentrasi O_3 dan NO_2 menggunakan data faktor polutan, meteorologi, dan geografi, seperti NO_2, CO, $PM_{2.5}$, PM_{10}, SO_2, O_3, suhu, kelembaban, kecepatan angin, dan arah angin. Penelitian ini menggunakan metode ResNet, Graph Convolutional Network (GCN), dan BiLSTM. Res-GCN-BiLSTM terdiri dari tiga komponen. Cabang 1 mengimplementasikan model ResNet untuk mengekstrak ketergantungan spasial antar stasiun dalam pola harian. Cabang 2 mengimplementasikan model GCN untuk menangkap informasi topologi seluruh jaringan pemantauan. Cabang 3 mengimplementasikan model BiLSTM untuk mempelajari korelasi temporal dari faktor eksternal, seperti zat polutan lainnya dan meteorologi. Kemudian, ketiga cabang tersebut digabungkan dan dilatih kembali dalam model BiLSTM yang sebelumnya diberi *weighted feature fusion* untuk melakukan ekstraksi fitur tingkat tinggi. Lalu, hasil prediksi diratakan (*flatten*) dan dimasukkan dalam *Fully Connected Network layer*. Prediksi pada penelitian ini memperoleh hasil MAE 11% dan 17% dibandingkan dengan model ResNet-LSTM untuk masing-masing konsentrasi O_3 dan NO_2.

Penelitian yang dilakukan oleh B. Zhang (2022) memprediksi salah satu zat polutan $PM_{2.5}$ dalam jangka pendek, yaitu selama 1 – 3 jam menggunakan 16 variabel data polutan dan meteorologi dari 14 kota. Target zat polutan yang digunakan adalah $PM_{2.5}$, sedangkan kota yang dipilih adalah Shanghai. Sebelum dilakukan pemodelan, data dianalisis baik dari dimensi spasial maupun temporal menggunakan *correlation coefficient Pearson*. Hasil analisis dimensi temporal menunjukkan bahwa ada korelasi antara data polutan dengan meteorologi. Hasil analisis dimensi spasial menunjukkan bahwa jarak antarkota berbanding terbalik dengan konsentrasi polutan pada kota yang menjadi acuan. *Missing value* pada *dataset* yang digunakan diisi menggunakan interpolasi *spatiotemporal*. Prediksi $PM_{2.5}$ pada penelitian ini menggunakan metode RCL (ResNet Conv-LSTM) *Learning*, di mana ResNet digunakan untuk mengekstrak fitur spasial dan Conv-LSTM digunakan untuk mengekstrak fitur *spatiotemporal* serta melakukan prediksi. Hasil akurasi pada penelitian ini sebesar 98%. Selain jangka pendek, penelitian juga melakukan prediksi jangka panjang selama 1 – 15 jam secara bertahap.

Penelitian yang dilakukan oleh Kalajdjieski (2020) menggunakan data primer berupa gambar yang diambil oleh kamera statis di Gunung Vodno dekat Skopje, Makedonia Utara. Selain itu, penulis juga menggunakan data meteorologi yang diambil dari stasiun meteorologi terdekat. Penelitian ini menggunakan empat arsitektur, dua diantaranya merupakan *transfer learning*, yaitu *convolutional neural network*, *residual neural network*, *basic inception network*, dan *custom inception-based model*. Inception V3 digunakan untuk mengidentifikasi fitur tingkat rendah dan dikombinasikan dengan arsitektur ResNet yang menghasilkan akurasi prediksi AQI sebesar 88%. *Conditional Generative Adversarial Networks* (CGAN) juga digunakan untuk mengatasi masalah ketidakseimbangan kelas pada data latih gambar dengan menghasilkan gambar baru berdasarkan gambar yang dilihat sebelumnya pada tahap *pre-processing*.

Penelitian yang dilakukan oleh Sheng (2023) menggunakan data *power loads* dan suhu. Penelitian ini menggunakan metode *Deep Residual Network* (DRN) dan LSTM. Model ini terbagi menjadi dua bagian, *Fully-Connected Neural Network* dan struktur *Residual* LSTM *block*. Bagian pertama, FCNN digunakan untuk memperoleh prediksi awal dalam waktu 24 jam dengan input power loads dan suhu dalam 1-3 bulan, 1-8 minggu, 7 hari, 24 jam sebelum hari prediksi, serta suhu hari setelahnya, musim, hari kerja, dan hari libur. Bagian kedua, *Residual* LSTM *block* yang terdiri dari 2 lapis LSTM, *batch normalization*, *dropout*, dan *dimension weighted unit* (DWU) digunakan untuk meningkatkan akurasi prediksi dan menghasilkan output final. LSTM digunakan untuk mempelajari relasi tersembunyi dari fitur dalam dimensi waktu, sedangkan DWU digunakan untuk memodelkan ketergantungan antar dimensi fitur. Prediksi ini memperoleh hasil MAPE sebesar 1.56 dan meningkat sebesar 12% menggunakan *Residual block* dan DWU.

Penelitian yang dilakukan oleh Qin (2019) menggunakan data polutan dan meteorologi dari tahun 2015 – 2017 di Shanghai. Penelitian ini menggunakan metode

CNN dan LSTM. Pada proses *preprocessing*, fitur polutan dan meteorologi dikonversi menjadi matriks dua dimensi. Model CNN digunakan untuk untuk menghilangkan redudansi data dan mengekstrak fitur spasial menggunakan *convolutional* dan *pooling layers*. Kemudian, *output* dari *pooling layer* terakhir CNN menjadi masukkan dalam lapisan LSTM. Model LSTM digunakan untuk mengesktrak fitur temporal. Hasil prediksi keluaran LSTM akhir didekodekan oleh lapisan *Fully Connected Layer*, lalu digunakan untuk memperoleh hasil prediksi akhir. Prediksi $PM_{2.5}$ perham memperoleh hasil RMSE sebesar 14.3, di mana lebih baik daripada metode pembandingnya, seperti CNN saja dan LSTM saja. Model ini mampu memproses data beberapa lokasi pemantauan dalam satu kota.

Tabel 2.1. *State of the Art*

No	Penulis	Metode	Hasil	*Gap Research*		
				VG*	ST*	P>1*
1.	(Q. Zhang et al., 2020)	AirRes-LSTM	Data inputan pada penelitian ini adalah data polusi udara (PM2.5, PM10, NO2, CO, O3), meteorologi (pressure, temp, wind direction, precipitation, wind speed), dan lalu lintas (status, speed, count) setiap satu jam. Metode Deep-Air ini memiliki akurasi prediksi kualitas udara sebesar 80%. Berdasarkan hasil peramalan zat polutan yang berbeda, model memberikan performa yang baik dalam meramalkan zat CO, tetapi kurang memuaskan pada zat O3.		✓	✓
2.	(Song et al., 2020)	ResNet-LSTM	Ekstraksi fitur spasial-temporal dari data gambar menggunakan ResNet18 memperoleh hasil akurasi prediksi PM2.5 sebesar 93% dan PM10 sebesar 89% dengan model ResNet-LSTM. Model Met-ResNet-LSTM-SP memperoleh akurasi PM2.5 sebesar 95% dan PM10 sebesar 91%.	✓	✓	
3.	(Cheng et al., 2022)	ResNet-LSTM and CORAL	Prediksi konsentrasi PM2.5 menggunakan model stacked ResNet-LSTM berdasarkan data historis kualitas udara dan data meteorologi menunjukkan hasil akurasi sebesar 80%. Kemudian, CORAL digunakan untuk mengatasi masalah ketersediaan data yang tidak mencukupi dan mampu meningkatkan prediksi PM2.5.		✓	

Tabel 2.2. Lanjutan *State of the Art*

No	Penulis	Metode	Hasil	*Gap Research*		
				VG*	ST*	P>1*
4.	(Wu et al., 2023)	ResNet, GCN, BiLSTM	Prediksi NO2 dan O3 pada penelitian ini memperoleh hasil MAE sebesar 11% dan 17% dibadingkan dengan ResNet-LSTM. Performa terbaik untuk pediksi O3 diraih oleh stasiun pemantauan lalu lintas, urban, dan terakhir suburban. Sedangkan untuk prediksi NO2 sebaliknya.	✓	✓	
5.	(B. Zhang et al., 2022)	ResNet and Conv-LSTM	Prediksi PM2.5 jangka pendek selama 1-3 jam menggunakan 16 variabel data polutan dan meteorologi dari 14 kota. Target polutan dan kota yang digunakan adalah PM2.5 dan Shanghai. Prediksi dengan metode RCL Learning ini memperoleh akurasi sebesar 98%. Penelitian ini juga melakukan prediksi jangka panjang selama 1-15 jam secara bertahap.	✓	✓	
6.	(Kalajdjieski et al., 2020)	ResNet and Inception V3 Pre-trained Model	Model pada penelitian ini menggunakan data gambar dan informasi cuaca. Inception V3 digunakan untuk mengidentifikasi fitur tingkat rendah dan dikombinasikan dengan arsitektur ResNet yang menghasilkan akurasi prediksi AQI sebesar 76%.	✓	✓	
7.	(Sheng et al., 2023)	DRN-LSTM	Prediksi pada penelitian ini memperoleh hasil MAPE sebesar 1.56 dengan nilai r = 16 dan jumlah layer LSTM = 20. Akurasi model ini meningkat sebanyak 12% menggunakan Residual block dan DWU.	✓		
8.	(Qin et al., 2019)	CNN-LSTM	Prediksi PM2.5 pada kota target, Shanghai, memperoleh hasil RMSE sebesar 14.3, di mana lebih baik daripada metode pembandingnya, seperti CNN saja atau LSTM saja. Model ini mampu memproses data beberapa lokasi pemantauan dalam satu kota.	✓	✓	

*VG = *Vanishing Gradient,* ST = data *statiotemporal*, P>1 = prediksi polutan lebih dari 1 zat

Penelitian-penelitian terdahulu yang telah dituliskan pada **Tabel 2.1** hingga **Tabel 2.2** telah memberikan kontribusi penting dalam memahami dan

mengembangkan metode prediksi polutan dan kualitas udara yang efektif. Namun, terdapat kesenjangan penelitian yang perlu diisi, seperti yang ditunjukkan pada kolom *gap research* dengan tiga parameter, yaitu VG (*Vanishing Gradient*), ST (Data *spatiotemporal*), dan P>1 (Prediksi polutan lebih dari 1 zat). Beberapa penelitian sebelumnya menggunakan data *spatiotemporal* yang terdiri dari data *time series* polutan dan meteorologi serta gambar. Untuk dapat melakukan ekstraksi fitur spasial dan temporal, tentunya struktur jaringan yang digunakan lebih dalam dan cukup kompleks sehingga dapat terjadi permasalahan *vanishing gradient*. Namun, beberapa penelitian tidak mengatasi permasalahan tersebut. Selain itu, beberapa penelitian sebelumnya cenderung memprediksi satu atau dua parameter saja, seperti $PM_{2.5}$, PM_{10}, dan AQI. Oleh karena itu, penelitian ini akan memfokuskan pada penyelesaian masalah *vanishing gradient* dalam pengembangan metode prediksi konsentrasi polutan PM_{10}, SO_2, CO, O_3, dan NO_2, sehingga diharapkan dapat memperoleh hasil prediksi yang akurat. Ringkasan *State-of-the-Art* dituliskan pada **Tabel 2.4**.

Tabel 2.3. Ringkasan *State-of-the-Art*

No	Penulis	*Vanishing Gradient*	Data *Spatiotemporal*	Prediksi Polutan >1
1.	(Q. Zhang et al., 2020) Deep-AIR: A Hybrid CNN-LSTM Framework for Fine-Grained Air Pollution Forecast		✓	✓
2.	(Song et al., 2020) ResNet-LSTM for Real-Time PM2.5and PM Estimation Using Sequential Smartphone Images	✓	✓	
3.	(Cheng et al., 2022) Stacked ResNet-LSTM and CORAL Model for Multi-Site Air Quality Prediction		✓	
4.	(Wu et al., 2023) A Hybrid Deep Learning Model for Regional O3 and NO2 Concentrations Prediction Based on Spatiotemporal Dependencies in Air Quality Monitoring Network	✓	✓	
5.	(B. Zhang et al., 2022) RCL-Learning: ResNet and Convolutional Long Short-Term Memory-Based Spatiotemporal Air Pollutant Concentration Prediction Model	✓	✓	
6.	(Kalajdjieski et al., 2020) Air Pollution Prediction with Multi-Modal Data and Deep Neural Networks	✓	✓	

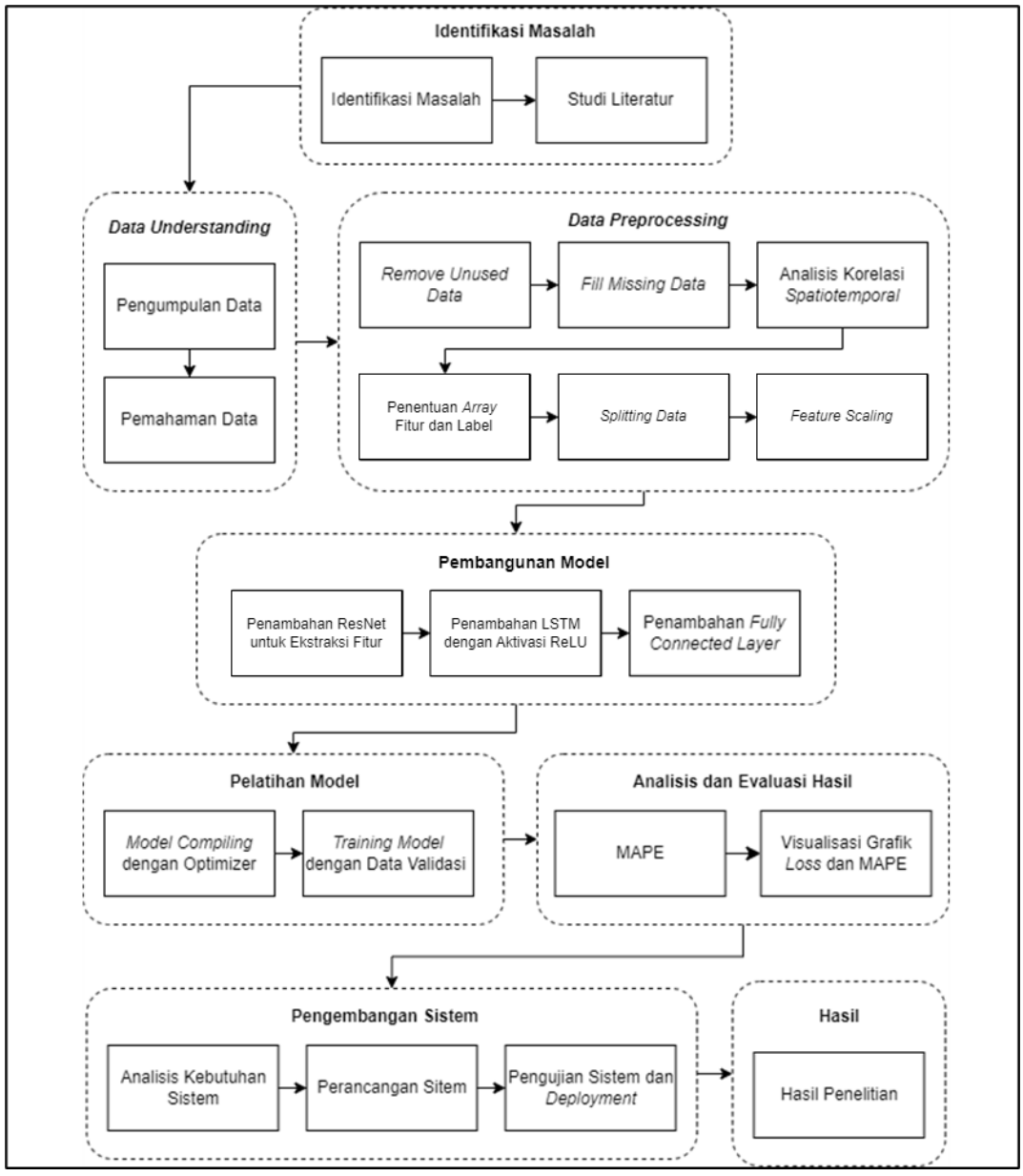
Tabel 2.4. Lanjutan Ringkasan *State-of-the-Art*

No	Penulis	***Vanishing Gradient***	**Data *Spatiotemporal***	**Prediksi Polutan >1**
7.	(Sheng et al., 2023) Residual LSTM Based Short-Term Load Forecasting	✓		
8.	(Qin et al., 2019) A Novel Combined Prediction Scheme Based on CNN and LSTM for Urban PM2.5 Concentration	✓	✓	

BAB III
METODOLOGI PENELITIAN DAN PENGEMBANGAN SISTEM

3.1 Metodologi Penelitian

Pada bagian ini akan dibahas mengenai metodologi yang akan dilakukan dalam penelitian ini, yaitu mengatasi *vanishing gradient* menggunakan CNN-LSTM pada prediksi konsentrasi polutan dalam kualitas udara. Penelitian ini menerapkan metode kuantitatif menggunakan data numerik dan statistik untuk menafsirkan informasi yang digunakan dalam pengujian hipotesis dengan teori yang sudah ada. Adapun data yang digunakan dalam penelitian ini adalah data historis zat polutan dan meteorologi Provinsi DKI Jakarta. Tahapan penelitian ditunjukkan pada Gambar 3.1.



Gambar 3.1. Tahapan Penelitian

3.1.1 Identifikasi Masalah

a. Identifikasi Masalah

Penelitian ini diawali dengan mengidentifikasi masalah yang akan diangkat dan diselesaikan dalam penelitian. Masalah yang diangkat adalah masalah *vanishing gradient* akibat meningkatnya kompleksitas jaringan pada prediksi data *spatiotemporal* yang menyebabkan akurasi menjadi tidak optimal.

b. Studi Literatur

Studi literatur dilakukan untuk memperoleh informasi yang relevan yang akan digunakan sebagai dasar atau acuan pada penelitian ini. Tahapan ini dilakukan dengan menelusuri beberapa sumber seperti buku, jurnal, dan penelitian terdahulu mengenai *vanishing gradient* dan prediksi konsentrasi polutan dalam kualitas udara. Informasi yang didapatkan digunakan sebagai dasar untuk menyelesaikan masalah dan mencapai tujuan penelitian. Informasi dari penelitian-penelitian terdahulu dapat dilihat dalam *State of the Art* pada Tabel 2.1 hingga Tabel 2.3.

3.1.2 *Data Understanding*

a. Pengumpulan Data

Penelitian ini menggunakan kombinasi data primer-sekunder, yaitu data historis konsentrasi polutan dari lima Stasiun Pemantauan Kualitas Udara dan data meteorologi dari lima Stasiun Meteorologi di DKI Jakarta. Masing-masing data diperoleh dari *website* Satu Data Jakarta (https://satudata.jakarta.go.id/) (Dinas Lingkungan Hidup, 2019, 2020, 2021) untuk data konsentrasi polutan dan Data Online Pusat Database BMKG (https://dataonline.bmkg.go.id/) (Badan Meteorologi, 2019, 2020, 2021) untuk data meteorologi dalam bentuk *excel*. Data ini dikumpulkan setiap 1 hari dari 1 Januari 2019 – 31 Desember 2021 sebanyak 5.480 data.

Pada penelitian ini, dipilih lima konsentrasi polutan, yaitu PM_{10}, SO_2, CO, O_3, dan NO_2 serta lima faktor meteorologi, yaitu suhu rata-rata, kelembaban rata-rata, kecepatan angin rata-rata, dan arah angin yang diambil dari Jakarta Pusat, Jakarta Barat, Jakarta Utara, Jakarta Selatan, dan Jakarta Timur.

b. Pemahaman Data

Data konsentrasi polutan dan meteorologi yang diperoleh memiliki beberapa parameter yang digunakan untuk melakukan prediksi menggunakan metode ResNet-LSTM. Detail dari setiap parameter dapat dilihat pada Tabel 3.1 untuk data meteorologi serta Tabel 3.2 dan Tabel 3.3 untuk data polutan.

Tabel 3.1. Detail Parameter Data Konsentrasi Polutan

No	Variabel	Deskripsi	Keterangan
1.	tanggal	Tanggal pengukuran konsentrasi polutan	Format penanggalan mm/dd/yyyy
2.	stasiun	Nomor/nama Stasiun Pemantauan Kualitas Udara (SPKU) di DKI Jakarta	Sebaran SPKU: 1. DKI 1 – Bundaran HI 2. DKI 2 – Kelapa Gading 3. DKI 3 – Jagakarsa 4. DKI 4 – Lubang Buaya 5. DKI 5 – Kebon Jeruk
3.	pm10	Rata-rata konsentrasi zat PM_{10} dalam periode waktu 1 hari pengukuran	Data yang tidak ada dikosongkan atau diberi tanda ---
4.	so2	Rata-rata konsentrasi zat SO_2 dalam periode waktu 1 hari pengukuran	Data yang tidak ada dikosongkan atau diberi tanda ---
5.	co	Rata-rata konsentrasi zat CO dalam periode waktu 1 hari pengukuran	Data yang tidak ada dikosongkan atau diberi tanda ---
6.	o3	Rata-rata konsentrasi zat O_3 dalam periode waktu 1 hari pengukuran	Data yang tidak ada dikosongkan atau diberi tanda ---
7.	no2	Rata-rata konsentrasi zat NO_2 dalam periode waktu 1 hari pengukuran	Data yang tidak ada dikosongkan atau diberi tanda ---
8.	max	Nilai konsentrasi polutan maksimum dalam 1 hari	Data yang tidak ada diberi nilai 0
9.	critical	Zat polutan dengan nilai konsentrasi maksimum dalam 1 hari	Data yang tidak ada dikosongkan
10.	categori	Kategori Indeks Standar Pencemar Udara (ISPU) dalam 1 hari	Kategori ISPU: 1. BAIK 2. SEDANG 3. TIDAK SEHAT 4. SANGAT TIDAK SEHAT 5. TIDAK ADA DATA

Tabel 3.2. Detail Parameter Data Meteorologi

No	Variabel	Deskripsi	Keterangan
1.	Tanggal	Tanggal pengukuran faktor meterologi	Format penanggalan dd-mm-yyy
2.	Tavg	Nilai suhu rata-rata dalam waktu 1 hari pengukuran	Data tidak terukur diberi nilai 8888, data tidak ada diberi nilai 9999 atau dikosongkan.
3.	RH_avg	Nilai kelembaban rata-rata dalam waktu 1 hari pengukuran	Data tidak terukur diberi nilai 8888, data tidak ada diberi nilai 9999 atau dikosongkan.

Tabel 3.3. Lanjutan Detail Parameter Data Meteorologi

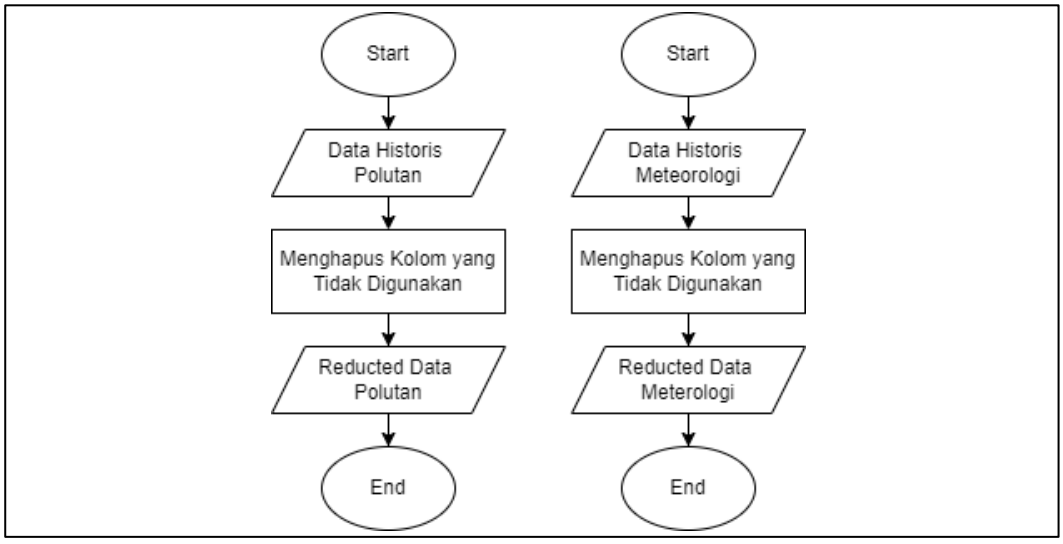
No	Variabel	Deskripsi	Keterangan
4.	RR	Nilai curah hujan dalam waktu 1 hari pengukuran	Data tidak terukur diberi nilai 8888, data tidak ada diberi nilai 9999 atau dikosongkan.
5.	ddd_x	Nilai arah angin dalam kecepatan maksimum dalam waktu 1 hari pengukuran	Data tidak terukur diberi nilai 8888, data tidak ada diberi nilai 9999 atau dikosongkan.
6.	ff_avg	Nilai kecepatan angin rata-rata dalam waktu 1 hari pengukuran	Data tidak terukur diberi nilai 8888, data tidak ada diberi nilai 9999 atau dikosongkan.

3.1.3 *Data Preprocessing*

Data yang telah didapatkan kemudian diproses agar menjadi lebih terstruktur sehingga dapat digunakan dalam pembuatan dan pelatihan model. Beberapa proses yang dilakukan pada tahapan ini, yaitu menghapus data-data yang tidak diperlukan, mengisi nilai yang hilang, menganalisis analisis korelasi *spatiotemporal*, dan membagi data.

a. *Remove Unused Data*

Proses pada tahapan ini dapat dilihat dalam *flowchart* pada Gambar 3.2. Penghapusan data dilakukan karena tidak semua variabel pada *dataset* akan digunakan dalam pembuatan model. Variabel yang dihapus meliputi max, critical, dan categori pada *dataset* konsentrasi polutan serta RR pada *dataset* meteorologi.

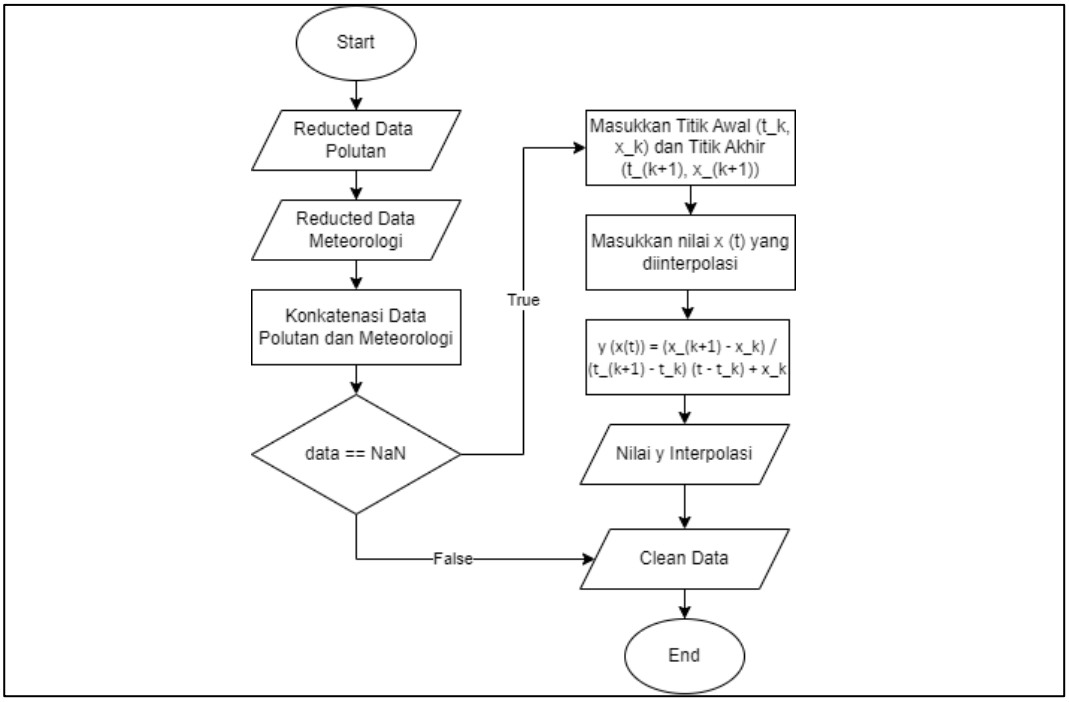


Gambar 3.2. *Flowchart Remove Unused Data*

b. *Fill Missing Value*

Setelah variabel yang tidak digunakan telah dihapus, selanjutnya adalah mengisi nilai yang hilang pada *dataset* konsentrasi polutan dan meteorologi.

