IMPLEMENTASI *LONG SHORT TERM MEMORY* (LSTM) UNTUK PREDIKSI KEBERANGKATAN HAJI DI KEMENTERIAN AGAMA KOTA MALANG

SKRIPSI

Oleh :
MUTIARA APRILLIA DZAKIROH
NIM. 210605110032

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM
MALANG
2025

IMPLEMENTASI *LONG SHORT TERM MEMORY* (LSTM) UNTUK PREDIKSI KEBERANGKATAN HAJI DI KEMENTERIAN AGAMA KOTA MALANG

SKRIPSI

Diajukan kepada:
Universitas Islam Negeri Maulana Malik Ibrahim Malang
Untuk memenuhi Salah Satu Persyaratan dalam
Memperoleh Gelar Sarjana Komputer (S.Kom)

Oleh :
MUTIARA APRILLIA DZAKIROH
NIM. 210605110032

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM MALANG
2025

HALAMAN PERSETUJUAN

IMPLEMENTASI *LONG SHORT TERM MEMORY* (LSTM) UNTUK PREDIKSI KEBERANGKATAN HAJI DI KEMENTERIAN AGAMA KOTA MALANG

SKRIPSI

Oleh :
MUTIARA APRILLIA DZAKIROH
NIM. 210605110032

Telah Diperiksa dan Disetujui untuk Diuji:
Tanggal: 23 Mei 2025

Pembimbing I,	Pembimbing II,
Supriyono, M.Kom	Johan Ericka Wahyu Prakasa, M.Kom
NIP. 19841010 201903 1 012	NIP. 19831213 201903 1 004

Mengetahui,
Ketua Program Studi Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang

Dr. Ir. Fachrul Kurniawan, M.MT., IPU
NIP. 19771020 200912 1 001

HALAMAN PENGESAHAN

IMPLEMENTASI *LONG SHORT TERM MEMORY* (LSTM) UNTUK PREDIKSI KEBERANGKATAN HAJI DI KEMENTERIAN AGAMA KOTA MALANG

SKRIPSI

Oleh :
MUTIARA APRILLIA DZAKIROH
NIM. 210605110032

Telah Dipertahankan di Depan Dewan Penguji Skripsi
dan Dinyatakan Diterima Sebagai Salah Satu Persyaratan
Untuk Memperoleh Gelar Sarjana Komputer (S.Kom)
Tanggal: 17 Juni 2025

Susunan Dewan Penguji

Ketua Penguji	: Fatchurrochman, M.Kom NIP. 19700731 200501 1 002	( )
Anggota Penguji I	: Nur Fitriyah Ayu Tunjung Sari, M.Cs NIP. 19911226 202012 2 001	( )
Anggota Penguji II	: Supriyono, M.Kom NIP. 19841010 201903 1 012	( )
Anggota Penguji III	: Johan Ericka Wahyu Prakasa, M.Kom NIP. 19831213 201903 1 004	( )

Mengetahui dan Mengesahkan,
Ketua Program Studi Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang

Dr. Ir. Fachrul Kurniawan, M.MT., IPU
NIP. 19771020 200912 1 001

PERNYATAAN KEASLIAN TULISAN

Saya yang bertanda tangan di bawah ini:

Nama	: Mutiara Aprillia Dzakiroh
NIM	: 210605110032
Fakultas / Jurusan	: Sains dan Teknologi / Teknik Informatika
Judul Skripsi	: Implementasi *Long Short Term Memory* (LSTM) untuk Prediksi Keberangkatan Haji di Kementerian Agama Kota Malang

Menyatakan dengan sebenarnya bahwa Skripsi yang saya tulis ini benar-benar merupakan hasil karya saya sendiri, bukan merupakan pengambil alihan data, tulisan, atau pikiran orang lain yang saya akui sebagai hasil tulisan atau pikiran saya sendiri, kecuali dengan mencantumkan sumber cuplikan pada daftar pustaka.

Apabila dikemudian hari terbukti atau dapat dibuktikan skripsi ini merupakan hasil jiplakan, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Malang, 23 Juni 2025
Yang membuat pernyataan,

Mutiara Aprillia Dzakiroh
NIM.210605110032

MOTTO

... Sebaik-baik manusia adalah yang bermanfaat bagi manusia lain ...

(HR. Ahmad, ath-Thabrani, ad-Daruqutni)

HALAMAN PERSEMBAHAN

Segala puji dan syukur hanya milik Allah Subhanahu wa Ta'ala, Tuhan semesta alam, yang dengan kasih dan rahmat-Nya, mengizinkanku melangkah sejauh ini. Dialah yang menenangkan hati di saat gundah, menguatkan jiwa saat lemah, dan membimbing langkah-langkahku yang rapuh menuju cahaya. Shalawat serta salam senantiasa tercurah kepada junjungan Nabi Muhammad Shallallahu 'alaihi wasallam, sosok mulia yang ajarannya menjadi cahaya dalam hidup ini, penuntun dalam setiap kesulitan, dan inspirasi dalam setiap perjuangan.

Kupersembahkan karya sederhana ini dalam bentuk skripsi kepada…

Kedua orang tuaku tercinta, Mama dan Papa

kalian adalah alasan terbesar aku terus berjuang. Dalam setiap tetes peluh dan doa yang kalian panjatkan, aku tumbuh. Dalam dekapan cinta yang tak pernah putus, aku belajar menjadi kuat. Meski aku anak bungsu yang sering cengeng dan merepotkan, kalian tetap mendekapku dengan sabar dan tanpa lelah. Terima kasih karena telah menjadi langit yang menaungiku dan bumi yang menjejak langkahku. Aku tak akan pernah cukup bisa membalas semua cinta, tapi aku akan terus mencoba menjadi anak yang kalian banggakan, dengan caraku yang sederhana.

