PREDIKSI PERUBAHAN LUAS HUTAN ALAM PER PROVINSI DI INDONESIA MENGGUNAKAN ARIMAX DAN GRU BERBASIS DATA DEFORESTASI 2001-2022

Laporan Akhir Tugas Besar Machine Learning

Oleh:
Bintang Fikri Fauzan 122140008
Cindy Nadila Putri 122140002
Ferdana Al Halkim 122140012
M. Fakhri Nur 122140034
Rafki Haykhal Alif 122140035
Arkan Hariz C. Liem 122140038
Naufal Haris N. 122140040

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI SUMATERA
LAMPUNG SELATAN
2025

DAFTAR ISI

List of Tables

List of Figures

Chapter 1

PENDAHULUAN

1.1 Latar Belakang Masalah

Hutan Indonesia, yang berperan vital sebagai penyerap karbon dan pelindung keanekaragaman hayati, terus terancam oleh deforestasi. Penyusutan luas hutan alam, tercatat hingga tingkat provinsi, disebabkan oleh alih fungsi lahan untuk perkebunan dan industri, serta diperburuk oleh kebakaran hutan dan lahan gambut [3]. Dampaknya sangat serius, meningkatkan emisi gas rumah kaca yang memperparah pemanasan global dan mengancam kelestarian spesies [3]. Diperlukan sistem pemantauan dan prediksi yang handal untuk mendukung mitigasi dan pengelolaan hutan berkelanjutan [4].

Untuk memahami dan mengantisipasi perubahan luas hutan alam secara lebih komprehensif, analisis deret waktu terhadap data historis menjadi sangat penting. Dataset utama yang digunakan dalam penelitian ini adalah data historis deret waktu dari tahun 2001 hingga 2022. Data ini terdiri dari dua komponen utama yang saling berkaitan: pertama, data tahunan luas hutan alam per provinsi di seluruh Indonesia, yang salah satu sumbernya dapat diakses melalui Trase Earth [5]. Kedua, data laju deforestasi tahunan per provinsi untuk periode yang sama, yang akan menjadi parameter penting dalam pemodelan. Seluruh rentang waktu tersebut akan dimanfaatkan sebagai dasar pelatihan dan evaluasi model prediktif untuk memperkirakan perubahan luas hutan alam di masa depan. Fokus utama penelitian ini adalah memprediksi perubahan luas hutan alam dengan memanfaatkan dua parameter utama tersebut, yaitu luas hutan alam historis dan laju deforestasi tahunan. Meskipun hanya menggunakan dua parameter, keduanya dianggap mewakili dinamika perubahan tutupan hutan secara signifikan. Keterbatasan metode pemantauan konvensional dalam merespons perubahan cepat di lapangan mendorong pemanfaatan teknologi canggih seperti model pembelajaran mesin untuk analisis prediktif [6, 4, 7].

Dalam konteks analisis data deret waktu dengan variabel eksogen, model statistik seperti ARIMAX (*Autoregressive Integrated Moving Average with Exogenous Variables*) menawarkan pendekatan yang mapan. ARIMAX, sebagai varian dari model ARIMA, mampu memodelkan dependensi temporal dari data luas hutan alam sekaligus memperhitungkan pengaruh variabel eksternal seperti laju deforestasi [7]. Di sisi lain, untuk menangkap pola non-linear yang mungkin kompleks dalam data deret waktu, model *deep learning* seperti GRU (*Gated Recurrent Unit*) menjadi pilihan yang relevan. GRU, sebagai varian dari *Recurrent Neural Networks* (RNN), efektif dalam memproses data sekuensial dan telah terbukti berkinerja baik, terkadang bahkan melampaui arsitektur lain seperti LSTM dengan struktur yang lebih sederhana dan efisiensi komputasi yang

lebih tinggi dalam beberapa studi prediksi kehutanan atau fenologi [6].

Oleh karena itu, penelitian ini bertujuan untuk mengembangkan dan membandingkan model prediksi perubahan luas hutan alam tahunan di tingkat provinsi di Indonesia menggunakan ARIMAX dan GRU. Model akan dilatih dan dievaluasi berdasarkan data luas hutan alam dan laju deforestasi dari tahun 2001 hingga 2022, untuk kemudian digunakan memproyeksikan perubahan hingga tahun 2030. Dengan mengintegrasikan kedua parameter ini dan membandingkan pendekatan statistik (ARIMAX) dengan pendekatan *deep learning* (GRU), diharapkan dapat dihasilkan model prediktif yang tidak hanya akurat tetapi juga memberikan wawasan mengenai metode terbaik untuk kasus ini. Hasil penelitian ini diharapkan dapat memberikan kontribusi signifikan berupa informasi prediktif yang lebih kaya dan akurat bagi pemerintah daerah dan pusat dalam merumuskan kebijakan pengelolaan hutan yang responsif dan berkelanjutan.

1.2 Rumusan Masalah

1. Bagaimana hasil prediksi luas hutan alam tahunan per Provinsi dapat digunakan sebagai informasi strategis dalam mendukung pengelolaan hutan yang berkelanjutan dan mitigasi perubahan iklim di Indonesia?
2. Model manakah antara ARIMAX dan GRU yang menunjukkan kinerja prediksi dan hasil evaluasi yang lebih unggul dan reliabel untuk mendukung upaya mitigasi serta pengelolaan hutan berkelanjutan di Indonesia pada tingkat provinsi?

1.3 Tujuan Percobaan

1. Menyajikan hasil prediksi luas hutan alam tahunan per Provinsi sebagai informasi strategis yang dapat mendukung upaya pengelolaan hutan berkelanjutan dan mitigasi perubahan iklim di Indonesia.
2. Menentukan model terbaik antara ARIMAX dan GRU yang memiliki kinerja prediksi dan hasil evaluasi paling unggul serta reliabel.

1.4 Batasan Masalah

Penelitian ini dibatasi oleh beberapa aspek berikut untuk menjaga fokus dan kelayakan implementasi:

1. Fokus spasial penelitian adalah pada tingkat administrasi Provinsi di seluruh wilayah Indonesia.
2. Periode waktu analisis data dan prediksi dibatasi dari tahun 2001 hingga 2022.
3. Variabel utama yang diprediksi adalah luas hutan alam tahunan.
4. Parameter temporal tambahan yang diintegrasikan ke dalam model terbatas pada data laju deforestasi.
5. Metode pemodelan utama yang dikembangkan dan dievaluasi adalah GRU dan ARIMAX

Chapter 2

DASAR TEORI

2.1 Hutan dan Deforestasi

2.1.1 Definisi dan Peran Hutan Alam

Hutan didefinisikan sebagai suatu kesatuan ekosistem berupa hamparan lahan berisi sumber daya alam hayati yang didominasi pepohonan dalam persekutuan alam lingkungannya, yang satu dengan lainnya tidak dapat dipisahkan. Hutan alam, secara khusus, merujuk pada hutan yang terbentuk melalui proses alami tanpa campur tangan manusia yang signifikan dalam komposisi dan strukturnya. Hutan Indonesia, dengan kekayaan tropisnya, memainkan peran krusial secara global dan nasional [8]. Secara ekologis, hutan berfungsi sebagai "paru-paru dunia" melalui kemampuannya menyerap sejumlah besar karbon dioksida (CO2) dari atmosfer dan melepaskan oksigen, sehingga membantu meregulasi iklim global. Hutan juga merupakan habitat bagi sebagian besar keanekaragaman hayati dunia, menjaga siklus hidrologi dengan mengatur tata air, mencegah erosi dan banjir, serta menjaga kesuburan tanah. Dari aspek sosial-ekonomi, hutan menyediakan sumber daya alam kayu dan non-kayu, serta menjadi tumpuan hidup bagi masyarakat yang tinggal di sekitar kawasan hutan.

2.1.2 Deforestasi: Definisi, Penyebab, dan Dampak di Indonesia

Deforestasi merupakan pengurangan atau hilangnya tutupan hutan akibat aktivitas manusia maupun faktor alam. Di Indonesia, penyebab utama deforestasi meliputi:

1. Alih fungsi hutan menjadi lahan perkebunan kelapa sawit, pertambangan, dan infrastruktur.
2. Kebakaran hutan yang semakin parah saat musim kemarau, terutama di lahan gambut yang menyimpan karbon dalam jumlah besar.
3. Praktik penebangan liar yang masih marak meskipun telah ada berbagai upaya penertiban.

Deforestasi di Indonesia menimbulkan dampak serius bagi lingkungan dan masyarakat. Sebagai salah satu penyumbang emisi gas rumah kaca terbesar di dunia, sekitar 80% emisi karbon Indonesia berasal dari kerusakan hutan [3]. Hilangnya tutupan hutan mengakibatkan kepunahan keanekaragaman hayati, meningkatnya bencana alam seperti banjir dan longsor, serta mengancam kehidupan masyarakat lokal yang bergantung pada hutan. Dampak ini memperparah krisis iklim global sekaligus merusak keseimbangan ekosistem.

2.2 Analisis Deret Waktu

Time Series Analysis (Analisis Deret Waktu) adalah metode statistik dan komputasi untuk menganalisis data yang direkam secara berurutan dalam waktu (misalnya, harian, bulanan, atau per detik). Tujuannya adalah memahami pola, tren, musiman, dan anomali dalam data, serta memprediksi nilai di masa depan [9].

Ciri-ciri utama deret waktu:

1. Ketergantungan Temporal: Nilai saat ini dipengaruhi oleh nilai sebelumnya.
2. Komponen pola: Tren, Musiman, Siklus, dan Variasi Acak.

Aplikasi analisis deret waktu memiliki peran penting dalam berbagai bidang kehidupan nyata. Contohnya, metode ini digunakan dalam prakiraan faktor meteorologi untuk prediksi cuaca, imputasi data yang hilang dalam proses penambangan data, serta deteksi anomali pada data pemantauan guna mendukung kegiatan pemeliharaan di sektor industri. Selain itu, analisis deret waktu juga diterapkan dalam klasifikasi lintasan untuk pengenalan tindakan serta dalam proses diagnosis medis. Pemanfaatan yang luas ini menunjukkan bahwa analisis deret waktu merupakan alat yang esensial dalam mendukung pengambilan keputusan berbasis data di berbagai sektor [10].

2.3 Autoregressive Integrated Moving Average (ARIMA)

Model *Autoregressive Integrated Moving Average* (ARIMA) adalah sebuah generalisasi dari model ARMA (*AutoRegressive Moving Average*) yang cocok untuk menangani data deret waktu non-stasioner [7]. Model ARMA klasik mengasumsikan stasioneritas data deret waktu yang dianalisis, sehingga untuk data non-stasioner, diperlukan transformasi melalui proses diferensiasi untuk menghilangkan tren dan musiman [7]. Model ARIMA pada dasarnya menangani sinyal waktu setelah dipisahkan dari *noise*, dan mengeluarkan prediksi untuk titik waktu berikutnya [7].

Komponen struktural dari model ARIMA adalah sebagai berikut [7]:

- **AR (Autoregression)**: Merupakan model regresi yang menggunakan hubungan dependensi antara sebuah observasi dengan sejumlah observasi yang tertinggal (observasi sebelumnya). Orde dari komponen AR dilambangkan dengan p [7].
- **I (Integrated)**: Merepresentasikan proses diferensiasi data observasi pada titik waktu yang berbeda untuk membuat data deret waktu menjadi stasioner. Orde diferensiasi dilambangkan dengan d. Sebuah data deret waktu dikatakan stasioner jika sifat statistiknya (rata-rata, varians, dll.) stabil sepanjang waktu [7].
- **MA (Moving Average)**: Pendekatan ini mempertimbangkan dependensi yang mungkin ada antara observasi dengan *error term* yang dihasilkan ketika model rata-rata bergerak (*moving average*) diterapkan pada observasi yang memiliki jeda waktu. Orde dari komponen MA dilambangkan dengan q [7].

Bentuk umum dari model ARIMA ditulis sebagai ARIMA(p, d, q). Pemilihan parameter p dan q yang optimal biasanya didasarkan pada pemeriksaan plot *Autocorrelation Function* (ACF) dan *Partial Autocorrelation Function* (PACF), sedangkan parameter d dipilih

untuk membuat deret waktu menjadi stasioner [7]. Kriteria informasi seperti *Akaike Information Criterion* (AIC) atau *Bayesian Information Criterion* (BIC) sering digunakan untuk memilih model ARIMA terbaik di antara beberapa kandidat model, di mana nilai yang lebih rendah menandakan model yang lebih baik. Perlu dicatat bahwa model ARIMA dalam bentuk dasarnya dirancang untuk menangani hubungan data linear dan umumnya digunakan untuk analisis univariat [11].

2.4 ARIMA with Exogenous Variables (ARIMAX)

Model ARIMAX (*Autoregressive Integrated Moving Average with Exogenous Variables*) adalah sebuah perluasan atau varian dari model ARIMA [12, 13]. Fitur utama yang membedakan ARIMAX adalah kemampuannya untuk menyertakan satu atau lebih variabel independen eksternal (variabel eksogen) yang diyakini dapat mempengaruhi variabel deret waktu yang sedang diprediksi [12, 13]. Penambahan variabel eksogen ini memungkinkan analisis multivariat, di mana beberapa parameter dapat mempengaruhi parameter yang diramal [13]. Dengan kata lain, model ARIMAX memperluas model ARIMA dengan menambahkan komponen variabel eksogen [12].

Penggunaan model ARIMAX relevan dalam berbagai konteks di mana faktor eksternal memainkan peran penting. Sebagai contoh, dalam peramalan curah hujan, variabel seperti kecepatan angin maksimum dapat dimasukkan sebagai variabel eksogen [12]. Dalam peramalan volume lalu lintas, ARIMAX memungkinkan integrasi faktor-faktor yang mempengaruhi seperti atribut cuaca [13]. Studi lain juga telah menggunakan ARIMAX untuk prediksi tingkat pengangguran dengan memanfaatkan data dari Google Trends sebagai variabel eksogen [14].

Keuntungan utama dari model ARIMAX adalah kemampuannya untuk memodelkan dampak dari faktor-faktor eksternal ini, yang berpotensi meningkatkan akurasi peramalan dibandingkan dengan model ARIMA univariat standar, terutama jika variabel eksogen tersebut memiliki pengaruh signifikan terhadap variabel dependen [12, 14]. Secara konseptual, model ARIMAX menggabungkan struktur dependensi temporal dari ARIMA dengan model regresi linear untuk variabel-variabel eksogen. Jika y_t adalah variabel dependen pada waktu t, dan $X_{k,t}$ adalah variabel eksogen ke-k pada waktu t, maka model ARIMAX secara umum dapat direpresentasikan setelah proses diferensiasi sebagai:

$$\phi(B)(1-B)^d y_t = c + \sum_{k=1}^{K} \beta_k (1-B)^d X_{k,t} + \theta(B)\epsilon_t \tag{2.1}$$

atau dalam bentuk yang lebih sederhana jika y'_t dan $X'_{k,t}$ adalah data yang telah didiferensiasi:

$$y'_t = c + \sum_{i=1}^{p} \phi_i y'_{t-i} + \sum_{j=0}^{q} \theta_j \epsilon_{t-j} + \sum_{k=1}^{M} \eta_k X'_{k,t-\delta_k} + \epsilon_t \tag{2.2}$$

dimana $\phi(B)$ dan $\theta(B)$ adalah polinomial operator *lag* untuk komponen AR dan MA, $(1-B)^d$ adalah operator diferensiasi, y_t adalah nilai observasi pada waktu t, $X_{k,t}$ adalah variabel eksogen ke-k, β_k atau η_k adalah koefisien untuk variabel eksogen, M adalah jumlah variabel eksogen, δ_k adalah jeda waktu (lag) untuk variabel eksogen, dan ϵ_t adalah *error term* [12, 13]. Meskipun demikian, implementasi praktis seringkali menentukan struktur yang lebih spesifik berdasarkan analisis data. Amelia et al. (2021) menunjukkan bahwa berdasarkan nilai AIC yang diperoleh, nilai ARIMAX masih lebih baik daripada ARIMA dalam kasus peramalan curah hujan mereka [12].

2.5 Reccurent Neural Network (RNN)

2.5.1 Pendahuluan Umum RNN

Recurrent Neural Networks (RNN) adalah arsitektur jaringan saraf dengan keadaan tersembunyi (*hidden state*) dan menggunakan *feedback loops* untuk memproses urutan data yang pada akhirnya menginformasikan output akhir. Oleh karena itu, model RNN dapat mengenali karakteristik sekuensial dalam data dan membantu memprediksi titik data berikutnya yang mungkin dalam urutan data. Dengan memanfaatkan kekuatan pemrosesan data sekuensial, kasus penggunaan RNN cenderung terhubung dengan model bahasa atau analisis data deret waktu [1].