Pengisian nilai yang hilang ini menggunakan metode interpolasi linear. Proses pada tahapan ini dapat dilihat dalam *flowchart* pada Gambar 3.3.



Gambar 3.3. *Flowchart Fill Missing Value*

Misalnya, pada *dataset* konsentrasi polutan terdapat nilai yang hilang pada tanggal 27 Januari 2019 seperti pada Tabel 3.4.

Tabel 3.4. Contoh *Missing Value* pada Data Konsentrasi Polutan

Index	**O_3**
1	118
2	NaN
3	64

Di, mana $(t_k, x_k) = (1, 118), (t_{k+1}, x_{k+1}) = (3, 64)$, dan $\left(t, \tilde{x}(t)\right) = (2, NaN)$, maka penerapan metode interpolasi linear menggunakan rumus (2.12) adalah sebagai berikut:

$$\tilde{x}(t) = \frac{x_{k+1} - x_k}{t_{k+1} - t_k}(t - t_k) + x_k$$

$$\tilde{x}(t) = \frac{64 - 118}{3 - 1}(2 - 1) + 118$$

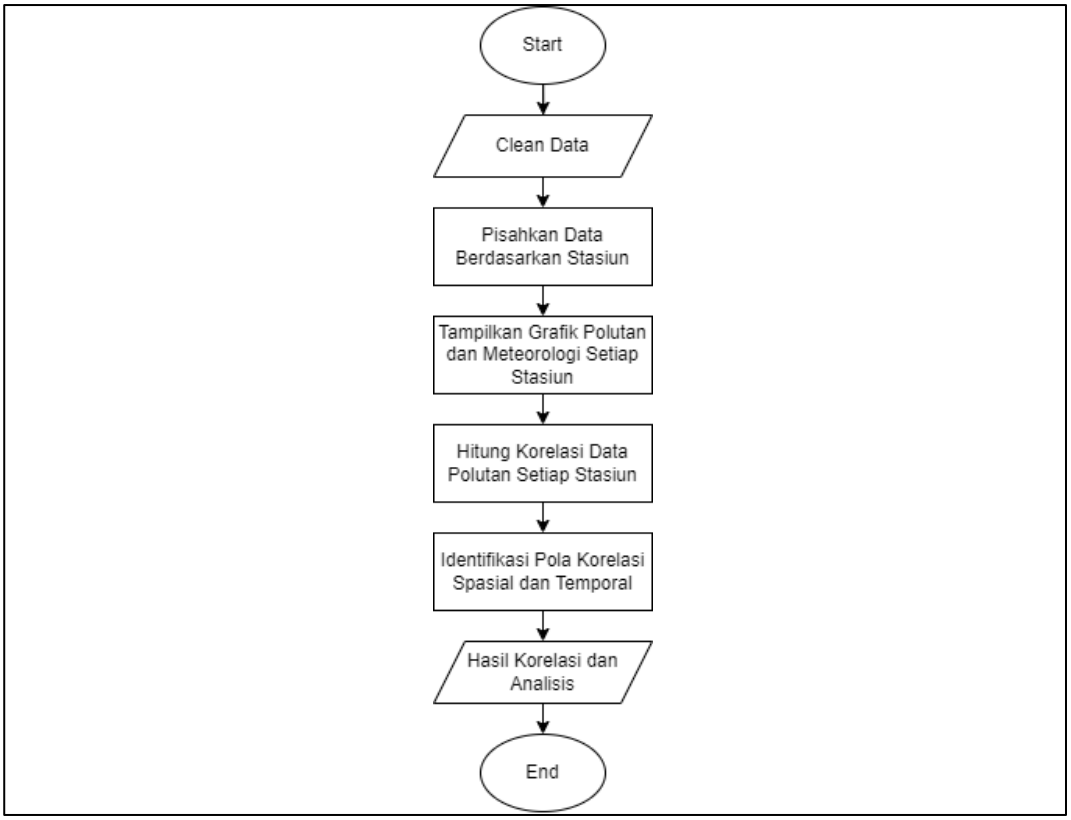
$$\tilde{x}(t) = \frac{-54}{2}(1) + 118$$

$$\tilde{x}(t) = (-27) + 118$$

$$\tilde{x}(t) = 91$$

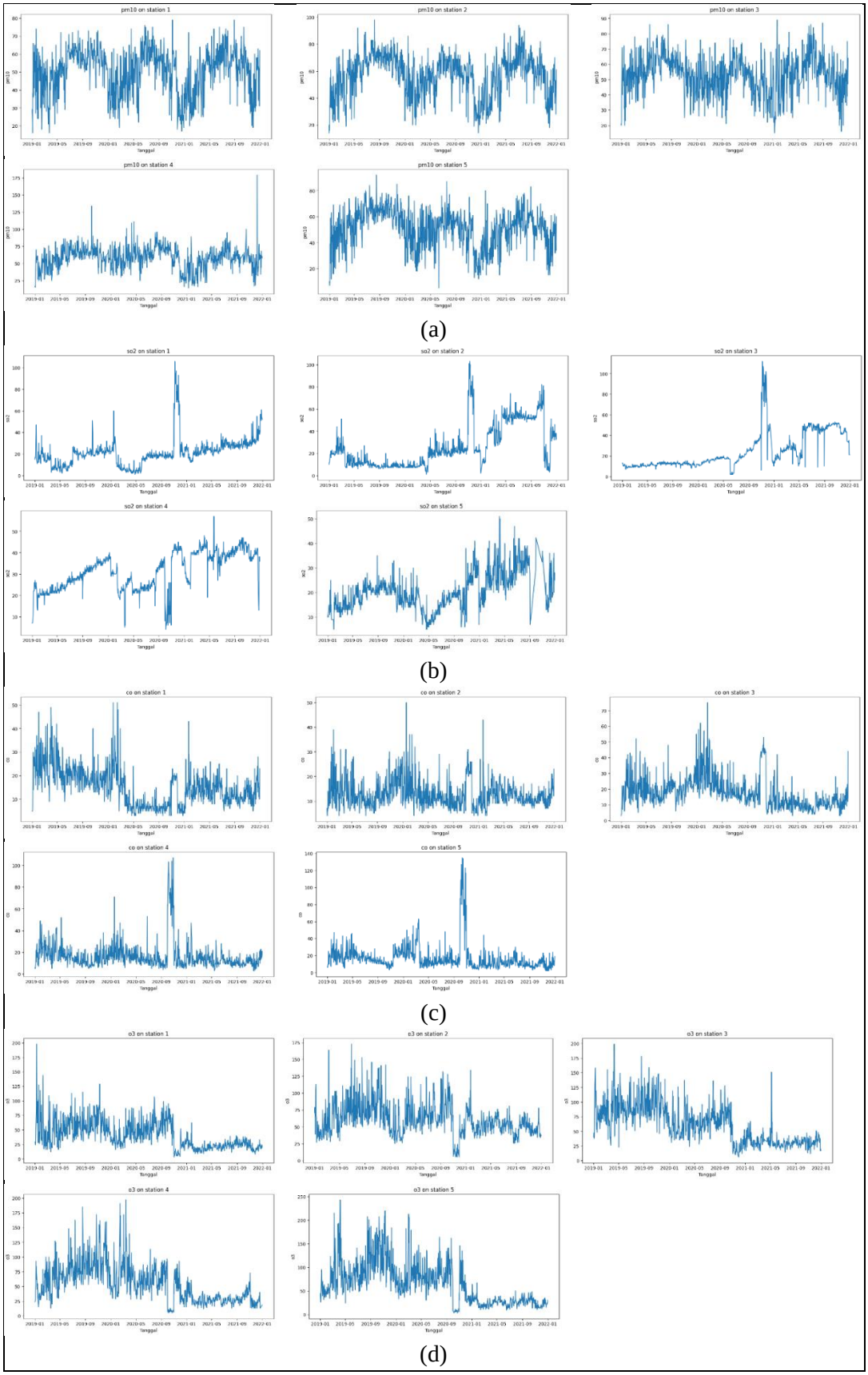
Sehingga, diperoleh hasil $\tilde{x}(t)$ yang merupakan nilai O_3 pada tanggal 27 Januari 2019 sebesar 91.

c. Analisis Korelasi *Spatiotemporal*

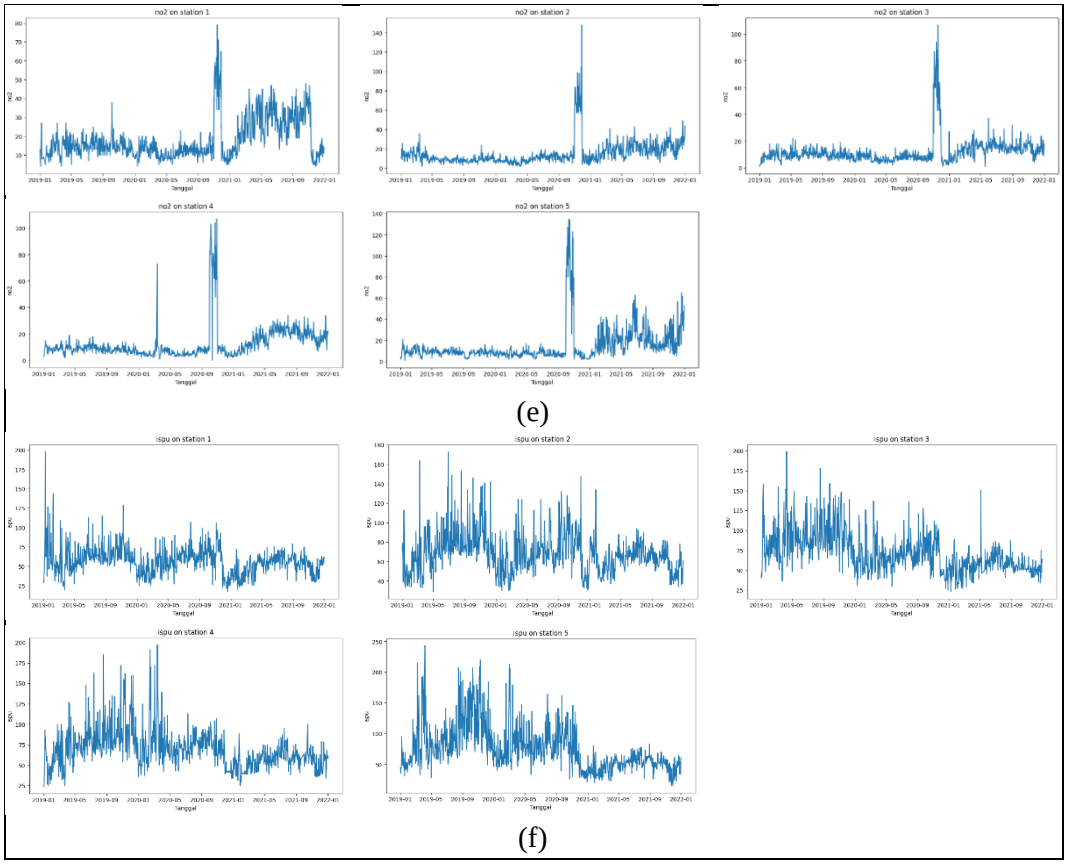


Gambar 3.4. *Flowchart* Analisis Korelasi Spatiotemporal

Korelasi aspek spasial dan temporal sangat penting untuk melakukan prediksi konsentrasi polutan dalam kualitas udara. Proses pada tahapan ini dapat dilihat dalam *flowchart* pada Gambar 3.4. Berdasarkan data historis ditemukan bahwa perubahan tren pada polutan dan faktor meteorologi secara umum konsisten, yang juga mencerminkan hubungan terkait antara keduanya. Gambar 3.5 dan Gambar 3.6 menunjukkan perubahan numerik dalam periode waktu 3 tahun setiap konsentrasi polutan dari 5 Stasiun Pemantauan Kualitas Udara, di mana (a) PM_{10}, (b) SO_2, (c) CO, (d) O_3, dan (e) NO_2. Setelah dilakukan perhitungan statistik, ditemukan bahwa antara tahun 2019 – 2021 sekitar 62% waktu konsentrasi PM_{10} lebih dari yang ditetapkan WHO, yaitu 50 $\mu g/m^3$. Konsentrasi PM_{10} mencapai rata-rata harian tertinggi yaitu 179 $\mu g/m^3$ terjadi pada 7 Desember 2021 di stasiun DKI 4 Lubang Buaya. Oleh karena itu, korelasi antara PM_{10} dengan polutan lainnya harus dipertimbangkan untuk memperoleh prediksi konsentrasi yang akurat dan mencegah dampak yang lebih buruk terhadap kesehatan masyarakat sejak dini.

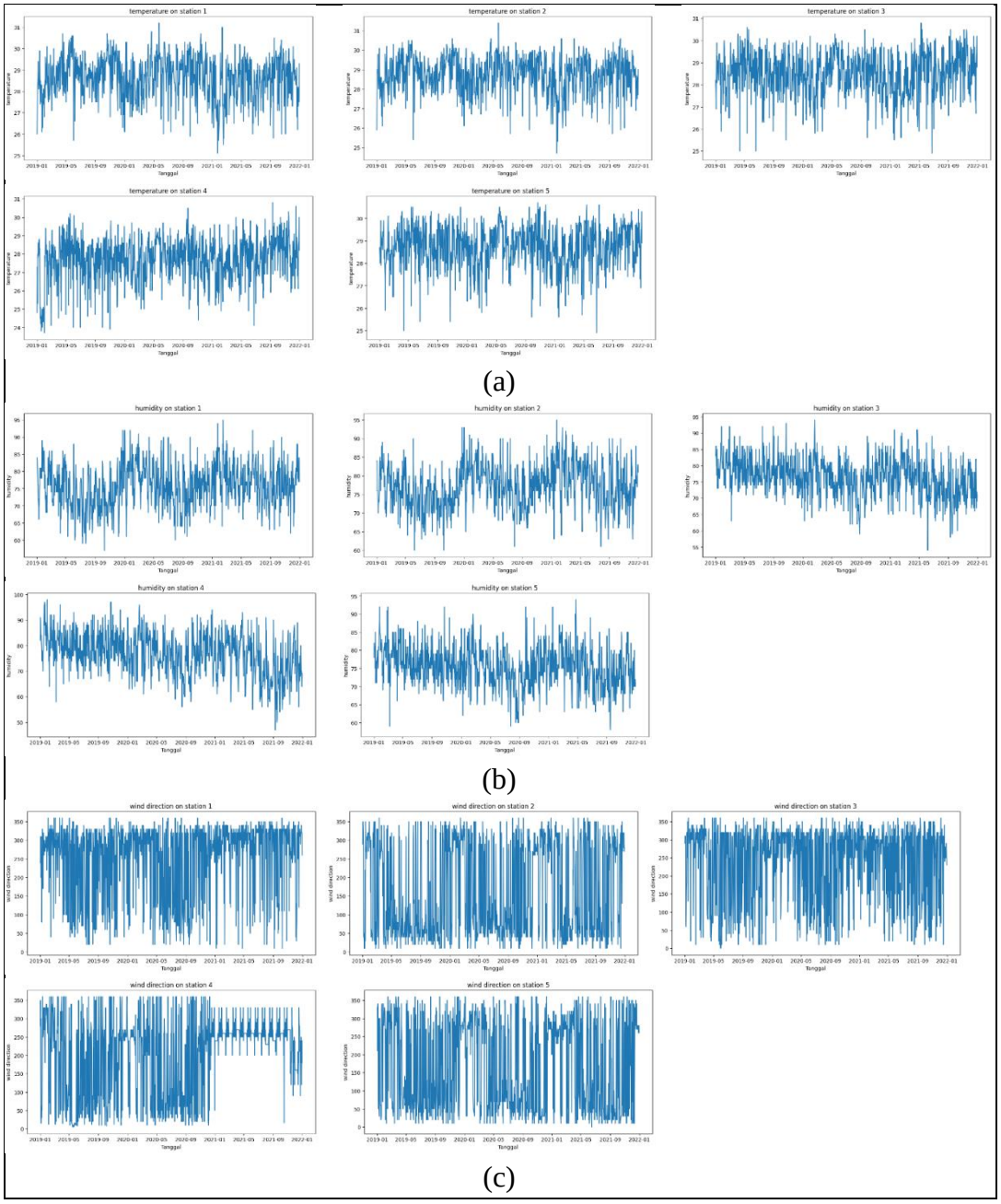


Gambar 3.5. Grafik Perubahan Konsentrasi Polutan Tahun 2019 – 2021 dari 5 Stasiun Pemantauan Kualitas Udara, (a) PM_{10}, (b) SO_2, (c) CO, dan (d) O_3

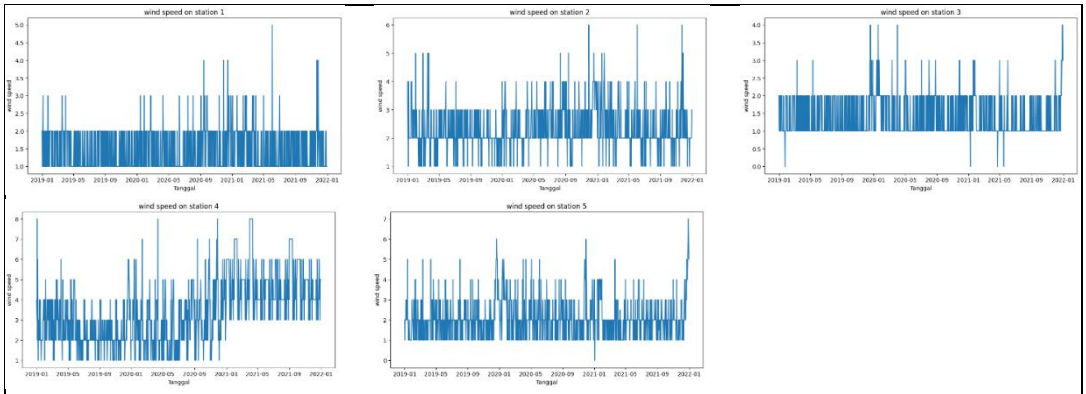


Gambar 3.6. Grafik Perubahan Konsentrasi Polutan Tahun 2019 – 2021 dari 5 Stasiun Pemantauan Kualitas Udara (e) NO_2, dan (f) ISPU

Gambar 3.7 menunjukkan perubahan numerik dalam periode waktu 3 tahun setiap faktor meteorologi dari 5 Stasiun Meteorologi, di mana (a) suhu rata-rata, dan (b) kelembaban rata-rata. Tipe numerik dan interval dari masing-masing faktor meteorologi sangat berbeda, tetapi tren perubahannya sangat mirip, yang menandakan bahwa mungkin terdapat pengaruh timbal balik antara faktor tersebut. Sebagai contoh, Gambar 3.7.(a) menunjukkan perubahan suhu yang cukup fluktuatif dibandingkan kelembaban pada Gambar 3.7.(b). Pada rentang bulan Juli – September, suhu dan kelembaban rata-rata cenderung mengalami penurunan. Sementara, pada rentang bulan Januari – Maret suhu rata-rata cenderung mengalami penurunan, sedangkan kelembaban rata-rata cenderung mengalami kenaikan. Faktor meteorologi juga konsisten dengan perubahan konsentrasi PM_{10}, yang menandakan terdapat korelasi antara polutan dan faktor meteorologi. Sebagai contoh, Pada rentang bulan Januari – Maret, konsentrasi PM_{10} cenderung mengalami penurunan, sedangkan Juli – September cenderung mengalami kenaikan. Hal ini berbanding terbalik dengan kelembaban rata-rata. Berdasarkan hasil analisis temporal yang telah dilakukan, faktor meteorologi digunakan sebagai salah satu *input* pada model untuk mengekstrak fitur tersembunyi antara polutan dan faktor meteorologi pada prediksi konsentrasi polutan dalam kualitas udara.



Gambar 3.7. Grafik Perubahan Faktor Meteorologi Tahun 2019 – 2021 dari 5 Stasiun Meteorologi (a) Suhu Rata-rata, (b) Kelembaban Rata-rata, dan (c) Arah Angin



(d)

Gambar 3.8. Grafik Perubahan Faktor Meteorologi Tahun 2019 – 2021 dari 5 Stasiun Meteorologi (d) Kecepatan Angin

Polutan PM_{10} mungkin juga memiliki beberapa karakteristik spasial, misalnya stasiun target adalah Stasiun DKI 1 Bundaran HI. Karakteristik spasial ini dipengaruhi oleh jarak antara Stasiun 1 dengan stasiun lainnya. Lokasi dari seluruh Stasiun Pemantauan Kualitas Udara DKI Jakarta ditunjukkan pada Gambar 3.9 dan jarak Stasiun 1 dengan stasiun lain ditunjukkan pada Tabel 3.5.

Tabel 3.5. Jarak Stasiun Target dan Stasiun Lainnya

	Stasiun 2	**Stasiun 3**	**Stasiun 4**	**Stasiun 5**
Stasiun 1	10.52 km	18.05 km	13.86 km	8.12 km

Gambar 3.9. Lokasi Stasiun Pemantauan Kualitas Udara DKI Jakarta

Dilakukan perhitungan *Pearson correlation coefficient* untuk menganalisis korelasi spasial antara Stasiun 1 dengan stasiun lain di sekitarnya. Data konsentrasi polutan PM_{10} sebagian ditunjukkan pada Tabel 3.6 dan dihitung

menggunakan rumus (2.14), di mana x_1 untuk Stasiun 1 dan y_2 untuk Stasiun 2 sebagai berikut:

Tabel 3.6. Konsentrasi PM_{10}

	Stasiun 1	**Stasiun 2**	**Stasiun 3**	**Stasiun 4**	**Stasiun 5**
0	29	14	21	16	10
1	24	20	20	17	7
2	16	16	23	15	9
:	:	:	:	:	:
1094	55	47	60	60	53
1095	62	61	64	58	60

$$r_{x_1y_2} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

$$r_{x_1y_2} = \frac{181380.3}{\sqrt{168788.9}\sqrt{256561.2}}$$

$$r_{x_1y_2} = \frac{181380.3}{(410.8392)(506.5187)}$$

$$r_{x_1y_2} = \frac{181380.3}{208097.8}$$

$$r_{x_1y_2} = 0.8716 \approx 0.872$$

Hasil perhitungan korelasi spasial PM_{10} antara Stasiun 1 dengan stasiun lainnya menggunakan *Pearson correlation coefficient* seperti yang ditunjukkan pada Tabel 3.7.

Tabel 3.7. *Pearson Correlation Coefficient* Polutan antara Stasiun 1 dengan Stasiun Lainnya

	PM_{10}	**SO_2**	**CO**	**O_3**	**NO_2**	**ISPU**
Stasiun 1 & 2	**0.872**	0.613	**0.620**	0.711	**0.741**	**0.830**
Stasiun 1 & 3	0.681	**0.703**	0.454	**0.747**	0.717	0.632
Stasiun 1 & 4	0.712	0.528	0.194	0.590	0.242	0.705
Stasiun 1 & 5	**0.847**	0.478	0.201	0.655	0.209	**0.810**

Kombinasi antara Gambar 3.9 dan Tabel 3.7 menunjukkan bahwa semakin pendek jarak stasiun dengan Stasiun 1, maka semakin tinggi tingkat korelasi polutan, seperti yang ditunjukkan pada tabel dengan huruf tebal. Selain itu, korelasi koefisien polutan PM_{10} secara umum lebih tinggi dibandingkan polutan lainnya. Sedangkan, semakin bertambahnya jarak stasiun dari Stasiun 1, korelasi koefisien polutan antara Stasiun 1 dengan stasiun lain akan semakin menurun. Pengaruh jarak menunjukkan bahwa di setiap wilayah stasiun memiliki relevansi spasial polutan udara. Oleh karena itu, perlu dilakukan pencegahan polutan lokal untuk mengurangi dampak buruk yang ditimbulkan dari zat polutan.

d. Penentuan *Array* Fitur dan Label

Proses ini dilakukan untuk menentukan *array* berisi fitur yang akan dilatih dalam model prediksi *time series* dan label yang menjadi target *output*. Proses pada tahapan ini dapat dilihat dalam *flowchart* pada Gambar 3.10.

Data asli diubah ke dalam bentuk *array* Numpy untuk memudahkan proses pengolahan data. Kemudian, *array* X dan y diinisialisasi untuk menyimpan data fitur (X) dan label (y). Iterasi dilakukan sebanyak jumlah baris pada data dikurangi *timesteps* atau *window size*. Dengan menggunakan satu hari data historis *spatiotemporal* polutan dan meteorologi, setiap sampel diubah menjadi satu *timesteps* (*window size* = 3), sehingga setiap fitur saat ini terdiri dari satu baris dengan tujuan untuk memprediksi ISPU di hari berikutnya. Satu nilai X sebelumnya digunakan untuk memprediksi nilai y saat ini, di mana

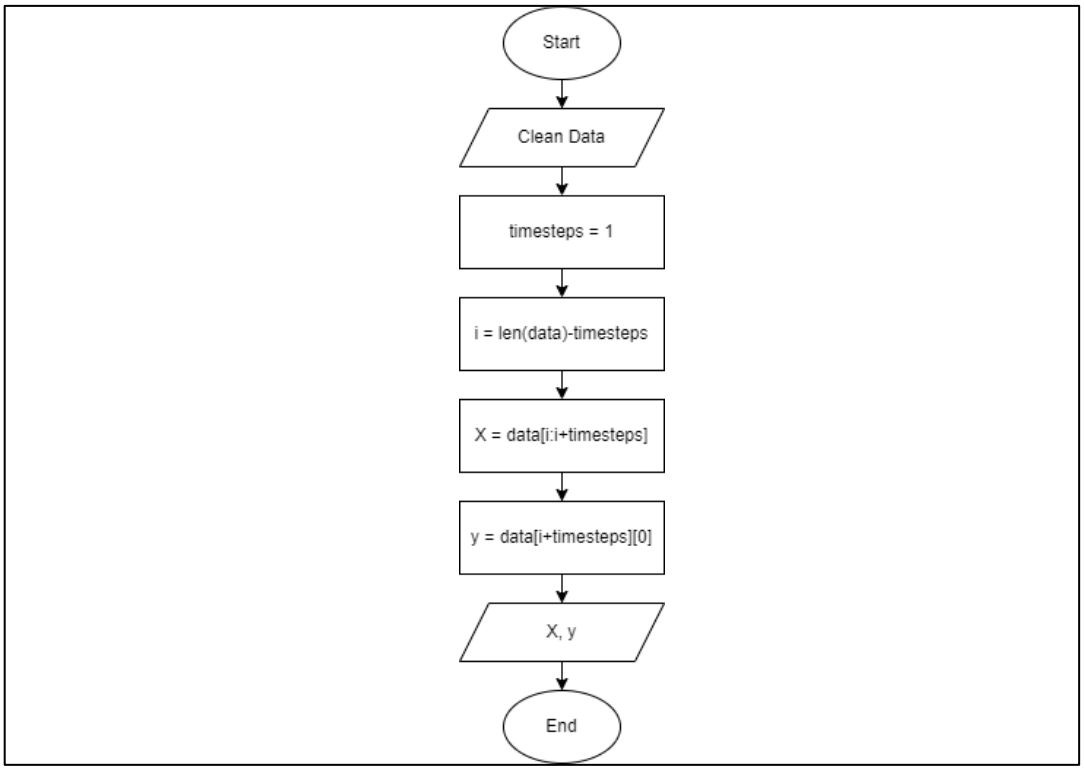
$$X = \left[[29,\ 29,\ 15,\ 5,\ 29,\ 13,\ 26,\ 84,\ 240,\ 2\ ]\right]$$

digunakan untuk memprediksi $y = [29]$, sehingga hasil penentuan *array* fitur dan label adalah sebagai berikut:

$$X = \begin{bmatrix} [[29, & 29, & 15, & 5, & 29, & 13, & 26, & 84, & 240, & 2\ ]], \\ [[29, & 24, & 17, & 5, & 29, & 6, & 28.3, & 75, & 290, & 2\ ]], \\ [[29, & 16, & 16, & 5, & 29, & 4, & 28.3, & 74, & 280, & 2\ ]], \\ & & & & [\ldots], & & & & & \\ [[54, & 31, & 54, & 10, & 24, & 11, & 27.5, & 84, & 320, & 1]], \\ [[55, & 55, & 53, & 16, & 23, & 14, & 28, & 84, & 320, & 1]], \\ [[62, & 62, & 52, & 23, & 20, & 14, & 29.3, & 77, & 260, & 1]] \end{bmatrix}$$

$$y = [29,\ \ 29,\ \ 38,\ \ \ldots,\ \ \ 55,\ \ 62,\ \ 61]$$

Setelah dibagi menjadi fitur dan label, *array* X memiliki bentuk (1096, 1, 10) dan *array* y memiliki bentuk (1096).