Diriku sendiri

terima kasih karena tidak menyerah, meski ada banyak malam yang terasa begitu gelap. Untuk setiap air mata yang jatuh diam-diam, dan semua ketakutan yang kau peluk sendirian—aku tahu kamu lelah, tapi kamu tetap bertahan. Kamu belajar berdamai dengan diri, menerima luka, dan tetap memilih untuk tumbuh. Kamu hebat, meski kadang tidak diakui. Dan meski kamu anak bungsu yang

sering merasa kecil, kamu punya hati yang besar untuk mencintai dan memperjuangkan mimpimu. Aku bangga padamu, untuk semua versi yang kamu lewati.

Keluarga besarku yang hangat

terima kasih karena selalu hadir sebagai rumah kedua dalam bentuk yang berbeda. Dalam canda, doa, dan perhatian kecil yang mungkin terlihat sepele, kalian menguatkan lebih dari yang kalian kira. Walau kadang terpisah jarak dan waktu, cinta kalian selalu sampai. Aku merasa dicintai, didukung, dan diterima—dan itu adalah hadiah terindah yang bisa dimiliki seseorang.

Kementerian Agama Kota Malang

terima kasih atas kesempatan luar biasa yang telah diberikan. Menjadi bagian dari lingkungan yang penuh semangat pelayanan dan dedikasi terhadap masyarakat adalah pengalaman yang sangat berharga. Semoga karya kecil ini bisa menjadi bentuk kontribusi nyata dari hasil belajar, sekaligus bentuk cinta terhadap pelayanan publik yang lebih baik dan bermakna.

Dosen pembimbingku yang luar biasa

terima kasih karena telah membimbing dengan kesabaran dan ketulusan. Setiap arahan dan koreksi Bapak/Ibu adalah pijakan penting dalam proses ini. Terima kasih sudah mempercayai potensi saya, bahkan ketika saya sendiri sempat ragu. Semoga Allah membalas setiap waktu, ilmu, dan perhatian yang Bapak/Ibu berikan, dengan keberkahan yang melimpah.

Teman-teman terdekat (Salma Chesha Putri Pramudya Wardani, Sarah Arelia Rahma, Fitria Susanti, Nurjihan Nabilah Ramadhani, Hajratul Aswad, Frisca Yuni Adilia Putri, Siti Rofidatus Saidah, Wafiq Aulia Zahra, Khumairoh Dewi Maqriza, Annisa Aulia Dzikrillah, Melinda Nur Rahmawati, Sri Rahayu Ningtyas, 张澳（Gao Gao), Zhang, Livy, 周维辕, Zhang Zhensheng, Bekhruz, Keysha dan teman sekamar tercinta (Aisya Gusti Savila)

kalian adalah aman di tengah badai. Untuk teman sekamarku yang paling tahu kapan aku sedang rapuh meski aku pura-pura kuat—terima kasih untuk peluk yang selalu tepat waktu, untuk sabar yang tak pernah habis, dan untuk pelan-pelan menuntunku keluar dari rasa gelapku sendiri. Untuk teman-teman terdekatku yang selalu siaga, meski aku sering jadi beban dengan segala tangis, curhat, dan drama—kalian tetap ada. Kalian tidak menuntut aku kuat, tapi kalian membuatku percaya bahwa aku bisa. Kalian menyelamatkanku, tanpa harus menjadi pahlawan. Terima kasih sudah tinggal, ketika kalian punya pilihan untuk pergi.

Terima kasih untuk semua cinta, waktu, doa dan kesabaran yang menyertai langkah ini. Dan untuk kalian—para pejuang skripsi di luar sana percaya lah, kamu tidak sendiri. Kita mungkin berjalan pelan, tapi kita tetap menuju akhir yang baik. Satu halaman setiap hari pun tetap berarti. Semangat ya, kita bisa sampai.

KATA PENGANTAR

Puji syukur penulis panjatkan ke hadirat Allah Subhanahu wa Ta'ala atas segala limpahan rahmat, hidayah, dan kemudahan-Nya, sehingga skripsi ini dapat diselesaikan dengan baik dan tepat waktu. Skripsi ini disusun sebagai salah satu syarat untuk menyelesaikan program Sarjana Strata-1 (S1) di Program Studi Teknik Informatika, Fakultas Sains dan Teknologi, Universitas Islam Negeri Maulana Malik Ibrahim Malang.

Selama proses penyusunan skripsi ini, penulis memperoleh banyak bantuan, dukungan, dan arahan dari berbagai pihak. Oleh karena itu, dengan penuh rasa hormat dan terima kasih yang sebesar-besarnya, penulis ingin menyampaikan apresiasi kepada:

1. Prof. Dr. H. M. Zainuddin, M.A., selaku Rektor UIN Maulana Malik Ibrahim Malang, atas segala kebijakan dan dukungan terhadap keberlangsungan pendidikan di kampus ini.
2. Prof. Dr. Hj. Sri Harini, M.Si., selaku Dekan Fakultas Sains dan Teknologi, yang telah memberikan ruang dan fasilitas terbaik bagi mahasiswa untuk tumbuh dan belajar.
3. Dr. Ir. Fachrul Kurniawan, M.MT., IPU., selaku Ketua Program Studi Teknik Informatika, atas motivasi dan arahannya selama masa perkuliahan hingga penyusunan skripsi ini.
4. Supriyono, M.Kom. selaku Dosen Pembimbing I dan Johan Ericka Wahyu Prakasa, M.Kom. selaku Dosen Pembimbing II, yang telah dengan sabar

memberikan bimbingan, masukan, dan arahan selama proses penelitian dan penulisan skripsi ini.