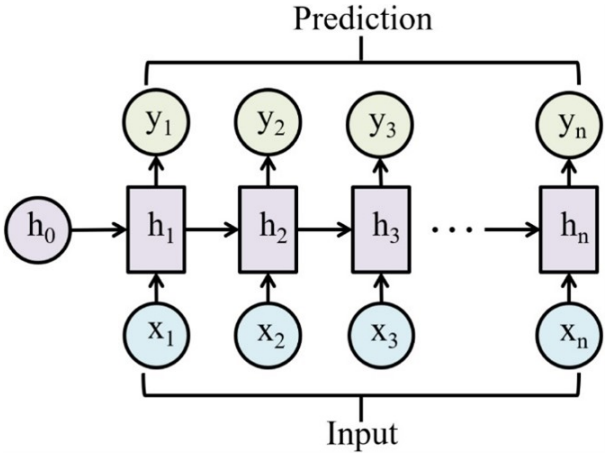

Figure 2.1: Visualisasi RNN menurut [1]

2.5.2 Arsitektur dan Prinsip Kerja RNN

Persamaan Dasar

Operasi RNN pada urutan input x dengan indeks langkah waktu t (mulai dari 1 hingga τ) dapat diilustrasikan. Indeks langkah waktu t tidak selalu merujuk pada berlalunya waktu di dunia nyata; ia bisa merujuk pada posisi dalam urutan. Pada setiap langkah waktu t, RNN mengambil vektor input x_t dan memperbarui keadaan tersembunyinya h_t menggunakan persamaan berikut.

$$h_t = \sigma_h(W_{xh}x_t + W_{hh}h_{t-1} + b_h) \tag{2.3}$$

di mana:

- h_t adalah vektor keadaan tersembunyi pada langkah waktu t.
- h_{t-1} adalah vektor keadaan tersembunyi pada langkah waktu $t-1$.
- x_t adalah vektor input pada langkah waktu t.
- W_{xh} adalah matriks bobot dari lapisan input ke lapisan tersembunyi [15].
- W_{hh} adalah matriks bobot dari lapisan tersembunyi ke lapisan tersembunyi (koneksi berulang) [15].

- b_h adalah vektor bias untuk lapisan tersembunyi [15].
- σ_h adalah fungsi aktivasi untuk lapisan tersembunyi, biasanya fungsi `tanh` atau ReLU [15].

Output y_t pada setiap langkah waktu t (jika diperlukan pada setiap langkah) dihitung sebagai berikut:

$$y_t = \sigma_y(W_{hy}h_t + b_y) \tag{2.4}$$

di mana:

- y_t adalah vektor output pada langkah waktu t.
- W_{hy} adalah matriks bobot dari lapisan tersembunyi ke lapisan output [1].
- b_y adalah vektor bias untuk lapisan output [1].
- σ_y adalah fungsi aktivasi untuk lapisan output (misalnya, sigmoid untuk klasifikasi biner, atau softmax untuk klasifikasi multikelas) [1].

Dalam beberapa literatur, bobot input-ke-hidden (W_{xh}) disebut sebagai U, bobot hidden-ke-hidden (W_{hh}) sebagai W, dan bobot hidden-ke-output (W_{hy}) sebagai V.

Fungsi Aktivasi

Pilihan fungsi aktivasi σ_h dan σ_y memainkan peran krusial dalam perilaku jaringan, memperkenalkan non-linearitas yang memungkinkan jaringan untuk mempelajari dan merepresentasikan pola kompleks dalam data [15].

- **Tanh (Hyperbolic Tangent)**: Sering digunakan untuk σ_h karena memetakan input ke rentang [-1, 1], menjadikannya zero-centered dan cocok untuk memodelkan urutan dengan nilai positif dan negatif.

$$\tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}} \tag{2.5}$$

- **ReLU (Rectified Linear Unit)**: Fungsi ini menghasilkan output input secara langsung jika positif, dan nol jika sebaliknya. Ini membantu mengurangi masalah *vanishing gradient*.

$$\text{ReLU}(z) = \max(0, z) \tag{2.6}$$

- **Sigmoid**: Sering digunakan untuk σ_y dalam tugas klasifikasi biner karena memetakan input ke rentang [0, 1], yang dapat diinterpretasikan sebagai probabilitas.

$$\text{sigmoid}(z) = \frac{1}{1 + e^{-z}} \tag{2.7}$$

- **Softmax**: Umumnya digunakan pada lapisan output dari jaringan klasifikasi multikelas untuk mengubah skor mentah menjadi probabilitas.

$$\text{softmax}(z_i) = \frac{e^{z_i}}{\sum_j e^{z_j}} \tag{2.8}$$

2.5.3 Pelatihan RNN

Pelatihan RNN dilakukan dengan menghitung gradien dari fungsi kerugian (*loss function*) terhadap parameter-parameter yang terlibat dalam propagasi maju (dari kiri ke kanan pada graf yang tidak digulirkan), diikuti oleh propagasi balik (*backpropagation*) yang bergerak dari kanan ke kiri melalui graf. Algoritma propagasi balik yang diterapkan pada RNN dikenal sebagai *Backpropagation Through Time* (BPTT).

Total kerugian untuk urutan nilai x dan nilai y yang sesuai diperoleh dengan menjumlahkan kerugian dari semua langkah waktu:

$$L = \sum_{t=1}^{\tau} L^{(t)} \tag{2.9}$$

di mana $L^{(t)}$ adalah kerugian pada langkah waktu t, dan τ adalah panjang total urutan. Untuk meminimalkan kerugian ini, gradien dari fungsi kerugian dihitung terhadap parameter-parameter terkait $(W_{xh}, W_{hh}, W_{hy}, b_h, b_y)$ [1].

Operasi komputasi ini mahal karena waktu proses tidak dapat dikurangi dengan paralelisasi, mengingat propagasi maju bersifat sekuensial. Keadaan-keadaan yang dihitung dalam propagasi maju disimpan hingga digunakan kembali dalam propagasi balik.

2.5.4 Tantangan dalam Pelatihan RNN

Meskipun RNN efektif dalam memodelkan data sekuensial, RNN standar (SimpleRNN) menghadapi beberapa tantangan signifikan, terutama ketika menangani dependensi jangka panjang.

Vanishing dan Exploding Gradient Problems

Saat melatih RNN, terutama untuk urutan yang panjang, masalah gradien yang menghilang (*vanishing gradient*) atau meledak (*exploding gradient*) sering muncul. Selama BPTT, gradien disebarkan mundur melalui waktu. Jika urutannya panjang, produk berulang dari matriks Jacobian yang terlibat dalam perhitungan gradien dapat menyebabkan nilai gradien menjadi sangat kecil (mendekati nol) atau sangat besar (meledak secara eksponensial).

Secara matematis, turunan parsial dari keadaan tersembunyi h_t terhadap keadaan tersembunyi h_k (untuk $k < t$) melibatkan produk dari banyak matriks Jacobian[:

$$\frac{\partial h_t}{\partial h_k} = \prod_{i=k+1}^{t} \frac{\partial h_i}{\partial h_{i-1}} \approx \prod_{i=k+1}^{t} W_{hh}^T \text{diag}[\sigma_h'(s_i)] \tag{2.10}$$

di mana $s_i = W_{xh}x_i + W_{hh}h_{i-1} + b_h$. Jika nilai eigen dari W_{hh} (atau norma dari matriks Jacobian) secara konsisten kurang dari 1, gradien akan menyusut secara eksponensial (vanishing). Sebaliknya, jika lebih besar dari 1, gradien dapat tumbuh secara eksponensial (exploding).

Masalah *vanishing gradient* membuat jaringan sulit mempelajari dependensi antara elemen-elemen yang berjauhan dalam urutan, karena sinyal gradien menjadi terlalu lemah untuk memperbarui bobot secara berarti untuk lapisan-lapisan awal atau langkah waktu yang lebih awal. Sebaliknya, *exploding gradient* dapat menyebabkan pembaruan bobot yang sangat besar, membuat pelatihan tidak stabil dan berpotensi menyebabkan luapan numerik (*numerical overflow*).

Kesulitan ini timbul karena bobot yang secara eksponensial lebih kecil diberikan pada interaksi jangka panjang dibandingkan dengan interaksi jangka pendek. Butuh waktu sangat lama untuk mempelajari dependensi jangka panjang karena sinyal dari dependensi ini cenderung tersembunyi oleh fluktuasi kecil yang timbul dari dependensi jangka pendek.

2.6 Gated Recurrent Unit

Gated Recurrent Unit (GRU) adalah salah satu varian dari *Recurrent Neural Network* (RNN) yang dirancang untuk mengatasi masalah *short-term memory* dan *vanishing gradient* yang sering muncul pada RNN standar [16, 2]. GRU memiliki struktur yang lebih sederhana dibandingkan dengan varian RNN lainnya seperti *Long Short-Term Memory* (LSTM) [16]. Keunggulan utama GRU terletak pada kemampuannya menggabungkan *forget gate* dan *input gate* yang ada pada LSTM menjadi satu *update gate* tunggal, serta menggabungkan *cell state* dan *hidden state* [2]. Hal ini menghasilkan jumlah tensor operasi yang lebih sedikit dan seringkali proses pelatihan yang lebih cepat dibandingkan LSTM [2].

Sebuah unit GRU terdiri dari dua komponen utama berupa gerbang (gate), yaitu *update gate* (z_t) dan *reset gate* (r_t), serta mekanisme untuk menghasilkan konten memori saat ini [16]. Gerbang-gerbang ini memungkinkan GRU untuk secara selektif memperbarui dan memanfaatkan informasi dari langkah waktu sebelumnya, sehingga mampu menangkap dependensi jangka panjang dalam data sekuensial [16].

1. Update Gate (z_t): *Update gate* menentukan seberapa banyak informasi dari langkah waktu sebelumnya (h_{t-1}) yang harus dipertahankan dan seberapa banyak informasi baru dari input saat ini (x_t) yang harus ditambahkan [16]. Gerbang ini mengatasi masalah *vanishing gradient* karena model mempelajari seberapa banyak informasi yang perlu diteruskan [2]. *Update gate* dihitung berdasarkan gabungan dari *hidden state* sebelumnya (h_{t-1}) dan input saat ini (x_t), yang kemudian dilewatkan melalui fungsi aktivasi sigmoid. Persamaannya adalah sebagai berikut:

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t] + b_z) \tag{2.11}$$

di mana W_z adalah matriks bobot untuk *update gate*, b_z adalah bias, dan σ adalah fungsi sigmoid. Operator $[h_{t-1}, x_t]$ menandakan konkatenasi dari h_{t-1} dan x_t.

2. Reset Gate (r_t): *Reset gate* memutuskan seberapa banyak informasi dari masa lalu yang harus dilupakan atau diabaikan. Perhitungannya mirip dengan *update gate*, juga menggunakan gabungan dari h_{t-1} dan x_t yang dilewatkan melalui fungsi sigmoid [16]:

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t] + b_r) \tag{2.12}$$

di mana W_r adalah matriks bobot untuk *reset gate* dan b_r adalah bias.

3. Candidate Hidden State ($\tilde{h}_t$): Konten memori saat ini atau *candidate hidden state* ($\tilde{h}_t$) dihitung berdasarkan *reset gate* dan gabungan dari *hidden state* sebelumnya yang telah di-reset ($r_t \odot h_{t-1}$) dengan input saat ini (x_t) [2]. Simbol $\odot$ menandakan perkalian elemen-demi-elemen (*element-wise product* atau *Hadamard product*). Hasilnya kemudian dilewatkan melalui fungsi aktivasi *hyperbolic tangent* (tanh) [2]:

$$\tilde{h}_t = \tanh(W_h \cdot [r_t \odot h_{t-1}, x_t] + b_h) \tag{2.13}$$

di mana W_h adalah matriks bobot dan b_h adalah bias untuk perhitungan *candidate hidden state*.

4. Final Hidden State (h_t): *Hidden state* akhir (h_t) pada langkah waktu t ditentukan oleh kombinasi linear antara *hidden state* sebelumnya (h_{t-1}) dan *candidate hidden state* ($\tilde{h}_t$), yang dikontrol oleh *update gate* (z_t). Persamaannya adalah:

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t \tag{2.14}$$

Jika z_t mendekati 0, maka sebagian besar konten $\tilde{h}_t$ akan diabaikan, dan jaringan akan meneruskan mayoritas informasi dari h_{t-1}, begitu pula sebaliknya. GRU tidak memiliki unit memori terpisah seperti LSTM (yaitu, *cell state*), dan ia mengekspos seluruh konten *hidden state* tanpa kontrol keluaran eksplisit seperti pada *output gate* LSTM [2].

Secara umum, GRU menawarkan alternatif yang lebih sederhana dibandingkan LSTM dengan jumlah parameter yang lebih sedikit, memungkinkan pelatihan yang lebih cepat pada beberapa kasus. Meskipun demikian, pilihan antara GRU dan LSTM sangat bergantung pada kasus penggunaan spesifik dan karakteristik data yang dihadapi [16]. Beberapa studi juga menunjukkan bahwa GRU cenderung berkinerja lebih baik daripada LSTM pada dataset pelatihan yang lebih kecil [2].

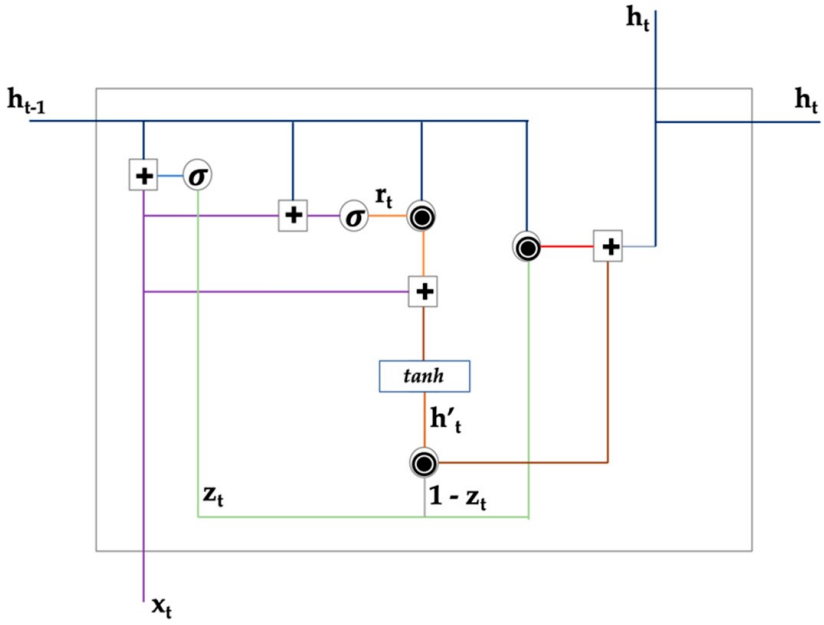

Figure 2.2: Struktur sebuah unit GRU (Sumber: Diadaptasi dari [2])

- **Reset Gate (r_t)**: Mengontrol seberapa banyak informasi dari hidden state sebelumnya (h_{t-1}) yang akan digunakan untuk menghitung kandidat hidden state baru
- **Update Gate (z_t)**: Menentukan seberapa banyak informasi dari hidden state sebelumnya yang akan dipertahankan dan seberapa banyak informasi baru yang akan ditambahkan

Alur Komputasi

1. **Reset Gate**: $r_t = \sigma(W_r \cdot [h_{t-1}, x_t])$
2. **Update Gate**: $z_t = \sigma(W_z \cdot [h_{t-1}, x_t])$

3. **Kandidat Hidden State**: $\tilde{h}_t = \tanh(W \cdot [r_t \odot h_{t-1}, x_t])$

4. **Hidden State Baru**: $h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t$

2.7 Preprocessing

Preprocessing data adalah tahap awal dalam proses analisis data yang bertujuan untuk meningkatkan kualitas data mentah sehingga dapat digunakan secara efektif dalam proses penambangan data dan pembelajaran mesin [17]. Dalam konteks *process mining* maupun *knowledge discovery*, *preprocessing* bukan hanya meningkatkan akurasi analisis, tetapi juga menyederhanakan representasi data agar lebih mudah diinterpretasikan dan diproses [18].

2.7.1 Teknik Preprocessing yang digunakan

Agregasi Data

Agregasi data adalah proses menyajikan data dalam bentuk yang diringkas (*summarized form*), yang direalisasikan dengan menggabungkan dua atau lebih atribut menjadi satu atribut. Agregasi berperan penting dalam mengonversi data yang dikumpulkan dari berbagai sumber ke dalam format yang sesuai. Selain itu, agregasi juga dapat mengurangi ukuran dataset, sehingga membuat penggunaan memori dan waktu menjadi lebih efisien [19].