Gambar 3.10. *Flowchart Reshaping Array*

e. *Data Splitting*

Tahapan *data splitting* dilakukan untuk membagi data menjadi dua bagian, yaitu data *training* dan data *testing* dengan rasio 7:3 atau 70% data *training* dan 30% data *testing* Bo Zhang (2022). Data dibagi berdasarkan jumlah stasiun, yaitu 5 stasiun, sehingga diperoleh total 1096 data untuk masing-masing stasiun. Masing-masing stasiun memiliki data *training* sebanyak 767 data dan data *testing* sebanyak 329 data. Proses pada tahapan ini dapat dilihat dalam *flowchart* pada Gambar 3.11. Berikut merupakan hasil pembagian data menjadi data *training* dan data *testing*.

Data *training*

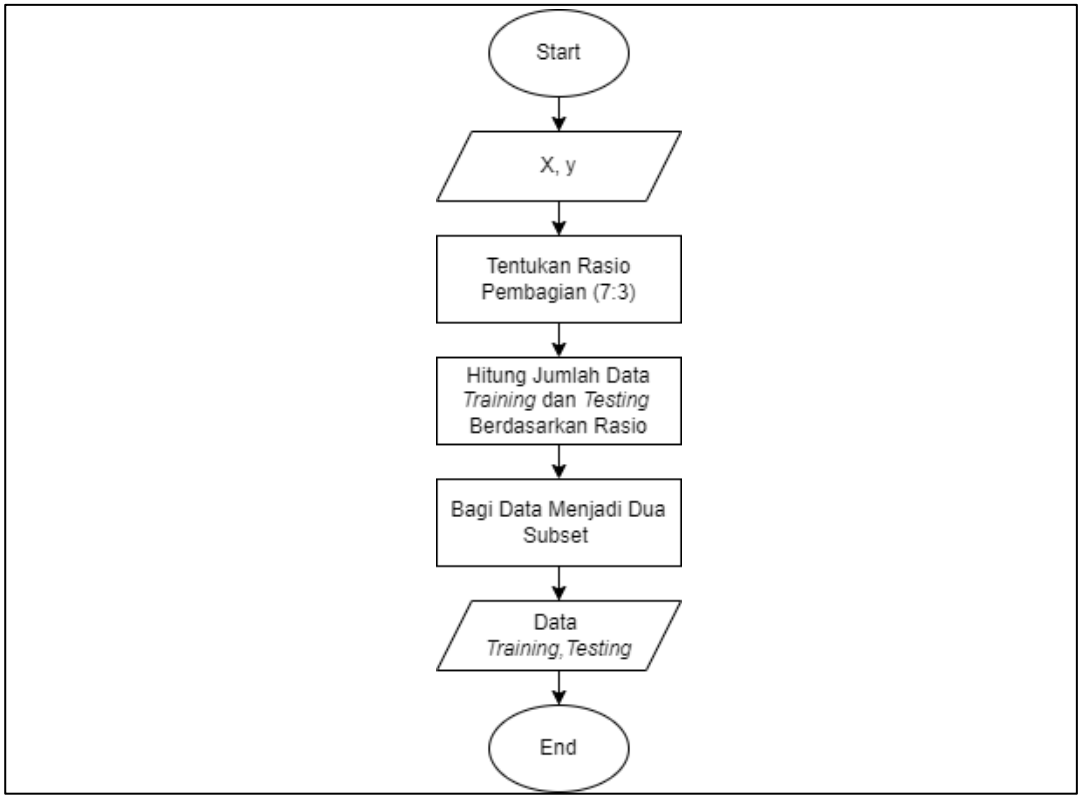
$$X = \begin{bmatrix} [[29, & 29, & 15, & 5, & 29, & 13, & 26, & 84, & 240, & 2\ ]], \\ [[29, & 24, & 17, & 5, & 29, & 6, & 28.3, & 75, & 290, & 2\ ]], \\ [[29, & 16, & 16, & 5, & 29, & 4, & 28.3, & 74, & 280, & 2\ ]], \\ & & & & & \dots, & & & & \\ [[23, & 23, & 19, & 11, & 14, & 19, & 26.5, & 89, & 290, & 1\ ]], \\ [[36, & 36, & 23, & 14, & 12, & 24, & 26.5, & 87, & 320, & 3\ ]], \\ [[29, & 29, & 20, & 14, & 12, & 21, & 27, & 85, & 280, & 3\ ]], \end{bmatrix}$$

$$y = [29, \quad 29, \quad 38, \quad 40.25, \quad 66, \quad 72.75, \quad \dots, \quad 36, \quad 29, \quad 27]$$

Data *testing*

$$X = \begin{bmatrix} [[27, & 27, & 21, & 8, & 21, & 15, & 28.2, & 80, & 300, & 2\ ]], \\ [[25, & 25, & 19, & 7, & 19, & 12, & 27.8, & 82, & 350, & 3\ ]], \\ [[24, & 24, & 19, & 10, & 18, & \ldots, & 17, & 28, & 83, & 330, & 2\ ]], \\ [[54, & 31, & 54, & 10, & 24, & 11, & 27.5, & 84, & 320, & 1]], \\ [[55, & 55, & 53, & 16, & 23, & 14, & 28, & 84, & 320, & 1]], \\ [[62, & 62, & 52, & 23, & 20, & 14, & 29.3, & 77, & 260, & 1]] \end{bmatrix}$$

$$y = [25 \quad 24 \quad 33 \quad 32 \quad 26 \quad 41 \quad \ldots \quad 55 \quad 62 \quad 61]$$



Gambar 3.11. *Flowchart Splitting Data*

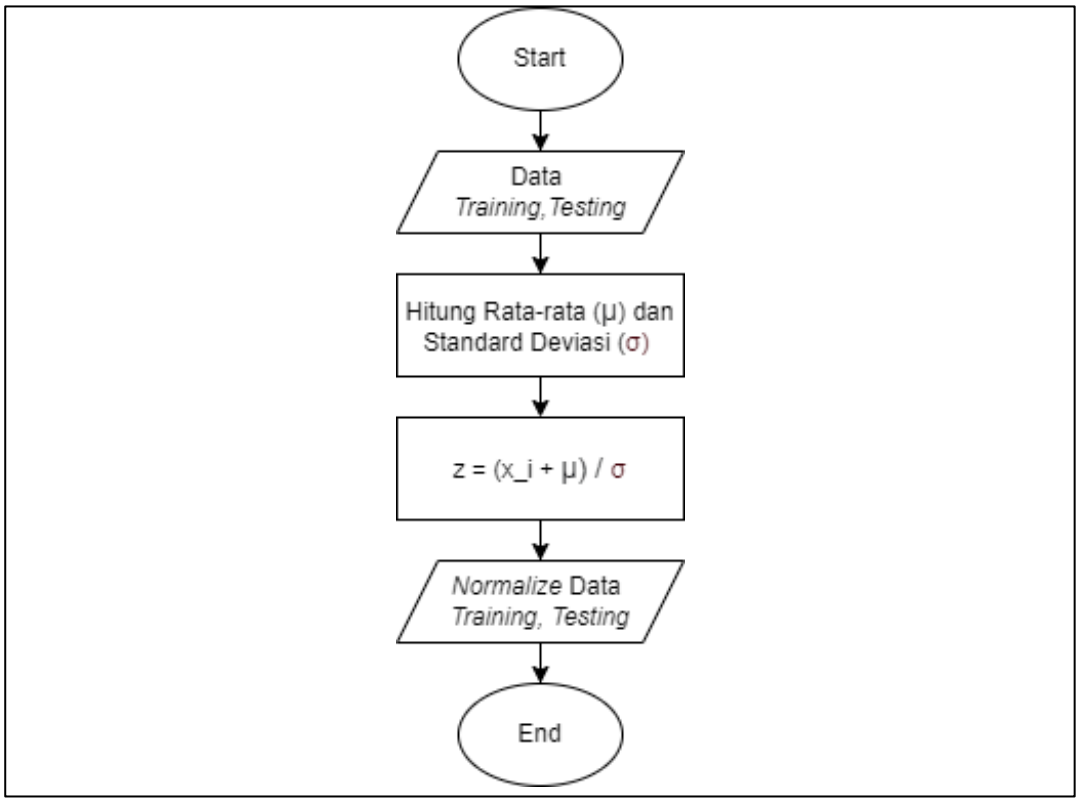
f. *Feature Scaling*

Setelah dibagi mejadi dua subset data *training* dan data *testing*, kemudian dilakukan normalisasi atau *feature scaling* menggunakan Z-*score normalization* untuk menyamakan rentang semua fitur. Nilai mean atau rata-rata sebesar 55.27 dan nilai *standard devation* sebesar 73.52. Proses normalisasi data dapat dilihat dalam *flowchart* pada Gambar 3.12. Perhitungan normalisasi menggunakan rumus (2.15), sebagai berikut:

$$z[0][0] = \frac{X_{[0][0]} - \mu}{\sigma}$$

$$z[0][0] = \frac{29 - 55.27}{73.52}$$

$z[0][0] = -1.64$



Gambar 3.12. *Flowchart Feature Scaling*

Data *training*

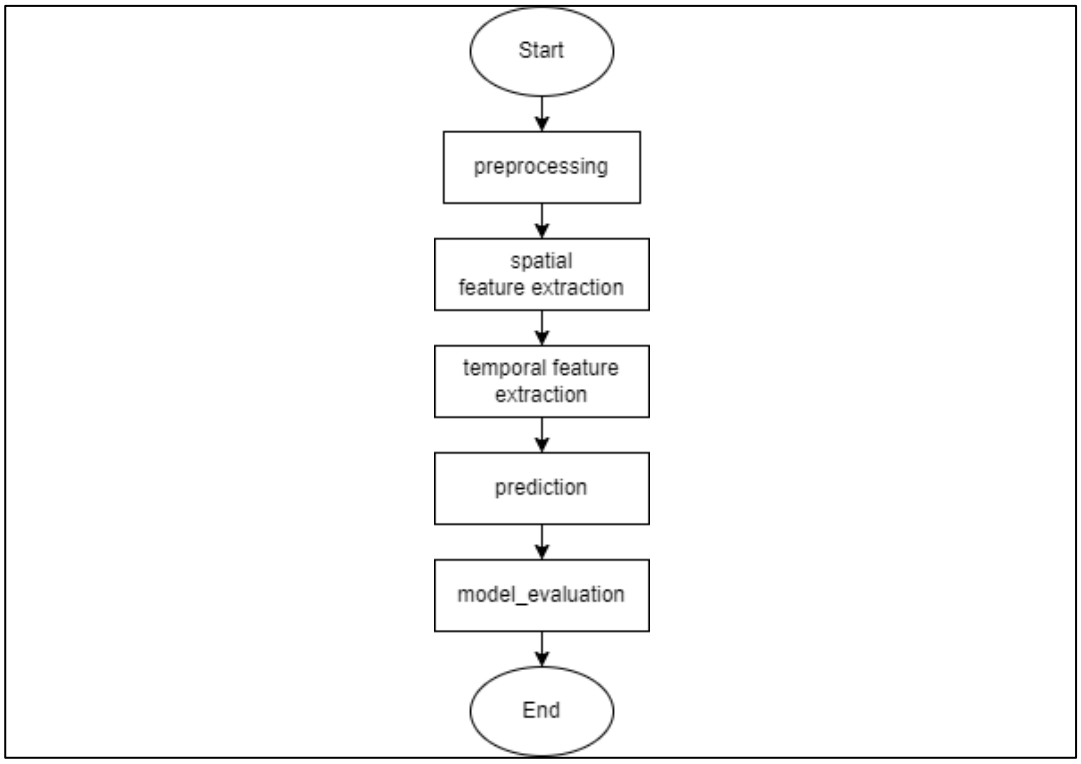
$$X = \begin{bmatrix} [[-0.35, -0.35, -0.54, -0.68, -0.35, -0.57, -0.39, 0.39, 2.51, -0.72\ ]], \\ [[-0.35, -0.42, -0.52, -0.68, -0.35, -0.67, -0.36, 0.26, 3.19, -0.72\ ]], \\ [[-0.35, -0.53, -0.53, -0.68, -0.35, -0.69, -0.36, 0.25, 3.05, -0.72\ ]], \\ \cdots, \\ [[-0.43, -0.43, -0.49, -0.60, -0.56, -0.49, -0.39, 0.45, 3.19, -0.73\ ]], \\ [[-0.26, -0.26, -0.43, -0.56, -0.58, -0.42, -0.39, 0.43, 3.60, -0.71\ ]], \\ [[-0.35, -0.35, -0.47, -0.56, -0.58, -0.46, -0.38, 0.40, 3.05, -0.71\ ]], \end{bmatrix}$$

Data *testing*

$$X = \begin{bmatrix} [[-0.38, -0.38, -0.46, -0.64, -0.46, -0.54, -0.36, 0.33, 3.32, -0.72\ ]], \\ [[-0.41, -0.41, -0.49, -0.65, -0.49, -0.58, -0.37, 0.36, 4.01, -0.71\ ]], \\ [[-0.42, -0.42, -0.49, -0.61, -0.51, -0.52, -0.37, 0.37, 3.73, -0.72\ ]], \\ \cdots, \\ [[-0.02, -0.33, -0.02, -0.61, -0.42, -0.60, -0.37, 0.39, 3.60, -0.73\ ]], \\ [[-0.003, -0.003, -0.03, -0.53, -0.43, -0.56, -0.37, 0.39, 3.60, -0.73\ ]], \\ [[0.09, 0.09, -0.04, -0.43, -0.47, -0.56, -0.35, 0.29, 2.78, -0.73\ ]], \end{bmatrix}$$

3.1.4 Pembangunan Model

Pada tahapan ini dilakukan pembuatan model *machine learning* dengan melatih semua data yang telah melalui tahap *preprocessing* menggunakan metode *Residual Network* dan *Long Short-Term Memory*. Arsitektur ResNet dan LSTM yang digunakan dalam penelitian ini merupakan hasil kustomisasi yang didasarkan pada arsitektur ResNet dan LSTM dalam penelitian Q. Zhang (2020). Optimasi terhadap model juga dilakukan untuk mencari model dengan performa terbaik dengan cara menguji parameter penting dalam pelatihan model. Adapun proses utama dalam sistem yang dibentuk pada penelitian ini tergambarkan pada Gambar 3.13. Terdapat lima buah proses utama yaitu, *preprocessing*, *spatial feature extraction, temporal feature extraction, prediction*, dan *model evaluation*.

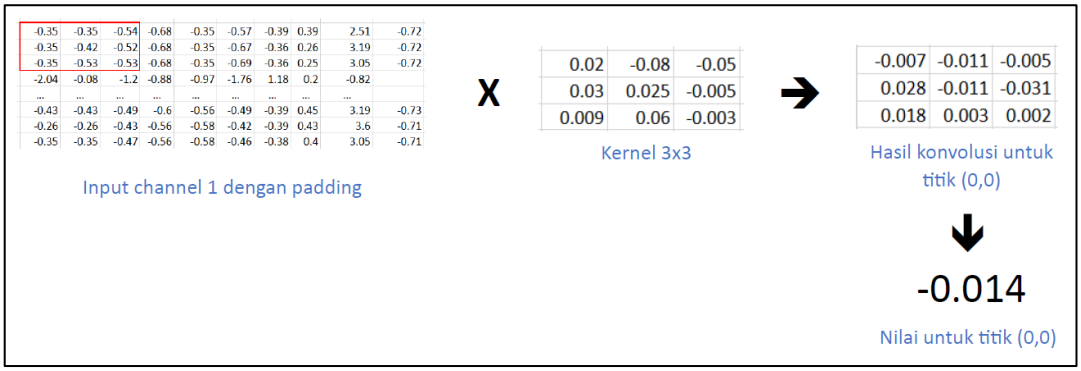


Gambar 3.13. *Flowchart* Utama

a. Residual Network (ResNet)

Penelitian ini menggunakan metode *Residual Network* (ResNet) untuk melakukan esktraksi fitur spasial pada data *spatiotemporal*. Proses pelatihan pada model ResNet menggunakan *dataset* berupa data historis satu dimensi yang diubah menjadi matriks dua dimensi, di mana n adalah jumlah lapisan yang dilatih dan m adalah *feature map*. Data yang digunakan merupakan data yang telah melalui tahap *preprocessing*. Agar dapat dilatih dalam model ResNet, perlu ditambahkan dimensi *channel* dengan nilai 1dalam struktur data X, baik pada data *training* maupun data *testing*. Dengan demikian, bentuk array akan menjadi (767, 1, 10, 1) untuk data *training* dan (329, 1, 10, 1) untuk data *testing*.

Lapisan konvolusi pertama pada penelitian ini memiliki *filter* sebanyak 3, *kernel* berukuran 3x3 *pixel*, *stride* sebesar 1, dan *padding "same"*. Dengan memberikan *padding "same"* dan *stride* sebesar 1, ukuran dimensi spasial pada *output* akan tetap sama dengan *input*. Perbedaan terjadi pada jumlah *channel* yang berubah menjadi 3 pada *feature map* sesuai dengan jumlah *filter*. Proses dari konvolusi pertama ini dapat dilihat pada Gambar 3.14.



Gambar 3.14. Perhitungan Konvolusi Channel 1 dengan Kernel

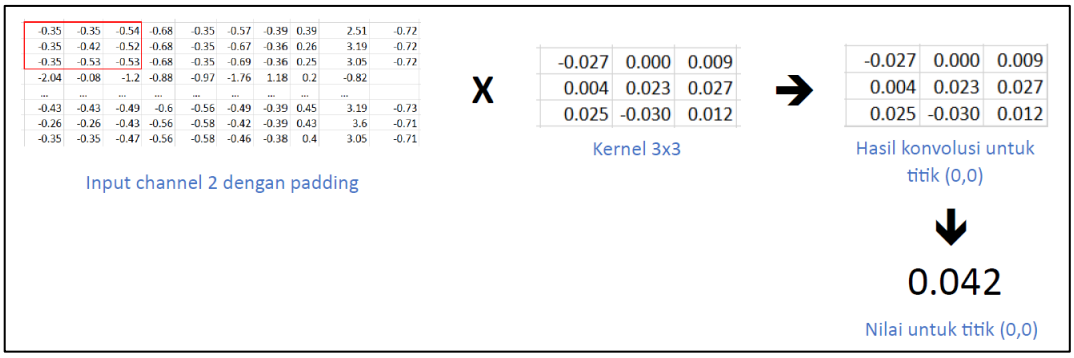
Pada Gambar 3.14, area dengan kotak merah pada *channel* 1 dikalikan dengan kernel untuk *channel* 1 dan menghasilkan bobot untuk pixel (0,0) yang disebut sebagai $C1_{0,0}$. Perhitungan $C1_{0,0}$ untuk menghasilkan nilai -0.014 adalah sebagai berikut.

$$C1_{0,0} = (x_{0,0} * k_{0,0}) + (x_{0,1} * k_{1,0}) + (x_{0,2} * k_{2,0}) + (x_{1,0} * k_{0,1}) + (x_{1,1} * k_{1,1}) + (x_{1,2} * k_{2,1}) + (x_{2,0} * k_{0,2}) + (x_{2,1} * k_{1,2}) + (x_{2,2} * k_{2,2})$$

$$C1_{0,0} = (-0.35 * 0.02) + (-0.35 * 0.03) + (-0.54 * 0.009) + (-0.35 * -0.08) + (-0.42 * 0.025) + (-0.52 * 0.06) + (-0.35 * -0.05) + (-0.53 * -0.0048) + (-0.53 * -0.003)$$

$$C1_{0,0} = -0.014$$

Perhitungan untuk *channel* 2 dan 3 dilakukan dengan cara yang sama untuk menghasilkan bobot $C1_{0,0}$. Ilustrasi perhitungan konvolusi pada *channel* 2 dan 3 dapat dilihat pada Gambar 3.15 yang menghasilan bobot $C2_{0,0}$ dan Gambar 3.16 yang menghasilkan bobot $C3_{0,0}$.

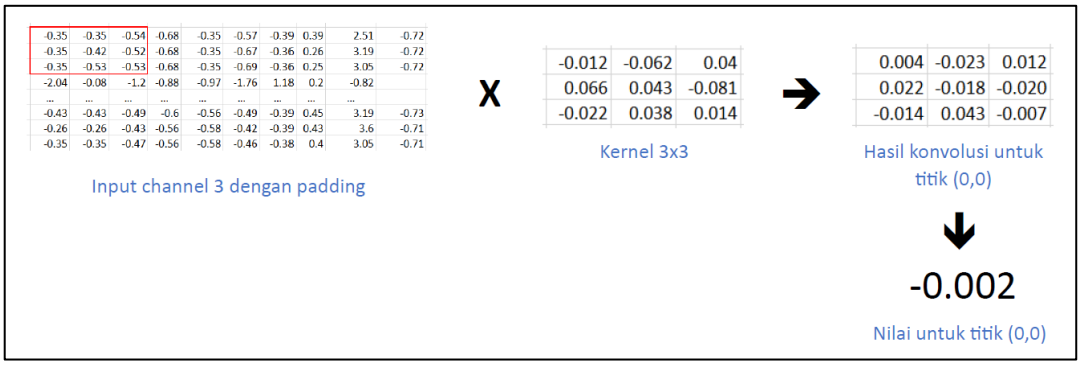


Gambar 3.15. Perhitungan Konvolusi Channel 2 dengan Kernel

$$C2_{0,0} = (x_{0,0} * k_{0,0}) + (x_{0,1} * k_{1,0}) + (x_{0,2} * k_{2,0}) + (x_{1,0} * k_{0,1}) + (x_{1,1} * k_{1,1}) + (x_{1,2} * k_{2,1}) + (x_{2,0} * k_{0,2}) + (x_{2,1} * k_{1,2}) + (x_{2,2} * k_{2,2})$$

$$C2_{0,0} = (-0.35 * 0.077) + (-0.35 * 0.0005) + (-0.54 * -0.016) + (-0.35 * -0.01) + (-0.42 * -0.055) + (-0.52 * -0.051) + (-0.35 * -0.071) + (-0.53 * 0.056) + (-0.53 * -0.023)$$

$$C2_{0,0} = 0.042$$



Gambar 3.16. Perhitungan Konvolusi Channel 3 dengan Kernel

$$C3_{0,0} = (x_{0,0} * k_{0,0}) + (x_{0,1} * k_{1,0}) + (x_{0,2} * k_{2,0}) + (x_{1,0} * k_{0,1}) + (x_{1,1} * k_{1,1}) + (x_{1,2} * k_{2,1}) + (x_{2,0} * k_{0,2}) + (x_{2,1} * k_{1,2}) + (x_{2,2} * k_{2,2})$$

$$C3_{0,0} = (-0.35 * -0.012) + (-0.35 * 0.066) + (-0.54 * -0.022) + (-0.35 * -0.062) + (-0.42 * 0.043) + (-0.52 * 0.038) + (-0.35 * 0.04) + (-0.53 * -0.081) + (-0.53 * 0.014)$$

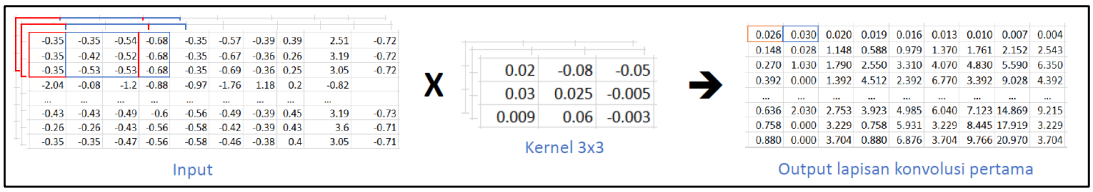
$$C3_{0,0} = -0.002$$

Kemudian, seluruh bobot dijumlahkan untuk memperoleh nilai pada titik (0,0) pada hasil luaran konvolusi lapisan pertama. Pada lapisan konvolusi ini,

tidak ada fungsi aktivasi yang digunakan, sehingga nilai dari hasil konvolusi akan tetap sama. Perhitungan dari penjumlahan bobot untuk pixel (0,0) sebagai berikut.

$$Y^1_{0,0} = C1_{0,0} + C2_{0,0} + C3_{0,0}$$
$$Y^1_{0,0} = -0.014 + 0042 + (-0.002)$$
$$Y^1_{0,0} = 0.026$$

Pada Gambar 3.14 hingga Gambar 3.16, kotak merah pada *input* adalah area yang dikonvolusi untuk menghasilkan pixel (0,0) pada *output*. Sedangkan, untuk menghasilkan pixel (0,1) pada *output*, perlu dilakukan konvolusi di area kotak biru pada *input* yang merupakan pergeseran sebanyak 1 pixel ke kanan sesuai dengan jumlah *stride*. Proses pada Gambar 3.17 diulang dengan *filter* yang berbeda sehingga terbentuklah lapisan konvolusi pertama dari arsitektur ResNet. Hasil konvolusi tersebut kemudian dimasukkan ke proses *batch normalization* untuk mengurangi risiko *overfitting* dan mempercepat proses pelatihan.



Gambar 3.17. Hasil Konvolusi Lapisan Pertama

Batch normalization dihitung menggunakan nilai rata-rata (*mean)* dan *variance* dari sebuah *batch.* Pada perhitungan *batch normalization* ini, satu *batch* hanya memiliki satu *channel* yang merupakan *output* dari konvolusi lapisan pertama pada Gambar 3.17. Perhitungan *mean* dan *variance* dari *output* lapisan konvolusi pertama adalah sebagai berikut.

$$\mu = \frac{1}{764}\sum_{i=1}^{764} x_i = \frac{0.026+0.030+0.020+\cdots+9.766+20.970+3.704}{767} = 3.468$$
$$\sigma^2 = \frac{1}{764}\sum_{i=1}^{764}(x_i - \mu)^2 = \frac{(0.026-3.468)^2+\cdots+(3.704-3.468)^2}{767} = 17.639$$

Setelah diperoleh nilai *mean* dan *varieance*, maka nilai normalisasi dapat dihitung. Epsilon (ε) dengan nilai 0.00001 digunakan dalam perhitungan normalisasi untuk menghindari pembagian dengan angka nol. Berikut adalah perhitungan *batch normalization* pada pixel (0,0).