5. Fatchurrochman, M.Kom. selaku Dosen Penguji I dan Nur Fitriyah Ayu Tunjung Sari, M.Cs. selaku Dosen Penguji II, atas pertanyaan dan masukan yang membangun sehingga skripsi ini dapat disempurnakan.
6. Achmad Shampton, S.HI., M.Ag., selaku Kepala Kementerian Agama Kota Malang, serta Nurul Istiqomah, S.Pd.I., M.Pd., selaku Kasubbag TU dan Dr. Subhan, M.Si., selaku Kasi Penyelenggaraan Haji dan Umrah (PHU), atas dukungan dan kepercayaan yang diberikan selama pelaksanaan penelitian.
7. Seluruh staf Penyelenggaraan Haji dan Umrah (PHU) dan Pelayanan Terpadu Satu Pintu (PTSP) Kementerian Agama Kota Malang, yang telah banyak membantu proses pengumpulan data dan kelancaran penelitian ini, khususnya, Elis Mufidah (staf PHU), Nur Hambali (staf PHU), Ari Yulianto (staf PHU), Adibah (staf PHU), Diana (staf PTSP), Sholichah (staf PTSP), terima kasih atas kerja sama dan bantuan teknis maupun administratif yang sangat membantu dalam proses penelitian ini.
8. Kedua orangtuaku tercinta, Aisah Maghfiroh dan M. Yusron Choirul Latif, atas doa, cinta, dan dukungan yang tak pernah terputus.
9. Kakakku satu-satunya, M. Syahril Yessa Febrian, atas semangat dan motivasi yang terus diberikan dalam setiap langkahku.
10. Keluarga besarku tercinta, Esti Khotimah, Toat Suprihadi, Yeni Kusuma Wardani, Widiah Galuh Candra Prawesti, Moh. Zafran Attoatlillah, dan Sadikyah, atas dukungan moril dan doa yang selalu menyertai.

11. Seluruh dosen dan staf Fakultas Sains dan Teknologi, khususnya Jurusan Teknik Informatika, atas ilmu, dedikasi, dan pengajaran yang sangat berarti selama masa studi. Ucapan terima kasih secara khusus penulis sampaikan kepada Nia Faricha, selaku staf administrasi program sarjana Jurusan Teknik Informatika, yang telah banyak membantu dalam proses administrasi akademik selama perkuliahan hingga penyusunan skripsi ini.
12. Teman-teman sebimbingan, teman-teman se-PKL dan teman-teman terdekatku, yang turut membantu dalam proses pengerjaan skripsi ini—mulai dari diskusi, bimbingan bersama, hingga formatting akhir. Terima kasih telah menjadi tempat bertukar pikiran, bahu untuk bersandar, dan semangat untuk terus melangkah.
13. Seluruh teman angkatan 2021 dan para Asisten Laboratorium (Aslab) angkatan 2020, atas kerja sama, dan bersedia berbagi informasi, dan berdiskusi sepanjang masa perkuliahan.

Penulis menyadari bahwa skripsi ini masih jauh dari sempurna. Oleh karena itu, penulis sangat terbuka terhadap segala bentuk kritik dan saran yang membangun demi perbaikan di masa mendatang. Semoga skripsi ini dapat memberikan manfaat dan menjadi salah satu bentuk kontribusi kecil dalam dunia pendidikan dan pelayanan publik.

Malang, 23 Juni 2025

Penulis.

DAFTAR ISI

HALAMAN PENGAJUAN ..... **ii**
HALAMAN PERSETUJUAN ..... **iii**
HALAMAN PENGESAHAN ..... **iv**
PERNYATAAN KEASLIAN TULISAN ..... **v**
MOTTO ..... **vi**
HALAMAN PERSEMBAHAN ..... **vii**
KATA PENGANTAR ..... **x**
DAFTAR ISI ..... **xiii**
DAFTAR GAMBAR ..... **xv**
DAFTAR TABEL ..... **xvi**
ABSTRAK ..... **xvii**
ABSTRACT ..... **xviii**
مستخلص البحث ..... **xix**
BAB I PENDAHULUAN **1**
1.1 Latar Belakang 1
1.2 Rumusan Masalah 3
1.3 Batasan Masalah 3
1.4 Tujuan Penelitian 4
1.5 Manfaat Penelitian 4
1.6 Metode Penelitian 5
BAB II STUDI PUSTAKA **5**
2.1 Penelitian Terkait 5
2.2 Pengelolaan Ibadah Haji 7
2.3 Alur Prediksi Keberangkatan Haji Reguler Secara Konvensional 9
2.4 *Long Short Term Memory* (LSTM) 10
2.5 Implementasi LSTM Dalam Prediksi Keberangkatan Haji 11
2.6 Implementasi LSTM Dalam Prediksi Keberangkatan Haji 13
BAB III DESAIN DAN IMPLEMENTASI **14**
3.1 Desain Sistem 14
3.2 Analisis Data 17
3.3 Implementasi Model dan Evaluasi 19
3.3.1 Pengumpulan dan Praproses Data 19
3.3.2 Pelatihan Model LSTM 23
3.3.3 Evaluasi Kinerja Model LSTM dan Sistem SISKOHAT 23
3.4 Cara Kerja LSTM dalam Memprediksi Keberangkatan 24
3.4.1 Normalisasi Data 25
3.4.2 Menghapus Entri Kosong (*Null Values*) 25
3.4.3 Proses *Windowing* 26
3.4.4 Proses Perhitungan LSTM 27
3.4.5 Proses Prediksi dengan LSTM 38
3.4.6 Mengembalikan Nilai Prediksi ke Bentuk Asli (*Denormalisasi*) 39
3.4.7 Menginterpresentasikan Hasil Prediksi 40
3.4.8 Perhitungan Konvensional SISKOHAT 41

3.4.9 Menghitung MAE dan RMSE 45
BAB IV HASIL DAN PEMBAHASAN..... 47
4.1 *Preprocessing* Data 47
4.1.1 Pembersihan Data 47
4.1.2 Pemilihan Fitur 49
4.1.3 Normalisasi Data..... 51
4.2 Pembagian Data untuk LSTM..... 54
4.3 Hasil *Windowing* 55
4.4 Membangun dan Melatih Model LSTM 58
4.4.1 Arsitektur Model LSTM 58
4.4.2 Konfigurasi *Hyperparameter*..... 64
4.4.3 Proses Pelatihan Model..... 65
4.5 Pengujian Model LSTM..... 66
4.5.1 Skenario Pengujian 1 68
4.5.2 Skenario Pengujian 2 71
4.5.3 Skenario Pengujian 3 75
4.5.4 Skenario Pengujian 4 79
4.5.5 Skenario Pengujian 5 83
4.5.6 Skenario Pengujian 6 88
4.5.7 Analisis dan Pembahasan Hasil Pengujian 92
4.6 Hasil Prediksi dan Evaluasi Prediksi..... 102
4.6.1 Evaluasi Hasil Prediksi LSTM dengan SISKOHAT (Konvensional)..... 105
4.7 Integrasi Islam 111
BAB V KESIMPULAN DAN SARAN 115
5.1 Kesimpulan..... 115
5.2 Saran..... 116
DAFTAR PUSTAKA
LAMPIRAN