Handling Outlier dengan Log Transformation

Transformasi logaritmik merupakan salah satu metode dalam data scaling yang bertujuan untuk mereduksi variabilitas ekstrem antar fitur data. Dengan memetakan data ke ruang baru menggunakan fungsi logaritma, perbedaan besar antar nilai dapat diminimalkan. Ini membuat data menjadi lebih stabil dan cocok untuk proses analisis atau pemodelan prediktif, terutama ketika data memiliki distribusi yang sangat miring atau mengandung outlier [17].

Normalisasi dengan Min-Max Scaler

Normalisasi merupakan salah satu teknik penting dalam *data preprocessing* yang bertujuan untuk menyetarakan skala antar fitur dalam dataset. Salah satu metode normalisasi yang paling umum digunakan adalah *Min-Max Normalization*. Teknik ini mengubah skala nilai suatu fitur menjadi rentang tertentu, biasanya antara $[0, 1]$ atau $[-1, 1]$, berdasarkan nilai minimum dan maksimum dari fitur tersebut.

Menurut Çetin dan Yıldız (2022), *Min-Max Normalization* membantu menghindari dominasi fitur yang memiliki skala besar terhadap fitur-fitur lainnya, yang dapat menurunkan kinerja model analisis atau klasifikasi [19]. Rumus Min-Max Normalization dinyatakan sebagai berikut:

$$v' = \frac{v - \min(v)}{\max(v) - \min(v)} \times (new_{max} - new_{min}) + new_{min} \tag{2.15}$$

di mana:

- v adalah nilai asli,

- $\min(v)$ adalah nilai minimum dari fitur,
- $\max(v)$ adalah nilai maksimum dari fitur,
- new_{min} dan new_{max} adalah batas bawah dan atas dari skala baru (misalnya 0 dan 1).

Fan et al. (2021) juga menegaskan bahwa teknik ini umum digunakan dalam konteks data operasional bangunan dan sistem energi, khususnya saat data tidak mengikuti distribusi normal [17]. Namun, mereka memperingatkan bahwa metode ini *sangat sensitif terhadap outlier*, karena kehadiran nilai ekstrem dapat secara signifikan memengaruhi nilai minimum dan maksimum, sehingga berdampak pada skala keseluruhan data.

2.8 Metrik Evaluasi

Evaluasi kinerja model regresi adalah langkah penting untuk memahami kemampuannya dalam membuat prediksi terhadap target kontinu. Bagian ini akan berfokus pada tiga metrik umum: *Mean Absolute Error* (MAE), *Root Mean Squared Error* (RMSE), dan Koefisien Determinasi (R^2).

2.8.1 Mean Absolute Error (MAE)

MAE mengukur rata-rata nilai absolut dari kesalahan prediksi, dihitung sebagai:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{2.16}$$

di mana y_i adalah nilai aktual, $\hat{y}_i$ adalah nilai prediksi, dan n adalah jumlah observasi. Nilai MAE berkisar dari 0 (prediksi sempurna) hingga $+\infty$. Karena menggunakan norma L1, MAE kurang sensitif terhadap nilai ekstrem (*outliers*) dibandingkan RMSE. MAE optimal untuk kesalahan yang terdistribusi Laplacian dan memiliki unit yang sama dengan variabel respons, sehingga mudah diinterpretasikan skalanya. Namun, karena tidak memiliki batas atas, interpretasi nilai MAE tunggal tanpa konteks bisa sulit [20].

2.8.2 Root Mean Squared Error (RMSE)

RMSE adalah akar kuadrat dari rata-rata kuadrat kesalahan, memberikan bobot lebih pada kesalahan besar karena penggunaan norma L2:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{2.17}$$

Nilai RMSE juga berkisar dari 0 (tidak ada kesalahan) hingga $+\infty$ dan memiliki unit yang sama dengan variabel respons. RMSE dianggap optimal untuk kesalahan prediksi yang terdistribusi normal (Gaussian). Seperti MAE, nilai RMSE tunggal juga sulit diinterpretasikan secara absolut tanpa pembanding atau konteks skala data. Pilihan antara RMSE dan MAE seringkali bergantung pada bagaimana peneliti ingin memperlakukan kesalahan besar dan asumsi distribusi kesalahan [21].

2.8.3 Koefisien Determinasi ($\mathbf{R^2}$)

$\mathbf{R^2}$ mengukur proporsi varians dalam variabel dependen yang dapat dijelaskan oleh variabel independen dalam model regresi. $\mathbf{R^2}$ dihitung sebagai:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{2.18}$$

di mana $\bar{y}$ adalah rata-rata dari nilai aktual y_i. Nilai $\mathbf{R^2}$ idealnya berkisar antara 0 dan 1.

- $R^2 = 1$ berarti prediksi sempurna.
- $R^2 = 0$ berarti model tidak lebih baik dari prediksi rata-rata $\bar{y}$[22].
- $R^2 < 0$ berarti model lebih buruk daripada prediksi rata-rata[22].

Karena hubungannya dengan MSE ($R^2 = 1 - \text{MSE/MST}$), $\mathbf{R^2}$ memiliki hubungan monotonik terbalik dengan MSE/RMSE. Keunggulan utama $\mathbf{R^2}$ adalah rentang nilai [0,1] untuk model yang layak, yang memberikan interpretasi yang lebih intuitif tentang kualitas model terlepas dari skala data. $\mathbf{R^2}$ dikatakan lebih informatif karena menghasilkan skor tinggi hanya jika sebagian besar elemen data aktual diprediksi dengan benar dengan mempertimbangkan distribusinya. $\mathbf{R^2}$ juga dapat dipartisi untuk analisis kontribusi prediktor individual (part $\mathbf{R^2}$) atau total (inclusive $\mathbf{R^2}$) menggunakan teknik tertentu. Berdasarkan perbandingan, $\mathbf{R^2}$ sering direkomendasikan sebagai metrik standar untuk evaluasi regresi karena lebih informatif dan interpretatif dibandingkan metrik dengan rentang tak terbatas [22].

Chapter 3

METODOLOGI PENELITIAN

3.1 Diagram Alir Penelitian

Untuk memberikan gambaran menyeluruh mengenai alur kerja penelitian ini, Gambar 3.1 menyajikan diagram alir yang mengilustrasikan secara visual tahapan-tahapan utama yang akan dilaksanakan.

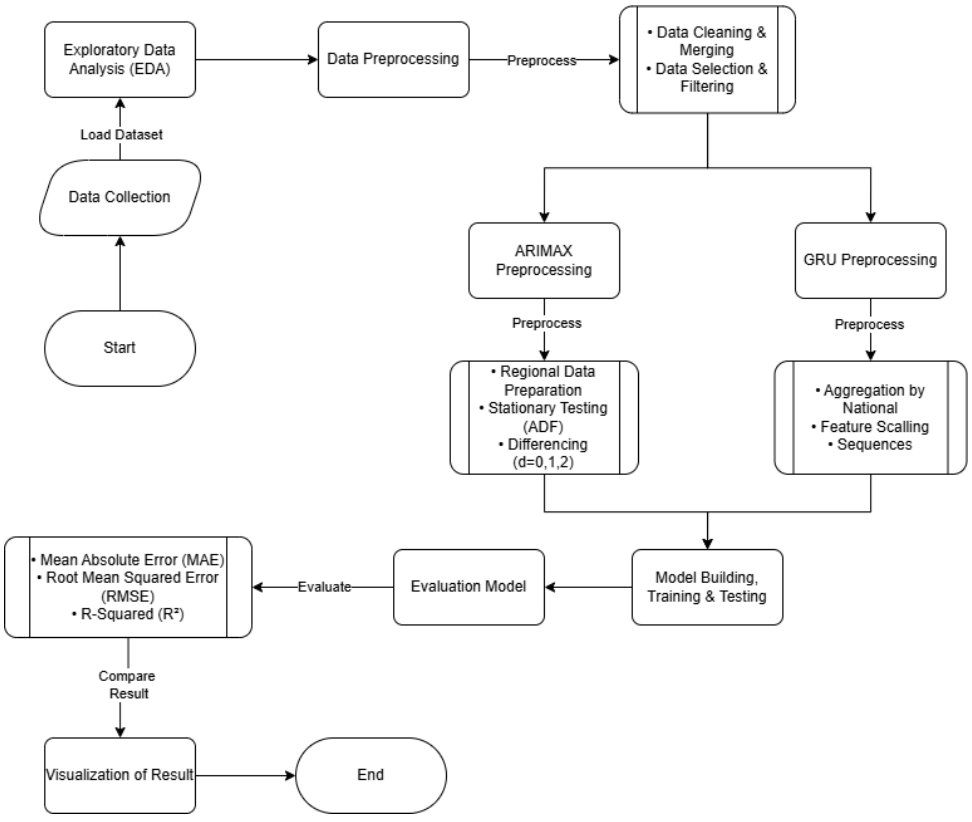

Figure 3.1: Alur Kerja Penelitian

Diagram alir di atas menggambarkan keseluruhan proses penelitian, dimulai dari pengumpulan data luas hutan dan deforestasi. Tahap selanjutnya adalah analisis data eksploratif (EDA) untuk memahami karakteristik data, diikuti dengan pra-pemrosesan data umum yang mencakup pembersihan, penggabungan, serta seleksi dan filter data. Setelah itu, alur terbagi dua untuk pra-pemrosesan spesifik model: untuk ARIMAX, dilakukan persiapan data regional, uji stasioneritas (ADF) dengan diferensiasi (d=0,1,2),

sedangkan untuk GRU, dilakukan agregasi data ke tingkat nasional, penskalaan fitur, dan pembuatan sekuens. Kedua set data yang telah diproses kemudian digunakan untuk membangun, melatih, dan menguji model masing-masing (ARIMAX dan GRU). Kinerja kedua model dievaluasi menggunakan metrik seperti *Mean Absolute Error* (MAE), *Root Mean Squared Error* (RMSE), dan *R-Squared* (R^2). Hasil evaluasi dari kedua model dibandingkan, dan akhirnya, hasil perbandingan serta prediksi divisualisasikan untuk ditarik kesimpulan.

3.2 Tahapan Percobaan

3.2.1 Pemilihan Dataset dan Pembagian Tugas

Pemilihan Dataset

Dataset inti yang digunakan adalah data historis tahunan mengenai luas hutan alam untuk setiap Provinsi di seluruh Indonesia. Data ini mencakup periode studi dari tahun 2001 hingga 2022 dan bersumber dari website trase earth [5]. Variabel kunci lainnya seperti year dari dataset ini juga akan dimanfaatkan untuk analisis temporal. Terdapat parameter temporal tambahan yang digunakan uuntuk meningkatkan kedalaman analisis, yaitu laju deforestasi tahunan dengan periode 2001-2022.

Pembagian Tugas Tim Penelitian

Untuk memastikan efektivitas dan kelancaran pelaksanaan proyek penelitian ini, telah dilakukan pembagian tugas yang terstruktur di antara anggota tim. Bintang Fikri Fauzan berperan sebagai Project Leader yang mengkoordinasikan keseluruhan jalannya penelitian, sekaligus sebagai Design Researcher yang merancang kerangka konseptual dan metodologis. Aspek analisis dan rekayasa data, yang krusial dalam menyiapkan dataset untuk pemodelan, ditangani oleh Cindy Nadila Putri dan Naufal Haris N. sebagai Data Analyst & Data Engineer. Pengembangan dan implementasi model machine learning LSTM menjadi tanggung jawab Ferdana Al Hakim dan Rafki Haykhal Alif selaku ML Engineer. Selanjutnya, evaluasi kinerja model dan visualisasi hasil penelitian dilakukan oleh Arkan Hariz Liem sebagai Evaluator & Results Visualizer. Terakhir, seluruh proses dokumentasi penelitian dan manajemen repositori kode pada platform GitHub dikelola oleh M. Fakhri Nur sebagai Documentation & GitHub Manager. Kolaborasi sinergis antar anggota tim dengan spesialisasi masing-masing ini diharapkan dapat menghasilkan penelitian yang komprehensif dan berkualitas.

3.2.2 Pra-pemrosesan Data (Preprocessing)

Sebelum data dapat digunakan untuk melatih model ARIMAX dan GRU, dilakukan serangkaian tahapan pra-pemrosesan. Tahapan ini krusial untuk memastikan kualitas data dan kesesuaian format data dengan kebutuhan masing-masing model. Berikut adalah langkah-langkah pra-pemrosesan yang dilakukan:

1. **Pembersihan Data**: Tahap awal melibatkan pemeriksaan dan pembersihan data dari anomali atau kesalahan pencatatan. Ini termasuk verifikasi konsistensi data antar tahun dan antar provinsi.

2. **Penggabungan Dataset**: Dua dataset utama, yaitu dataset luas hutan alam per provinsi dan dataset laju deforestasi per provinsi (keduanya dari tahun 2001-2022), digabungkan menjadi satu dataset terstruktur berdasarkan provinsi dan tahun. Penggabungan ini memungkinkan analisis keterkaitan antara luas hutan dan laju deforestasi secara langsung.

3. **Seleksi Kolom**: Kolom-kolom yang relevan untuk prediksi, yaitu tahun, nama provinsi, luas hutan alam, dan laju deforestasi, dipilih. Kolom lain yang tidak memberikan informasi signifikan untuk pemodelan dihilangkan guna menyederhanakan proses.

4. **Penanganan Nilai Hilang (*Missing Values*)**: Dilakukan pemeriksaan terhadap adanya nilai yang hilang dalam dataset. Jika terdapat nilai hilang, diterapkan strategi penanganan yang sesuai, misalnya dengan imputasi menggunakan rata-rata, median, atau metode interpolasi berdasarkan karakteristik data deret waktu dan data per provinsi, untuk memastikan kelengkapan data.

5. **Transformasi Format Waktu**: Data waktu (tahun) dipastikan memiliki format yang konsisten dan sesuai untuk analisis deret waktu.

6. **Normalisasi Data (*Scaling*) untuk GRU**: Khusus untuk model GRU, dilakukan normalisasi data pada fitur luas hutan alam dan laju deforestasi. Proses ini bertujuan untuk mengubah rentang nilai data ke skala tertentu (misalnya, antara 0 dan 1) menggunakan metode seperti Min-Max Scaling. Normalisasi penting untuk meningkatkan stabilitas dan kecepatan konvergensi model *deep learning* seperti GRU.

7. **Pembuatan Sekuens (*Sequence Generation*) untuk GRU**: Untuk model GRU yang memproses data dalam bentuk sekuens, data deret waktu diubah menjadi format pasangan input-output. Input terdiri dari urutan data historis (luas hutan dan laju deforestasi) dari beberapa langkah waktu sebelumnya, dan output adalah nilai luas hutan pada langkah waktu berikutnya yang akan diprediksi.

8. **Pembagian Data Latih dan Uji (*Train-Test Split*)**: Dataset dibagi menjadi dua bagian: data latih (*training data*) dan data uji (*testing data*). Data dari tahun 2001 hingga beberapa tahun sebelum 2022 (misalnya, hingga 2019 atau 2020) digunakan sebagai data latih untuk melatih kedua model, sedangkan sisa data hingga tahun 2022 digunakan sebagai data uji untuk mengevaluasi kinerja model pada data yang belum pernah dilihat sebelumnya.

9. **Diferensiasi untuk Stasioneritas (*Differencing for Stationarity*) untuk ARIMAX**: Khusus untuk model ARIMAX, dilakukan uji stasioneritas pada data deret waktu luas hutan alam (variabel endogen) dan laju deforestasi (variabel eksogen). Jika data terbukti tidak stasioner, dilakukan proses diferensiasi (pengambilan selisih antara nilai pada waktu t dengan waktu $t-1$) hingga data menjadi stasioner. Stasioneritas merupakan asumsi penting dalam pemodelan ARIMAX.