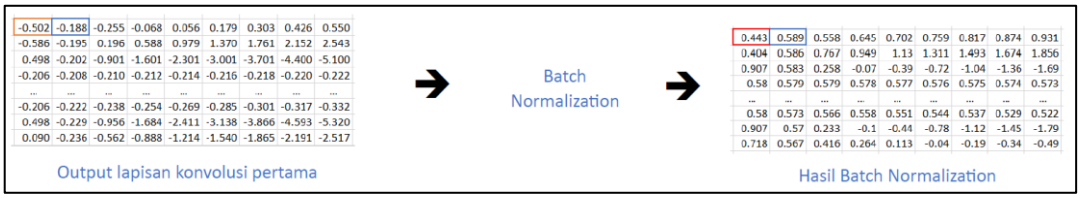
$$\hat{x}_{(0,0)} = \frac{x_i - \mu}{\sqrt{\sigma^2 - \varepsilon}} = \frac{0.026-3.468}{\sqrt{17.639-0.00001}} = -0.819$$

Nilai yang sudah dinormalisasi kemudian dikalikan dengan gamma (γ) dan ditambahkan dengan beta (β). Operasi ini disebut *scale and shift*, di mana gamma

dan beta merupakan parameter yang dinamis dan akan diperbarui selama proses pelatihan. Pada contoh ini, diasumsikan nilai gamma adalah 0.747 dan nilai beta adalah 0.32. Perhitungan untuk menghasilkan $Y_{0,0}^{1}$ yang merupakan hasil akhir dari perhitungan *batch normalization* adalah sebagai berikut

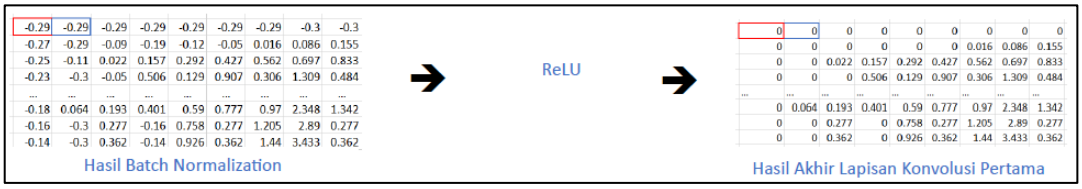
$$Y_{0,0}^{1} = \gamma \hat{x}_{(0,0)} + \beta = (0.747)(-0.819) + (0.32) = -0.291$$

Proses perhitungan normalisasi ini diimplementasikan ke seluruh pixel dalam *batch* tersebut. Hasil perhitungan *batch normalization* dari Gambar 3.17 dapat dilihat pada Gambar 3.18.



Gambar 3.18. Ilustrasi *Batch Normalization*

Hasil *batch normalizaion* kemudian dimasukkan ke dalam fungsi aktivasi ReLU sehingga tidak ada nilai negatif. Aktivasi ReLU dari hasil *bacth normalization* pada Gambar 3.18, dapat dilihat pada Gambar 3.19.



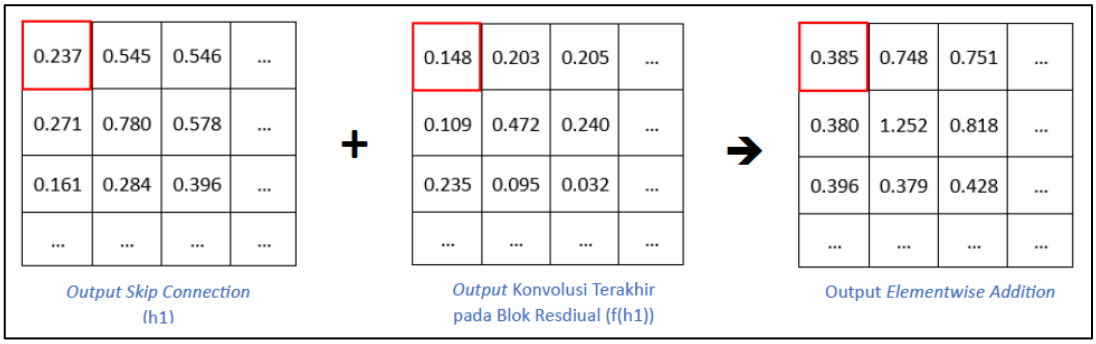
Gambar 3.19. Ilustrasi Fungsi Aktivasi ReLU

Hasil dari operasi fungsi aktivasi ReLU menjadi hasil akhir lapisan konvolusi pertama dan selanjutnya akan dilakukan konvolusi kembali dengan detail sebagai berikut.

Tabel 3.8. *Layer* Konvolusi ResNet

Nama Layer	Ukuran Output	Blok Residual
Conv_2x	$(1, 10, 9)$	$\begin{bmatrix} 3 \times 3, & 9 \\ 1 \times 1 & 9 \end{bmatrix}$
Conv_3x	$(1, 10, 16)$	$\begin{bmatrix} 3 \times 3, & 16 \\ 1 \times 1 & 16 \end{bmatrix}$
Conv_4x	$(1, 10, 64)$	$\begin{bmatrix} 3 \times 3, & 64 \\ 1 \times 1 & 64 \end{bmatrix}$
Conv_5x	$(1, 10, 128)$	$\begin{bmatrix} 3 \times 3, & 128 \\ 1 \times 1 & 128 \end{bmatrix} \times 2$

Input blok residual merupakan operasi penjumlahan antara *output* residual blok dengan *skip connection* yang disebut operasi *elementwise addition*. Pada penelitian ini, *output* residual blok dan *skip connection* telah dinormalisasi dengan *batch normalization* dan diaktivasi dengan ReLU. Apabila tidak ada penyesuaian dimensi yang diperlukan, maka dapat menggunakan *identity shortcut*, yaitu nilai dari *skip connection* langsung ditambahkan ke *output* dari blok residual.



Gambar 3.20. Ilustrasi Operasi *Elementwise Addition*

Pada operasi *elementwise addition, output* dari *skip connection* yang telah dinormaliasi dan diaktivasi akan dijumlahkan dengan *output* dari konvolusi terakhir pada blok residual yang telah dinormalisasi dan diaktivasi pula. Operasi ini dilakukan pada semua saluran (*channel*), sehingga jumlah saluran pada *output skip connection* dan konvolusi terakhir pada blok residual harus sama. Perhitungan dari *elementwise addition* pada pixel (0,0) pada Gambar 3.20. sebagai berikut.

$$Y^1_{0,0} = h1^1_{0,0} + f(h^1_{0,0}) = (0.237) + (0.148) = 0.385$$

Operasi *elementwise addition* pada blok residual kelima menghasilkan *output* dengan ukuran (1, 10, 128). Kemudian, *output* dari ResNet ini dibentuk kembali (*reshape*) agar sesuai dengan bentuk masukan LSTM menjadi (1, 1280), di mana 1 adalah *timestep* dan 1280 adalah jumlah fitur yang diperoleh dari perkalian jumlah fitur dan *channel* pada lapisan konvolusi.

b. Long Short-Term Memory (LSTM)

Lapisan LSTM dengan 64 unit digunakan untuk mengolah urutan *input* dari proses jaringan konvolusi dan residual dan mengembalikan urutan *output* untuk setiap *timestep*. Kemudian, hasil pengolahan data pada lapisan LSTM dinormalisasi menggunakan *batch normalization* dan diaktivasi menggunakan fungsi aktivasi ReLU. Pemilihan 64 unit pada lapisan LSTM memberikan kapasitas yang cukup untuk menangkap pola dan keterkaitan fitur dalam urutan *input*. Parameter *return_sequences "true"* membuat lapisan ini mengembalikan *output* untuk setiap *timestep* dalam urutan *input*.

Lapisan LSTM memerlukan nilai *weight* (bobot) dan *bias* untuk melakukan proses kalkulasi. Kedua nilai ini didapatkan salaam proses inisialisasi model dan diperbarui selama proses pelatihan menggunakan algoritma *Stochastic Gradient Descent* (SGD). Lapisan LSTM memiliki empat *gate* untuk setiap unitnya, sehingga diperlukan empat nilai *weight* berbeda untuk setiap *gate* dalam satu unit tersebut. Lapisan LSTM dalam penelitian ini akan memiliki total 768 nilai *weight*, di mana 256 *input*, 256 *hidden state*, dan 256 *cell state*. Berikut simulasi kalkulasi nilai pada lapisan LSTM dengan kondisi di mana:

a. Nilai *input* yang didapatkan dari satu *keypoint* pada hasil *output* ResNet adalah 0.5.
b. Nilai *input weight, hidden state weight,* dan *cell state weight* untuk setiap *gate* dalam satu unit adalah sama.
c. Lapisan LSTM hanya memiliki satu unit.
d. Nilai *mean, variance, gamma,* dan *beta* secara berurutan adalah 3.5, 15, 0.747, dan 0.32

adalah sebagai berikut,

1. Inisialisasi nilai:
   Bobot *input* ke unit LSTM ($W_{xi}, W_{xf}, W_{xc}, W_{xo}$) = -1.82
   Bobot *hidden state* ke unit LSTM ($W_{hi}, W_{hf}, W_{hc}, W_{ho}$) = -1.65
   Bobot *cell state* ke unit LSTM ($W_{ci}, W_{cf}, W_{cc}, W_{co}$) = -5.79
2. Perhitungan:

$$x_1 = 0.5$$

$$\begin{aligned} i_1 &= \sigma(W_{xi}x_1 + W_{hi}h_0 + W_{ci}c_0 + b_i) \\ &= \sigma(-1.82 * 0.5 + (-1.65) * 0 + (-5.79) * 0 + 0) \\ &= \sigma(-0.91) \\ &= 0.286 \end{aligned}$$

$$\begin{aligned} f_1 &= \sigma(W_{xf}x_1 + W_{hf}h_0 + W_{cf}c_0 + b_f) \\ &= \sigma(-1.82 * 0.5 + (-1.65) * 0 + (-5.79) * 0 + 0) \\ &= \sigma(-0.91) \\ &= 0.286 \end{aligned}$$

$$\begin{aligned} \tilde{c}_1 &= \tanh(W_{xc}x_1 + W_{hc}h_0 + b_f) \\ &= \tanh(-1.82 * 0.5 + (-1.65) * 0 + 0) \\ &= \sigma(-0.91) \\ &= -0.721 \end{aligned}$$

$$\begin{aligned} c_1 &= f_1c_0 + i_1 \circ \tilde{c}_1 \\ &= 0.286 * 0 + 0.286 * (-0.721) \\ &= -0.206 \end{aligned}$$

$$\begin{aligned} o_1 &= \sigma(W_{xo}x_1 + W_{ho}h_0 + W_{co}c_1 + b_o) \\ &= \sigma(-1.82 * 0.5 + (-1.65) * 0 + (-5.79) * (-0.206) + 0) \\ &= \sigma(0.281) \\ &= 0.570 \end{aligned}$$

$$h_1 = o_1 \tanh(c_1)$$
$$= 0.570 * \tanh(-0.206)$$
$$= 0.570 * (-0.203)$$
$$= -0.115$$
$$bn = \frac{h_1 - \mu}{\sqrt{\sigma^2 - \epsilon}} * 0.747 + 0.32$$
$$= \frac{-0.115 - 3.5}{\sqrt{(15 - 0.00001)}} * 0.747 + 0.32$$
$$= -0.377$$
$$= ReLU(-0.377)$$
$$h_1 = 0$$

c. *Fully Connected Layer* (FCN)

Selain lapisan LSTM, lapisan Dense juga memerlukan nilai *weight* dan *bias*. Namun, pada lapisan ini, hanya akan terdapat satu nilai *weight* dan *bias* pada setiap unit.

1. Inisialisasi nilai:
   Bobot ke unit Dense *output* (*W*) = -0.44
   Bias ke unit Dense *output* (*b*) = 1.90
2. Perhitungan:

$$x = h_1$$
$$= 0$$
$$y = W * x_4 + b$$
$$= -0.44 * 0 + 1.90$$
$$= 1.90$$

3.1.5 Pelatihan Model

Pelatihan model dilakukan melalui dua tahap, yaitu *model compiling* dan *model fitting*. Tahapan *model compiling* dilakukan menggunakan optimasi *Stochastic Gradient Descent* (SGD) dan fungsi *loss* dan *metric Mean Absolute Percentage Error*. Tahapan *model fitting* dilakukan dengan nilai *epoch* sebesar 100. Pada tahapan ini didekalarasikan *validation split* yang digunakan untuk membagi data pelatihan menjadi subset pelatihan dan validasi sebesar 20%, di mana sebesar 80% atau 613 data untuk data latih dan 20% atau 154 data untuk data validasi.

Penggunaan subset validasi sangat penting untuk memonitor kinerja model selama pelatihan, di mana pada setiap *epoch* model dievaluasi menggunakan data validasi yang tidak digunakan dalam pelatihan. Dengan demikian, metrik evaluasi dapat dipantau pada data yang tidak digunakan untuk melatih model dan mengevaluasi sejauh mana model mampu menggeneralisasi pada data baru.

Nilai *batch size* yang digunakan dalam pelatihan data sebesar 64. *Batch size* menunjukkan jumlah sampel atau data yang diproses secara bersamaan dalam satu perhitungan. Dengan menggunakan *batch size* 64, total *batch* yang digunakan

untuk melatih seluruh data pada model ini adalah 10 yang diperoleh dari hasil pembagian jumlah data latih dengan ukuran *batch size* (613 / 64 = 9.5 ≈ 10).

3.1.6 Analisis dan Evaluasi Hasil

Hasil prediksi menggunakan ResNet-LSTM selanjutnya dievaluasi untuk mengetahui tingkat kesalahan relatif dari model dalam memprediksi nilai aktual. Evaluasi pada penelitian ini menggunakan MAPE dengan hasil *training error* sebesar 14.7% dan *testing error* menggunakan data validasi sebesar 23%. Sedangkan, *testing error* menggunakan data *testing* sebesar 15.4%.

Terdapat lima skenario yang akan diuji dalam penelitian ini, yaitu prediksi konsentrasi polutan menggunakan LSTM, CNN, ResNet, CNN-LSTM, dan ResNet-LSTM. Kelima skenario tersebut digunakan untuk melihat hasil akurasi prediksi pada model yang model yang diusulkan optimal untuk mengatasi *vanishing gradient*.

3.2 Metodologi Pengembangan Sistem

Metode pengembangan sistem pada penelitian ini menggunakan metode *waterfall* dengan tahapan sebagai berikut.

3.2.1 *Requirements*

Tahapan pertama dari pengembangan sistem ini adalah menganalisis kebutuhan sistem baik secara fungsional maupun non-fungsional. Berdasarkan analisis yang dilakukan, kebutuhan sistem pada penelitian ini secara fungsional, yaitu:

1. Sistem dapat menerima masukan berupa tanggal dan lokasi prediksi.
2. Sistem dapat melakukan proses prediksi konsentrasi polutan dalam kualitas udara.
3. Sistem dapat menampilkan hasil prediksi konsentrasi polutan.

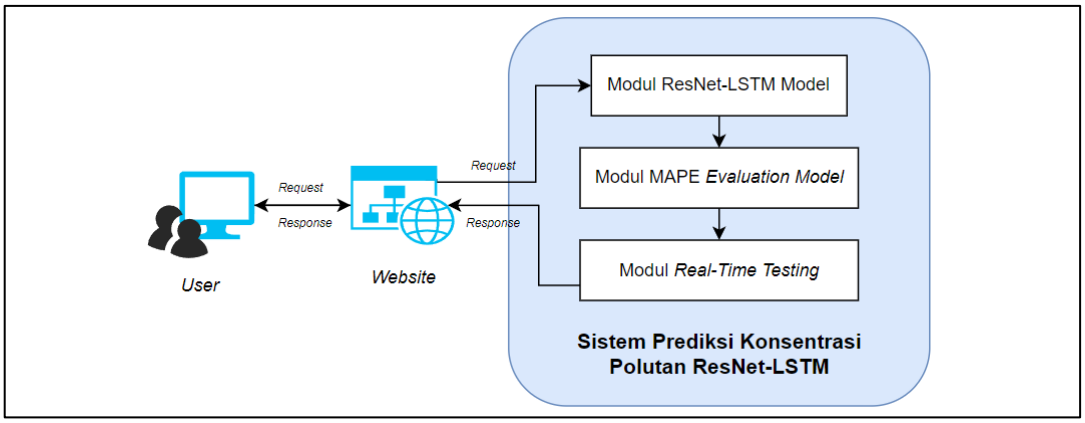
Selanjutnya, kebutuhan sistem pada penelitian ini secara non-fungsional berupa kebutuhan perangkat lunak dan perangkat keras, yaitu:

1. Laptop pribadi dengan spesifikasi sebagai berikut:
   a. Processor AMD Ryzen 5 4500U.
   b. RAM 8 GB.
   c. Sistem Operasi Windows 11.
2. IDE Visual Studio Code
3. IDE Google Colab
4. Bahasa pemrograman Python versi 3.10.12 dengan *library* sebagai berikut
   a. Tensorflow versi 2.15.0
   b. Pandas versi 1.5.3
   c. Numpy versi 1.25.2

3.2.2 *Design*

a. Perancangan Arsitektur

Tahapan selanjutnya adalah merancang arsitektur sistem agar hasil implementasi sesuai dengan kebutuhan sistem yang telah diidentifikasi. Arsitektur sistem dapat dilihat pada Gambar 3.21.



Gambar 3.21. Arsitektur Sistem

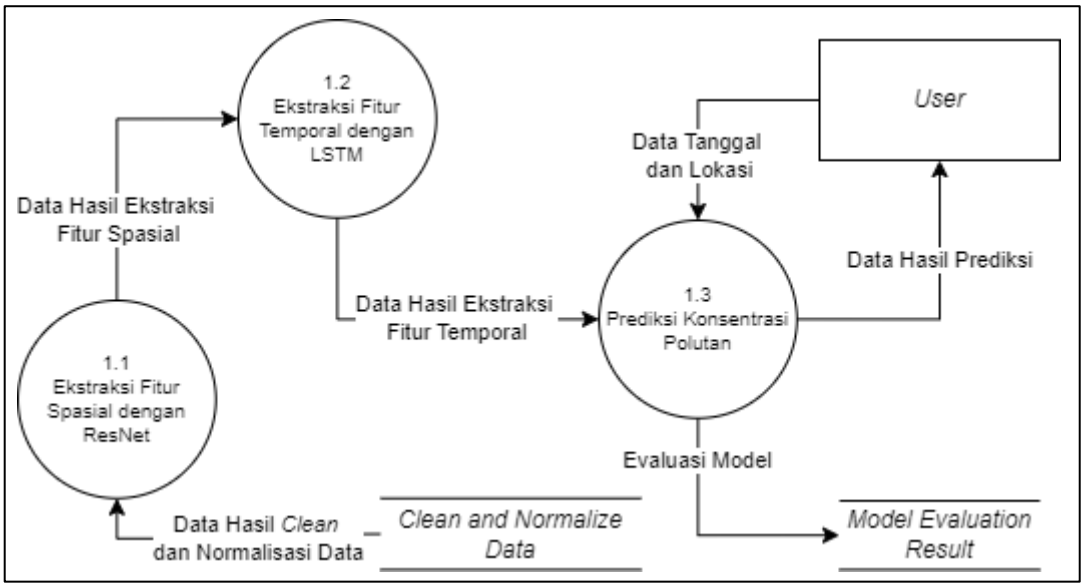
Arsitektur sistem terdiri dari tiga komponen utama, yaitu *user*, *website*, dan sistem prediksi konsentrasi polutan ResNet-LSTM. *User* atau pengguna adalah orang yang mengoperasikan sistem aplikasi prediksi konsentrasi polutan. *User* dapat memasukkan tanggal dan lokasi yang ingin diprediksi konsentrasi polutannya. Sistem akan menggunakan dan mempelajari pola data historis polutan dan meteorologi yang tersedia untuk menghasilkan prediksi pada tanggal dan lokasi yang dipilih pengguna. Proses *preprocessing* dan prediksi akan dilakukan menggunakan model prediksi konsentrasi polutan ResNet-LSTM yang telah dilatih sebelumnya.

b. Perancangan Proses

Perancangan selanjutnya adalah perancangan proses yang dilakukan dengan membuat *Data Flow Diagram* (DFD) level 0, level 1, dan level 2. Rancangan DFD level 0 dapat dilihat pada Gambar 3.22.

Gambar 3.22. DFD Level 0

Pada DFD level 0, terdapat satu entitas dan satu proses yang berinteraksi. Sistem prediksi akan menerima masukan berupa tanggal dan lokasi yang ingin diprediksi dan kemudian memberikan luaran berupa hasil prediksi berdasarkan tanggal dan lokasi yang dipilih. Luaran tersebut berasal dari beberapa proses yang digambarkan lebih lanjut dalam DFD level 1 pada Gambar 3.23.



Gambar 3.23. DFD Level 1

Proses prediksi konsentrasi polutan dalam kualitas udara dengan metode ResNet-LSTM akan menghasilkan prediksi konsentrasi polutan sesuai tanggal dan lokasi yang diinputkan oleh *user* seperti pada Gambar 3.24. Terdapat tiga proses dalam pembangunan model, yaitu ekstraksi fitur spasial menggunakan ResNet, esktraksi fitur temporal menggunakan LSTM, dan prediksi konsentrasi polutan menggunakan *Fully Connected Layer*. Data bersih yang sudah dinormalisasi dan di*reshape*, kemudian dimasukkan ke dalam model ResNet untuk diekstrak fitur atau informasi spasial dari tiap *timesteps*. ResNet secara

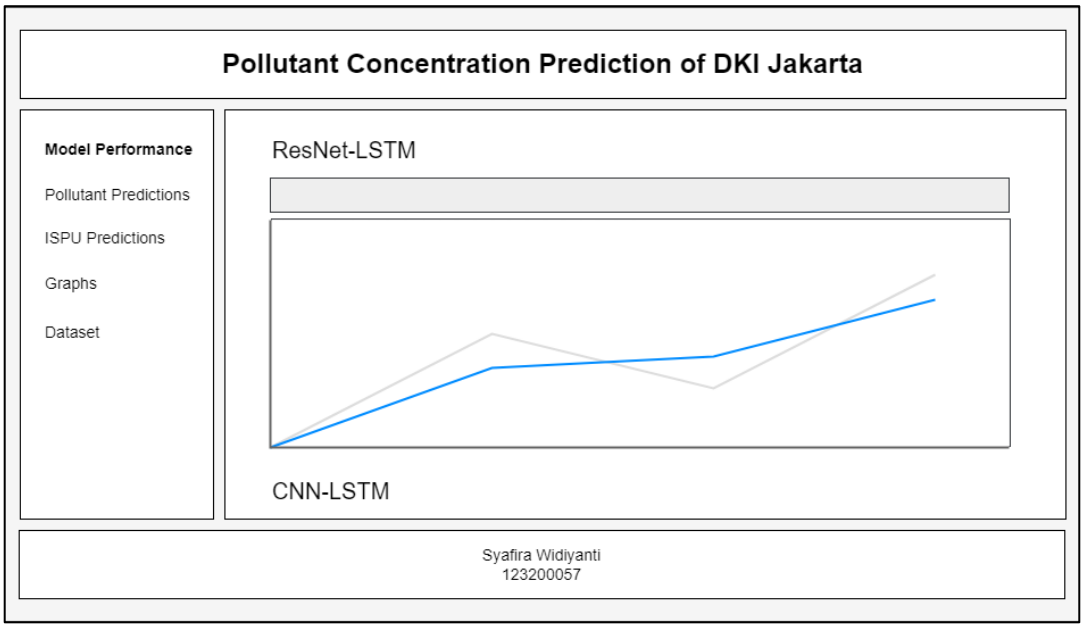
otomatis akan mempelajari fitur dari data masukan, sehingga tidak perlu melakukan ekstraksi fitur sebelum pelatihan model. Selanjutnya, hasil luaran ResNet dijadikan sebagai input LSTM untuk diekstrak fitur temporalnya. Terakhir, hasil prediksi keluaran LSTM didekodekan oleh *Fully Connected Layer* dan diperoleh hasil akhir prediksi konsentrasi polutan.

c. Perancangan Desain Antarmuka

Perancangan terakhir adalah desain antarmuka sistem. Perancangan antarmuka menjadi gambaran bagaimana sistem akan dibangun. Tahapan ini menjadi acuan dalam pembuatan *user interface* saat proses *development* dimulai. Beberapa rincian dari rancangan antarmuka adalah sebagai berikut.

1. Rancangan Halaman *Model Performance*

   Halaman *model performance* akan menampilkan grafik hasil prediksi menggunakan ResNet-LSTM pada data dibandingkan nilai aktual konsentrasi polutan. Pada halaman ini juga ditampilkan hasil evaluasi berupa nilai MAPE dari *testing loss* selama proses pelatihan. Selain itu, pada halaman ini juga akan menampilkan hasil evaluasi MAPE dan grafik metode CNN-LSTM, ResNet, CNN, dan LSTM. Tampilan halaman ditunjukkan pada Gambar 3.24.

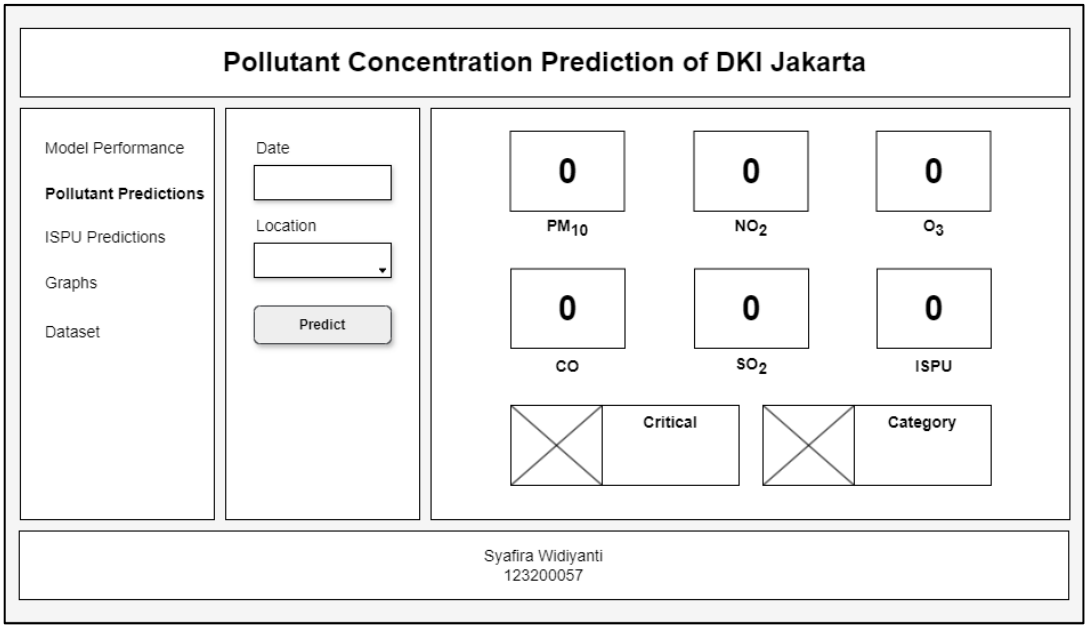


Gambar 3.24. Rancangan Halaman *Modeling*

2. Rancangan Halaman *Pollutant Predictions*

   Halaman prediksi kota / wilayah akan menampilkan hasil prediksi konsentrasi polutan, seperti PM_{10}, NO_2, O_3, CO, SO_2, dan ISPU berdasarkan tanggal dan lokasi yang diinputkan oleh *user*. Selain itu, juga akan ditampilkan jenis polutan dengan nilai yang tertinggi pada kolom *critical* dan

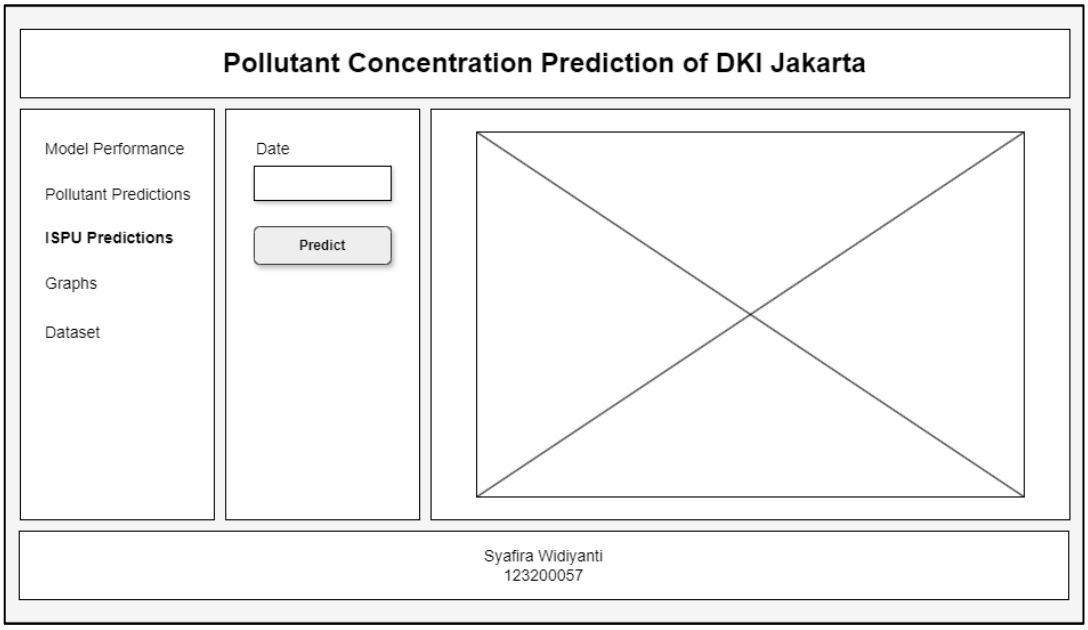
kategori ISPU pada kolom *category*. Tampilan halaman ditunjukkan pada Gambar 3.25.