DAFTAR GAMBAR

Gambar 3.1 Flowchart Penelitian dengan LSTM 14
Gambar 3.2 Arsitektur LSTM 16
Gambar 3.3 Flowchart SISKOHAT menentukan Estimasi Keberangkatan Haji 42
Gambar 4.1 Grafik Evaluasi Training Loss dan MAE Skenario 1 69
Gambar 4.2 Grafik Evaluasi Model Pengujian Skenario 1 70
Gambar 4.3 Grafik Distribusi Error Skenario 1 71
Gambar 4.4 Grafik Evaluasi Training Loss dan MAE Skenario 2 73
Gambar 4.5 Grafik Evaluasi Model Pengujian Skenario 2 74
Gambar 4.6 Grafik Distribusi Error Skenario 2 75
Gambar 4.7 Grafik Evaluasi Training Loss dan MAE Skenario 3 77
Gambar 4.8 Grafik Evaluasi Model Pengujian Skenario 3 78
Gambar 4.9 Grafik Distribusi Error Skenario 3 79
Gambar 4.10 Grafik Evaluasi Training Loss dan MAE Skenario 4 81
Gambar 4.11 Grafik Evaluasi Model Pengujian Skenario 4 82
Gambar 4.12 Grafik Distribusi Error Skenario 4 83
Gambar 4.13 Grafik Evaluasi Training Loss dan MAE Skenario 5 85
Gambar 4.14 Grafik Evaluasi Model Pengujian Skenario 5 87
Gambar 4.15 Grafik Distribusi Error Skenario 5 88
Gambar 4.16 Grafik Evaluasi Training Loss dan MAE Skenario 6 90
Gambar 4.17 Grafik Evaluasi Model Pengujian Skenario 6 91
Gambar 4.18 Grafik Distribusi Error Skenario 6 92
Gambar 4.19 Grafik Evaluasi MAE dan RMSE Data Testing LSTM Skenario 1..... 96
Gambar 4.20 Grafik Evaluasi MAE dan RMSE Data Testing LSTM Skenario 2..... 97
Gambar 4.21 Grafik Evaluasi MAE dan RMSE Data Testing LSTM Skenario 3..... 98
Gambar 4.22 Grafik Evaluasi MAE dan RMSE Data Testing LSTM Skenario 4..... 99
Gambar 4.23 Grafik Evaluasi MAE dan RMSE Data Testing LSTM Skenario 5..... 100
Gambar 4.24 Grafik Evaluasi MAE dan RMSE Data Testing LSTM Skenario 6..... 101
Gambar 4.25 Grafik Akurasi model LSTM 108
Gambar 4.26 Grafik Akurasi Model Konvensional 109
Gambar 4.27 Grafik Hasil Evaluasi MAE dan RMSE LSTM 109
Gambar 4.28 Grafik Akurasi model LSTM 110

DAFTAR TABEL

Tabel 2.1 Penelitian Terkait 5
Tabel 3.1 Sampel Data 20
Tabel 3.2 Sample Data Bobot dan Bias 28
Tabel 4.1 Hasil dari missing value 48
Tabel 4.2 Hasil pengecekan format data 49
Tabel 4.3 Pemilihan Fitur X 50
Tabel 4.4 Nilai minimum dan maksimum setiap fitur sebelum Normalisasi 51
Tabel 4.5 Nilai minimum dan maksimum setiap fitur setelah Normalisasi 52
Tabel 4.6 Hasil Normalisasi 53
Tabel 4.7 Hasil Pembagian Dataset 54
Tabel 4.8 Hasil Windowing – Data Training 56
Tabel 4.9 Hasil Windowing – Data Testing 56
Tabel 4.10 Windowing pada Fitur Usia Daftar (X_train) 56
Tabel 4.11 Model Skuensial LSTM 62
Tabel 4.12 Sample Hasil Prediksi LSTM Skenario 1 70
Tabel 4.13 Sample Hasil Prediksi LSTM Skenario 2 74
Tabel 4.14 Sample Hasil Prediksi LSTM Skenario 3 77
Tabel 4.15 Sample Hasil Prediksi LSTM Skenario 4 81
Tabel 4.16 Sample Hasil Prediksi LSTM Skenario 5 86
Tabel 4.17 Sample Hasil Prediksi LSTM Skenario 6 90
Tabel 4.18 Hasil Evaluasi Pelatihan Skenario 93
Tabel 4.19 Hasil Evaluasi Pengujian Skenario LSTM 94
Tabel 4.20 Hasil Prediksi LSTM dari Data Uji 103
Tabel 4.21 Sample Baris Pertama Data Hasil Prediksi (Normalisasi) 104
Tabel 4.22 Lima Sample Hasil Prediksi LSTM Tepat 106
Tabel 4.23 Lima Sample Hasil Prediksi LSTM Tidak Tepat 107

ABSTRAK

Dzakiroh, Mutiara Aprillia. 2025. **Implementasi *Long Short Term Memory* (LSTM) untuk Prediksi Keberangkatan Haji di Kementerian Agama Kota Malang.** Skripsi. Program Studi Teknik Informatika Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang. Pembimbing: (I) Supriyono, M.Kom (II) Johan Ericka Wahyu Prakasa, M.Kom.

Kata kunci: Evaluasi Model, Keberangkatan Haji, LSTM, MAE, Prediksi, RMSE, SISKOHAT.