3.2.3 Penerapan Model

Dalam penelitian ini, dua pendekatan model digunakan untuk memprediksi luas hutan alam Indonesia, yaitu model ARIMAX dan model GRU. Berikut adalah tahapan utama

dalam pembuatan masing-masing model:

Proses Pembuatan Model ARIMAX

Model ARIMAX dikembangkan dengan mengikuti langkah-langkah berikut untuk setiap provinsi secara individual sebelum hasilnya diagregasi ke tingkat nasional:

1. **Persiapan dan Uji Stasioneritas per Provinsi**:
    - Data untuk setiap provinsi diproses secara terpisah. Dilakukan pengecekan jumlah data minimum untuk memastikan kecukupan data per provinsi.
    - *Uji Stasioneritas (ADF Test)*: Data deret waktu luas hutan alam diuji stasioneritasnya menggunakan *Augmented Dickey-Fuller (ADF) Test*. Jika data tidak stasioner, dilakukan proses *differencing* (maksimal dua kali) hingga data mencapai stasioneritas.
2. **Pemodelan ARIMAX per Provinsi**:
    - *Fitting Model*: Model ARIMAX diterapkan pada data setiap provinsi dengan parameter awal $p = 1$, d (orde diferensiasi dari uji stasioneritas), dan $q = 1$. Variabel laju deforestasi ('deforestation_hectares') digunakan sebagai variabel eksogen.
    - *Optimasi Parameter*: Dilakukan pencarian *grid search* sederhana untuk menemukan kombinasi parameter (p, q) terbaik berdasarkan nilai *Akaike Information Criterion* (AIC) terendah.
3. **Peramalan (Forecasting) per Provinsi**:
    - Model ARIMAX yang telah dioptimasi digunakan untuk melakukan peramalan luas hutan alam dari tahun 2023 hingga 2030.
    - Untuk variabel eksogen (laju deforestasi) di masa depan, nilainya diasumsikan konstan menggunakan nilai observasi terakhir yang tersedia.
    - Hasil peramalan beserta interval kepercayaannya disimpan untuk setiap provinsi.
4. **Evaluasi Model ARIMAX**:
    - *Evaluasi Nasional*: Hasil prediksi dari setiap provinsi diagregasi untuk mendapatkan prediksi luas hutan alam tingkat nasional.
    - Perubahan prediksi, tren historis, dan tingkat ketidakpastian dari hasil prediksi dianalisis.
    - *Visualisasi*: Hasil prediksi nasional dan interval kepercayaannya divisualisasikan dalam bentuk plot.

Proses Pembuatan Model GRU

Model GRU dikembangkan untuk memprediksi luas hutan alam pada tingkat nasional dengan tahapan sebagai berikut:

1. **Pembagian Data Latih dan Uji**:

- Data dibagi menjadi data latih (tahun ≤ 2019) dan data uji (tahun ≥ 2020).

2. **Pembentukan Input GRU**:

   - Data sekuens (X) dibentuk dari dua tahun terakhir (mencakup fitur luas hutan dan laju deforestasi), dan target prediksi (y) adalah luas hutan pada tahun berikutnya.

3. **Pembangunan Model GRU**:

   - *Arsitektur Model*: Model GRU dibangun dengan satu lapisan GRU (32 unit), diikuti oleh satu lapisan *Dense* (16 unit), dan satu lapisan *Dense* output (1 unit) untuk prediksi.
   - *Kompilasi Model*: Model dikompilasi menggunakan optimizer 'Adam' dan fungsi kerugian (*loss function*) 'MSE' (*Mean Squared Error*).
   - *Callbacks*: Mekanisme 'EarlyStopping' dan 'ReduceLROnPlateau' diterapkan selama pelatihan untuk mencegah *overfitting* dan menyesuaikan laju pembelajaran secara dinamis.

4. **Pelatihan Model GRU**:

   - Model dilatih menggunakan data latih dengan ukuran *batch* yang kecil dan tanpa pengacakan urutan data (*shuffle=False*) karena merupakan data deret waktu.

5. **Pengujian dan Peramalan GRU**:

   - *Prediksi Data Uji*: Model digunakan untuk memprediksi luas hutan pada data uji (2020-2022). Hasil prediksi kemudian di-inverse transform untuk dikembalikan ke skala nilai aslinya.
   - *Peramalan 2023-2030*:
     - Laju deforestasi untuk periode 2023-2030 diproyeksikan terlebih dahulu menggunakan model ARIMA.
     - Hasil proyeksi laju deforestasi tersebut digunakan sebagai input fitur eksogen untuk model GRU dalam melakukan prediksi luas hutan alam secara iteratif (*rolling forecast*) hingga tahun 2030.

Evaluasi dan Visualisasi Hasil

Kinerja model GRU dan ARIMAX yang telah dilatih akan dievaluasi secara kuantitatif pada set data uji, yang merupakan data yang belum pernah dilihat oleh model selama proses pelatihan dan validasi.

1. Metrik Kuantitatif:

   - Mean Absolute Error (MAE)
   - Root Mean Squared Error (RMSE)
   - Koefisien Determinasi (R^2)

2.
   - Visualisasi Hasil: Plot perbandingan antara nilai luas hutan alam aktual dengan nilai prediksi dari model GRU dan ARIMAX akan dibuat secara kumulatif untuk serluruh provinsi dan dijadikan untuk prediksi satu Indonesia. Visualisasi ini akan membantu dalam memahami bagaimana model menangkap tren dan fluktuasi data secara grafis.
   - Visualisasi Perbandingan: Plot perbandingan evaluasi matriks antara model GRU dan ARIMAX untuk memahami kinerja model dalam mengolah data.

3.2.4 Analisis Hasil

Tahap terakhir adalah analisis dan interpretasi hasil evaluasi model.

1. Interpretasi Kinerja Model: Hasil dari metrik evaluasi akan diinterpretasikan untuk menilai seberapa baik model GRU dan ARIMAX dalam memprediksi luas hutan alam tahunan.

2. Pembahasan Keterbatasan: Keterbatasan penelitian, termasuk potensi bias dalam data atau batasan model, akan dibahas.

3. Kesimpulan dan Implikasi: Kesimpulan akan ditarik berdasarkan temuan penelitian, dan implikasi hasil prediksi untuk kebijakan pengelolaan hutan serta penelitian di masa depan akan diuraikan.

Chapter 4

HASIL DAN PEMBAHASAN

4.1 Pemilihan dan Pemrosesan Data

4.1.1 Pemilihan Data

Penelitian ini menggunakan dua set data utama yang mencakup informasi kehutanan di Indonesia pada tingkat provinsi. Dataset pertama adalah data luas hutan alam tahunan per provinsi, dan dataset kedua adalah data laju deforestasi tahunan per provinsi. Kedua dataset ini mencakup periode waktu dari tahun 2001 hingga 2022. Kedua dataset bersumber dari website trase earth [5].

Informasi awal dari dataset luas hutan alami menunjukkan adanya 748 entri data dengan 8 kolom, meliputi year, country, country_iso2, region (provinsi), region_trase_id, parent_region (wilayah induk/pulau), parent_region_trase_id, dan natural_forest. Demikian pula, dataset deforestasi memiliki struktur dan jumlah entri yang sama, dengan kolom target adalah deforestation_hectares. Kedua dataset ini mencakup 34 provinsi unik di seluruh Indonesia.

4.1.2 Analisis Data Eksploratif

Analisis data eksploratif (EDA) dilakukan untuk memahami pola, tren, dan karakteristik data sebelum dilakukan pra-pemrosesan lebih lanjut. Hasil EDA dari kedua dataset utama disajikan di bawah ini.

Statistik Deskriptif dan Nilai Hilang Awal

Sebelum penggabungan, kedua dataset, yaitu data luas hutan alami dan data deforestasi, masing-masing terdiri dari 748 entri dan 8 kolom, yang mencakup 34 provinsi unik di Indonesia dari tahun 2001 hingga 2022. Berdasarkan pemeriksaan awal menggunakan fungsi ‘.info()‘ dan ‘.isnull().sum()‘ pada kedua dataset terpisah, tidak ditemukan adanya nilai yang hilang (NaN) pada kolom-kolom fitur utama yang akan digunakan.

```
print("\n\n Analisis Data Eksploratif: Luas Hutan Alami per Provinsi ")
print("-" * 60)

print("\n Informasi Dataset Luas Hutan Alami:")
forest_df.info()

print("\n Statistik Deskriptif Luas Hutan Alami:")
display(forest_df.describe(include='all'))

print("\n Pengecekan Nilai Hilang Luas Hutan Alami:")
display(forest_df.isnull().sum())
```

Figure 4.1: Potongan Code Statistik Deskriptif

Untuk dataset luas hutan alami, statistik deskriptif (‘forest_df) menunjukkan bahwa nilai rata-rata luas hutan alam adalah sekitar 2.705.803 hektar per entri provinsi per tahun, dengan standar deviasi yang tinggi (sekitar 4.683.202 hektar), mengindikasikan variasi luas hutan yang signifikan antar provinsi dan waktu. Sementara itu, untuk dataset deforestasi (‘deforest_df), nilai rata-rata deforestasi tercatat sebesar 13.898 hektar per entri provinsi per tahun, dengan standar deviasi sekitar 28.894 hektar, yang juga menunjukkan variabilitas yang cukup besar.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 748 entries, 0 to 747
Data columns (total 8 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   year                    748 non-null    int64
 1   country                 748 non-null    object
 2   country_iso2            748 non-null    object
 3   region                  748 non-null    object
 4   region_trase_id         748 non-null    object
 5   parent_region           748 non-null    object
 6   parent_region_trase_id  748 non-null    object
 7   deforestation_hectares  748 non-null    float64
dtypes: float64(1), int64(1), object(6)
memory usage: 46.9+ KB
```

Figure 4.2: Pengecekan Nilai Null Dataset

	year	country	country_iso2	region	region_trase_id	parent_region	parent_region_trase_id	deforestation_hectares
count	748.000000	748	748	748	748	748	748	748.000000
unique	NaN	1	1	34	34	7	7	NaN
top	NaN	INDONESIA	ID	ACEH	ID-11	SUMATERA	ID-SM	NaN
freq	NaN	748	748	22	22	220	220	NaN
mean	2011.500000	NaN	NaN	NaN	NaN	NaN	NaN	13898.697182
std	6.348534	NaN	NaN	NaN	NaN	NaN	NaN	28894.803285
min	2001.000000	NaN	NaN	NaN	NaN	NaN	NaN	0.000000
25%	2006.000000	NaN	NaN	NaN	NaN	NaN	NaN	417.120441
50%	2011.500000	NaN	NaN	NaN	NaN	NaN	NaN	2950.012412
75%	2017.000000	NaN	NaN	NaN	NaN	NaN	NaN	14626.305000
max	2022.000000	NaN	NaN	NaN	NaN	NaN	NaN	300781.703647

Figure 4.3: Statistik Deskriptif Dataset

Visualisasi Tren Luas Hutan Alami

- **Tren Tahunan per Wilayah Induk**: Visualisasi tren tahunan luas hutan alami berdasarkan pengelompokan wilayah induk (pulau besar seperti Sumatera, Kalimantan, Papua, dll.) menunjukkan bahwa wilayah Papua secara konsisten memiliki total luas hutan alami terbesar, diikuti oleh Kalimantan. Sebagian besar wilayah induk lainnya menunjukkan adanya tren penurunan luas hutan alami selama periode 2001 hingga 2022.

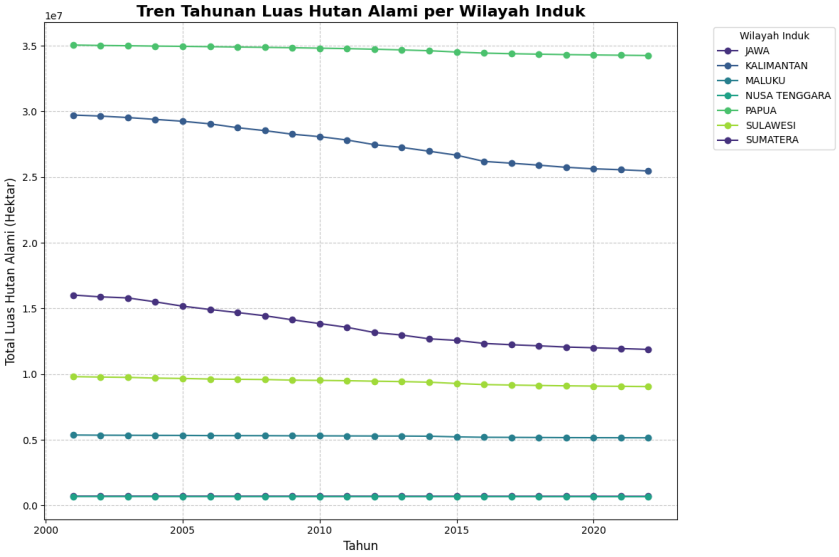

Figure 4.4: Tren Tahunan Luas Hutan Alami per Wilayah Induk

- **Distribusi pada Tahun Terakhir (2022)**: Grafik batang luas hutan alami per provinsi pada tahun 2022 (Top 15) mengonfirmasi dominasi provinsi-provinsi di Papua dan Kalimantan, seperti Papua, Papua Barat, Kalimantan Barat, dan Kalimantan Tengah, sebagai wilayah dengan tutupan hutan terluas.

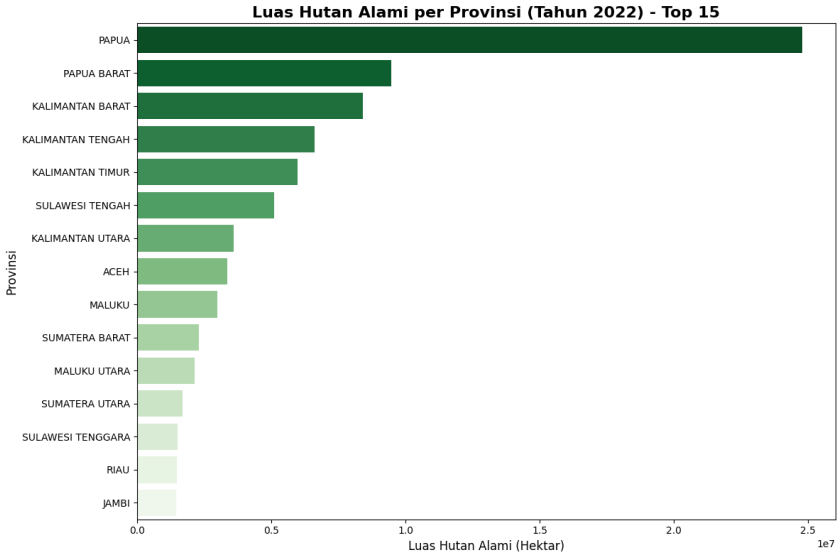

Figure 4.5: Distribusi Luas Hutan Alami pada Tahun Terakhir

- **Distribusi per Tahun (Boxplot)**: Boxplot distribusi luas hutan alami untuk setiap tahun dari 2001 hingga 2022 mengindikasikan adanya penurunan nilai median dan penyempitan sebaran (interquartile range) dari waktu ke waktu. Hal ini secara visual memperkuat adanya tren penyusutan luas hutan secara umum di tingkat provinsi.

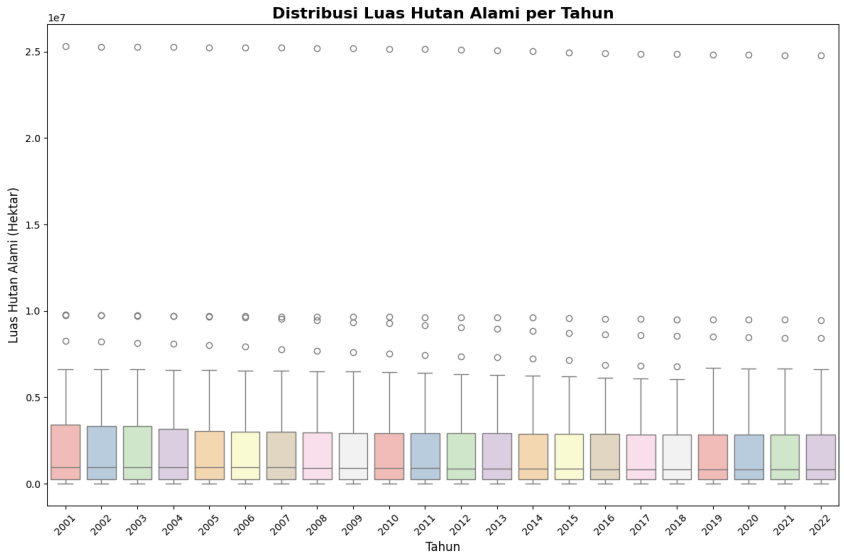

Figure 4.6: Distribusi Luas Hutan Alami per Tahun dengan Boxplot

Visualisasi Tren Deforestasi

- **Tren Tahunan per Wilayah Induk**: Plot tren tahunan total area deforestasi per wilayah induk menunjukkan adanya fluktuasi yang signifikan dari tahun ke tahun. Wilayah Kalimantan dan Sumatera tercatat mengalami beberapa periode dengan puncak deforestasi yang tinggi selama rentang waktu observasi.

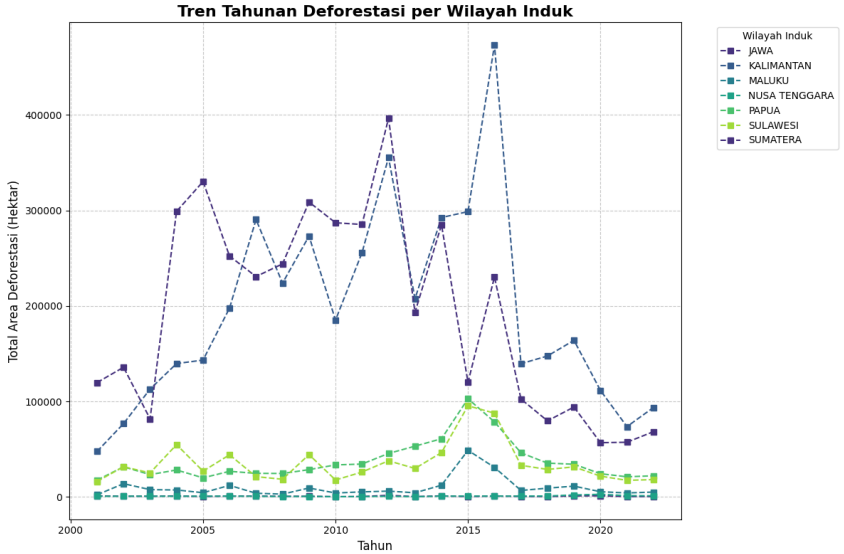

Figure 4.7: Tren Tahunan Deforestasi per Wilayah Induk

- **Distribusi pada Tahun Terakhir (2022)**: Grafik batang deforestasi per provinsi pada tahun 2022 (Top 15) menyoroti beberapa provinsi, khususnya di Kalimantan (seperti Kalimantan Barat dan Kalimantan Tengah) dan Sumatera (seperti Riau dan Sumatera Barat), sebagai kontributor deforestasi yang signifikan pada tahun tersebut.

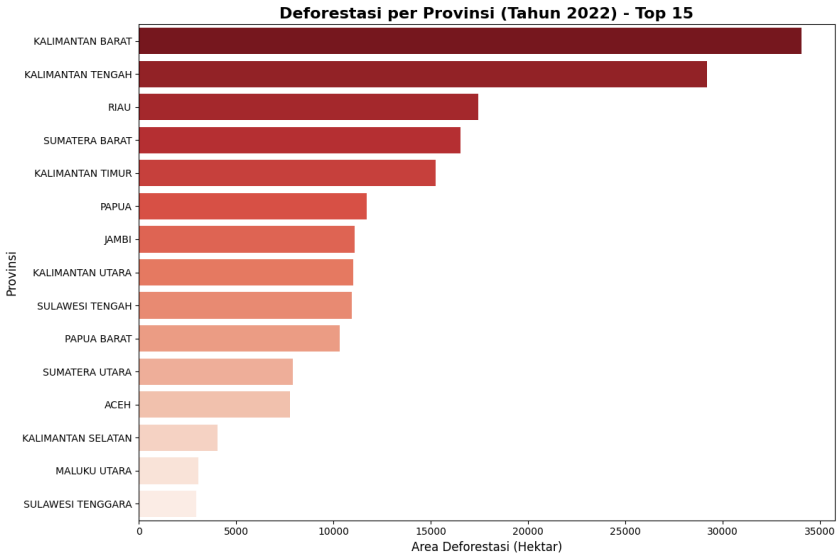

Figure 4.8: Distribusi Deforestasi pada Tahun Terakhir

- **Distribusi per Tahun (Boxplot)**: Boxplot distribusi area deforestasi per tahun menunjukkan variabilitas yang besar antar tahun, dengan beberapa tahun menunjukkan nilai ekstrem (outlier) yang tinggi. Ini mengindikasikan adanya periode-periode dengan intensitas deforestasi yang jauh di atas rata-rata.

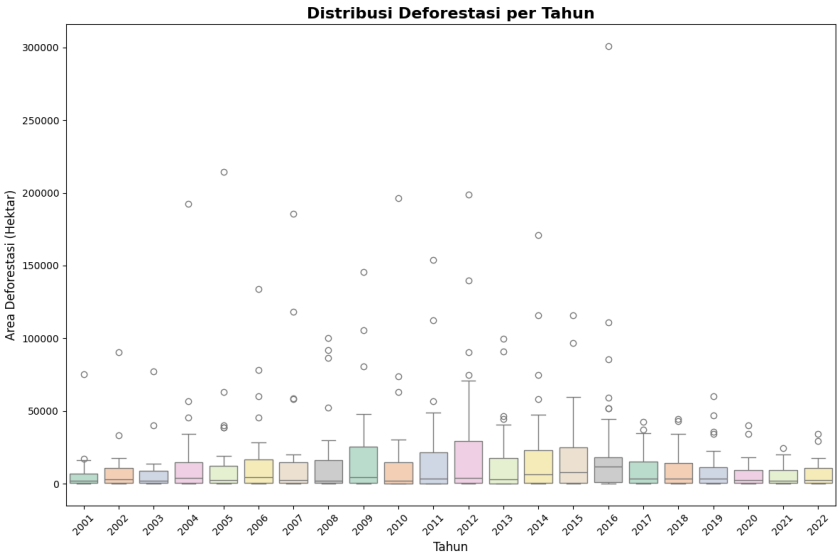

Figure 4.9: Distribusi Deforestasi per Tahun dengan Boxplot

Hasil EDA ini memberikan landasan pemahaman mengenai dinamika perubahan tutupan hutan dan laju deforestasi di Indonesia, yang selanjutnya menginformasikan langkah-langkah pra-pemrosesan dan pemilihan model.

4.1.3 Preprocessing

Pra-pemrosesan data merupakan langkah krusial untuk menyiapkan data agar sesuai dengan kebutuhan model ARIMAX dan GRU. Langkah-langkah ini mencakup pembersihan, penggabungan, transformasi, dan penyesuaian format data.

Pengolahan Data Awal (Gabungan)

Langkah awal pra-pemrosesan data dilakukan untuk menggabungkan dan membersihkan kedua dataset sumber (luas hutan alami dan laju deforestasi per provinsi) menjadi satu dataset yang siap untuk analisis lebih lanjut:

1. **Konversi Tipe Data Penggabungan Dataset**: Kolom target numerik dikonversi menjadi tipe data numerik untuk memastikan konsistensi kemudian kedua dataset tersebut kemudian digabungkan menjadi satu *DataFrame* (‘merged_df‘) berdasarkan kolom ‘year‘ dan ‘region‘ (provinsi) sebagai kunci gabungan.

2. **Penyortiran Data**: Data hasil gabungan diurutkan berdasarkan ‘region‘ dan kemudian ‘year‘ untuk memastikan urutan temporal yang benar untuk setiap provinsi.

3. **Penanganan Nilai Hilang Awal**: Setelah penggabungan, baris yang memiliki nilai hilang (NaN) pada kolom ‘natural_forest_area_hectares‘ atau ‘deforestation_hectares‘ dihapus (‘merged_df.dropna()‘) untuk memastikan integritas data input model.