Gambar 3.25. Rancangan Halaman *Predict City/Area*

3. Rancangan Halaman *ISPU Predictions*

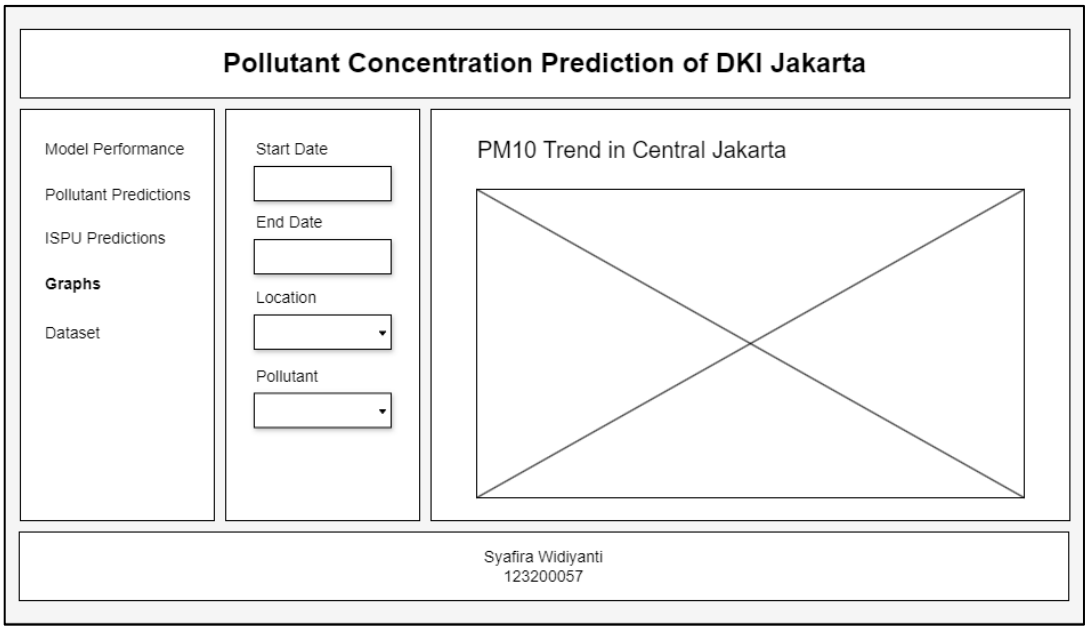
Halaman ini akan menampilkan hasil prediksi ISPU seluruh kota/kabupaten di DKI Jakarta dan ISPU provinsi DKI Jakarta berdasarkan tanggal yang diinputkan oleh *user*. Tampilan halaman ditunjukkan pada Gambar 3.26.



Gambar 3.26. Rancangan Halaman *Predict All City*

4. Rancangan Halaman *Graphs*

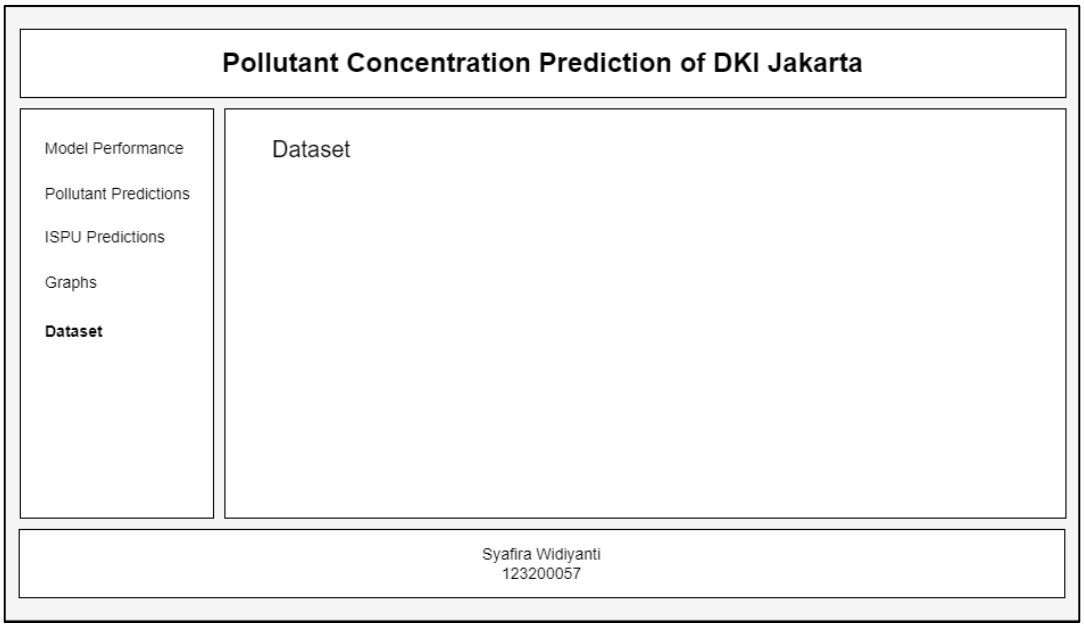
Halaman grafik akan menampilkan grafik dari masing-masing polutan berdasarkan tanggal mulai, tanggal selesai, dan lokasi yang diinputkan oleh *user*. Tampilan halaman ditunjukkan pada Gambar 3.27.



Gambar 3.27. Rancangan Halaman *Trend Graphic*

5. Rancangan Halaman *Dataset*

Halaman *dataset* akan menampilkan *dataset* yang digunakan dalam pelatihan model. *Dataset* yang ditampilkan merupakan *dataset* yang sudah bersih, di mana data yang tidak digunakan telah dihilangkan dan nilai yang hilang telah diisi dengan interpolasi linear. Tampilan halaman ditunjukkan pada Gambar 3.28.



Gambar 3.28. Rancangan Halaman *Dataset*

3.2.3 *Development*

Tahapan ini mengimplementasikan atau menerapkan rancangan sistem yang telah dibuat sebelumnya menjadi sistem seutuhnya menggunakan bahasa pemrograman. Pada penelitian ini, bahasa pemrograman yang digunakan adalah Python dan *framework* Streamlit.

3.2.4 *Testing*

Tahapan pengujian dilakukan agar sistem aplikasi yang dibangun berfungsi secara penuh tanpa ada *error* atau *bug* saat dioperasikan. Dalam penelitian ini, pengujian dilakukan dengan mengukur kinerja dan fungsionalitas aplikasi yang dibuat menggunakan metode pengujian *black box.* Skenario pengujian fungsionalitas aplikasi yang dibangun ditunjukkan dalam Tabel 3.9 dan Tabel 3.10.

Tabel 3.9. Skenario Pengujian *Black Box*

No	Halaman	Pengujian	Hasil	
			Sukses	Gagal
1	*Model Performance*	Menampilkan evaluasi model ResNet-LSTM, CNN-LSTM, ResNet, CNN, LSTM		
2	*Pollutant Predictions*	Menampilkan hasil prediksi PM_{10}, NO_2, O_3, CO, SO_2, ISPU berdasarkan tanggal dan lokasi yang dipilih		
		Menampilkan jenis polutan dengan nilai tertinggi		
		Menampilkan kategori ISPU		

Tabel 3.10. Lanjutan Skenario Pengujian *Black Box*

No	Halaman	Pengujian	Hasil	
			Sukses	Gagal
3	ISPU *Predictions*	Menampilkan hasil prediksi ISPU seluruh kota/kabupaten di DKI Jakarta berdasarkan tanggal yang dipilih dalam bentuk angka		
		Menampilkan hasil prediksi dan kategori ISPU DKI Jakarta berdasarkan tanggal yang dipilih		
4	*Graphs*	Menampilkan grafik polutan berdasarkan rentang tanggal dan lokasi yang dipilih		
5	*Dataset*	Menampilkan seluruh *dataset* yang sudah bersih		

BAB IV
HASIL DAN PEMBAHASAN

4.1 Hasil Penelitian

Pada bagian ini dijelaskan secara lengkap mengenai proses rancangan pada bab metodologi penelitian dan pengembangan sistem yang dimulai dari tahap *data understanding*, *data preprocessing*, pembangunan model, pelatihan model, analisis dan evaluasi hasil, serta pengujian sistem. Dalam pembangunan model dan pengembangan sistem, kode program dibuat dan ditulis menggunakan beberapa *library open-source*. Berikut adalah *library* yang digunakan dalam proses implementasi.

Source Code* 1: Proses *Import Library

```
import tensorflow as tf
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from keras import layers
from keras.layers import InputLayer, ConvLSTM2D
from keras.callbacks import EarlyStopping, TensorBoard, ModelCheckpoint,
EarlyStopping
from keras.losses import MeanAbsolutePercentageError
from keras.metrics import MeanAbsolutePercentageError
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from livelossplot import PlotLossesKeras
from datetime import datetime, timedelta
```

Modul Program 4.1. Proses *Import Library*

Berikut adalah tampilan aplikasi prediksi konsentrasi polutan DKI Jakarta yang dapat dilihat pada Gambar 4.1.

Gambar 4.1. Tampilan Aplikasi Prediksi Konsentrasi Polutan DKI Jakarta

Terdapat lima menu utama pada aplikasi prediksi kualitas udara. Menu pertama adalah *Model Peformance* yang menampilkan hasil evaluasi MAPE dan grafik metode ResNet-LSTM, CNN-LSTM, ResNet, CNN, dan LSTM. Menu kedua adalah *Pollutant Predictions* yang menampilkan hasil prediksi konsentrasi polutan, jenis

polutan tertinggi dan kategori ISPU berdasarkan data tanggal dan lokasi yang dipilih. Menu ketiga adalah *ISPU Predictions* yang menampilkan hasil prediksi ISPU seluruh kota/kabupaten di DKI Jakarta dan prediksi ISPU DI Provinsi DKI Jakarta berdasarkan tanggal yang dipilih. Menu keempat adalah *graphs* yang menampilkan grafik konsentrasi polutan berdasarkan tanggal dan lokasi yang dipilih. Menu kelima adalah *dataset* yang menampikan data asli hasil *cleaning* yang digunakan dalam pelatihan model.

4.1.1 *Data Understanding*

Tahapan pertama adalah *data understanding* yang terbagi menjadi dua proses yaitu pengumpulan data dan pemahaman data.

1) Pengumpulan Data

Tahapan pertama dari *data understanding* adalah melakukan *import* data polutan dan meteorologi yang telah diperoleh dari *website* dan disimpan dalam Google Drive. Proses *import data* dapat dilakukan sebagai berikut.

Source Code* 2: Proses *Import Data

```
polutan_raw = pd.read_excel('/content/drive/MyDrive/dataset/Dataset Zat
Polutan.xlsx')
meteorologi_raw = pd.read_excel('/content/drive/MyDrive/dataset/Dataset
Meteorologi.xlsx')
```

Modul Program 4.2. Proses *Import Data*

2) Pemahaman Data

Tahapan selanjutnya adalah membuat *dataframe* menggunakan data polutan dan meteorologi yang sudah di-*import* agar dapat dengan mudah menganalisis dan memanipulasi data yang akan digunakan. Proses pembuatan *dataframe* dapat dilakukan sebagai berikut.

Source Code* 3: Proses Pembuatan *Dataframe

```
df_polutan = pd.DataFrame(polutan_raw)
df_meteorologi = pd.DataFrame(meteorologi_raw)
```

Modul Program 4.3. Proses Pembuatan *Dataframe*

Berikut adalah hasil dari proses *import data* dan pembuatan *dataframe* dapat dilihat pada Gambar 4.2 untuk data polutan dan Gambar 4.3 untuk data meteorologi.

	tanggal	stasiun	pm10	so2	co	o3	no2	max	critical	categori
0	2019-01-01	DKI1 (Bunderan HI)	29	15	5	NaN	13	29.0	PM10	BAIK
1	2019-01-02	DKI1 (Bunderan HI)	24	17	5	NaN	6	24.0	PM10	BAIK
2	2019-01-03	DKI1 (Bunderan HI)	16	16	5	29	4	29.0	O3	BAIK
3	2019-01-04	DKI1 (Bunderan HI)	38	18	8	24	NaN	38.0	PM10	BAIK
4	2019-01-05	DKI1 (Bunderan HI)	37	29	16	NaN	16	37.0	PM10	BAIK

Gambar 4.2. Proses *Import Data* dan Pembuatan *Dataframe* Data Polutan

	Tanggal	Stasiun	Tavg	RH_avg	RR	ddd_x	ff_avg
0	01-01-2019	1	26.0	84.0	8.5	240.0	2.0
1	02-01-2019	1	28.3	75.0	2.3	290.0	2.0
2	03-01-2019	1	28.3	74.0	0.4	280.0	2.0
3	04-01-2019	1	29.3	67.0	0.0	250.0	3.0
4	05-01-2019	1	28.9	71.0	0.0	280.0	1.0

Gambar 4.3. Proses *Import Data* dan Pembuatan *Dataframe* Data Meteorologi

4.1.2 *Data Preprocessing*

Tahapan kedua adalah *data preprocessing* yang terbagi menjadi enam proses yaitu *remove unused data, fill missing data,* analisis korelasi spasial temporal, penentuan *array* fitur dan label, *splitting data*, *feature scaling*.

1) *Remove Unused Data*

Tahapan pertama dari *data preprocessing* adalah menghapus kolom-kolom yang tidak digunakan. Tahapan ini diperlukan karena tidak semua data yang diperoleh dari sumber data digunakan dalam pembangunan dan pelatihan model. Kolom yang tidak diperlukan pada data polutan adalah ‘max’, ‘critical’, dan ‘categori’, sedangkan pada data meteorologi adalah kolom ‘RR’ yang berisi data curah hujan rata-rata. Proses penghapusan kolom-kolom dapat dilakukan sebagai berikut.

Source Code* 4: Proses *Remove Unused Data

```
df_polutan = df_polutan.drop(['max', 'critical', 'categori'], axis=1)
df_meteorologi = df_meteorologi.drop('RR', axis=1)
```

Modul Program 4.4. Proses *Remove Unused Data*

Berikut adalah hasil dari proses *remove unused data* dapat dilihat pada Gambar 4.4 untuk data polutan dan Gambar 4.5 untuk data meteorologi.

	tanggal	stasiun	pm10	so2	co	o3	no2
0	2019-01-01	DKI1 (Bunderan HI)	29.0	15.0	5.0	NaN	13.0
1	2019-01-02	DKI1 (Bunderan HI)	24.0	17.0	5.0	NaN	6.0
2	2019-01-03	DKI1 (Bunderan HI)	16.0	16.0	5.0	29.0	4.0
3	2019-01-04	DKI1 (Bunderan HI)	38.0	18.0	8.0	24.0	NaN
4	2019-01-05	DKI1 (Bunderan HI)	37.0	29.0	16.0	NaN	16.0

Gambar 4.4. Proses *Remove Unused Data* Polutan

	Tanggal	Stasiun	Tavg	RH_avg	ddd_x	ff_avg
0	01-01-2019	1	26.0	84.0	240.0	2.0
1	02-01-2019	1	28.3	75.0	290.0	2.0
2	03-01-2019	1	28.3	74.0	280.0	2.0
3	04-01-2019	1	29.3	67.0	250.0	3.0
4	05-01-2019	1	28.9	71.0	280.0	1.0

Gambar 4.5. Proses *Remove Unused Data* Meteorologi

2) *Fill Missing Data*

Tahapan kedua adalah menggabungkan data polutan dengan data meteorologi dan mengisi nilai yang hilang pada *dataset*. Pertama, nilai yang hilang pada data polutan dan meteorologi diganti dengan nilai *Not a Number* atau NaN. Nilai yang hilang pada data polutan ditunjukkan dengan nilai "---", sedangkan pada data meteorologi ditunjukkan dengan nilai "8888". Kemudian, kolom tanggal pada data polutan dan meteorologi diubah menjadi tipe data *datetime*. Hal ini dilakukan untuk mempermudah manipulasi data waktu.

Pada data polutan, nilai stasiun diganti dengan kode yang sesuai dan diubah menjadi tipe data *integer*, yaitu 1 untuk DKI 1, 2 untuk DKI 2, 3 untuk DKI 3, 4 untuk DKI 4, dan 5 untuk DKI 5. Hal ini dilakukan agar data stasiun menjadi seragam dan mudah diinterpretasikan. Pada data meteorologi, nama kolom diubah agar sesuai dengan format yang diinginkan serta memudahkan penggunaan dan penggabungan kolom, seperti kolom 'Tanggal' menjadi 'tanggal', 'Stasiun' menjadi 'stasiun', 'Tavg' menjadi 'temperature', 'RH_avg' menjadi 'humidity', 'ddd_x' menjadi 'wind direction', dan 'ff_avg' menjadi 'wind speed'. Setelah kedua *dataframe* memiliki format kolom 'tanggal' dan 'stasiun' yang sama, tahapan selanjutnya adalah menggabungkan kedua *dataframe* berdasarkan kedua kolom tersebut. Berikut adalah hasil proses konkatenasi data polutan dan meteorologi dapat dilihat pada Gambar 4.6.

	tanggal	stasiun	pm10	so2	co	o3	no2	temperature	humidity	wind direction	wind speed
0	2019-01-01	1	29.0	15.0	5.0	NaN	13.0	26.0	84.0	240.0	2.0
1	2019-01-02	1	24.0	17.0	5.0	NaN	6.0	28.3	75.0	290.0	2.0
2	2019-01-03	1	16.0	16.0	5.0	29.0	4.0	28.3	74.0	280.0	2.0
3	2019-01-04	1	38.0	18.0	8.0	24.0	NaN	29.3	67.0	250.0	3.0
4	2019-01-05	1	37.0	29.0	16.0	NaN	16.0	28.9	71.0	280.0	1.0

Gambar 4.6. Proses Konkatenasi Data

Proses penggabungan data polutan dan meteorologi dapat dilakukan sebagai berikut.

***Source Code* 5: Proses Konkatenasi Data**

```
df_polutan.replace('---', np.nan, inplace=True)
df_polutan['tanggal'] = pd.to_datetime(df_polutan['tanggal'])
df_polutan.replace(['DKI1 (Bunderan HI)', 'DKI2 (Kelapa Gading)', 'DKI3
(Jagakarsa)', 'DKI4 (Lubang Buaya)', 'DKI5 (Kebon Jeruk)', 'DKI5 (Kebon
Jeruk) Jakarta Barat'], [1,2,3,4,5,5], inplace=True)
df_polutan['stasiun'] = df_polutan['stasiun'].astype(str).astype(int)

df_meteorologi.replace('8888', np.nan, inplace=True)
df_meteorologi['Tanggal'] = pd.to_datetime(df_meteorologi['Tanggal'],
format='%d-%m-%Y')
df_meteorologi.rename(columns = {'Tanggal':'tanggal',
'Stasiun':'stasiun', 'Tavg':'temperature', 'RH_avg':'humidity',
'ddd_x':'wind direction', 'ff_avg':'wind speed'}, inplace = True)

df_join =
df_polutan.join(df_meteorologi.set_index(['tanggal','stasiun']),
on=['tanggal','stasiun'])
```

Modul Program 4.5. Proses Konkatenasi Data

Kemudian, nilai yang hilang atau NaN pada *dataframe* gabungan diisi menggunakan metode interpolasi linear. Proses pengisian nilai yang hilang dapat dilakukan sebagai berikut.

Source Code* 6: Proses *Fill Missing Data

```
df_join = df_join.interpolate(method='linear')
df_join['o3'].bfill(inplace = True)
```

Modul Program 4.6. Proses *Fill Missing Data*

Berikut adalah hasil proses *fill missing data* dapat dilihat pada Gambar 4.7.

	tanggal	stasiun	pm10	so2	co	o3	no2	temperature	humidity	wind direction	wind speed	ispu
0	2019-01-01	1	29.0	15.0	5.0	29.00	13.0	26.0	84.0	240.0	2.0	29.00
1	2019-01-02	1	24.0	17.0	5.0	29.00	6.0	28.3	75.0	290.0	2.0	29.00
2	2019-01-03	1	16.0	16.0	5.0	29.00	4.0	28.3	74.0	280.0	2.0	29.00
3	2019-01-04	1	38.0	18.0	8.0	24.00	10.0	29.3	67.0	250.0	3.0	38.00
4	2019-01-05	1	37.0	29.0	16.0	40.25	16.0	28.9	71.0	280.0	1.0	40.25

Gambar 4.7. Proses *Fill Missing Data*

3) Analisis Korelasi Spasial Temporal

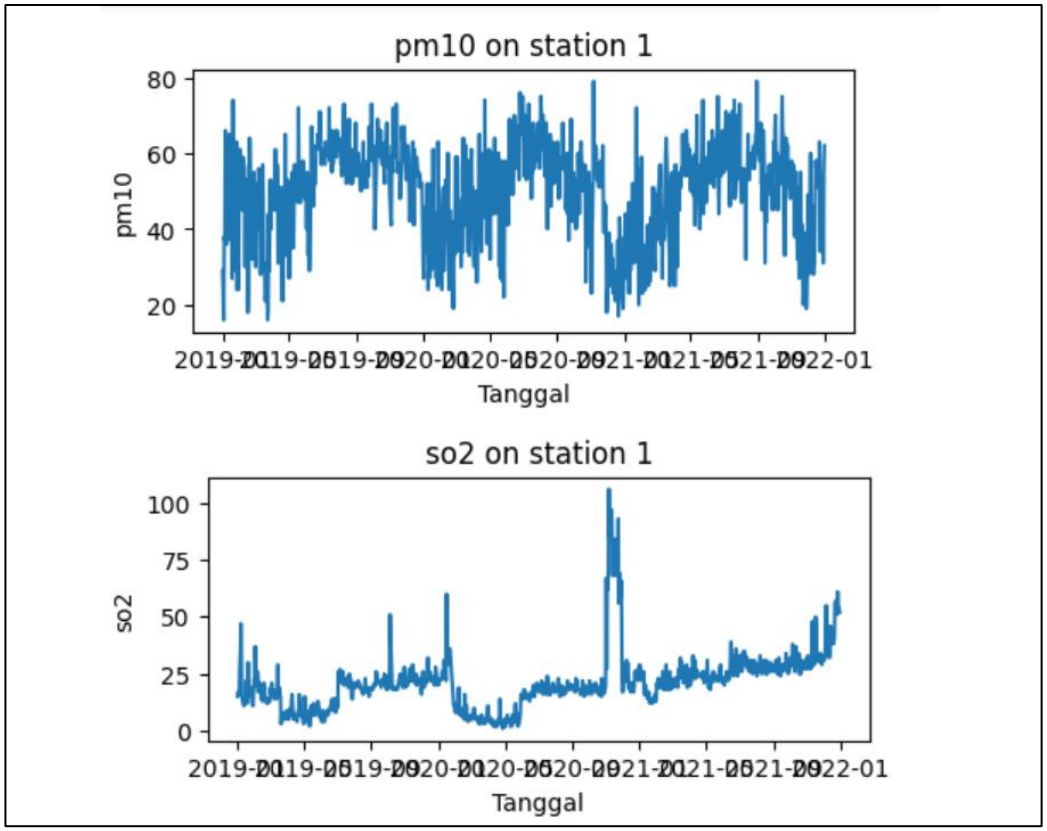
Tahapan ketiga adalah melakukan analisis korelasi spasial temporal pada data gabungan polutan dan meteorologi. Proses analisis temporal dilakukan dengan menampilkan grafik tren setiap polutan dan faktor meteorologi dari kelima stasiun tahun 2019 – 2021. Proses penampilan grafik dapat dilakukan sebagai berikut.

***Source Code* 7: Proses Analisis Grafik Polutan dan Meteorologi**

```
def graph_per_station(number):
  st = pd.DataFrame(df_join.loc[(df_join['stasiun']==number)])
  cols = st.columns
  par = cols[2:]
  for col in par :
    plt.figure(figsize=(10,5))
    plt.plot(st['tanggal'], st[col])
    plt.title('{0} on station {1}'.format(col, number))
    plt.xlabel('Tanggal')
    plt.ylabel(col)
    plt.show()

for number in range(1,6):
  graph_per_station(number)
```

Modul Program 4.7. Proses Analisis Grafik Polutan dan Meteorologi

Berikut adalah hasil proses analisis grafik polutan dan meteorologi dapat dilihat pada Gambar 4.8.



Gambar 4.8. Proses Analisis Grafik Polutan dan Meteorologi

Kemudian, analisis spasial dilakukan dengan menghitung nilai korelasi polutan tiap stasiun menggunakan metode perhitungan *pearson correlation coefficient*. Proses penghitungan korelasi dapat dilakukan sebagai berikut.

***Source Code* 8: Proses Analisis Korelasi Data Polutan**

```
def calculate_polutan_correlation(df, polutan):
    stasiun_data = []
    for stasiun in range(1, 6):
        st_data = pd.DataFrame(df.loc[df['stasiun'] ==
stasiun][polutan]).rename(columns={polutan: f'St
{stasiun}'}).reset_index().drop('index', axis=1)
        stasiun_data.append(st_data)
    polutan_concat = pd.concat(stasiun_data, axis=1)
    return polutan_concat.corr(method='pearson')

ispu_corr = calculate_polutan_correlation(df_join, 'ispu')
pm10_corr = calculate_polutan_correlation(df_join, 'pm10')
so2_corr = calculate_polutan_correlation(df_join, 'so2')
co_corr = calculate_polutan_correlation(df_join, 'co')
o3_corr = calculate_polutan_correlation(df_join, 'o3')
no2_corr = calculate_polutan_correlation(df_join, 'no2')
```

Modul Program 4.8. Proses Analisis Korelasi Data Polutan

Berikut adalah hasil proses analisis korelasi data polutan untuk data ISPU dapat dilihat pada Gambar 4.9.