Penelitian ini bertujuan untuk mengevaluasi kinerja model *Long Short-Term Memory* (LSTM) dalam memprediksi keberangkatan haji dan membandingkannya dengan metode konvensional yang saat ini diterapkan. Penelitian ini menggunakan data historis dari Sistem Informasi dan Komputerisasi Haji Terpadu (SISKOHAT) dan melakukan enam skenario pengujian dengan variasi parameter seperti jumlah *epoch* dan ukuran *batch* untuk mengukur efektivitas dan efisiensi model LSTM. Metode yang digunakan adalah eksperimen dengan membagi data menjadi data pelatihan (80%) dan data pengujian (20%) untuk melatih model dan mengukur hasilnya menggunakan metrik MAE dan RMSE. Hasil penelitian menunjukkan bahwa Skenario 5, dengan konfigurasi 200 *epoch* dan *batch size* 16, memberikan kinerja terbaik dengan akurasi 79,5% dan 157 prediksi yang sesuai target, serta nilai *error* yang terkecil, yaitu 0,0609 dan 0,1404, menunjukkan model mempelajari pola temporal dan demografis dengan baik. Dibandingkan dengan metode konvensional yang hanya memiliki akurasi 24% dan 76% kesalahan, model LSTM terbukti lebih efektif dan efisien dalam memprediksi keberangkatan haji. Penelitian ini memberikan wawasan mengenai optimasi model LSTM untuk prediksi yang lebih akurat dan stabil, serta menekankan pentingnya pengaturan parameter yang tepat untuk memperoleh hasil terbaik.

ABSTRACT

Dzakiroh, Mutiara Aprillia. 2025. **Implementation of Long Short-Term Memory (LSTM) for Hajj Departure Prediction at the Ministry of Religious Affairs of Malang City.** Thesis. Department of Informatics Engineering, Faculty of Science and Technology, Maulana Malik Ibrahim State Islamic University Malang. Supervisors: (I) Supriyono, M.Kom (II) Johan Ericka Wahyu Prakasa, M.Kom.

Key words: Model Evaluation, Hajj Departure, LSTM, MAE, Prediction, RMSE, SISKOHAT.

This study aims to evaluate the performance of the Long Short-Term Memory (LSTM) model in predicting Hajj departure dates and compare it with the conventional method currently used. The study utilizes historical data from the Integrated Hajj Information and Computerization System (SISKOHAT) and conducts six testing scenarios with variations in parameters such as the number of epochs and batch size to assess the effectiveness and efficiency of the LSTM model. The research method involves an experiment by splitting the data into 80% training and 20% testing, and measuring the results using MAE and RMSE metrics. The results show that Scenario 5, with a configuration of 200 epochs and a batch size of 16, provides the best performance with an accuracy of 79.5% and 157 correct predictions, along with small error values of 0.0609 and 0.1404, indicating that the model effectively learns temporal and demographic patterns. Compared to the conventional method, which only achieves 24% accuracy and 76% error, the LSTM model proves to be more effective and efficient in predicting Hajj departure dates. This study provides insights into optimizing the LSTM model for more accurate and stable predictions, emphasizing the importance of proper parameter tuning to achieve the best results.

مستخلص البحث

ذَاكِرَة، مُتَيَارَة أَبْرِيلِّيَة. ٢٠٢٥. تنفيذ الذاكرة طويلة المدى (LSTM) لتنبؤ مواعيد مغادرة الحجاج في وزارة الشؤون الدينية بَمدينة مالانغ. أطروحة. قسم هندسة المعلوماتية. كلية العلوم والتكنولوجيا . جامعة مولانا مالك إبراهيم الإسلامية الحكومية مالانج. المشرف: (أ) سُبْرِيونُو، الماجستير (ب) جوهان إِرِيكَا وَاهْيُو بْرَكَا سَة، الماجستير.

.

الكلمات المفتاحية: تقييم النموذج، مغادرة الحجاج، LSTM، MAE، التنبؤ، RMSE، SISKOHAT.

تهدف هذه الدراسة إلى تقييم أداء نموذج الذاكرة قصيرة وطويلة المدى (LSTM) في التنبؤ بتواريخ مغادرة الحجاج ومقارنته مع الطريقة التقليدية المستخدمة حالياً. تستخدم الدراسة بيانات تاريخية من نظام المعلومات والحج الحوسبة المتكاملة (SISKOHAT) وتجري ستة سيناريوهات اختبار مع اختلافات في المعلمات مثل عدد الدورات وحجم الدفعة لتقييم فعالية وكفاءة نموذج .LSTM تتضمن طريقة البحث تجربة من خلال تقسيم البيانات إلى ٪80 للتدريب و ٪20 للاختبار، وقياس النتائج باستخدام مقاييس MAE و RMSE. تظهر النتائج أن السيناريو 5، مع تكوين 200 دورة تدريبية و حجم دفعة 16، يوفر أفضل أداء مع دقة تصل إلى ٪79.5 و 157 تنبؤًا دقيقًا، مع قيم خطأ صغيرة 0.0609 و 0.1404، مما يشير إلى أن النموذج يتعلم الأنماط الزمنية والديموغرافية بشكل فعال. مقارنة بالطريقة التقليدية التي تحقق دقة ٪24 و ٪76 خطأ، يثبت نموذج LSTM أنه أكثر فعالية وكفاءة في التنبؤ بتواريخ مغادرة الحجاج. توفر هذه الدراسة رؤى حول تحسين نموذج LSTM للحصول على تنبؤات أكثر دقة واستقرارا، مع التأكيد على أهمية ضبط المعلمات بشكل صحيح لتحقيق أفضل النتائج.