```
# Data preprocessing
forest_df['natural_forest_area_hectares'] = pd.to_numeric(forest_df['natural_forest_area_hectares'], errors='coerce')
deforest_df['deforestation_hectares'] = pd.to_numeric(deforest_df['deforestation_hectares'], errors='coerce')

# Merge datasets
merged_df = pd.merge(forest_df, deforest_df, on=['year', 'region'], suffixes=('_forest', '_deforest'))
merged_df = merged_df.sort_values(['region', 'year']).reset_index(drop=True)
merged_df = merged_df.dropna(subset=['natural_forest_area_hectares', 'deforestation_hectares'])
```

Figure 4.10: Konversi Tipe Data Penggabungan Dataset

4. **Seleksi Kolom dan Deduplikasi**: Kolom-kolom yang relevan untuk analisis selanjutnya dipilih, termasuk ‘year‘, ‘region‘, ‘parent_region_forest‘, ‘natural_forest_area_hectares‘, dan ‘deforestation_hectares‘. Data duplikat yang mungkin muncul setelah penggabungan dan seleksi kolom kemudian dihilangkan (‘filtered_df.drop_duplicates()‘).

```
selected_columns = [
    'year',
    'country_forest',
    'country_iso2_forest',
    'region',
    'region_trase_id_forest',
    'parent_region_forest',
    'parent_region_trase_id_forest',
    'natural_forest_area_hectares',
    'deforestation_hectares'
]

# Filter hanya kolom yang diperlukan
filtered_df = merged_df[selected_columns].drop_duplicates()
```

Figure 4.11: Seleksi Kolom dan Deduplikasi

Pra-pemrosesan Spesifik untuk Model ARIMAX (per Provinsi)

Untuk model ARIMAX, data diproses secara individual untuk setiap provinsi:

1. **Pemuatan dan Iterasi Data**: Dataset ‘combined_spatial_data.csv‘ yang telah digabungkan dimuat. Proses kemudian dilakukan secara iteratif untuk setiap provinsi unik yang terdapat dalam dataset.

```
file_path = 'combined_spatial_data.csv'

# Memuat dataset
try:
    df = pd.read_csv(file_path)
    print(f"Dataset loaded from '{file_path}'.")
except FileNotFoundError:
    print(f"Error: File '{file_path}' tidak ditemukan. Pastikan nama file dan path sudah benar.")
    exit()

# --- Persiapan Data untuk Satu Wilayah (Contoh: ACEH) ---
nama_wilayah = 'ACEH'
df_region = df[df['region'] == nama_wilayah].copy()

if df_region.empty:
    print(f"\nError: Tidak ada data untuk wilayah '{nama_wilayah}'.")
    exit()

print(f"\nData untuk wilayah: {nama_wilayah}")
display(df_region.head(3)) # Tampilkan beberapa baris saja
```

Figure 4.12: Pemuatan dan Iterasi Data

2. **Transformasi Format Waktu**: Untuk setiap provinsi, kolom ‘year‘ dikonversi menjadi objek *datetime* dan ditetapkan sebagai indeks dari *DataFrame* provinsi tersebut. Hal ini penting untuk analisis deret waktu.

```
# Mengatur 'year' sebagai index datetime
df_region['date'] = pd.to_datetime(df_region['year'], format='%Y')
df_region.set_index('date', inplace=True)

# Memilih variabel target (y) dan variabel eksogen (X)
target_variable = 'natural_forest_area_hectares'
exog_variable = 'deforestation_hectares'
```

Figure 4.13: Transformasi Format Waktu

3. **Penanganan Nilai Hilang (Lanjutan per Provinsi)**: Meskipun telah dilakukan penghapusan NaN secara umum, dilakukan pengecekan dan penanganan nilai hilang tambahan pada data luas hutan alam dan laju deforestasi untuk setiap provinsi menggunakan metode *forward fill* (‘ffill‘) yang dilanjutkan dengan *backward fill* (‘bfill‘).

```
# Pengecekan nilai NaN
print(f"\nJumlah NaN di '{target_variable}' ({nama_wilayah}): {df_region[target_variable].isnull().sum()}")
print(f"Jumlah NaN di '{exog_variable}' ({nama_wilayah}): {df_region[exog_variable].isnull().sum()}")

y = df_region[target_variable]
X = df_region[[exog_variable]] # Variabel eksogen harus dalam bentuk DataFrame 2D

print(f"\nVariabel Target (y) - {nama_wilayah} (awal):")
display(y.head(3))
print(f"Variabel Eksogen (X) - {nama_wilayah} (awal):")
display(X.head(3))

print(f"\nPanjang deret waktu y: {len(y)}")
print(f"Panjang deret waktu X: {len(X)}")

if len(y) != len(X):
    print("Peringatan: Panjang y dan X tidak sama!")
```

Figure 4.14: Penanganan Nilai Hilang

4. **Uji Stasioneritas dan Diferensiasi**: Stasioneritas data deret waktu luas hutan alam (‘natural_forest_area_hectares‘) diuji menggunakan *Augmented Dickey-Fuller (ADF) Test*. Jika data terbukti non-stasioner (umumnya dengan p-value > 0.05), maka dilakukan proses diferensiasi (pengambilan selisih antara nilai pada waktu t dengan waktu $t - 1$) secara bertahap (maksimal dua kali) hingga data mencapai stasioneritas. Orde diferensiasi (‘d‘) yang optimal untuk setiap provinsi dicatat untuk digunakan dalam model ARIMAX.

```
# Uji Stasioneritas dengan Augmented Dickey-Fuller Test
print(f"\nHasil Uji ADF untuk '{target_variable}' di {nama_wilayah} (Original):")
adf_result = adfuller(y)
print(f'\nADF Statistic: {adf_result[0]}')
print(f'p-value: {adf_result[1]}')
print('Critical Values:')
for key, value in adf_result[4].items():
    print(f'\t{key}: {value}')

d = 0
y_differenced = y.copy()
```

Figure 4.15: Uji Stasioneritas dan Diferensiasi

Pra-pemrosesan Spesifik untuk Model GRU (Data Nasional)

Untuk model GRU, data diolah pada tingkat nasional:

1. **Agregasi ke Tingkat Nasional**: Data luas hutan alam dan laju deforestasi dari dataset ‘combined_spatial_data.csv‘ diagregasi dengan menjumlahkan nilai-nilai tersebut dari seluruh provinsi untuk setiap tahunnya. Hasilnya adalah dua deret waktu nasional: ‘natural_forest_area‘ dan ‘deforestation_rate‘.

```
# Agregasi data ke tingkat nasional per tahun
df_nasional_gru = df_full.groupby('year').agg(
    total_natural_forest_area_hectares=('natural_forest_area_hectares', 'sum'),
    total_deforestation_hectares=('deforestation_hectares', 'sum')
).reset_index()
```

Figure 4.16: Agregasi ke Tingkat Nasional

2. **Normalisasi Data (*Scaling*)**: Kedua fitur nasional tersebut (‘natural_forest_area‘ dan ‘deforestation_rate‘) kemudian dinormalisasi menggunakan ‘MinMaxScaler‘ untuk mengubah skala nilainya menjadi rentang antara 0 dan 1. Proses normalisasi ini

penting untuk meningkatkan kinerja dan stabilitas model GRU. Objek 'scaler' yang digunakan disimpan untuk dapat melakukan transformasi balik (*inverse transform*) pada hasil prediksi.

```
# Mendefinisikan fitur dan melakukan scaling
features = ['natural_forest_area', 'deforestation_rate']
data_for_scaling_gru = df_nasional_gru[features].values

scaler_gru = MinMaxScaler(feature_range=(0, 1))
scaled_data_gru = scaler_gru.fit_transform(data_for_scaling_gru)

joblib.dump(scaler_gru, 'scaler_gru.pkl')
print("\nScaler GRU telah disimpan ke 'scaler_gru.pkl'")
```

Figure 4.17: Normalisasi Data

3. **Pembuatan Sekuens (*Sequence Generation*)**: Data deret waktu nasional yang telah dinormalisasi diubah menjadi format sekuens input-output yang sesuai untuk model GRU. Dalam penelitian ini, panjang sekuens input ('n_steps_gru') ditetapkan sebanyak 2 langkah waktu (dua tahun), yang berarti dua observasi terakhir dari kedua fitur digunakan untuk memprediksi nilai luas hutan alam pada tahun berikutnya.

```
# Fungsi untuk membuat sekuens data untuk GRU
def create_gru_sequences(input_data, n_steps_in):
    X, y = [], []
    for i in range(len(input_data)):
        end_ix = i + n_steps_in
        if end_ix >= len(input_data):
            break
        seq_x = input_data[i:end_ix, :]
        seq_y = input_data[end_ix, 0] # Target fitur pertama ('natural_forest_area')
        X.append(seq_x)
        y.append(seq_y)
    return np.array(X), np.array(y)

# Menentukan jumlah langkah waktu untuk sekuens
n_steps_gru = 2

# Membuat sekuens untuk data latih
X_train_gru, y_train_gru = create_gru_sequences(scaled_train_data_gru, n_steps_gru)

# Membuat sekuens untuk data uji
# Menggabungkan bagian akhir data latih dengan data uji untuk membentuk sekuens input pertama unt
if len(scaled_train_data_gru) >= n_steps_gru:
    temp_for_X_test_gru = np.concatenate((scaled_train_data_gru[-n_steps_gru:], scaled_test_data_
    X_test_gru, _ = create_gru_sequences(temp_for_X_test_gru, n_steps_gru)
    # Menyesuaikan panjang X_test_gru agar sesuai dengan jumlah target aktual di set uji
    X_test_gru = X_test_gru[:len(actual_test_forest_area_gru)]
else:
    X_test_gru = np.array([]) # Jika data latih tidak cukup untuk membuat sekuens awal
    print("\nData latih tidak cukup untuk membuat sekuens awal bagi data uji dengan n_steps_gru y
```

Figure 4.18: Pembuatan Sekuens

4. **Pembagian Data Latih dan Uji**: Dataset nasional yang telah berbentuk sekuens kemudian dibagi menjadi data latih dan data uji. Data dari tahun 2001 hingga 2019 digunakan sebagai data latih, sementara data dari tahun 2020 hingga 2022 digunakan sebagai data uji untuk mengevaluasi model.