	St 1	St 2	St 3	St 4	St 5
St 1	1.000000	0.650449	0.598643	0.523329	0.571584
St 2	0.650449	1.000000	0.570631	0.558027	0.595707
St 3	0.598643	0.570631	1.000000	0.627601	0.670598
St 4	0.523329	0.558027	0.627601	1.000000	0.657647
St 5	0.571584	0.595707	0.670598	0.657647	1.000000

Gambar 4.9. Proses Analisis Korelasi Data Polutan

4) Penentuan *Array* Fitur dan Label

Tahapan keempat adalah menentukan *array* berisi fitur yang akan dilatih dalam model prediksi dan label yang menjadi target *output*. Proses penentuan *array* fitur dan label dapat dilakukan sebagai berikut.

***Source Code* 9: Proses Penentuan Array Fitur dan Label**

```
def df_to_X_y(df, window_size=1):
  df_as_np = df.to_numpy()
  X = []
  y = []
  for i in range(len(df_as_np)-window_size):
    row = [r for r in df_as_np[i:i+window_size]]
    X.append(row)
    label = df_as_np[i+window_size][0]
    y.append(label)
  return np.array(X), np.array(y)

X, y = df_to_X_y(df_st1)
```

Modul Program 4.9. Proses Penentuan *Array* Fitur dan Label

Berikut adalah hasil proses penentuan *array* fitur dan label dapat dilihat pada Gambar 4.10.

```
Array X :  (1096, 1, 10)
[[[ 29.  29.  15. ...  84. 240.   2.]]

 [[ 29.  24.  17. ...  75. 290.   2.]]

 [[ 29.  16.  16. ...  74. 280.   2.]]

 ...

 [[ 54.  31.  54. ...  84. 320.   1.]]

 [[ 55.  55.  53. ...  84. 320.   1.]]

 [[ 62.  62.  52. ...  77. 260.   1.]]]
Array y :  (1096,)
[29. 29. 38. ... 55. 62. 61.]
```

Gambar 4.10. Proses Penentuan *Array* Fitur dan Label

5) *Splitting Data*

Tahapan kelima adalah membagi data menjadi dua bagian, yaitu data *training* dan data *testing* untuk masing-masing *array* X dan y. Rasio pembagian data adalah 7:3, di mana 70% untuk data *training* dan 30% untuk data *testing*. Proses *splitting data* dapat dilakukan sebagai berikut.

Source Code* 10: Proses *Splitting Data

```
train_size = int(X.shape[0]*0.7)
X_train, y_train = X[:train_size], y[:train_size]
X_test, y_test = X[train_size:], y[train_size:]
```

Modul Program 4.10. Proses *Splitting Data*

Berikut adalah hasil proses *splitting data* dapat dilihat pada Gambar 4.11.

```
X Train :  (767, 1, 10)
[[[ 29.    24.    17.     5.    29.     6.    28.3   75.   290.     2.  ]]

 [[ 29.    16.    16.     5.    29.     4.    28.3   74.   280.     2.  ]]

 [[ 38.    38.    18.     8.    24.    10.    29.3   67.   250.     3.  ]]

 [[ 40.25  37.    29.    16.    40.25  16.    28.9   71.   280.     1.  ]]]
y Train :  (767,)
[29.   38.   40.25 66.  ]
X Test :  (329, 1, 10)
[[[ 25.   25.   19.    7.   19.   12.   27.8  82.  350.    3. ]]

 [[ 24.   24.   19.   10.   18.   17.   28.   83.  330.    2. ]]

 [[ 33.   33.   22.   12.   14.   22.   27.2  87.  270.    1. ]]

 [[ 32.   32.   21.   10.   12.   16.   28.1  80.  200.    2. ]]]
y Test :  (329,)
[24. 33. 32. 26.]
```

Gambar 4.11. Proses *Splitting Data*

6) *Feature Scaling*

Tahapan terakhir dari *data preprocessing* adalah melakukan normalisasi atau *feature scaling* menggunakan Z-*score normalization*. Hal ini dilakukan untuk menyamakan rentang semua fitur. Perhitungan Z-*score normalization* memerlukan nilai rata-rata dan *standard deviation*. Proses *feature scaling* dapat dilakukan sebagai berikut.

Source Code* 11: Proses *Feature Scaling

```
scaler = StandardScaler()
scaler.fit(X_train)
scaler.fit(X_test)

X_train = scaler.transform(X_train)
X_test = scaler.transform(X_test)
```

Modul Program 4.11. Proses *Feature Scaling*

Berikut adalah hasil proses *feature scaling* dapat dilihat pada Gambar 4.12.

```
X Train :  (767, 10)
[[-1.55700582 -2.04444659 -0.08259977 -1.24466389 -0.88217759 -0.97948306
  -0.39132435 -0.07911362  0.50431597  0.89560625]
 [-1.55700582 -2.67811453 -0.15550429 -1.24466389 -0.88217759 -1.21221648
  -0.39132435 -0.23390992  0.40262546  0.89560625]
 [-1.08308045 -0.93552769 -0.00969525 -0.9082801  -1.09589528 -0.5140162
   0.62888563 -1.31748401  0.09755394  2.621561  ]
 [-0.96459911 -1.01473618  0.79225441 -0.01125665 -0.40131278  0.18418408
   0.22080164 -0.69829882  0.40262546 -0.83034851]]
X Test :  (329, 10)
[[-2.35646858 -2.20540051 -1.29423776 -1.41853171 -0.34769453 -1.39430384
  -0.78827879  0.9994553   0.84638063  2.51654213]
 [-2.44512246 -2.28983863 -1.29423776 -0.71212868 -0.50679258 -0.90305736
  -0.57482716  1.18377189  0.62755216  0.87379935]
 [-1.64723756 -1.52989554 -0.87513912 -0.24119333 -1.14318479 -0.41181088
  -1.42863365  1.92103825 -0.02893325 -0.76894343]
 [-1.73589144 -1.61433366 -1.01483867 -0.71212868 -1.46138089 -1.00130666
  -0.46810135  0.63082213 -0.79483289  0.87379935]]
```

Gambar 4.12. Proses *Feature Scaling*

4.1.3 Pembangunan Model

Setelah data dibersihkan dan dinormalisasi pada tahapan *data preprocessing*, selanjutnya data digunakan dalam pembuatan model. Arsitektur model dibangun menggunakan Keras untuk melakukan prediksi konsentrasi polutan dengan mengimplementasikan arsitektur *Residual Network* (ResNet) dan metode *Long Short-Term Memory* (LSTM). Masukan atau *input* untuk *block residual* memerlukan *input* 3 dimensi yang terdiri dari *height* (panjang data), *width* (lebar data), dan *channel*. Bentuk dimensi dari *array* X *train* terdiri dari jumlah data, *height*, dan *width*. Oleh karena itu, dimensi masukan perlu dibentuk ulang menggunakan fungsi *shape* untuk menambahkan *channel* sejumlah 1, atau hitam putih, sehingga bentuk *input* data menjadi (1, 10, 1).

Layer pertama dari ResNet-LSTM adalah *convolutional layer* 2 dimensi dengan jumlah *filter* sebanyak 3, ukuran *kernel* 3x3 *pixel*, *strides* sebesar 1, dan *padding* "*same*" yang ditunjukkan oleh operasi *tf.keras.layers.Conv2D(filters=3, kernel_size=(3,3),strides=(1,1),padding='same').* Hasil konvolusi kemudian dinormalisasi menggunakan *batch normalization*, lalu diaktivasi menggunakan fungsi aktviasi ReLU.

Hasil konvolusi pertama yang sudah diaktivasi kemudian dimasukkan ke *block residual* pertama yang terdiri dari dua *layer*. *Layer* pertama memiliki filter sebanyak 9, ukuran *kernel* 3x3 *pixel*, *strides* sebesar 1, dan *padding* "*same*" yang ditunjukkan dengan operasi *tf.keras.layers.Conv2D(filters=9, kernel_size=(3,3), strides=(1,1), padding='same')*. *Layer* kedua memiliki filter sebanyak 9, ukuran *kernel* 1x1 *pixel*, *strides* sebesar 1, dan *padding* "*same*". Pada tiap *layer*, hasil konvolusi akan dinormalisasi menggunakan *batch normalization*, lalu diaktivasi menggunakan fungsi aktviasi ReLU. Gradien konvolusi yang telah diaktivasi pada *layer* pertama (*x*) dijumlahkan dengan gradien pada *layer* kedua (*skip*)

ditunjukkan dengan operasi *tf.keras.layers.Add()([x,skip])*. Operasi ini disebut dengan *skip connection* atau *shortcut connection*.

Hasil *block residual* pertama dengan dimensi (1,10,9) dijadikan sebagai *input* pada *block residual* kedua yang terdiri dari dua *layer*. Detail dari setiap *layer* hampir sama seperti *block residual* pertama, yang membedakan adalah jumlah *filter* yang digunakan sebanyak 16. Setelah dilakukan operasi *skip connection*, hasil *block residual* kedua dengan dimensi (1,10,64) dijadikan sebagai *input* pada *block residual* ketiga yang memiliki *filter* sebanyak 64. Begitu pula dengan blok keempat dan kelima dengan *filter* sebanyak 128.

Operasi *tf.keras.layers.Reshape((-1,x.shape[2]*x.shape[3]))* berfungsi untuk membentuk kembali dimensi hasil *block residual* kelima agar sesuai dengan bnetuk masukan LSTM, yaitu menjadi (1, 1280), di mana 1 adalah *timestep* dan 1280 adalah jumlah fitur yang diperoleh dari perkalian jumlah fitur dan *channel* pada *block residual*. *Layer* LSTM yang digunakan hanya 1 dengan jumlah *unit* sebanyak 64. Hasil pengolahan data kemudian dinormalisasi menggunakan *batch normalization*, lalu diaktivasi menggunakan fungsi aktivasi ReLU. Terakhir, hasil prediksi diperoleh dari *fully connected layer* yang ditunjukkan dengan operasi *tf.keras.layers.Dense(1)*, di mana nilai *output* yang dihasilkan pada *layer* ini sejumlah 1.

***Source Code* 12: ResNet-LSTM**

```
def create_model(X_train, y_train, X_test, y_test):
    input = tf.keras.layers.Input(shape=(X_train.shape[1],
X_train.shape[2], 1))

    x = tf.keras.layers.Conv2D(filters=3, kernel_size=(3,3),
strides=(1,1), padding='same',
kernel_initializer='TruncatedNormal')(input)
    x = tf.keras.layers.BatchNormalization()(x)
    x = tf.keras.layers.ReLU()(x)

  #1
    x = tf.keras.layers.Conv2D(filters=9, kernel_size=(3,3),
strides=(1,1), padding='same', kernel_initializer='TruncatedNormal')(x)
    x = tf.keras.layers.BatchNormalization()(x)
    x = tf.keras.layers.ReLU()(x)
    skip = tf.keras.layers.Conv2D(filters=9, kernel_size=(1,1),
strides=(1,1), padding='same')(x)
    skip = tf.keras.layers.BatchNormalization()(skip)
    skip = tf.keras.layers.ReLU()(skip)
    x = tf.keras.layers.Add()([x,skip])

  # 2
    x = tf.keras.layers.Conv2D(filters=16, kernel_size=(3,3),
strides=(1,1), padding='same', kernel_initializer='TruncatedNormal')(x)
    x = tf.keras.layers.BatchNormalization()(x)
    x = tf.keras.layers.ReLU()(x)
    skip = tf.keras.layers.Conv2D(filters=16, kernel_size=(1,1),
strides=(1,1), padding='same')(x)
    skip = tf.keras.layers.BatchNormalization()(skip)
    skip = tf.keras.layers.ReLU()(skip)
    x = tf.keras.layers.Add()([x,skip])
```

Modul Program 4.12. ResNet-LSTM

***Source Code* 13: Lanjutan ResNet-LSTM**

```
    #3
    x = tf.keras.layers.Conv2D(filters=64, kernel_size=(3,3),
strides=(1,1), padding='same', kernel_initializer='TruncatedNormal')(x)
    x = tf.keras.layers.BatchNormalization()(x)
    x = tf.keras.layers.ReLU()(x)
    skip = tf.keras.layers.Conv2D(filters=64, kernel_size=(1,1),
strides=(1,1), padding='same')(x)
    skip = tf.keras.layers.BatchNormalization()(skip)
    skip = tf.keras.layers.ReLU()(skip)
    x = tf.keras.layers.Add()([x,skip])

    #4
    x = tf.keras.layers.Conv2D(filters=128, kernel_size=(3,3),
strides=(1,1), padding='same', kernel_initializer='TruncatedNormal')(x)
    x = tf.keras.layers.BatchNormalization()(x)
    x = tf.keras.layers.ReLU()(x)
    skip = tf.keras.layers.Conv2D(filters=128, kernel_size=(1,1),
strides=(1,1), padding='same')(x)
    skip = tf.keras.layers.BatchNormalization()(skip)
    skip = tf.keras.layers.ReLU()(skip)
    x = tf.keras.layers.Add()([x,skip])

    #5
    x = tf.keras.layers.Conv2D(filters=128, kernel_size=(3,3),
strides=(1,1), padding='same', kernel_initializer='TruncatedNormal')(x)
    x = tf.keras.layers.BatchNormalization()(x)
    x = tf.keras.layers.ReLU()(x)
    skip = tf.keras.layers.Conv2D(filters=128, kernel_size=(1,1),
strides=(1,1), padding='same')(x)
    skip = tf.keras.layers.BatchNormalization()(skip)
    skip = tf.keras.layers.ReLU()(skip)
    x = tf.keras.layers.Add()([x,skip])

    x = tf.keras.layers.Reshape((-1, x.shape[2] * x.shape[3]))(x)
    x = tf.keras.layers.LSTM(64)(x)
    x = tf.keras.layers.BatchNormalization()(x)
    x = tf.keras.layers.ReLU()(x)

    output = tf.keras.layers.Dense(1)(x)
    create_model = tf.keras.Model(inputs=input, outputs=output,
name='resnet-lstm')
    return create_model
```

Modul Program 4.13. Lanjutan ResNet-LSTM

4.1.4 Pelatihan Model

Tahapan pelatihan model adalah langkah-langkah dalam menentukan *hyperparameter*, *model compiling*, dan *model fitting*. Setelah model diinisialisasi, langkah selanjutnya adalah menentukan *hyperparameter* yang digunakan selama pelatihan model terdiri dari *epochs*, *batch size*, dan *learning rate*. Pada penelitian ini, *epochs* yang digunakan adalah 50, 100, 150, 200, 250, dan 300, sedangkan *learning rate* yang digunakan adalah 0.001 dan 0.0001. *Optimizer* yang digunakan dalam pelatihan model adalah *Stochastic Gradient Descent* (SGD) dengan konfigurasi *learning rate*, *momentum* sebesar 0.9, *decay* 0, dan jenis momentum *Nesterov*.

Source Code* 14: *Hyperparameter Tuning

```
model = create_model(X_train, y_train, X_test, y_test)

#Hyperparameter Tuning
epochs = 100
bs = 64
learning_rate = 0.001

sgd = tf.keras.optimizers.legacy.SGD(learning_rate=learning_rate,
momentum=0.9, decay=0, nesterov=True)
```

Modul Program 4.14. *Hyperparameter Tuning*

Setelah menentukan *hyperparameter* dan *optimizer*, model siap di-*compile* untuk persiapan pelatihan. Fungsi *loss* dan *metric* yang digunakan adalah *Mean Absolute Percentage Error* (MAPE).

Source Code* 15: *Model Compiling

```
model.compile(optimizer='sgd', loss =
tf.keras.losses.MeanAbsolutePercentageError(),
metrics=[tf.keras.metrics.MeanAbsolutePercentageError()])
```

Modul Program 4.15. *Model Compiling*

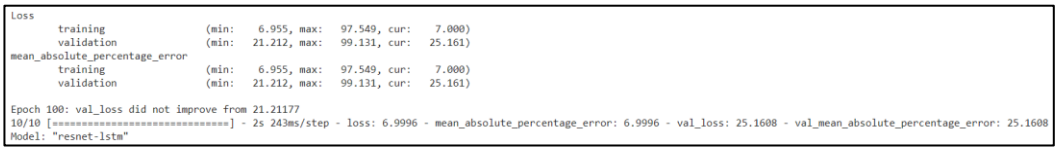
Selanjutnya, model yang telah di-*compile* kemudian dipelajari menggunakan data latih (X *train* dan y *train*). Pada tahapan ini dideklarasikan *validation split* yang digunakan untuk membagi data latih X *train* menjadi subset pelatihan dan validasi sebesar 20%, di mana 80% untuk data latih dan 20% untuk data validasi.

Source Code* 16: *Model Fitting

```
model.fit(X_train, y_train, epochs=epochs, batch_size=bs,
validation_split=0.2, verbose=1, callbacks=[PlotLossesKeras()])
```

Modul Program 4.16. *Model Fitting*

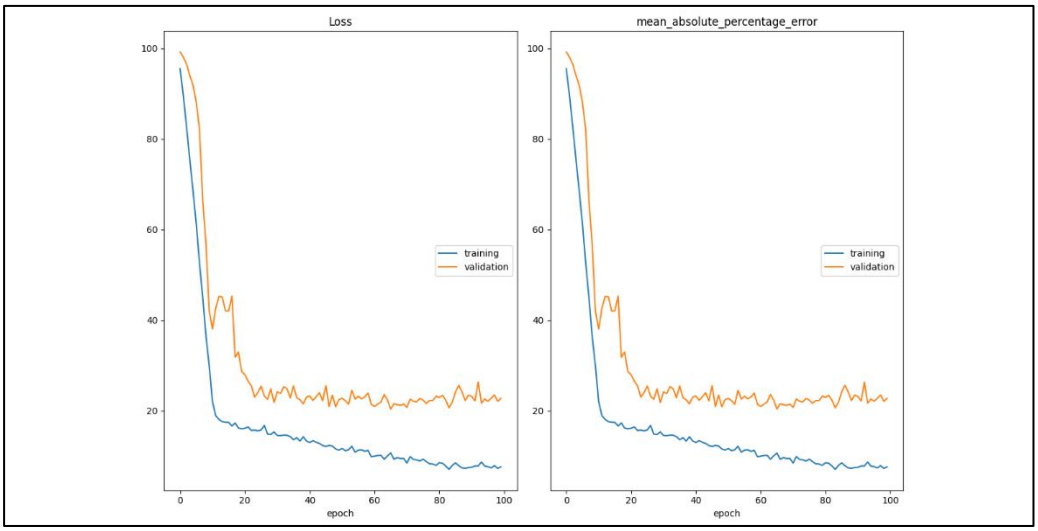
Model yang dilatih menghasilkan nilai MAPE sebesar 7.5% untuk *training error* dan 22.7% untuk *validation error.* Berikut adalah hasil proses pelatihan model setelah dilakukan *hyperparameter tuning*, *model compiling,* dan *model fitting* yang dapat dilihat pada Gambar 4.13.



Gambar 4.13. Proses Pelatihan Model

4.1.5 Analisis dan Evaluasi Hasil

Hasil *model fitting* divisualisasikan dalam bentuk grafik *loss* dan *metric* yang ditampilkan menggunakan *library* PlotLossesKeras. Hasil *loss* dan *metric* pada model ini serupa karena sama-sama menggunakan evaluasi MAPE dengan *epoch* 100 dan *learning rate* 0.001. Grafik *loss* dan *metric* evaluasi ResNet-LSTM dapat dilihat pada Gambar 4.14.



Gambar 4.14. Grafik *Loss* dan *Metric* Evaluasi ResNet-LSTM

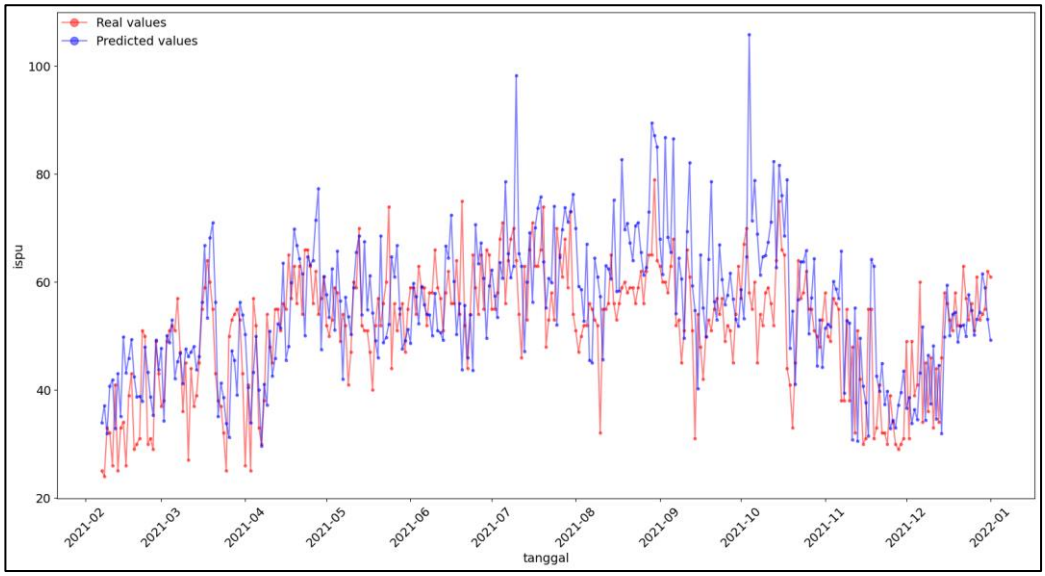
Setelah model dilatih, selanjutnya adalah melakukan pengujian menggunakan data uji X *test*. Hasil evaluasi MAPE menunjukkan *loss* dan *metric* sebesar 19.1%.

Source Code* 17: *Model Evaluation

```
y_pred = model.predict(X_test)
y_true = y_test.reshape(y_test.shape[0],1)

plt.plot(pd.to_datetime(x.index), y_true, '.-', color='red', label='Real
values', alpha=0.5)
plt.plot(pd.to_datetime(x.index), y_pred, '.-', color='blue',
label='Predicted values', alpha=0.5)

mape_result = model.evaluate(X_test, y_test)
```

Modul Program 4.17. *Model Evaluation*

Kemudian, hasil pengujian ditampilkan dalam bentuk grafik yang dapat dilihat pada Gambar 4.15.



Gambar 4.15. Grafik Evaluasi Model ResNet-LSTM

Nilai MAPE pada pelatihan model menunjukkan hasil yang berbeda-beda sesuai dengan *epochs* dan *learning rate* yang diinisialisasi. Hasil *training error*, *validation error*, dan *testing error* berdasarkan *epochs* dan *learning rate* yang digunakan dapat dilihat pada Tabel 4.1 hingga Tabel 4.3.

Tabel 4.1. *Training Error Value*

Epoch/LR	0.001	0.0001
50	11.5%	11.6%
100	**7.5%**	6.9%
150	5.1%	5.8%
200	4.9%	3.7%
250	3.5%	4.1%
300	3.8%	3.1%

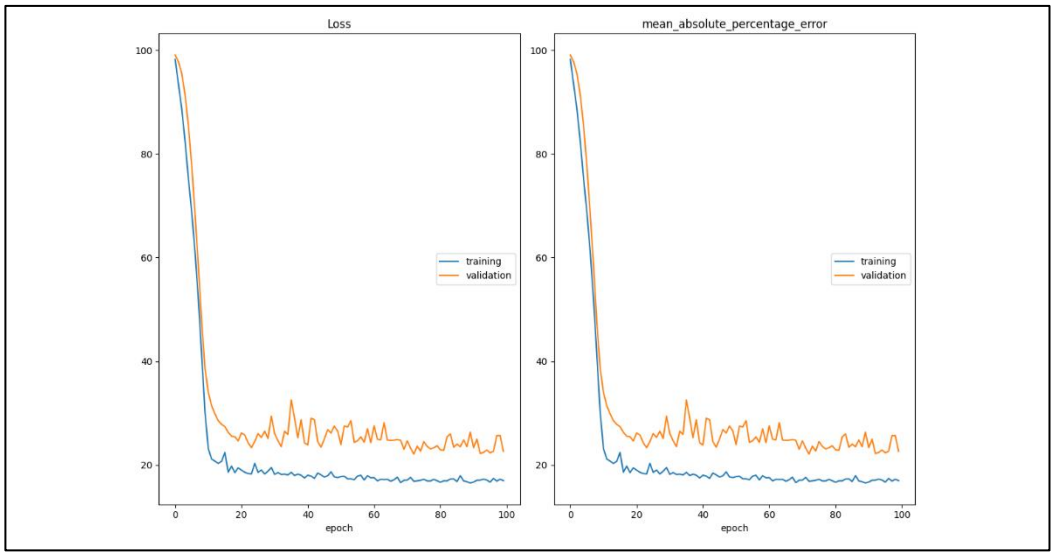
Tabel 4.2. *Validation Error Value*

Epoch/LR	0.001	0.0001
50	23.4%	24.3%
100	**22.7%**	24.2%
150	27.7%	24.1%
200	23.4%	24.7%
250	24%	24.5%
300	27.7%	24.9%

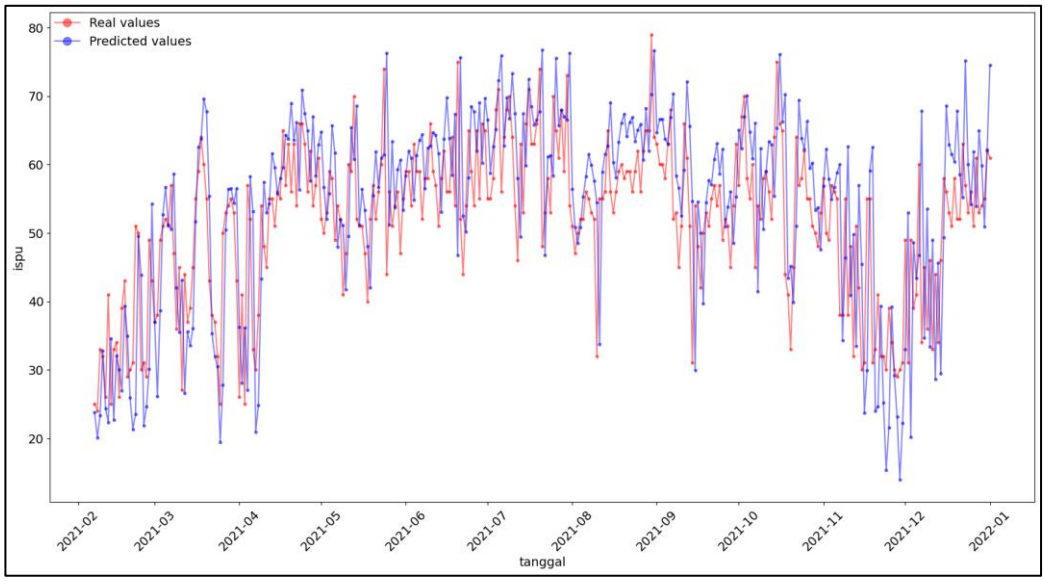
Tabel 4.3. *Testing Error Value*

Epoch/LR	0.001	0.0001
50	17.0%	17.5%
100	**19.1%**	18.1%
150	20.0%	19.1%
200	19.3%	19.4%
250	20.6%	22.0%
300	21.7%	18.7%

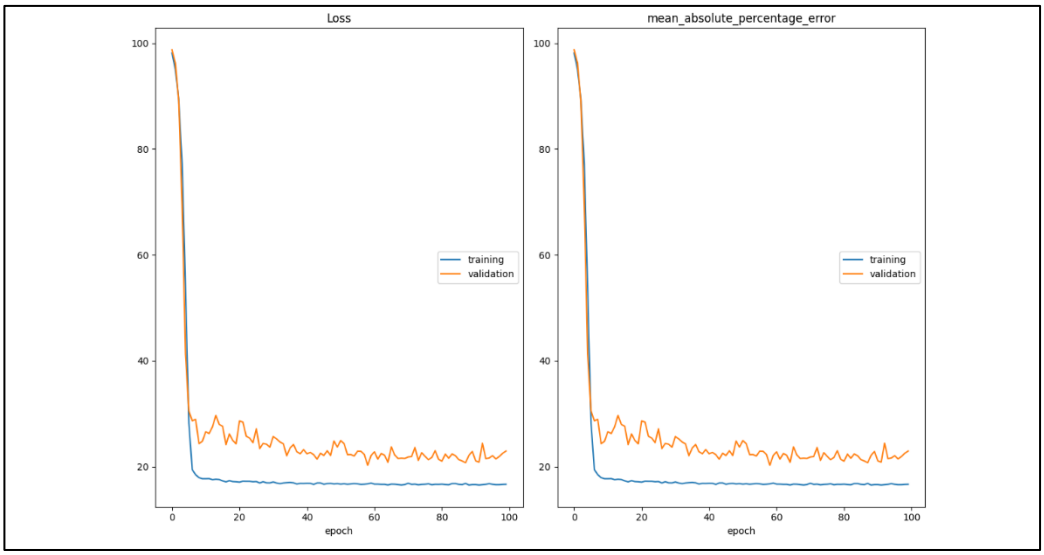
Evaluasi juga dilakukan dengan membandingkan hasil nilai MAPE pada pelatihan model dengan metode yang berbeda, yaitu LSTM, CNN, ResNet, CNN-LSTM, dan ResNe-LSTM menggunakan *epochs* sebesar 100 dan *learning rate* 0.001. Grafik *loss* dan *metric*, serta hasil pengujian masing-masing metode dapat dilihat pada Gambar 4.16 hingga Gambar 4.23.