BAB I

PENDAHULUAN

1.1 Latar Belakang

Haji merupakan salah satu rukun Islam yang wajib dilaksanakan oleh setiap Muslim yang mampu, baik secara fisik, mental, maupun finansial (Al-Zahrani, 2022). Sebagai ibadah yang sangat sakral, haji memiliki dimensi spiritual yang mendalam, di mana umat Muslim berusaha mendekatkan diri kepada Allah SWT melalui serangkaian ritual yang ditentukan. Allah SWT berfirman dalam Al-Qur'an:

﴿ وَاَتِمُّوا الْحَجَّ وَالْعُمْرَةَ لِلّٰهِ ۗ فَاِنْ اُحْصِرْتُمْ فَمَا اسْتَيْسَرَ مِنَ الْهَدْيِ ۚ وَلَا تَحْلِقُوْا رُءُوْسَكُمْ حَتّٰى يَبْلُغَ الْهَدْيُ
مَحِلَّهٗ ۗ فَمَنْ كَانَ مِنْكُمْ مَّرِيْضًا اَوْ بِهٖٓ اَذًى مِّنْ رَّأْسِهٖ فَفِدْيَةٌ مِّنْ صِيَامٍ اَوْ صَدَقَةٍ اَوْ نُسُكٍ ۚ فَاِذَآ اَمِنْتُمْ ۗ فَمَنْ تَمَتَّعَ
بِالْعُمْرَةِ اِلَى الْحَجِّ فَمَا اسْتَيْسَرَ مِنَ الْهَدْيِ ۚ فَمَنْ لَّمْ يَجِدْ فَصِيَامُ ثَلٰثَةِ اَيَّامٍ فِى الْحَجِّ وَسَبْعَةٍ اِذَا رَجَعْتُمْ ۗ تِلْكَ
عَشَرَةٌ كَامِلَةٌ ۗ ذٰلِكَ لِمَنْ لَّمْ يَكُنْ اَهْلُهٗ حَاضِرِى الْمَسْجِدِ الْحَرَامِ ۗ وَاتَّقُوا اللّٰهَ وَاعْلَمُوْٓا اَنَّ اللّٰهَ شَدِيْدُ الْعِقَابِ ﴿١٩٦﴾

"Sempurnakanlah ibadah haji dan umrah karena Allah. Akan tetapi, jika kamu terkepung (oleh musuh), (sembelihlah) hadyu yang mudah didapat dan jangan mencukur (rambut) kepalamu sebelum hadyu sampai di tempat penyembelihannya. Jika ada di antara kamu yang sakit atau ada gangguan di kepala (lalu dia bercukur), dia wajib berfidyah, yaitu berpuasa, bersedekah, atau berkurban. Apabila kamu dalam keadaan aman, siapa yang mengerjakan umrah sebelum haji (tamatu'), dia (wajib menyembelih) hadyu yang mudah didapat. Akan tetapi, jika tidak mendapatkannya, dia (wajib) berpuasa tiga hari dalam (masa) haji dan tujuh (hari) setelah kamu kembali. Itulah sepuluh hari yang sempurna. Ketentuan itu berlaku bagi orang yang keluarganya tidak menetap di sekitar Masjidilharam. Bertakwalah kepada Allah dan ketahuilah bahwa Allah Maha Keras hukuman-Nya. Hadyu adalah hewan ternak yang disembelih di tanah haram Makkah pada Iduladha dan hari-hari tasyrik karena menjalankan haji tamattu' atau qiran, meninggalkan salah satu manasik haji atau umrah, mengerjakan salah satu larangan manasik, atau murni ingin mendekatkan diri kepada Allah Swt. sebagai ibadah sunah. Fidyah (tebusan) karena tidak dapat menyempurnakan manasik haji dengan alasan tertentu." (Q.S. Al Baqarah : 196).

Dalam tafsirnya, Ibnu Kathir menyebut QS. *Al-Baqarah* ayat 196 sebagai perintah untuk menyempurnakan haji dan umrah dengan niat ikhlas karena Allah. Jika terdapat halangan seperti sakit atau musuh, maka disyariatkan bertahallul dan menunaikan fidyah berupa puasa, sedekah, atau menyembelih hewan sebagai bentuk ketaatan kepada syariat (Jasmi, 2023).

Sebagai negara dengan populasi Muslim terbesar di dunia, Indonesia menghadapi tantangan besar dalam mengelola keberangkatan jutaan jamaah haji setiap tahunnya (Farhan, 2020). Kementerian Agama (Kemenag), melalui Sistem Informasi dan Komputerisasi Haji Terpadu (SISKOHAT), bertanggung jawab untuk mengatur pendaftaran, penjadwalan, dan administrasi keberangkatan haji (Fahmi, 2022). Namun, dengan jumlah jamaah yang begitu besar, penjadwalan keberangkatan haji menjadi proses yang kompleks dan sering kali menemui berbagai kendala yang memerlukan solusi yang lebih canggih dan akurat (Nandavita & Islahuddin, 2022).

Dalam beberapa tahun terakhir, teknologi kecerdasan buatan (AI) telah berkembang pesat dan diterapkan di berbagai bidang untuk meningkatkan efisiensi dan akurasi prediksi, salah satunya adalah LSTM (Elsheikh et al., 2021). LSTM, yang merupakan bagian dari jaringan saraf dalam (*deep learning*), dirancang khusus untuk memproses dan memprediksi data berurutan, seperti data seri waktu (Jin et al., 2020). Model ini memiliki kemampuan untuk mengingat informasi jangka panjang, sehingga sangat efektif digunakan dalam konteks prediksi berdasarkan data historis (C. Y. Lin & Lobo Marques, 2024). Penerapan LSTM dalam memprediksi keberangkatan haji berdasarkan data dari tahun-tahun sebelumnya

dan faktor lainnya menawarkan potensi besar untuk mengatasi masalah yang dihadapi saat ini.

Penelitian ini bertujuan untuk mengevaluasi kinerja model LSTM dalam memprediksi keberangkatan haji dan membandingkan nilai *error* prediksinya dengan metode konvensional. Upaya tersebut diharapkan tidak hanya memberikan manfaat teknis, tetapi juga mencerminkan nilai-nilai kemanusiaan dalam memberikan pelayanan terbaik kepada jamaah haji dan mendukung petugas operasional secara lebih efektif.

1.2 Rumusan Masalah

Berdasarkan latar belakang penelitian, terdapat rumusan masalah yang diangkat dalam penelitian ini adalah bagaimana kinerja model LSTM dibandingkan dengan metode konvensional yang saat ini digunakan untuk memprediksi keberangkatan haji?