```
# Membagi data menjadi set latih dan uji berdasarkan tahun
train_years_condition_gru = df_nasional_gru['year'] <= 2019
test_years_condition_gru = df_nasional_gru['year'] >= 2020

scaled_train_data_gru = scaled_data_gru[train_years_condition_gru.values]
scaled_test_data_gru = scaled_data_gru[test_years_condition_gru.values]

# Menyimpan nilai aktual untuk evaluasi
actual_test_forest_area_gru = df_nasional_gru.loc[test_years_condition_gru, 'natural_forest_area']
```

Figure 4.19: Pembagian Data Latih dan Uji

4.2 Pembangunan dan Pelatihan Model

4.2.1 GRU

Model GRU (Gated Recurrent Unit) digunakan untuk memprediksi nilai luas hutan alam berdasarkan data historis dua tahun terakhir serta pengaruh dari laju deforestasi. Model ini dipilih karena memiliki keunggulan dalam mempelajari pola deret waktu jangka pendek hingga menengah, dengan arsitektur yang efisien secara komputasi dibandingkan LSTM.

1. **Menyusun Data Input Time Series**
   Sebelum memulai pembangunan model GRU, langkah pertama yang dilakukan adalah menyusun data input dalam format yang sesuai untuk pembelajaran deret waktu. Model GRU mengharuskan data input memiliki dimensi tiga, yaitu (jumlah sampel, panjang sekuens, jumlah fitur). Pada eksperimen ini, panjang sekuens ditetapkan dua tahun, dan jumlah fitur adalah dua, yaitu luas hutan alam (`natural_forest_area_hectares`) dan deforestasi (`deforestation_hectares`). Dengan demikian, setiap sampel input terdiri dari dua tahun berturut-turut data historis hutan dan deforestasi, dan target (label) dari sampel tersebut adalah nilai luas hutan pada tahun ketiga. Data ini kemudian dibagi menjadi dua bagian, yakni `X_train_gru` untuk data input, dan `y_train_gru` untuk data target. Proses ini memungkinkan GRU untuk mengenali pola perubahan nilai luas hutan berdasarkan tren jangka pendek dan tekanan deforestasi.

```
# Fungsi untuk membuat sekuens data untuk GRU
def create_gru_sequences(input_data, n_steps_in):
    X, y = [], []
    for i in range(len(input_data)):
        end_ix = i + n_steps_in
        if end_ix >= len(input_data):
            break
        seq_x = input_data[i:end_ix, :]
        seq_y = input_data[end_ix, 0] # Target fitur pertama ('natural_forest_area')
        X.append(seq_x)
        y.append(seq_y)
    return np.array(X), np.array(y)
```

Figure 4.20: Menyusun Data Input Time Series

2. **Membangun Arsitektur Model GRU**
   Setelah data disiapkan, tahap selanjutnya adalah membangun arsitektur jaringan saraf menggunakan pustaka Keras. Model dibuat dalam bentuk `Sequential` dengan tiga lapisan utama. Lapisan pertama adalah Input, yang menerima data dalam bentuk sekuens 2 tahun $\times$ 2 fitur. Lapisan kedua adalah GRU dengan 32 unit memori

dan fungsi aktivasi `tanh`. GRU dipilih karena kemampuannya mengingat informasi dari waktu ke waktu, serta memiliki mekanisme gate yang lebih sederhana dibanding LSTM namun tetap efektif. Parameter `return_sequences=False` digunakan karena kita hanya memerlukan output prediksi satu tahun ke depan, bukan urutan output. Setelah lapisan GRU, ditambahkan satu lapisan Dense dengan 16 neuron dan aktivasi `tanh` untuk memproses representasi hasil GRU secara non-linear. Terakhir, terdapat output layer berupa Dense dengan satu neuron untuk memproduksi nilai akhir prediksi regresi berupa luas hutan.

```
# Definisi model GRU
model_gru = Sequential([
    Input(shape=(X_train_gru.shape[1], X_train_gru.shape[2]), name='input_layer'),
    GRU(units=32, activation='tanh', return_sequences=False, name='gru_layer_1'),
    # Dropout(0.1, name='dropout_1'), # Dapat diaktifkan jika terjadi overfitting
    Dense(units=16, activation='tanh', name='dense_layer_1'),
    Dense(units=1, name='output_layer')
])
```

Figure 4.21: Membangun arsitektur model GRU

3. **Kompilasi Model**
   Model yang telah dibangun perlu dikompilasi sebelum digunakan untuk pelatihan. Pada tahap ini, digunakan optimizer `Adam` dengan learning rate sebesar 0.001, dan fungsi loss `mean_squared_error` (MSE). Optimizer Adam dipilih karena efisiensinya dalam mempercepat konvergensi dan kemampuannya menyesuaikan learning rate secara adaptif. Sementara itu, MSE digunakan karena model ini bertujuan melakukan prediksi nilai kontinu (regresi), dan MSE memberikan penalti besar pada prediksi yang jauh dari nilai aktual, sehingga cocok untuk mengurangi error prediksi luas hutan.

```
# Kompilasi model
optimizer_gru = tf.keras.optimizers.Adam(learning_rate=0.001)
model_gru.compile(optimizer=optimizer_gru, loss='mean_squared_error')

model_gru.summary()
```

Figure 4.22: Kompilasi model

4. **Menyiapkan Callback untuk Pelatihan**
   Agar pelatihan model lebih efisien dan menghindari overfitting, digunakan dua callback penting: `EarlyStopping` dan `ReduceLROnPlateau`. `EarlyStopping` memantau nilai loss pada data latih dan menghentikan pelatihan jika dalam 20 epoch tidak ada perbaikan signifikan. Ini membantu mencegah pembelajaran berlebih (overfitting) dan menghemat waktu komputasi. Sementara itu, `ReduceLROnPlateau` akan menurunkan learning rate secara otomatis jika selama 10 epoch loss tidak membaik, dengan tujuan memperhalus proses pembelajaran dan membantu model keluar dari local minimum. Penggunaan callback ini sangat penting dalam pelatihan jaringan saraf, terutama untuk data time series yang sensitif terhadap parameter pelatihan.

```python
# Callbacks untuk optimasi pelatihan
early_stopping_gru = EarlyStopping(
    monitor='loss',
    patience=20,
    restore_best_weights=True,
    verbose=1
)

reduce_lr_gru = ReduceLROnPlateau(
    monitor='loss',
    factor=0.2,
    patience=10,
    min_lr=0.00001,
    verbose=1
)
```

Figure 4.23: Callbacks untuk optimasi pelatihan

5. **Melatih Model GRU**
   Dengan data, model, dan callback yang telah disiapkan, proses pelatihan model dilakukan menggunakan fungsi `fit()`. Model dilatih maksimal selama 100 epoch dengan ukuran batch sebesar 4. Nilai `shuffle=False` digunakan karena dalam konteks time series, urutan data harus dipertahankan agar pola temporal dapat dipelajari dengan benar. Callback yang telah didefinisikan sebelumnya juga disertakan untuk memastikan pelatihan berlangsung optimal. Dari hasil pelatihan, tercatat bahwa loss model mengalami penurunan yang konsisten, misalnya dari 0.388 pada epoch awal menjadi sekitar 0.052 pada epoch ke-10, menandakan model belajar dengan baik dari data historis.

```python
# Melatih model GRU
print("\nMemulai pelatihan model GRU...")
history_gru = model_gru.fit(
    X_train_gru, y_train_gru,
    epochs=100,
    batch_size=4, # Batch size kecil cocok untuk dataset yang tidak terlalu besar
    verbose=1,
    callbacks=[early_stopping_gru, reduce_lr_gru],
    shuffle=False # Penting untuk data time series
)

print("\nPelatihan model GRU selesai.")
```

Figure 4.24: Melatih model GRU

6. **Visualisasi Proses Pelatihan**
   Setelah proses pelatihan selesai, dilakukan visualisasi nilai loss selama epoch untuk memantau proses pembelajaran model. Kurva loss divisualisasikan dengan Matplotlib, dan idealnya menunjukkan penurunan yang konsisten tanpa lonjakan besar.

Visualisasi ini berguna untuk mengevaluasi apakah model mengalami overfitting, underfitting, atau sudah optimal. Pada eksperimen ini, kurva loss menunjukkan penurunan yang halus dan stabil, yang mengindikasikan proses pelatihan berjalan dengan baik dan model mampu belajar dari data tanpa kehilangan generalisasi.

```
# Plot loss pelatihan
plt.figure(figsize=(10, 6))
plt.plot(history_gru.history['loss'], label='Training Loss GRU')
plt.title('Loss Model GRU Selama Pelatihan')
plt.xlabel('Epoch')
plt.ylabel('Loss (MSE)')
plt.legend()
plt.grid(True)
plt.show()
```

Figure 4.25: Plot loss pelatihan

7. **Menyimpan Model GRU**
   Sebagai langkah terakhir, model GRU yang telah selesai dilatih disimpan dalam format `.h5` menggunakan fungsi `save()` dari Keras. Penyimpanan ini penting agar model dapat digunakan kembali untuk evaluasi atau deployment tanpa harus dilatih ulang dari awal. File yang dihasilkan menyimpan seluruh informasi arsitektur, bobot model, dan konfigurasi pelatihan, sehingga dapat langsung digunakan pada tahap prediksi.

```
# Menyimpan model GRU
model_gru.save('model_gru_forest.h5')
print("Model GRU telah disimpan ke 'model_gru_forest.h5'")
```

Figure 4.26: Menyimpan model GRU

4.2.2 ARIMAX

Model ARIMAX (Autoregressive Integrated Moving Average with Exogenous Variables) digunakan sebagai pendekatan statistik klasik untuk memodelkan dan memprediksi luas hutan alam per tahun berdasarkan nilai historis dan variabel eksternal berupa deforestasi. Berbeda dengan model GRU yang berbasis neural network, ARIMAX menangkap hubungan linier antar waktu dan memperhitungkan pengaruh variabel eksogen secara eksplisit. Model ini dibangun dan dilatih per provinsi, dimulai dengan studi kasus Provinsi Aceh, dan kemudian diperluas ke seluruh Indonesia.

1. **Menyiapkan Data dan Uji Stasioneritas**
   Langkah pertama adalah menyiapkan data target dan variabel eksogen, kemudian melakukan uji stasioneritas menggunakan *Augmented Dickey-Fuller Test* (ADF Test). Data target `natural_forest_area_hectares` disusun sebagai deret waktu, dan `deforestation_hectares` sebagai input variabel eksogen. Uji ADF bertujuan memeriksa apakah data target stasioner atau tidak. Jika *p-value* ¿ 0.05, maka dilakukan proses *differencing* hingga data menjadi stasioner. Hasil uji ADF menentukan parameter d pada model ARIMAX.

```
# Uji Stasioneritas dengan Augmented Dickey-Fuller Test
print(f"\nHasil Uji ADF untuk '{target_variable}' di {nama_wilayah} (Original):")
adf_result = adfuller(y)
print(f'\nADF Statistic: {adf_result[0]}')
print(f'p-value: {adf_result[1]}')
print('Critical Values:')
```

Figure 4.27: Uji Stasioneritas

2. **Menentukan Orde dan Membentuk Model ARIMAX**
   Setelah memastikan data target stasioner, langkah berikutnya adalah membentuk model ARIMAX. Variabel target (y) dan eksogen (X) digunakan sebagai input dalam fungsi `ARIMA()` dari pustaka `statsmodels`. Model ini menerima parameter orde (p, d, q) yang ditentukan berdasarkan hasil plot ACF/PACF dan hasil uji ADF. Sebagai permulaan, digunakan nilai $p = 1$, $d = 0$ atau 1, dan $q = 1$.

```
# ARIMAX Model Fitting
from statsmodels.tsa.arima.model import ARIMA

try:
    model = ARIMA(endog=y_fit, exog=X_fit, order=(p, d, q))
    results = model.fit()
```

Figure 4.28: Membuat dan melatih model ARIMAX

3. **Menampilkan Ringkasan Model dan Koefisien**
   Setelah model dilatih, ditampilkan ringkasan hasil *fitting*, termasuk koefisien regresi, AR, MA, dan variabel eksogen. Informasi ini penting untuk mengetahui kekuatan dan arah pengaruh deforestasi terhadap perubahan luas hutan.

```
print("\n" + "="*60)
print("-"*19 + "RINGKASAN MODEL ARIMAX" + "-"*19)
print("="*60)
print(results.summary())
```

Figure 4.29: Ringkasan Model

4. **Diagnostik dan Validasi Model**
   Untuk memastikan kualitas model, dilakukan diagnostik residual. Fungsi `plot_diagnostics()` dari `statsmodels` digunakan untuk memvisualisasikan distribusi error, autokorelasi, dan normalitas residual. Visualisasi ini penting untuk memverifikasi apakah asumsi ARIMA terpenuhi.

```python
# Model Diagnostics Visualization
print("\n" + "="*60)
print("-"*19 + "PLOT DIAGNOSTIK MODEL" + "-"*19)
print("="*60)

results.plot_diagnostics(figsize=(15, 10))
plt.tight_layout()
plt.show()
```

Figure 4.30: Diagnostik model

5. **Melakukan Prediksi dan Simulasi**
   Setelah model dinyatakan valid, dilakukan prediksi luas hutan pada masa depan. Karena ARIMAX memerlukan input variabel eksogen, maka nilai deforestasi masa depan diasumsikan konstan menggunakan data tahun terakhir.

```python
# Forecast Configuration
n_forecast_steps = 2030 - 2022
print(f"Jumlah periode peramalan: {n_forecast_steps} tahun (2023-{2022 + n_forecast_steps})")

# Future Exogenous Variables Preparation
last_known_exog_value = X[exog_variable].iloc[-1]
print(f"Nilai eksogen terakhir ({exog_variable}) yang diketahui (tahun 2022): {last_known_exog_value}")

future_years = pd.date_range(start=y.index[-1] + pd.DateOffset(years=1), periods=n_forecast_steps, freq='AS-JAN')
X_future = pd.DataFrame(index=future_years, columns=[exog_variable])

# Menggunakan nilai konstan (nilai terakhir)
X_future[exog_variable] = last_known_exog_value
print(f"Contoh X_future (menggunakan nilai konstan terakhir dari {exog_variable}):")
print(X_future)
```

Figure 4.31: Melakukan prediksi

6. **Agregasi Nasional dan Evaluasi**
   Setelah prediksi dilakukan untuk setiap provinsi, hasil dikompilasi dan dijumlahkan untuk menghasilkan estimasi luas hutan nasional. Model dievaluasi dengan metrik RMSE, MAE, dan R^2 pada data historis tahun 2020–2022. Visualisasi hasil dilakukan dengan grafik prediksi vs aktual, serta garis tren ke depan hingga tahun 2030.

```python
# --- Menghitung Metrik Evaluasi Kumulatif Nasional ---
print("\n\n--- Hasil Evaluasi Kumulatif Nasional dengan Model ARIMA (pada Test Set) ---")
rmse_nasional = np.sqrt(mean_squared_error(national_actuals_aligned, national_predictions_aligned))
mae_nasional = mean_absolute_error(national_actuals_aligned, national_predictions_aligned)
r2_nasional = r2_score(national_actuals_aligned, national_predictions_aligned)

print(f"RMSE Nasional Kumulatif: {rmse_nasional:,.2f}")
print(f"MAE Nasional Kumulatif : {mae_nasional:,.2f}")
print(f"R2 Nasional Kumulatif  : {r2_nasional:.4f} (Perhatikan interpretasi R2 untuk deret waktu)")
```

Figure 4.32: Evaluasi dengan metrik RMSE

4.3 Visualisasi Hasil Model

4.3.1 Visualisasi Perbandingan Hasil Peramalan Model

Untuk memvisualisasikan dan membandingkan hasil peramalan dari kedua model, yaitu GRU *(Gated Recurrent Unit)* dan ARIMAX *(Autoregressive Integrated Moving Average with Exogenous Variables)*, serangkaian langkah dilakukan menggunakan pustaka

matplotlib.pyplot. Tujuan utama dari visualisasi ini adalah untuk menyajikan perbandingan antara data historis aktual luas hutan alami nasional dengan prediksi yang dihasilkan oleh masing-masing model hingga tahun 2030, serta menyertakan interval kepercayaan untuk model ARIMAX.