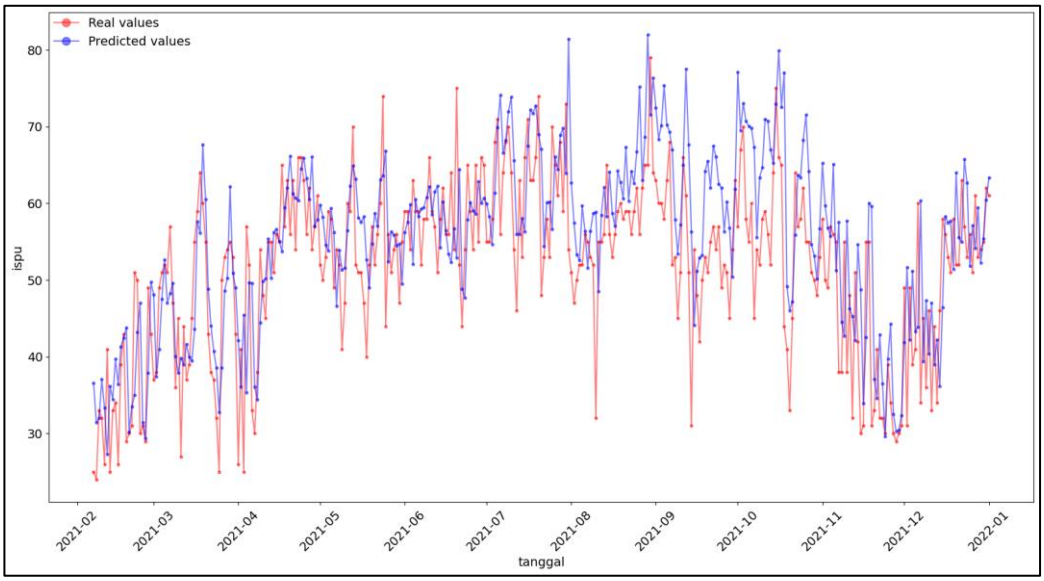
Gambar 4.16. Grafik *Loss* dan *Metric* Evaluasi LSTM



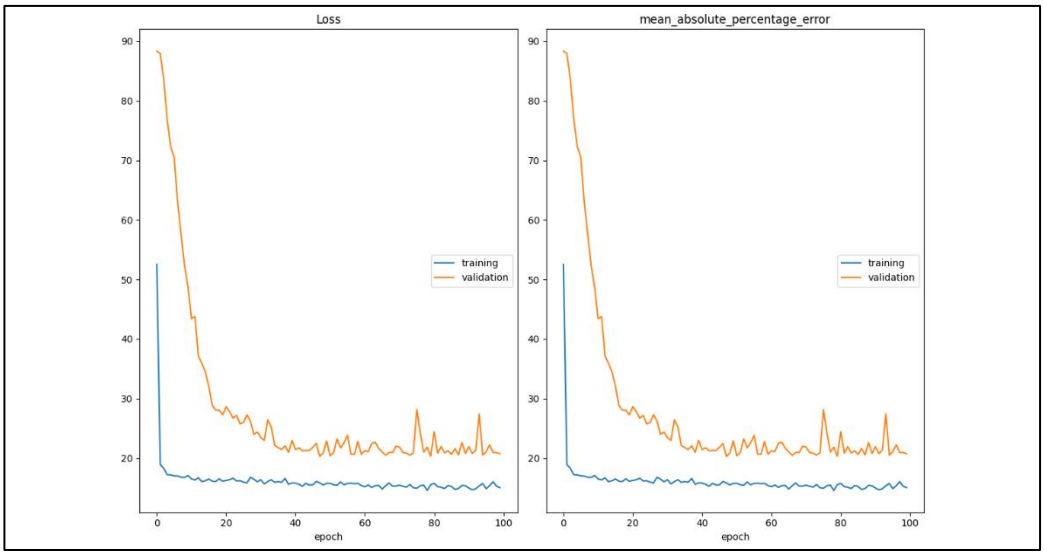
Gambar 4.17. Grafik Evaluasi Model LSTM



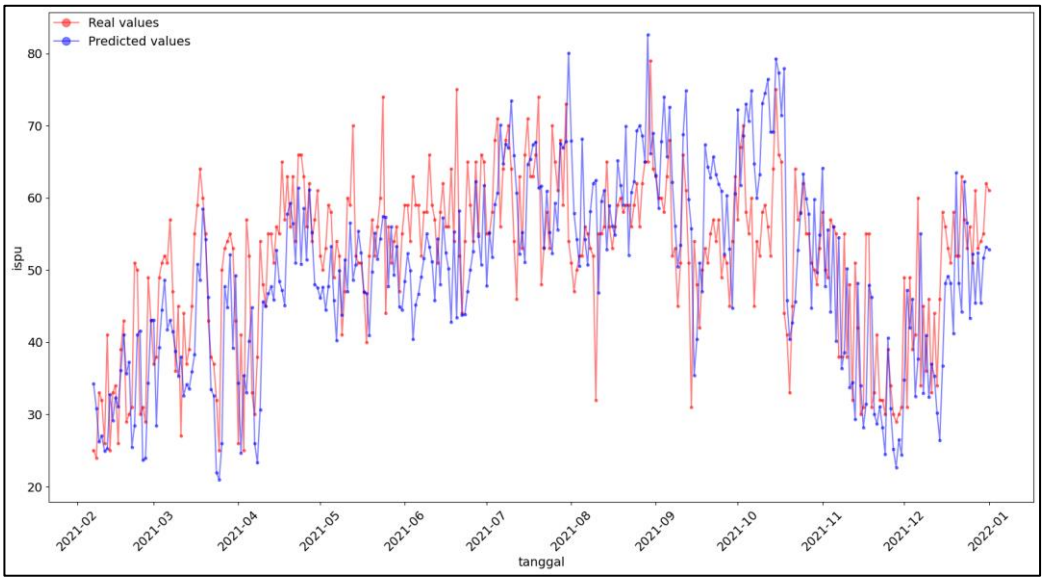
Gambar 4.18. Grafik *Loss* dan *Metric* Evaluasi CNN



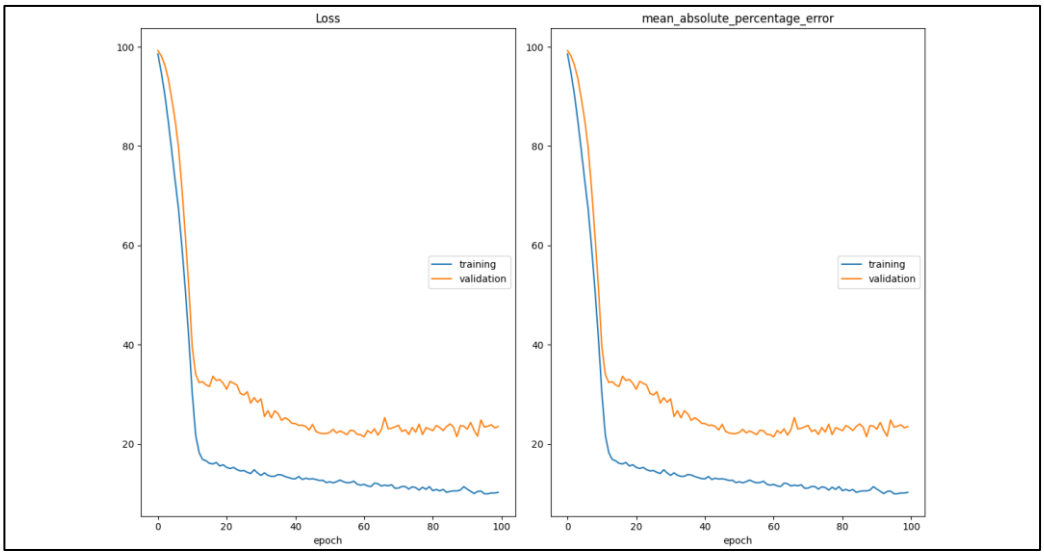
Gambar 4.19. Grafik Evaluasi Model CNN



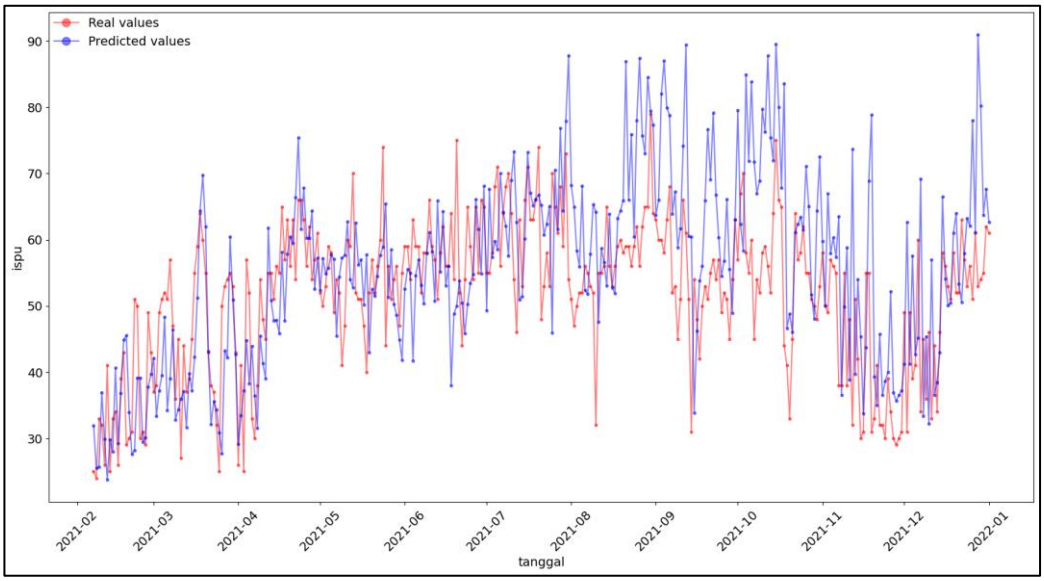
Gambar 4.20. Grafik *Loss* dan *Metric* Evaluasi ResNet



Gambar 4.21. Grafik Evaluasi Model ResNet



Gambar 4.22. Grafik *Loss* dan *Metric* Evaluasi CNN-LSTM



Gambar 4.23. Grafik Evaluasi Model CNN-LSTM

Hasil *training error*, *validation error*, dan *testing error* masing-masing metode LSTM, CNN, ResNet, CNN-LSTM, dan ResNe-LSTM dapat dilihat pada Tabel 4.4.

Tabel 4.4. *Model Comparison*

Metode	***Training Error***	***Validation Error***	***Testing Error***
LSTM	16.9%	22.6%	18%
CNN	16.6%	22.9%	15.5%
ResNet	15%	20.7%	16.5%
CNN-LSTM	10.2%	23.4%	19.7%
ResNet-LSTM	**7.5%**	**22.7%**	**19.1%**

Berdasarkan hasil pengujian MAPE, metode ResNet-LSTM menunjukkan nilai *training error* yang lebih rendah dan nilai *validation error* rata-rata yang setara dengan metode lainnya. Namun, nilai *testing error* pada metode ResNet-LSTM lebih tinggi dibandingkan dengan metode yang tidak dikombinasikan, tetapi lebih rendah dibandingkan dengan metode CNN-LSTM.

Selain pengujian menggunakan data uji, model juga diuji secara langsung dengan *input* berupa tanggal. Tanggal yang di-*input* harus terdapat dalam variabel *data indices* untuk diambil *key* yang digunakan sebagai indeks untuk mengambil data *input* dari *array* X berdasarkan indeks tanggal sebelum tanggal *input*. Proses *real-time testing* dapat dilakukan sebagai berikut.

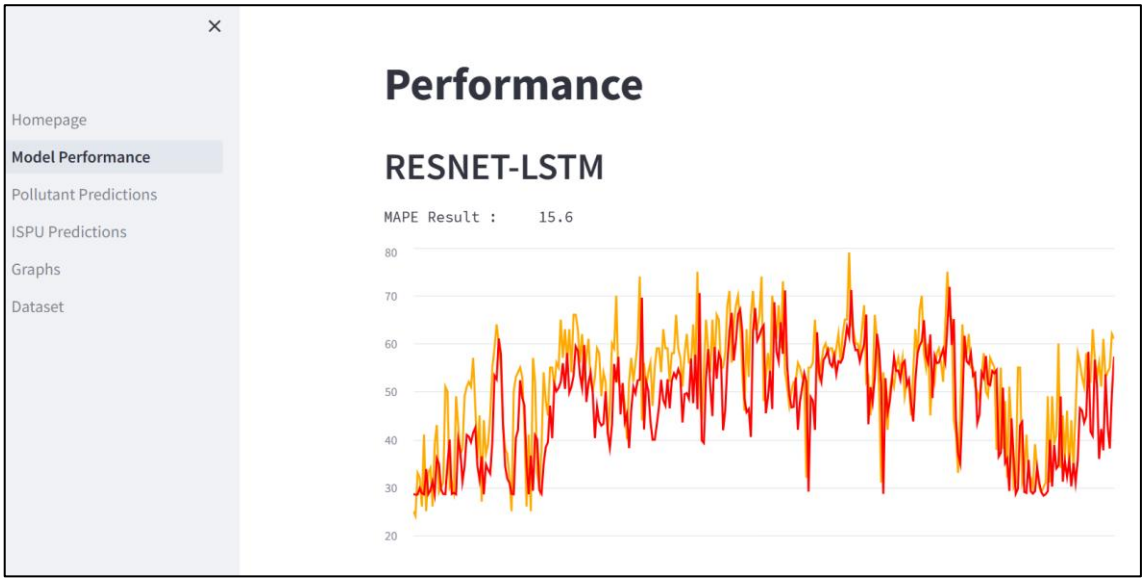
Source Code* 18: *Real-Time Testing

```
current_date = datetime.strptime('2022-01-02', '%Y-%m-%d')
end_date = datetime.strptime('2022-01-03', '%Y-%m-%d')

while current_date < end_date:
    dates_hat = np.append(dates_hat, current_date.strftime('%Y-%m-%d'))
    current_date += timedelta(days=1)

date_indices = {date: i for i, date in enumerate(dates_hat)}

input_date = '2022-01-01'
input_index = date_indices[input_date]
input_data = np.array(X[input_index-1])

prediction = model.predict(input_data)
```

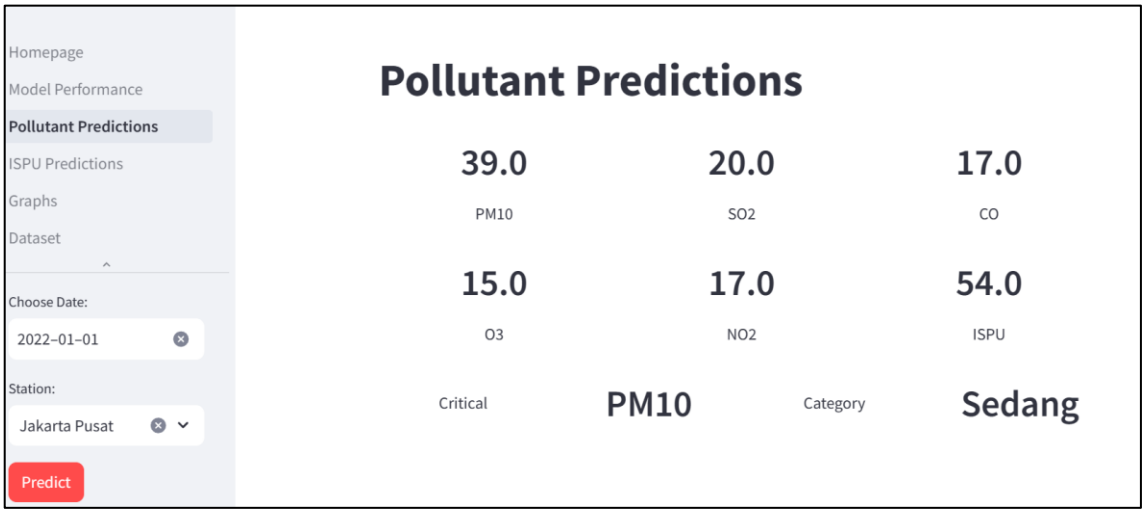
Modul Program 4.18. *Real-Time Testing*

4.1.6 Pengembangan Sistem

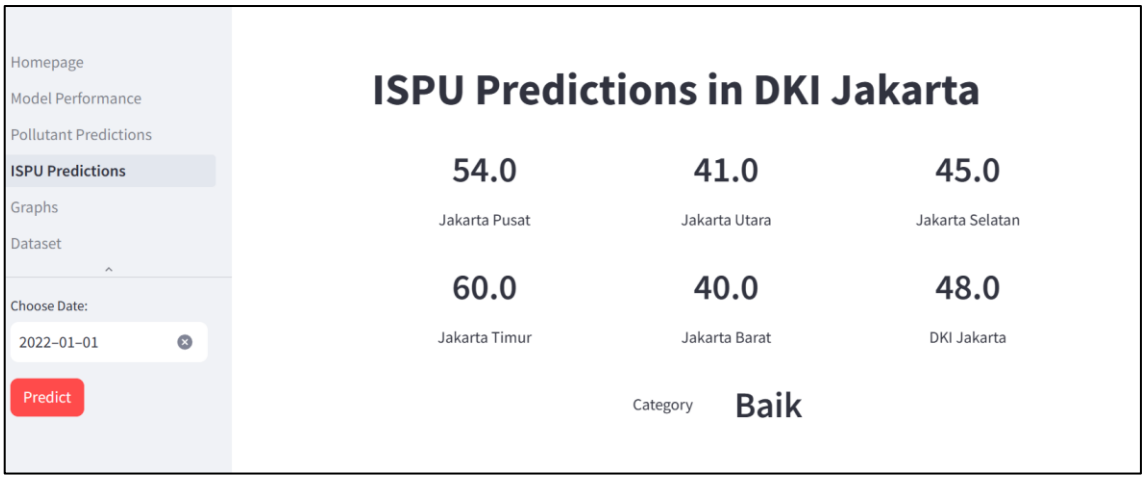
Rancangan aplikasi dikembangkan menggunakan *platform* berbasis *website* dengan bahasa pemrograman *Python* dan *framework Streamlit*. *Framework* tersebut diimplementasikan untuk membangun *user interface* dari aplikasi yang telah dirancang sebelumnya. Terdapat lima buah halaman menu yang dibangun dalam aplikasi ini. Hasil *user interface* dari masing-masing menu yang telah dibangun dapat dilihat pada Gambar 4.16 hingga Gambar 4.20.



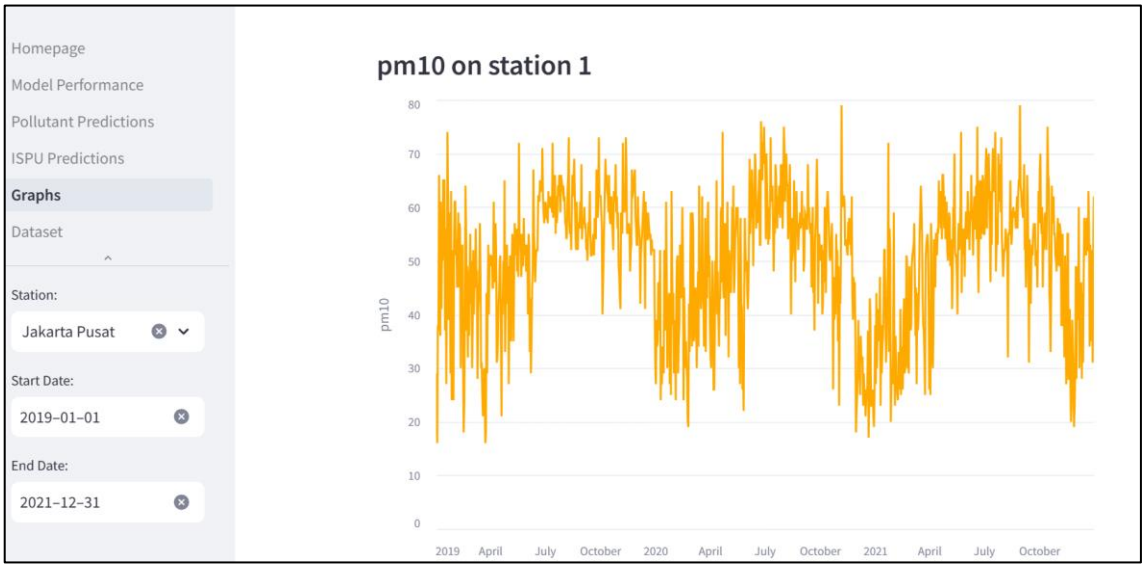
Gambar 4.24. Halaman *Model Performance*



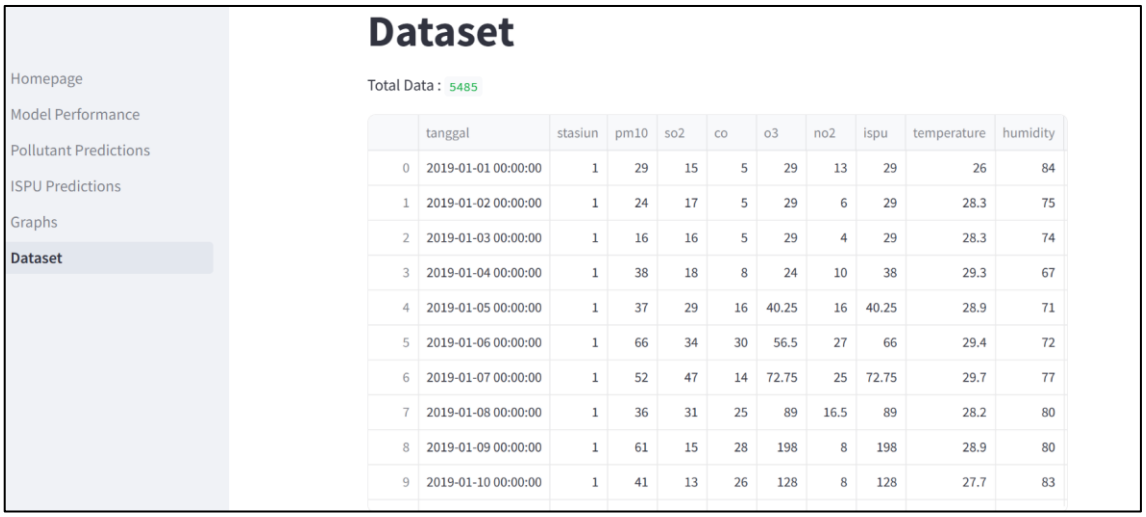
Gambar 4.25. Halaman *Pollutant Predictions*



Gambar 4.26. Halaman ISPU *Predictions*



Gambar 4.27. Halaman *Graphs*



	tanggal	stasiun	pm10	so2	co	o3	no2	ispu	temperature	humidity
0	2019-01-01 00:00:00	1	29	15	5	29	13	29	26	84
1	2019-01-02 00:00:00	1	24	17	5	29	6	29	28.3	75
2	2019-01-03 00:00:00	1	16	16	5	29	4	29	28.3	74
3	2019-01-04 00:00:00	1	38	18	8	24	10	38	29.3	67
4	2019-01-05 00:00:00	1	37	29	16	40.25	16	40.25	28.9	71
5	2019-01-06 00:00:00	1	66	34	30	56.5	27	66	29.4	72
6	2019-01-07 00:00:00	1	52	47	14	72.75	25	72.75	29.7	77
7	2019-01-08 00:00:00	1	36	31	25	89	16.5	89	28.2	80
8	2019-01-09 00:00:00	1	61	15	28	198	8	198	28.9	80
9	2019-01-10 00:00:00	1	41	13	26	128	8	128	27.7	83

Gambar 4.28. Halaman *Dataset*

Fitur dan komponen yang ada pada sistem dibangun sesuai dengan rancangan *user interface* yang telah dibuat. Pengujian sistem menggunakan metode *black box testing* dilakukan untuk memverifikasi bahwa sistem yang dibangun telah berjalan dan berperilaku sesuai dengan yang diharapkan. Sistem yang berjalan diuji secara langsung untuk mengamati bagaimana sistem berfungsi dan berperilaku saat menerima *input* dan mengeluarkan *output* dengan tepat. Berdasarkan pengujian yang telah dilakukan, sistem telah memenuhi seluruh skenario pengujian, yaitu:

Tabel 4.5. Hasil Skenario Pengujian *Black Box*

No	Halaman	Pengujian	Hasil	
			Sukses	**Gagal**
1	*Model Performance*	Menampilkan evaluasi model ResNet-LSTM, CNN-LSTM, ResNet, CNN, LSTM	✓	
2	*Pollutant Predictions*	Menampilkan hasil prediksi PM_{10}, NO_2, O_3, CO, SO_2, ISPU berdasarkan tanggal dan lokasi yang dipilih	✓	
		Menampilkan jenis polutan dengan nilai tertinggi	✓	
		Menampilkan kategori ISPU	✓	
3	ISPU *Predictions*	Menampilkan hasil prediksi ISPU seluruh kota/kabupaten di DKI Jakarta berdasarkan tanggal yang dipilih dalam bentuk angka	✓	
		Menampilkan hasil prediksi dan kategori ISPU DKI Jakarta berdasarkan tanggal yang dipilih	✓	

Tabel 4.6. Lanjutan Hasil Skenario Pengujian *Black Box*

No	Halaman	Pengujian	Hasil	
			Sukses	Gagal
4	*Graphs*	Menampilkan grafik polutan berdasarkan rentang tanggal dan lokasi yang dipilih	✓	
5	*Dataset*	Menampilkan seluruh *dataset* yang sudah bersih	✓	

4.2 Pembahasan

Bagian ini menjelaskan hasil penelitian yang diperoleh untuk menjawab masalah penelitian yang diangkat. Masalah pada penelitian ini yaitu *vanishing gradient* akibat meningkatnya kompleksitas jaringan mengakibatkan akurasi menjadi tidak optimal yang diatasi dengan model ResNet-LSTM. Selain itu, penelitian ini juga akan membandingkan hasil akurasi prediksi dengan atau tanpa ResNet atau LSTM.