1.3 Batasan Masalah

Berdasarkan rumusan masalah, penelitian ini dibatasi pada beberapa aspek berikut:

1. Penelitian ini terbatas pada Kemenag Kota Malang dan tidak mencakup wilayah lain di Indonesia. Hasil penelitian mungkin tidak sepenuhnya dapat digeneralisasi untuk konteks nasional atau internasional.
2. Penelitian ini menggunakan data pendaftaran haji periode daftar tahun 2008-2019.

1.4 Tujuan Penelitian

Berikut adalah tujuan penelitian yaitu mengevaluasi kinerja model LSTM untuk memprediksi keberangkatan haji, dan membandingkannya dengan metode konvensional yang saat ini diterapkan, untuk menentukan efektivitas dan efisiensi dari masing-masing metode.

1.5 Manfaat Penelitian

Manfaat dari penelitian ini meliputi beberapa aspek yang penting, diantaranya sebagai berikut:

1. Manfaat Praktis: Penelitian ini memberikan panduan praktis untuk Kemenag Kota Malang dalam mengevaluasi kinerja metode LSTM dibandingkan dengan metode konvensional yang saat ini digunakan untuk memprediksi keberangkatan haji. Diharapkan hasil penelitian ini dapat membantu dalam menentukan metode yang paling akurat dan efisien.
2. Manfaat Akademis: Penelitian ini memberikan kontribusi bagi pengembangan ilmu pengetahuan di bidang kecerdasan buatan dan prediksi seri waktu, terutama dalam penerapan model LSTM untuk kasus prediksi berbasis data historis. Hasil penelitian ini juga diharapkan dapat memperkaya literatur dan membuka peluang pengembangan model prediksi lainnya dalam konteks yang serupa.

1.6 Metode Penelitian

Pada metode penelitian yang digunakan dalam studi ini mencakup beberapa tahapan utama sebagai berikut:

1. Mengumpulkan data historis keberangkatan haji dari SISKOHAT dan literatur terkait untuk memahami penerapan model LSTM dalam prediksi.
2. Merancang dan melatih model LSTM menggunakan data historis yang telah dikumpulkan.

BAB II

STUDI PUSTAKA

2.1 Penelitian Terkait

Untuk mengetahui tentang efektivitas Sistem Informasi dan Komputerisasi Haji Terpadu (SISKOHAT) dan berbagai aspek manajemen haji, sejumlah penelitian terdahulu telah memberikan kontribusi signifikan dalam memahami permasalahan serta menawarkan solusi terkait. Beberapa penelitian yang relevan mengenai penerapan teknologi dalam pengelolaan haji, sistem pendaftaran, prediksi jadwal keberangkatan, hingga kepuasan jamaah haji disajikan dalam Tabel 2.1. Penelitian-penelitian ini memberikan dasar yang kuat untuk memahami kelemahan dan keunggulan dari berbagai pendekatan yang telah diterapkan.

Tabel 2.1 Penelitian Terkait

No	Peneliti	Metodologi	Hasil Utama	Kebaruan	Kelemahan
1.	(Wardana & Suhartini, 2023)	Kualitatif, fenomenologi; studi pengelolaan dana haji.	SISKOHAT Gen 3 meningkatkan transparansi dan akuntabilitas dengan empat dimensi akuntabilitas.	Analisis mendalam akuntabilitas dana haji melalui SISKOHAT Gen 3.	Sistem belum optimal menjangkau daerah terpencil.
2.	(Fredricka & Ihsan, 2020)	*Weighted Product* (WP) untuk prioritas jadwal keberangkatan haji.	WP memprioritaskan keberangkatan berdasarkan skor tertinggi (contoh: 5.11).	Otomatisasi prioritas keberangkatan haji dengan WP.	Masih ada manualisasi; kurang fleksibel terhadap perubahan kriteria.
3.	(Salihoglu, 2023)	Model LSTM, *sliding window* untuk prediksi destinasi penerbangan.	LSTM prediksi destinasi dengan akurasi tinggi, hasil konsisten pada dataset beragam.	Pendekatan *sliding window* untuk prediksi destinasi, lintas sektor.	Bergantung pada kualitas data; hasil terpengaruh jika data buruk.
4.	(J. Li et al., 2024)	*Hybrid* model: NAM, HD, dan LSTM, rasio data	Prediksi banjir akurat, RMSE rendah dengan rasio data 2:1.	Integrasi model fisik dan LSTM dengan data campuran.	Kualitas data simulasi kritis; komputasi

No	Peneliti	Metodologi	Hasil Utama	Kebaruan	Kelemahan
		simulasi 2:1, indikator performa seperti MAE, RMSE, dan PRE.			tinggi untuk *real-time* prediksi.
5.	(Hafiz & Nurdianty, 2024)	Kualitatif deskriptif, analisis SISKOHAT untuk haji khusus.	SISKOHAT efektif memfasilitasi pendaftaran online jamaah haji khusus.	Penekanan integrasi SISKOHAT untuk pendaftaran haji khusus.	Ketergantungan pada internet, gangguan koneksi hambat proses.
6.	(Datta & Faroughi, 2023)	*Multihead* LSTM untuk prediksi kelembapan tanah.	Prediksi akurasi tinggi hingga sebulan, R^2: 95.04%, RMSE: 0.2007.	*Multihead* LSTM menangani berbagai skala waktu sekaligus.	Perlu data besar; akurasi menurun pada horizon panjang.
7.	(Prameswari et al., 2021)	Kualitatif deskriptif, evaluasi SISKOHAT Generasi 2	SISKOHAT Gen 2 sistem "*first come, first served*".	Evaluasi lokal integrasi SISKOHAT di Malang	Ketergantungan jaringan, *server* bermasalah; kurang pelatihan SDM.
8.	(H. Liu et al., 2020)	LSTM-ANN	LSTM-ANN tingkatkan akurasi prediksi waktu kedatangan di berbagai kondisi jalan.	Gabungan LSTM dan ANN menangani ketergantungan jarak jauh.	Bergantung pada data historis dan *real-time*; pelatihan komputasi tinggi.
9.	(Kiramy et al., 2024)	RNN dan LSTM, CRISP-DM, data 8 tahun	LSTM lebih akurat dari RNN, RMSE: 0.1758, MAPE: 0.4846.	Aplikasi LSTM/RNN untuk prediksi jumlah jamaah Umrah	Data terbatas akibat pandemi; LSTM membutuhkan komputasi lebih tinggi.
10.	(Ge et al., 2024)	LSTM-SVR untuk prediksi keberangkatan bus	LSTM-SVR lebih akurat dibanding model tunggal pada dataset kecil.	Kombinasi LSTM (*temporal*) dan SVR (*non-linear*)	Parameter kompleks; pemisahan dataset kritis untuk pelatihan dan pengujian seimbang.