```
# Create the subplot figure
fig, axes = plt.subplots(1, 2, figsize=(18, 7), sharey=True) # sharey=True to have consistent y-ax

# --- Plot 1: GRU Prediction Comparison (until 2030) ---
axes[0].plot(historical_actual_plot.index, historical_actual_plot['Actual_Data'], label='Data Aktu
axes[0].plot(gru_combined_predictions.index, gru_combined_predictions['GRU_Prediction'], label='Pr

axes[0].axvline(x=2022.5, color='gray', linestyle=':', linewidth=1.5, label='Batas Data Historis/P

axes[0].set_title('Prediksi Luas Hutan Alami Nasional: GRU vs Aktual (hingga 2030)', fontsize=16)
axes[0].set_xlabel('Tahun', fontsize=12)
axes[0].set_ylabel('Luas Hutan Alam (Hektar)', fontsize=12)
axes[0].legend(fontsize=10)
axes[0].grid(True, which='both', linestyle='--', linewidth=0.5)
axes[0].set_xlim(historical_actual_plot.index.min(), gru_combined_predictions.index.max())


# --- Plot 2: ARIMAX Prediction Comparison (until 2030) ---
axes[1].plot(historical_actual_plot.index, historical_actual_plot['Actual_Data'], label='Data Aktu
axes[1].plot(arimax_combined_predictions.index, arimax_combined_predictions['ARIMAX_Prediction'],
axes[1].fill_between(arimax_combined_predictions.index,
                     arimax_combined_predictions['ARIMAX_Lower_CI'],
                     arimax_combined_predictions['ARIMAX_Upper_CI'],
                     color='salmon', alpha=0.3, label='ARIMAX Interval Kepercayaan 95%')
```

Figure 4.33: Potongan Code Visualisasi

4.3.2 Plot Perbandingan Evaluasi Matriks

Visualisasi perbandingan metrik evaluasi antara model GRU dan ARIMAX diimplementasikan menggunakan grafik batang (bar chart) yang dikelompokkan, dihasilkan melalui pustaka matplotlib dan seaborn.

```
try:
    # Memuat metrik GRU
    df_gru_metrics_raw = pd.read_csv('evaluation_metrics_gru.csv')
    gru_metrics = df_gru_metrics_raw.iloc[0]
    data_gru = {
        'Metrik': ['MAE', 'RMSE', 'R-squared'],
        'Model': ['GRU', 'GRU', 'GRU'],
        'Nilai': [gru_metrics['MAE'], gru_metrics['RMSE'], gru_metrics['R-squared']]
    }
    df_gru_metrics = pd.DataFrame(data_gru)

    # Memuat metrik ARIMAX
    df_arimax_metrics_raw = pd.read_csv('evaluasi_kumulatif_nasional_arima.csv')
    df_arimax_metrics = df_arimax_metrics_raw.rename(columns={'Metric': 'Metrik', 'Value_ARIMA':
    df_arimax_metrics['Model'] = 'ARIMAX'
    df_arimax_metrics['Metrik'] = df_arimax_metrics['Metrik'].replace({'R2': 'R-squared'})


    # Menggabungkan kedua DataFrame metrik
    df_metrics_combined = pd.concat([df_gru_metrics, df_arimax_metrics], ignore_index=True)