Data yang digunakan dalam penelitian ini adalah data primer-sekunder, yaitu data polutan yang diperoleh dari *website* Satu Data Jakarta dan data meteorologi dari *website* Data Online Pusat Database BMKG. Data diambil setiap hari dari 1 Januari 2019 hingga 31 Desember 2021 yang berjumlah 5.480 data secara keseluruhan dan 1.096 data untuk masing-masing dari kelima stasiun. Data-data tersebut kemudian dilakukan *preprocessing* yang terbagi menjadi enam tahap agar data yang digunakan menjadi bersih dan siap digunakan dalam pembuatan model. Data yang tidak diperlukan dihilangkan dan nilai yang hilang pada data diisi dengan metode interpolasi linear. Analisis data juga dilakukan untuk melihat korelasi spasial dan temporal pada data polutan dan meteorologi. Lalu, data dibagi menjadi fitur yang akan dilatih dalam model prediksi dan label yang menjadi target *output*. Data-data *array* ini kemudian dibagi menjadi data *training* dan data *testing* dengan rasio sebesar 7:3, di mana 70% untuk data *training* dan 30% untuk data *testing*.

Model dibangun menggunakan arsitektur ResNet dan metode LSTM dengan satu *layer* konvolusi dengan *filter* 3 dan *kernel size* 3x3 *pixel*, lima *block residual* dengan *filter* 9, 16, 64, dan dua block 128 yang masing-masing terdiri dari dua *layer*, di mana *layer* pertama memiliki *kernel size* 3x3 *pixel* dan *layer* kedua memiliki *kernel size* 1x1 *pixel*, serta satu *layer* LSTM yang memiliki 64 *unit*. *Optimizer* yang digunakan dalam pelatihan ini adalah SGD dengan *batch size* 64 serta masing-masing *epochs* dan *learning rate* yang optimal sebesar 100 dan 0.001. Hasil evaluasi model ResNet-LSTM menghasilkan *training error* sebesar 7.5%, *validation error* sebesar 22.7%, dan *testing error* sebesar 19.1%. Berdasarkan hasil evaluasi model tersebut, metode ResNet-LSTM memiliki nilai *error* yang lebih rendah dibandingkan metode CNN-LSTM. Hal tersebut menunjukkan bahwa metode ResNet-LSTM masih lebih baik dalam mengatasi *vanishing gradient* karena memiliki nilai *error* yang lebih rendah.

Hasil pengujian sistem menggunakan metode *black box testing* yang dilakukan secara langsung juga menunjukkan hasil yang sangat baik di mana sistem berhasil melakukan prediksi konsentrasi polutan pada tiap kota/kabupaten di DKI Jakarta dan prediksi ISPU di DKI Jakarta berdasarkan tanggal dan lokasi yang diinputkan oleh pengguna. Hasil pengujian model membuktikan bahwa kombinasi arsitektur ResNet dan metode LSTM mampu mengatasi masalah *vanishing gradient* akibat meningkatkan kompleksitas jaringan pada pelatihan model, tetapi belum mampu mengoptimalkan akurasi prediksi konsentrasi polutan karena terjadi *overfitting*. Meskipun demikian, keberhasilan pengujian secara langsung dapat dipengaruhi oleh beberapa kondisi, seperti ketersediaan data *input* berdasarkan indeks tanggal sebelum tanggal *input* dan data pendukung seperti data jumlah penggunaan kendaraan, data jumlah pabrik industri, dan lainnya.

BAB V
PENUTUP

5.1 Kesimpulan

Berdasarkan penelitian yang telah dilakukan, dapat diperoleh kesimpulan sebagai berikut:

1. Kombinasi ResNet-LSTM memiliki nilai *error* yang lebih rendah dibandingkan metode CNN-LSTM, sehingga mampu mengatasi masalah *vanishing gradient* selama proses pelatihan model, tetapi tetap mengalami masalah *overfitting*.
2. Hasil evaluasi MAPE pada model dengan masing-masing *epochs* dan *learning rate* yang optimal sebesar 100 dan 0.001 menghasilkan *training error* sebesar 7.5%, *validation error* sebesar 22.7%, dan *testing error* sebesar 19.1%.

5.2 Saran

Berdasarkan penelitian yang telah dilakukan, terdapat kekurangan dan berpotensi untuk dikembangkan dalam penelitian selanjutnya. Adapun saran yang dapat diberikan sebagai berikut:

1. Optimalisasi model ResNet-LSTM dengan mencari kemungkinan *hyperparameter* dan arsitektur untuk menurukan nilai *error* serta mengatasi masalah *overfitting*.
2. Penambahan data pendukung seperti data jumlah penggunaan kendaraan, data jumlah pabrik industri, dan lainnya.
3. Penambahan ketersediaan data *input* berdasarkan indeks tanggal sebelum tanggal *input* agar pengguna dapat memilih tanggal di atas 1 Januari 2022.
4. Mengembangkan *user interface* yang lebih responsi dan men-*deploy* sistem ke platform yang lebih sesuai untuk memastikan sistem dapat memberikan manfaat secara optimal.

DAFTAR PUSTAKA

Abdi, H., & Lynne J., W. (2010). *Normalizing data*. Encyclopedia of Research Design. http://www.utd.edu/

Ackerman, S. A., & Knox, J. A. (2011). *Meteorology* (3rd ed.). Jones & Bartlett Publishers.

Alizanovic, V. (2023). *Jenis-jenis Data dalam Data Mining*. Https://Pacmann.Io/Blog/Jenis-Data-Dalam-Data-Mining.

Allenbrand, C. (2023). Supervised and unsupervised learning models for pharmaceutical drug rating and classification using consumer generated reviews. *Healthcare Analytics*, *5*. https://doi.org/10.1016/j.health.2023.100288

Amalia, A., Zaidiah, A., Isnainiyah, I. N., Program,), S1, S., Informasi, S., Komputer, I., Veteran, U., Jl, J. R., Fatmawati, P., & Labu, J. (2022). Prediksi Kualitas Udara Menggunakan Algoritma K-Nearest Neighbor. *JIPI (Jurnal Ilmiah Penelitian Dan Pembelajaran Informatika) Volume 07, 7*(2). https://data.jakarta.go.id/.

Amran, A., Islami, Muh. I., Jaya, A. K., & Bakri, B. (2020). Spatio-Temporal Model of Rainfall Data Using Kalman Filter and Expectation-Maximization Algorithm. *Jurnal Matematika, Statistika Dan Komputasi*, *17*(2), 304–313. https://doi.org/10.20956/jmsk.v17i2.11918

Arif, A. (2019). *Polusi Udara Ancaman Serius bagi Kesehatan*. Https://Www.Kompas.Id/Baca/Utama/2019/04/15/Polusi-Udara-Ancaman-Serius-Bagi-Kesehatan.

Asgari, M., Yang, W., & Farnaghi, M. (2022). Spatiotemporal data partitioning for distributed random forest algorithm: Air quality prediction using imbalanced big spatiotemporal data on spark distributed framework. *Environmental Technology and Innovation*, *27*. https://doi.org/10.1016/j.eti.2022.102776

Ashshiddiqi, H., Jati, P., & Lelono, D. (2013). Deteksi dan Monitoring Polusi Udara Berbasis Array Sensor Gas. *IJEIS*, *3*(2), 147–156.

Basodi, S., Ji, C., Zhang, H., & Pan, Y. (2020). Gradient amplification: An efficient way to train deep neural networks. *Big Data Mining and Analytics*, *3*(3), 196–207. https://doi.org/10.26599/BDMA.2020.9020004

Bengio, Y., Simard, P., & Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, *5*(2), 157–166. https://doi.org/10.1109/72.279181

Badan Meteorologi, K. dan G. (2019). *Data Iklim Harian*. Https://Dataonline.Bmkg.Go.Id/Dashboard_user.

Badan Meteorologi, K. dan G. (2020). *Data Iklim Harian*. Https://Dataonline.Bmkg.Go.Id/Dashboard_user.

Badan Meteorologi, K. dan G. (2021). *Data Iklim Harian*. Https://Dataonline.Bmkg.Go.Id/Dashboard_user.

Chaniago, D., Zahara, A., & Ramadhani, I. S. (2020a, September 24). *INDEKS STANDAR PENCEMAR UDARA (ISPU) SEBAGAI INFORMASI MUTU UDARA AMBIEN DI INDONESIA*. Https://Ditppu.Menlhk.Go.Id/Portal/Read/Indeks-Standar-Pencemar-Udara-Ispu-Sebagai-Informasi-Mutu-Udara-Ambien-Di-Indonesia.

Chaniago, D., Zahara, A., & Ramadhani, I. S. (2020b, September 24). *INDEKS STANDAR PENCEMAR UDARA (ISPU) SEBAGAI INFORMASI MUTU UDARA*

AMBIEN DI INDONESIA. Https://Ditppu.Menlhk.Go.Id/Portal/Read/Indeks-Standar-Pencemar-Udara-Ispu-Sebagai-Informasi-Mutu-Udara-Ambien-Di-Indonesia.

Cheng, X., Zhang, W., Wenzel, A., & Chen, J. (2022). Stacked ResNet-LSTM and CORAL model for multi-site air quality prediction. *Neural Computing and Applications*, *34*(16), 13849–13866. https://doi.org/10.1007/s00521-022-07175-8

Chung, J., Gulcehre, C., Cho, K., & Bengio, Y. (2014). Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. *ArXiv*. http://arxiv.org/abs/1412.3555

de Myttenaere, A., Golden, B., Le Grand, B., & Rossi, F. (2016). Mean Absolute Percentage Error for regression models. *Neurocomputing*, *192*, 38–48. https://doi.org/10.1016/j.neucom.2015.12.114

Dewi, S. P., Nurwati, N., & Rahayu, E. (2022). Penerapan Data Mining untuk Prediksi Penjualan Produk Terlaris Menggunakan Metode K-Nearest Neighbor. *Building of Informatics, Technology and Science (BITS)*, *3*(4), 639–648. https://doi.org/10.47065/bits.v3i4.1408

Dinas Lingkungan Hidup. (2019). *Indeks Standar Pencemaran Udara (ISPU) 2019*. Https://Satudata.Jakarta.Go.Id/Open-Data/Detail?Kategori=dataset&page_url=indeks-Standar-Pencemaran-Udara-Ispu-Tahun-2019&data_no=1.

Dinas Lingkungan Hidup. (2020). *Indeks Standar Pencemaran Udara (ISPU) 2020*. Https://Satudata.Jakarta.Go.Id/Open-Data/Detail?Kategori=dataset&page_url=indeks-Standar-Pencemaran-Udara-Ispu-Tahun-2020&data_no=1.

Dinas Lingkungan Hidup. (2021). *Indeks Standar Pencemaran Udara (ISPU) 2021*. Https://Satudata.Jakarta.Go.Id/Open-Data/Detail?Kategori=dataset&page_url=indeks-Standar-Pencemaran-Udara-Ispu-Tahun-2021&data_no=1.

Duan, J., Zuo, H., Bai, Y., Duan, J., Chang, M., & Chen, B. (2021). Short-term wind speed forecasting using recurrent neural networks with error correction. *Energy*, *217*. https://doi.org/10.1016/j.energy.2020.119397

Dun, A., Yang, Y., & Lei, F. (2022). Dynamic graph convolution neural network based on spatial-temporal correlation for air quality prediction. *Ecological Informatics*, *70*. https://doi.org/10.1016/j.ecoinf.2022.101736

Eko H., R. (2017). *INDEKS STANDAR PENCEMAR UDARA*. Https://Lingkungan.Itats.Ac.Id/Indeks-Standar-Pencemar-Udara/.

Firman, Nurhidayah, P., & Pratama, R. J. (2023, September 22). *INDEKS STANDAR PENCEMAR UDARA (ISPU)*. Https://Arcgis.Jabarprov.Go.Id/Portal/Apps/Storymaps/Stories/5cbe0dd241cc4630bb2ce92eabf3772c.

Gnauck, A. (2004). Interpolation and approximation of water quality time series and process identification. *Analytical and Bioanalytical Chemistry*, *380*.

Gupta, P., Varshney, A., Khan, M. R., Ahmed, R., Shuaib, M., & Alam, S. (2022). Unbalanced Credit Card Fraud Detection Data: A Machine Learning-Oriented Comparative Study of Balancing Techniques. *Procedia Computer Science*, *218*, 2575–2584. https://doi.org/10.1016/j.procs.2023.01.231

Hamdi, A., Shaban, K., Erradi, A., Mohamed, A., Rumi, S. K., & Salim, F. D. (2022). Spatiotemporal data mining: a survey on challenges and open problems. *Artificial*

Intelligence Review, *55*(2), 1441–1488. https://doi.org/10.1007/s10462-021-09994-y

Hartatik, Kwintiana, B., Nengsih, T. A., Baradja, A., Harto, B., Robet, Sudipa, I. G. I., Handika, I. P. S., Adhicandra, I., Gugat, R. M. D., & Terttiaavini. (2023). *DATA SCIENCE FOR BUSINESS : Pengantar & Penerapan Berbagai Sektor* (Efitra & Sepriano, Eds.). PT. Sonpedia Publishing Indonesia.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). *Deep Residual Learning for Image Recognition*. http://image-net.org/challenges/LSVRC/2015/

Heldayani, E., Setianto, H., & Adji Nugroho, Y. (2021). VISUALISASI SPATIO TEMPORAL KASUS COVID-19 DI KOTA PALEMBANG. *Jurnal Pendidikan Geografi Undiksha*, *9*(2), 56–67. https://doi.org/10.23887/jjpg.v9i2.30177

Hernawati, R., Reza, M., Kusuma, C., & Darmawan, S. (2020). *Correlation Analysis of PM10 Air Pollution with NDVI (Normalized Difference Vegetation Index) Based on Landsat-8 and Sentinel-2A Satellite Images. (Case Study: Bandung City, West Correlation Analysis of PM10 Air Pollution with NDVI (Normalized Difference Vegetation Index) Based on Landsat-8 and Sentinel-2A Satellite Images. (Case Study: Bandung City, West Java)*. https://www.researchgate.net/publication/377404321

Hidayat, A., & Anov, D. (2023). Manajemen Proyek Menggunakan Jira pada Information Security Division di Bank BRI. *Jurnal Manajemen Informatika*, *9*(2), 64–65.

Higienis. (2023). *Index Kualitas Udara (AQI): Pedoman Kualitas Udara*. Https://Www.Higienis.Com/Blog/Index-Kualitas-Udara-Aqi-Pedoman-Kualitas-Udara/.

Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory . *Neural Computation*, *9*(8).

Janiesch, C., Zschech, P., & Heinrich, K. (2021). Machine learning and deep learning. *Electronic Markets*, *31*. https://doi.org/10.1007/s12525-021-00475-2/Published

Jimenez-Mesa, C., Arco, J. E., Martinez-Murcia, F. J., Suckling, J., Ramirez, J., & Gorriz, J. M. (2023). Applications of machine learning and deep learning in SPECT and PET imaging: General overview, challenges and future prospects. In *Pharmacological Research* (Vol. 197). Academic Press. https://doi.org/10.1016/j.phrs.2023.106984

Kalajdjieski, J., Zdravevski, E., Corizzo, R., Lameski, P., Kalajdziski, S., Pires, I. M., Garcia, N. M., & Trajkovik, V. (2020). Air pollution prediction with multi-modal data and deep neural networks. *Remote Sensing*, *12*(24), 1–19. https://doi.org/10.3390/rs12244142

Kartasapoetra, A. G. (2004). *Klimatologi Pengaruh Iklim Terhadap Tanah dan Tanaman*. PT. Bumi Aksara.

Keita, Z. (2023, November). *An Introduction to Convolutional Neural Networks (CNNs)*. Https://Www.Datacamp.Com/Tutorial/Introduction-to-Convolutional-Neural-Networks-Cnns.

Khair, U., Fahmi, H., Hakim, S. Al, & Rahim, R. (2017). Forecasting Error Calculation with Mean Absolute Deviation and Mean Absolute Percentage Error. *Journal of Physics: Conference Series*, *930*(1). https://doi.org/10.1088/1742-6596/930/1/012002

Kim, H. G., Cho, K. H., & Recknagel, F. (2023). Time-series modelling of harmful cyanobacteria blooms by convolutional neural networks and wavelet generated

time-frequency images of environmental driving variables. *Water Research*, *246*. https://doi.org/10.1016/j.watres.2023.120662

Kow, P. Y., Wang, Y. S., Zhou, Y., Kao, I. F., Issermann, M., Chang, L. C., & Chang, F. J. (2020). Seamless integration of convolutional and back-propagation neural networks for regional multi-step-ahead PM2.5 forecasting. *Journal of Cleaner Production*, *261*. https://doi.org/10.1016/j.jclepro.2020.121285

Lakitan, B. (2002). *Dasar-dasar Klimatologi*. PT. Raja Grafindo Persada.

Lara-Benítez, P., Carranza-García, M., García-Gutiérrez, J., & Riquelme, J. C. (2020). Asynchronous dual-pipeline deep learning framework for online data stream classification. *Integrated Computer-Aided Engineering*, *27*(2), 101–119. https://doi.org/10.3233/ICA-200617

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436–444. https://doi.org/10.1038/nature14539

Lepot, M., Aubin, J. B., & Clemens, F. H. L. R. (2017). Interpolation in time series: An introductive overview of existing methods, their performance criteria and uncertainty assessment. In *Water (Switzerland)* (Vol. 9, Issue 10). MDPI AG. https://doi.org/10.3390/w9100796

Li, G., Guo, S., Li, X., & Cheng, C. (2023). Short-term Forecasting Approach Based on bidirectional long short-term memory and convolutional neural network for Regional Photovoltaic Power Plants. *Sustainable Energy, Grids and Networks*, *34*. https://doi.org/10.1016/j.segan.2023.101019

Linsley, R. K., Kohler, M. A., & Paulhus, J. L. (1996). *Hidrologi untuk Insinyur*.

Liu, W., Guo, G., Chen, F., & Chen, Y. (2019). Meteorological pattern analysis assisted daily PM2.5 grades prediction using SVM optimized by PSO algorithm. *Atmospheric Pollution Research*, *10*(5), 1482–1491.

Lu, X., Ma, C., & Qiao, Y. (2021). Short-term demand forecasting for online car-hailing using ConvLSTM networks. *Physica A: Statistical Mechanics and Its Applications*, *570*. https://doi.org/10.1016/j.physa.2021.125838

Mandlekar, A., Ramos, F., Boots, B., Savarese, S., Fei-Fei, L., Garg, A., & Fox, D. (2019). *IRIS: Implicit Reinforcement without Interaction at Scale for Learning Control from Offline Robot Manipulation Data*. http://arxiv.org/abs/1911.05321

Millah, H. R., Sudiadnyana, I. W., Aryana, I. K., & Sali, I. W. (2022). Hubungan Faktor Meteorologis dan Kepadatan Lalu Lintas dengan Kualitas Udara di Kota Tabanan. *Jurnal Kesehatan Lingkungan*, *12*(2).

Mobarak, M. H., Mimona, M. A., Islam, M. A., Hossain, N., Zohura, F. T., Imtiaz, I., & Rimon, M. I. H. (2023). Scope of machine learning in materials research—A review. In *Applied Surface Science Advances* (Vol. 18). Elsevier B.V. https://doi.org/10.1016/j.apsadv.2023.100523

Mufadhol, N. (2022, April 12). *Partikulat Matter (PM2.5)*. Https://Staklim-Sumsel.Bmkg.Go.Id/Partikulat-Matter-Pm2-5/#:~:Text=Partikulat%20(PM%202.5)%20adalah%20partikel,BMKG%20dimulai%20sejak%20tahun%202015.

Nurmaini, S., Darmawahyuni, A., Sapitri, A. I., Rachmatullah, M. N., Firdaus, & Tutuko, B. (2021). *Pengenalan Deep Learning dan Implementasinya* (A. Darmawahyuni, Ed.). UPT. Penerbit dan Percetakan.

Onan, A., & Toçoğlu, M. A. (2021). A Term Weighted Neural Language Model and Stacked Bidirectional LSTM Based Framework for Sarcasm Identification. *IEEE Access*, *9*, 7701–7722. https://doi.org/10.1109/ACCESS.2021.3049734

Patro, S. G. K., & sahu, K. K. (2015). Normalization: A Preprocessing Stage. *IARJSET*, 20–22. https://doi.org/10.17148/iarjset.2015.2305

Pichka, E. (2023, January 12). *The Best Resources to Learn Reinforcement Learning*. Https://Towardsdatascience.Com/Best-Free-Courses-and-Resources-to-Learn-Reinforcement-Learning-Ed6633608cb2.

Portal-Porras, K., Fernandez-Gamiz, U., Zulueta, E., Irigaray, O., & Garcia-Fernandez, R. (2023). Hybrid LSTM+CNN architecture for unsteady flow prediction. *Materials Today Communications*, *35*. https://doi.org/10.1016/j.mtcomm.2023.106281

Pressman, R. S. (2015). *Rekayasa Perangkat Lunak: Pendekatan Praktisi Buku I* (7th ed.). Andi Offset.

Qin, D., Yu, J., Zou, G., Yong, R., Zhao, Q., & Zhang, B. (2019). A Novel Combined Prediction Scheme Based on CNN and LSTM for Urban PM2.5 Concentration. *IEEE Access*, *7*, 20050–20059. https://doi.org/10.1109/ACCESS.2019.2897028

Ramadhan P, R. (2021). *Indeks Standar Pencemar Udara (ISPU) Berbasis Android : "ISPU Net."* Https://Lingkunganhidup.Jogjakota.Go.Id/Detail/Index/330.

Ren, B. (2020). The use of machine translation algorithm based on residual and LSTM neural network in translation teaching. *PLoS ONE*, *15*(11 November). https://doi.org/10.1371/journal.pone.0240663

Ridhovan, A., & Suharso, A. (2022). PENERAPAN METODE RESIDUAL NETWORK (RESNET) DALAM KLASIFIKASI PENYAKIT PADA DAUN GANDUM. *JIPI (Jurnal Ilmiah Penelitian Dan Pembelajaran Informatika)*, *7*(1).

Roddick, J. F., & Spiliopoulou, M. (1999). *A Bibliography of Temporal, Spatial and Spatio-Temporal Data Mining Research*.

Rozadi, M. (2019). *Peta Penggunaan Lahan*. Https://Muhammadrozadi.Wordpress.Com/2019/08/26/Peta-Penggunaan-Lahan/.

Samet, H. (1995). *Spatial Data Structures*. Addison WesleyyACM Press.

Sari, N. K. (2015). PENENTUAN KORELASI CURAH HUJAN DAN KETINGGIAN LAPISAN INVERSI DAN HUBUNGANNYA DENGAN KUALITAS UDARA AMBIEN KOTA SURABAYA. *Jurnal Teknik ITS*, *4*(1).

Schlegel, S., Korn, N., & Scheuermann, G. (2012). On the Interpolation of Data with Normally Distributed Uncertainty for Visualization. *IEEE Transactions on Visualization and Computer Graphics*, *18*(12), 2305–2314. https://doi.org/10.1109/TVCG.2012.249

Serlina, Y. (2020). Pengaruh Faktor Meteorologi Terhadap Konsentrasi NO 2 di Udara Ambien (Studi Kasus Bundaran Hotel Indonesia DKI Jakarta). *Serambi Engineering*, *V*(3).

Sheng, Z., An, Z., Wang, H., Chen, G., & Tian, K. (2023). Residual LSTM based short-term load forecasting. *Applied Soft Computing*, *144*. https://doi.org/10.1016/j.asoc.2023.110461

Song, S., Lam, J. C. K., Han, Y., & Li, V. O. K. (2020). ResNet-LSTM for Real-Time PM2.5and PM Estimation Using Sequential Smartphone Images. *IEEE Access*, *8*, 220069–220082. https://doi.org/10.1109/ACCESS.2020.3042278

Suryanto, W., & Luthfian, A. (2019). *Pengantar Meteorologi Dasar-dasar Ilmu Tentang Cuaca*. Gadjah Mada University Press.

Szegedy, C., Ioffe, S., Vanhoucke, V., & Alemi, A. A. (2017). *Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning*. www.aaai.org

Taylor, R. (1990). Interpretation of the Correlation Coefficient A Basic Review. *Journal of Diagnostic Medical Sonography*, *6*(1).
Tjasjono, B. (1999). *Klimatologi Umum* (2nd ed.). Penerbit ITB.
Tjasjono, B. (2004). *Klimatologi*. Penerbit ITB.
Wang, H. (2021). *Searching by learning : exploring artificial general intelligence on small board games by deep reinforcement learning*.
Wang, J., Li, X., Li, J., Sun, Q., & Wang, H. (2022). NGCU: A New RNN Model for Time-Series Data Prediction. *Big Data Research*, *27*. https://doi.org/10.1016/j.bdr.2021.100296
Wang, J., Xue, M., Culhane, R., Diao, E., Ding, J., & Tarokh, V. (2020). Speech Emotion Recognition with Dual-Sequence LSTM Architecture. *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 6474–6478. https://doi.org/10.1109/ICASSP40776.2020.9054629
Wang, Z., Zheng, W., Song, C., Zhang, Z., Lian, J., Yue, S., & Ji, S. (2019). Air Quality Measurement Based on Double-Channel Convolutional Neural Network Ensemble Learning. *IEEE Access*, *7*, 145067–145081. https://doi.org/10.1109/ACCESS.2019.2945805
Weigend, A. S. (2018). *Time series prediction: forecasting the future and understanding the past*. Routledge.
Wibawana, W. A. (2023, August 15). *5 Kategori Indeks Kualitas Udara di Indonesia Menurut ISPU KLHK* . Https://News.Detik.Com/Berita/d-6877417/5-Kategori-Indeks-Kualitas-Udara-Di-Indonesia-Menurut-Ispu-Klhk.
Wiharja. (2002). *Identifikasi Kualitas Gas So2 Di Daerah Industri Pengecoran Logam Ceper*.
World Health Organization. (2010). *Global Influenza Programme*. Https://Www.Who.Int/Tools/Flunet.
Wu, C. lin, He, H. di, Song, R. feng, Zhu, X. hang, Peng, Z. ren, Fu, Q. yan, & Pan, J. (2023). A hybrid deep learning model for regional O3 and NO2 concentrations prediction based on spatiotemporal dependencies in air quality monitoring network. *Environmental Pollution*, *320*. https://doi.org/10.1016/j.envpol.2023.121075
Zhang, B., Zou, G., Qin, D., Ni, Q., Mao, H., & Li, M. (2022). RCL-Learning: ResNet and convolutional long short-term memory-based spatiotemporal air pollutant concentration prediction model. *Expert Systems with Applications*, *207*. https://doi.org/10.1016/j.eswa.2022.118017
Zhang, L., Li, D., & Guo, Q. (2020). Deep Learning from Spatio-Temporal Data Using Orthogonal Regularizaion Residual CNN for Air Prediction. *IEEE Access*, *8*, 66037–66047. https://doi.org/10.1109/ACCESS.2020.2985657
Zhang, L., Zhu, G., Shen, P., Song, J., Afaq Shah, S., & Bennamoun, M. (2017). *Learning Spatiotemporal Features using 3DCNN and Convolutional LSTM for Gesture Recognition*.
Zhang, Q., Li, V. O., Ck Lam, J., & Han, Y. (2020). *Deep-AIR: A Hybrid CNN-LSTM Framework for Fine-Grained Air Pollution Forecast*.
Zheng, Y., Liu, Q., Chen, E., Ge, Y., & Zhao, J. L. (2014). Time Series Classification Using Multi-Channels Deep Convolutional Neural Networks. *International Conference on Web-Age Information Management*.
Zipfel, J., Verworner, F., Fischer, M., Wieland, U., Kraus, M., & Zschech, P. (2023). Anomaly detection for industrial quality assurance: A comparative evaluation of

unsupervised deep learning models. *Computers and Industrial Engineering*, *177*. https://doi.org/10.1016/j.cie.2023.109045