except FileNotFoundError as e:
    print(f"Error: Salah satu file CSV metrik tidak ditemukan. Pastikan file berikut ada di direk
    print("- evaluation_metrics_gru.csv")
    print("- evaluasi_kumulatif_nasional_arima.csv")
    print(f"Detail error: {e}")
    exit()
except Exception as e:
    print(f"Terjadi kesalahan saat memuat atau memproses file CSV: {e}")
    exit()
```

Figure 4.34: Potongan Code Evaluasi

4.4 Evaluasi Model

Evaluasi model dilakukan untuk membandingkan performa model ARIMAX dan GRU dalam memprediksi luas hutan alam di Indonesia. Evaluasi menggunakan tiga metrik utama: Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), dan koefisien determinasi (R^2). Data uji yang digunakan adalah periode 2020-2022.

4.4.1 Root Mean Squared Error (RMSE)

Hasil RMSE Model ARIMAX

Model ARIMAX menghasilkan nilai RMSE sebesar **116.211,63 hektar** pada periode testing 2020-2022. Nilai ini menunjukkan bahwa rata-rata kesalahan prediksi model ARIMAX adalah sekitar 116 ribu hektar ketika memperhitungkan bobot yang lebih besar untuk kesalahan yang signifikan.

Hasil RMSE Model GRU

Model GRU menunjukkan performa yang superior dengan nilai RMSE sebesar **66.698,91 hektar**. Nilai ini menunjukkan peningkatan akurasi yang signifikan sebesar 42,6% dibandingkan dengan model ARIMAX.

Table 4.1: Breakdown RMSE ARIMAX per Tahun

Tahun	Aktual (Hektar)	Prediksi (Hektar)	Error (Hektar)	Squared Error
2020	87.562.827,96	87.510.350,00	-52.477,96	2.753.936.484
2021	87.388.158,16	87.266.680,00	-121.478,16	14.756.784.356
2022	87.179.926,83	87.028.260,00	-151.666,83	23.002.644.889
RMSE				**116.211,63**

Table 4.2: Breakdown RMSE GRU per Tahun

Tahun	Aktual (Hektar)	Prediksi (Hektar)	Error (Hektar)	Squared Error
2020	87.562.827,96	87.611.376,11	48.548,15	2.356.930.304
2021	87.388.158,16	87.285.427,07	-102.731,09	10.553.471.716
2022	87.179.926,83	87.200.798,68	20.871,85	435.047.456
RMSE				**66.698,91**

Perbandingan RMSE

Perbandingan nilai RMSE menunjukkan bahwa model GRU memiliki akurasi prediksi yang lebih baik dengan selisih 49.512,72 hektar. Dalam konteks skala nasional dimana total luas hutan mencapai sekitar 87 juta hektar, perbedaan ini sangat signifikan dan menunjukkan kemampuan model GRU dalam menghasilkan prediksi yang lebih presisi.

4.4.2 Mean Absolute Error (MAE)

Hasil MAE Model ARIMAX

Model ARIMAX menghasilkan nilai MAE sebesar **108.543,44 hektar**. Nilai ini menunjukkan bahwa rata-rata kesalahan absolut prediksi model adalah sekitar 108 ribu hektar, atau setara dengan sekitar 0,124% dari total luas hutan nasional.

Table 4.3: Detail MAE ARIMAX per Tahun

Tahun	Aktual (Hektar)	Prediksi (Hektar)	Absolute Error (Hektar)
2020	87.562.827,96	87.510.350,00	52.477,96
2021	87.388.158,16	87.266.680,00	121.478,16
2022	87.179.926,83	87.028.260,00	151.666,83
MAE			**108.543,44**

Hasil MAE Model GRU

Model GRU menunjukkan performa yang lebih baik dengan nilai MAE sebesar **57.383,70 hektar**. Hal ini menunjukkan peningkatan akurasi sebesar 47,1% dibandingkan dengan model ARIMAX.

Interpretasi Perbandingan MAE

Perbedaan MAE sebesar 51.159,74 hektar antara kedua model menunjukkan bahwa model GRU konsisten menghasilkan prediksi yang lebih mendekati nilai aktual. Dalam konteks perencanaan kehutanan nasional, akurasi ini sangat penting untuk pengambilan keputusan yang tepat.

Table 4.4: Detail MAE GRU per Tahun

Tahun	Aktual (Hektar)	Prediksi (Hektar)	Absolute Error (Hektar)
2020	87.562.827,96	87.611.376,11	48.548,15
2021	87.388.158,16	87.285.427,07	102.731,09
2022	87.179.926,83	87.200.798,68	20.871,85
MAE			**57.383,70**

4.4.3 Koefisien Determinasi (R^2)

Hasil R^2 Model ARIMAX

Model ARIMAX menghasilkan nilai R^2 sebesar **0,4487**. Ini berarti model dapat menjelaskan sekitar 44,87% variabilitas dalam data luas hutan, sementara 55,13% sisanya disebabkan oleh faktor-faktor lain yang tidak tertangkap oleh model.

Table 4.5: Analisis R^2 ARIMAX

Komponen	Nilai	Persentase
Explained Variance	0,4487	44,87%
Unexplained Variance	0,5513	55,13%

Hasil R^2 Model GRU

Model GRU menunjukkan performa yang sangat baik dengan nilai R^2 sebesar **0,8184**. Ini menunjukkan bahwa model dapat menjelaskan sekitar 81,84% variabilitas data, yang merupakan peningkatan substansial sebesar 82,4% dibandingkan ARIMAX.

Table 4.6: Analisis R^2 GRU

Komponen	Nilai	Persentase
Explained Variance	0,8184	81,84%
Unexplained Variance	0,1816	18,16%

Interpretasi R^2

Perbedaan R^2 sebesar 0,3697 menunjukkan bahwa model GRU jauh lebih efektif dalam menangkap pola dan tren dalam data historis. Nilai R^2 GRU yang mendekati 0,8 menunjukkan tingkat kepercayaan yang tinggi terhadap kemampuan prediksi model.

4.4.4 Ringkasan Perbandingan Metrik Evaluasi

Table 4.7: Perbandingan Komprehensif Metrik Evaluasi

Metrik Evaluasi	ARIMAX	GRU	Selisih	Improvement (%)	Kategori Perform.
RMSE (Hektar)	116.211,63	66.698,91	49.512,72	**42,6%**	Sangat Baik
MAE (Hektar)	108.543,44	57.383,70	51.159,74	**47,1%**	Sangat Baik
R^2	0,4487	0,8184	0,3697	**82,4%**	Excellent

Keterangan Kategori Performa:

- **Sangat Baik**: Improvement 40-80%
- **Excellent**: Improvement ¿80%

4.5 Analisis Hasil

4.5.1 Visualisasi Hasil Model

Plot Peramalan Luas Hutan Alam Indonesia

Visualisasi hasil prediksi kedua model menunjukkan pola yang konsisten dalam memproyeksikan tren penurunan luas hutan alam Indonesia hingga tahun 2030.

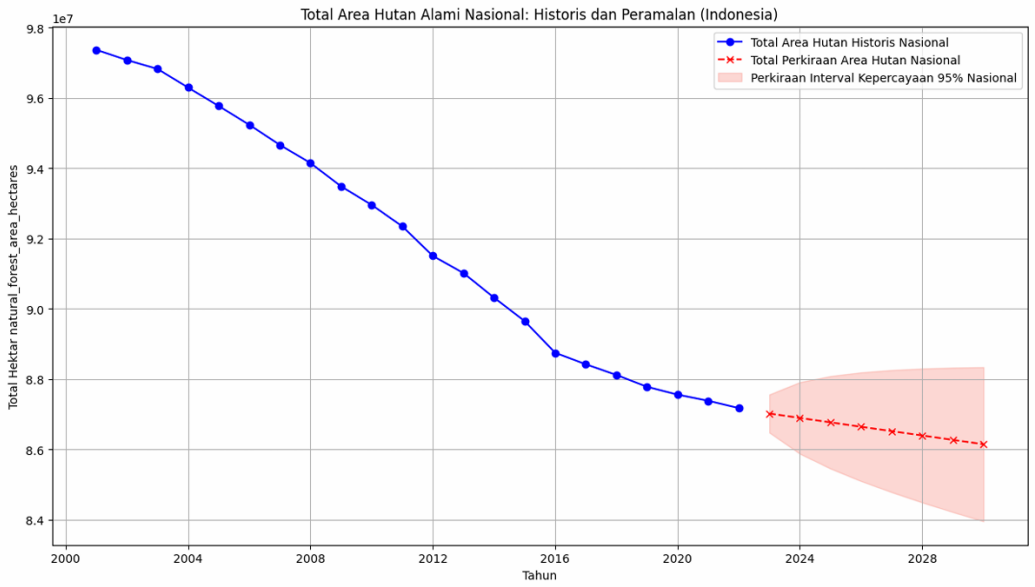

Figure 4.35: Plot Peramalan Arimax

Grafik peramalan ARIMAX menampilkan hasil peramalan luas hutan alam nasional dari tahun 2001 hingga 2030. Grafik ini terdiri dari tiga komponen utama:

1. **Garis Biru Solid (Data Historis 2001-2022)**: Menunjukkan data aktual luas hutan yang mengalami penurunan konsisten dari sekitar 97 juta hektar pada 2001 menjadi 87 juta hektar pada 2022
2. **Garis Merah Putus-putus (Prediksi ARIMAX 2023-2030)**: Proyeksi model menunjukkan kelanjutan tren penurunan dengan pola yang relatif linear
3. **Area Pink (Interval Kepercayaan 95%)**: Menunjukkan rentang ketidakpastian prediksi yang semakin melebar pada tahun-tahun ke depan

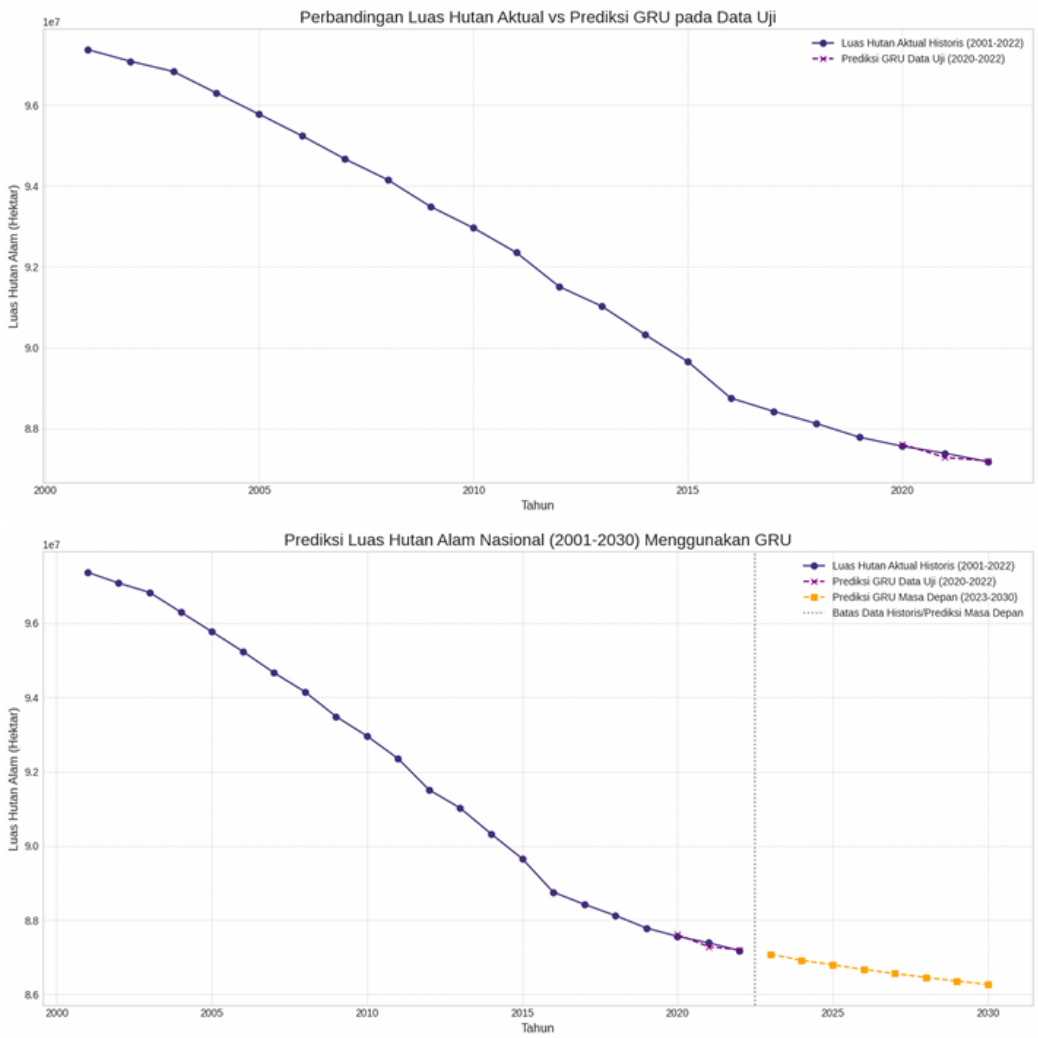

Figure 4.36: Plot Peramalan Gru

Grafik peramalan GRU menampilkan dua visualisasi utama:

- **Grafik Atas**: Perbandingan data uji (2020-2022) menunjukkan kecocokan yang sangat baik antara prediksi GRU dengan data aktual
- **Grafik Bawah**: Prediksi jangka panjang (2001-2030) dengan tren penurunan yang lebih smooth dan gradual dibandingkan ARIMAX

Plot Perbandingan Evaluasi Metrik

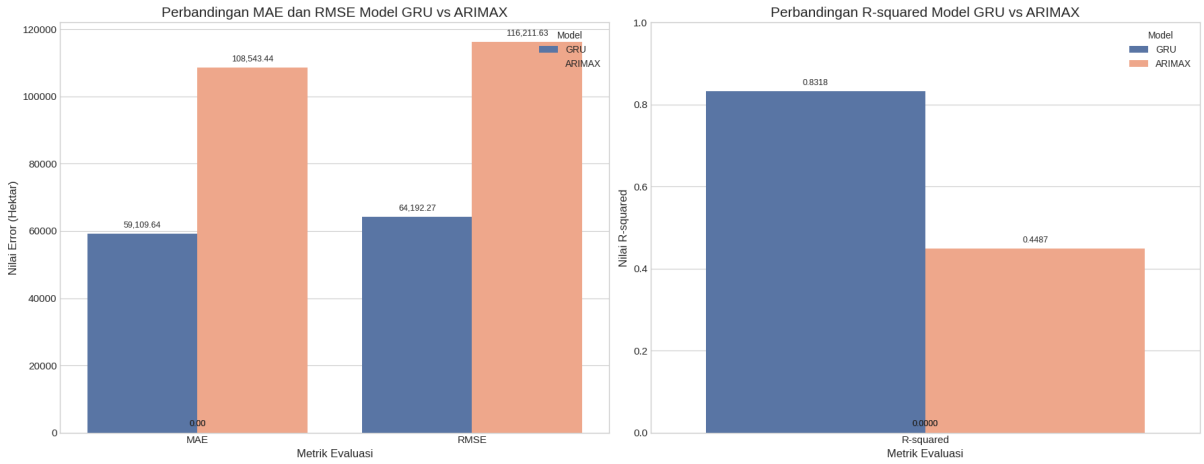

Figure 4.37: Grafik Perbandingan Metrik Evaluasi

Visualisasi perbandingan metrik menunjukkan keunggulan model GRU dalam semua aspek evaluasi dengan improvement yang signifikan di setiap metrik.

4.5.2 Interpretasi dan Implikasi Hasil

Keunggulan Model GRU

Hasil evaluasi menunjukkan bahwa model GRU memiliki keunggulan signifikan dalam beberapa aspek:

1. **Akurasi Prediksi**: Dengan RMSE 42,6% lebih rendah dan MAE 47,1% lebih baik, GRU menghasilkan prediksi yang jauh lebih akurat
2. **Kemampuan Eksplanasi**: R^2 sebesar 0,8184 menunjukkan GRU dapat menjelaskan lebih dari 80% variabilitas data
3. **Konsistensi**: Pola prediksi GRU lebih stabil dan realistis dibandingkan ARIMAX

Faktor-faktor yang Mempengaruhi Performa

Model GRU unggul karena:

- Kemampuan menangkap pola non-linear dalam data deforestasi
- Arsitektur neural network yang dapat mempelajari dependensi temporal yang kompleks
- Mekanisme gate yang efektif dalam mengelola informasi jangka panjang

Keterbatasan Model ARIMAX:

- Asumsi linearitas yang tidak sesuai dengan kompleksitas data deforestasi
- Sensitivitas terhadap outlier dan variabilitas tinggi dalam data
- Keterbatasan dalam menangkap pola temporal yang kompleks

Proyeksi Jangka Panjang

Kedua model menghasilkan proyeksi yang konsisten menunjukkan tren penurunan:

- **ARIMAX**: Penurunan 873.455 hektar (2023-2030)
- **GRU**: Penurunan 816.280 hektar (2023-2030)

Perbedaan proyeksi sekitar 57.175 hektar menunjukkan bahwa meskipun menggunakan pendekatan berbeda, kedua model mengidentifikasi tren yang serupa dengan magnitude yang relatif konsisten.

Proyeksi Total Area Hutan Nasional

Table 4.8: Proyeksi Luas Hutan Nasional 2023–2030

Tahun	ARIMAX (Hektar)	CI Bawah 95%	CI Atas 95%	GRU (Hektar)	Selisih Model
2023	87.024.287	86.482.060	87.566.520	87.083.810	59.523
2024	86.898.750	86.391.850	86.915.300	86.915.300	16.550
2025	86.824.000	86.056.800	86.798.060	86.808.260	15.740
2026	86.688.490	86.091.500	86.671.010	86.712.010	23.520
2027	86.523.750	86.226.200	86.560.490	86.560.490	36.740
2028	86.399.900	86.099.480	86.454.470	86.455.000	55.100
2029	86.272.930	86.282.950	86.357.690	86.282.950	9.980
2030	86.150.832	86.395.851	88.834.316	86.267.530	116.698

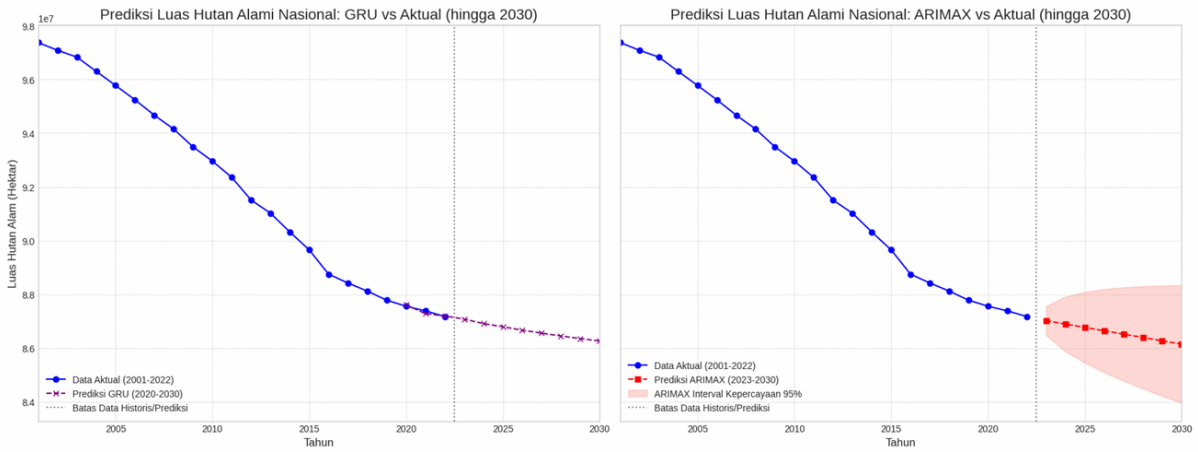

Figure 4.38: Perbandingan Prediksi GRU vs ARIMAX (2001-2030)

Visualisasi side-by-side menunjukkan:

- Model GRU menghasilkan prediksi yang lebih stabil dan konsisten
- ARIMAX memiliki interval kepercayaan yang sangat lebar (menunjukkan ketidakpastian tinggi)
- Kedua model konsisten memprediksi tren penurunan luas hutan

Table 4.9: Analisis Residual dan Distribusi Error

Statistik	ARIMAX	GRU	Interpretasi
Mean Error	-108.540,98	11.096,71	GRU lebih balanced (mendekati 0)
Std Dev Error	49.605,12	41.133,89	GRU lebih konsisten
Min Error	-151.666,83	-102.731,09	Range error GRU lebih kecil
Max Error	-52.477,96	48.548,15	
Median Error	-121.478,16	20.871,85	GRU tidak bias sistematis

Analisis Residual dan Distribusi Error

Interpretasi Kunci:

- **Mean error negatif ARIMAX**: Kecenderungan underestimate secara konsisten
- **Mean error mendekati nol GRU**: Prediksi lebih seimbang antara over dan underestimate
- **Standard deviation lebih kecil GRU**: Konsistensi error yang lebih baik

Keterbatasan dan Rekomendasi

Keterbatasan Penelitian:

- Dataset terbatas pada periode 22 tahun
- Hanya menggunakan variabel deforestasi sebagai input eksogen utama
- Tidak mempertimbangkan faktor eksternal seperti kebijakan dan perubahan iklim

Rekomendasi:

1. Implementasi model GRU untuk aplikasi praktis prediksi luas hutan
2. Pengembangan ensemble model yang menggabungkan kekuatan kedua pendekatan
3. Integrasi variabel tambahan untuk meningkatkan akurasi prediksi
4. Validasi berkelanjutan dengan data terbaru untuk memastikan relevansi model

4.5.3 Kesimpulan Evaluasi Model

Berdasarkan hasil evaluasi komprehensif menggunakan metrik RMSE, MAE, dan R^2, model GRU menunjukkan performa yang secara konsisten lebih baik dibandingkan model ARIMAX dalam memprediksi luas hutan alam Indonesia. Dengan peningkatan akurasi yang signifikan di semua metrik evaluasi, model GRU terbukti lebih suitable untuk aplikasi prediksi jangka panjang dalam konteks pengelolaan hutan nasional.

Chapter 5

KESIMPULAN

5.1 Ringkasan Hasil Percobaan

Penelitian ini bertujuan untuk membandingkan dua pendekatan model prediksi, yaitu model statistik **ARIMAX** dan model deep learning **GRU**, dalam memproyeksikan perubahan luas hutan alam di Indonesia berdasarkan data deret waktu luas hutan dan laju deforestasi tahunan pada tingkat provinsi dari tahun 2001 hingga 2022. Berdasarkan hasil eksperimen, dapat disimpulkan beberapa poin utama sebagai berikut:

1. **Model GRU (Gated Recurrent Unit) menunjukkan performa yang unggul dibandingkan ARIMAX.** GRU berhasil mengungguli ARIMAX secara signifikan pada tiga metrik evaluasi utama:

   - RMSE GRU: 66.698 hektar vs ARIMAX: 116.211 hektar.
   - MAE GRU: 57.384 hektar vs ARIMAX: 108.543 hektar.
   - R^2 GRU: 0,8184 vs ARIMAX: 0,4487.

   Hal ini menunjukkan bahwa GRU mampu menangkap pola temporal yang kompleks dan non-linear dalam data deret waktu secara lebih efektif dibandingkan ARIMAX.

2. **ARIMAX tetap memberikan kelebihan dalam hal interpretabilitas dan kesederhanaan.** Model ini dapat dengan mudah dijelaskan secara statistik dan cocok untuk data yang bersifat linear dan stasioner. Namun, dalam studi ini, ARIMAX tidak mampu mengatasi pola kompleks dari data deret waktu yang digunakan.

3. **GRU menunjukkan hasil proyeksi yang lebih halus dan realistis hingga tahun 2030.** Model GRU lebih stabil dalam memetakan tren penurunan luas hutan alam di masa depan, sementara ARIMAX menunjukkan fluktuasi yang kurang konsisten. Hal ini membuat GRU lebih cocok untuk digunakan dalam analisis kebijakan jangka panjang.

4. **Secara umum, model GRU direkomendasikan untuk digunakan dalam studi prediksi lingkungan yang kompleks,** terutama jika data memiliki komponen non-linear dan ketergantungan jangka panjang.

5.2 Saran

Berdasarkan hasil analisis dan evaluasi model, berikut beberapa saran untuk pengembangan dan penelitian selanjutnya:

- **Penambahan variabel eksogen tambahan** seperti data curah hujan, kebakaran hutan, penggunaan lahan, dan indeks kebijakan dapat meningkatkan akurasi dan kemampuan generalisasi model.
- **Peningkatan resolusi spasial ke tingkat kabupaten/kota** dapat memberikan hasil prediksi yang lebih detail dan mendukung pengambilan kebijakan yang lebih spesifik.
- **Eksplorasi model hybrid** yang menggabungkan kekuatan statistik dari ARIMAX dengan kemampuan pembelajaran dalam GRU dapat menjadi pendekatan yang menjanjikan untuk masa depan.

Daftar Pustaka

[1] I. D. Mienye, T. G. Swart, and G. Obaido, "Recurrent neural networks: A comprehensive review of architectures, variants, and applications," *Information*, vol. 15, no. 9, p. 517, 2024.

[2] A. Dutta, S. Kumar, and M. Basu, "A gated recurrent unit approach to bitcoin price prediction," *Journal of Risk and Financial Management*, vol. 13, p. 23, February 2020.

[3] H. Wahyuni and Suranto, "Dampak deforestasi hutan skala besar terhadap pemanasan global di indonesia," *JIIP: Jurnal Ilmiah Ilmu Pemerintahan*, vol. 6, pp. 148–162, Mar. 2021.

[4] V. O. d. Costa, "Multistep forecast amazon deforestation using regression and recurrent neural network approaches," Master's thesis, Programa de Pós-graduação em Matemática, Pontifícia Universidade Católica do Rio de Janeiro (PUC-Rio), September 2024. Dissertação de Mestrado.

[5] Trase, "Spatial metrics - indonesia - remaining forest," 2024. Diakses pada 24 Mei 2025.

[6] P. Guan and Y. Zheng, "Research on forest phenology prediction based on LSTM and GRU model," *Journal of Resources and Ecology*, vol. 14, pp. 25–34, January 2023.

[7] V. I. Kontopoulou, A. D. Panagopoulos, I. Kakkos, and G. K. Matsopoulos, "A review of arima vs. machine learning approaches for time series forecasting in data driven networks," *Future Internet*, vol. 15, p. 255, July 2023.

[8] S. S. A. Ansar, A. Rahmawati, and R. D. Arrahman, "Peninjauan bencana alam akibat deforestasi hutan dan tantangan penegakkan hukum mengenai kebijakan penebangan hutan berskala besar di indonesia," *Indonesian Journal of Law and Justice*, vol. 1, no. 4, pp. 11–11, 2024.

[9] K. Zhang, Q. Wen, C. Zhang, R. Cai, M. Jin, Y. Liu, J. Y. Zhang, Y. Liang, G. Pang, D. Song, *et al.*, "Self-supervised learning for time series analysis: Taxonomy, progress, and prospects," *IEEE transactions on pattern analysis and machine intelligence*, 2024.

[10] H. Wu, T. Hu, Y. Liu, H. Zhou, J. Wang, and M. Long, "Timesnet: Temporal 2d-variation modeling for general time series analysis," *arXiv preprint arXiv:2210.02186*, 2022.

[11] C. Tarmanini, N. Sarma, C. Gezegin, and O. Ozgonenel, "Short term load forecasting based on ARIMA and ANN approaches," *Energy Reports*, vol. 9, pp. 550–557, 2023. Published by Elsevier Ltd. This is an open access article under the CC BY-NC-ND license.

[12] R. Amelia, D. Y. Dalimunthe, E. Kustiawan, and I. Sulistiana, "ARIMAX model for rainfall forecasting in pangkalpinang, indonesia," in *IOP Conference Series: Earth and Environmental Science*, vol. 926, p. 012034, IOP Publishing, 2021.

[13] B. Dissanayake, O. Hemachandra, N. Lakshitha, D. Haputhanthri, and A. Wijayasiri, "A comparison of ARIMAX, VAR and LSTM on multivariate short-term traffic volume forecasting," in *PROCEEDING OF THE 28TH CONFERENCE OF FRUCT ASSOCIATION*, (Sri Lanka), p. 2, FRUCT Association, 2021.

[14] W. K. Adu, P. Appiahene, and S. Afrifa, "VAR, ARIMAX and ARIMA models for nowcasting unemployment rate in Ghana using Google trends," *Journal of Electrical Systems and Information Technology*, vol. 10, no. 1, p. 12, 2023.

[15] S. Das, A. Tariq, T. Santos, S. S. Kantareddy, and I. Banerjee, "Recurrent neural networks (rnns): architectures, training tricks, and introduction to influential research," *Machine learning for Brain disorders*, pp. 117–138, 2023.

[16] F. M. Shiri, T. Perumal, N. Mustapha, and R. Mohamed, "A comprehensive overview and comparative analysis on deep learning models," *Journal on Artificial Intelligence*, November 2024.

[17] C. Fan, M. Chen, X. Wang, J. Wang, and B. Huang, "A review on data preprocessing techniques toward efficient and reliable knowledge discovery from building operational data," *Frontiers in energy research*, vol. 9, p. 652801, 2021.

[18] H. M. Marin-Castro and E. Tello-Leal, "Event log preprocessing for process mining: a review," *Applied Sciences*, vol. 11, no. 22, p. 10556, 2021.

[19] V. Çetin and O. Yıldız, "A comprehensive review on data preprocessing techniques in data analysis," *Pamukkale Üniversitesi Mühendislik Bilimleri Dergisi*, vol. 28, no. 2, pp. 299–312, 2022.

[20] T. O. Hodson, "Root mean square error (rmse) or mean absolute error (mae): When to use them or not," *Geoscientific Model Development Discussions*, vol. 2022, pp. 1–10, 2022.

[21] D. Chicco, M. J. Warrens, and G. Jurman, "The coefficient of determination r-squared is more informative than smape, mae, mape, mse and rmse in regression analysis evaluation," *Peerj computer science*, vol. 7, p. e623, 2021.

[22] M. A. Stoffel, S. Nakagawa, and H. Schielzeth, "partr2: Partitioning r2 in generalized linear mixed models," *PeerJ*, vol. 9, p. e11414, 2021.