Dari Tabel 2.1, terlihat berbagai metode, seperti penggunaan algoritma, teknologi *deep learning*, serta optimasi sistem, dapat membantu mengatasi kendala dalam manajemen haji. Namun, beberapa penelitian juga menyoroti kelemahan yang memerlukan perbaikan lebih lanjut. Bab berikutnya membahas lebih dalam mengenai makna dan hikmah ibadah haji, sebagai fondasi spiritual dalam pelaksanaan ibadah yang terorganisir secara teknis melalui sistem yang terus berkembang.

2.2 Pengelolaan Ibadah Haji

Pengelolaan ibadah haji di Indonesia semakin kompleks seiring dengan meningkatnya jumlah pendaftar setiap tahunnya (Khan & Shambour, 2023). Keterbatasan kuota haji yang diberikan oleh pemerintah Arab Saudi menyebabkan sistem daftar tunggu menjadi sangat panjang, sering kali mencapai puluhan tahun. Kondisi ini menimbulkan kekhawatiran, khususnya bagi jamaah lansia yang memiliki risiko kesehatan lebih tinggi selama masa tunggu (Huda & Haeba, 2021). Situasi ini menuntut Kemenag untuk mengembangkan solusi yang efektif guna mengatasi tantangan ini secara menyeluruh.

Sebagai respon terhadap kebutuhan tersebut, Kemenag Republik Indonesia mengembangkan SISKOHAT yang pertama kali diluncurkan dalam bentuk Generasi pertama (Gen-1) pada tahun 2010 (Anwar, 2020). Pada tahun 2014, SISKOHAT diperbarui menjadi Generasi kedua (Gen-2) dengan peningkatan signifikan dalam fitur dan kapabilitas sistem. Salah satu perbedaan utama antara Gen-1 dan Gen-2 adalah kemampuan integrasi yang lebih baik antara berbagai pihak yang terlibat dalam penyelenggaraan haji, seperti Kemenag, bank penerima

setoran haji, serta kantor-kantor kementerian agama di seluruh provinsi dan kabupaten di Indonesia.

Selain itu, SISKOHAT Gen-2 juga meningkatkan kemampuan dalam pemrosesan dokumen, pengawasan operasional di Arab Saudi, dan pemantauan kepulangan jamaah ke Tanah Air (Suzami et al., 2021). Sistem ini mempermudah akses informasi bagi calon jamaah haji dan menjamin keamanan data mereka. Namun, meskipun telah mengalami banyak peningkatan, SISKOHAT masih memiliki kelemahan, terutama dalam hal estimasi daftar jamaah haji reguler yang berangkat pada tahun-tahun tertentu. Ketidakmampuan ini disebabkan oleh keterbatasan sistem dalam memproses data sekuensial dan menyesuaikan dengan variasi data yang dinamis, terutama terkait perubahan kuota dan kebijakan pemerintah.

Keterbatasan ini juga tercermin dalam beberapa metode analisis lain yang digunakan untuk memprioritaskan keberangkatan jamaah haji. Salah satu contohnya adalah metode *Weighted Product* yang digunakan untuk menganalisis sistem keberangkatan calon jamaah haji (Fredricka & Ihsan, 2020). Meskipun metode ini dapat membantu dalam menentukan prioritas berdasarkan kriteria seperti usia dan lama waktu pendaftaran, hasil yang diperoleh sering kali kurang optimal. Hal ini disebabkan oleh ketidakmampuan metode tersebut untuk menangani variasi dan dinamika data secara efektif, terutama ketika terjadi perubahan mendadak dalam jumlah pendaftar atau penyesuaian kuota. Kekurangan lain dari metode ini adalah kurangnya fleksibilitas dalam menyesuaikan prediksi

secara *real-time*, yang sering kali menyebabkan ketidakpuasan di kalangan jamaah karena ketidakadilan dalam penentuan prioritas keberangkatan (Shan et al., 2024).

2.3 Alur Prediksi Keberangkatan Haji Reguler Secara Konvensional

Proses prediksi keberangkatan haji reguler di Kementerian Agama (Kemenag) Kota Malang dimulai dengan proses input data calon jamaah melalui SISKOHAT. Data utama seperti nama, tanggal daftar, nomor porsi, usia, dan informasi lain dimasukkan ke dalam sistem untuk diolah lebih lanjut. Setelah data diverifikasi, SISKOHAT menghitung masa tunggu berdasarkan tanggal pendaftaran dan kuota tahunan yang diterima. Kuota tahunan, yang ditetapkan melalui kesepakatan antara Pemerintah Arab Saudi dan Kemenag Indonesia, menjadi variabel utama dalam proses ini. Sebagai contoh, pada tahun 2025, Indonesia menerima kuota sebanyak 221.000 jamaah, yang terdiri dari 203.320 untuk haji reguler dan sisanya untuk haji khusus. Kuota ini kemudian didistribusikan secara proporsional ke setiap provinsi berdasarkan jumlah pendaftar dan pertimbangan lainnya, seperti prioritas jamaah lansia.

Selanjutnya, sistem secara otomatis mengurutkan calon jamaah berdasarkan nomor porsi, sesuai prinsip *first come*, *first served* (Mohd et al., 2024). Pada tahap ini, kebijakan prioritas, seperti untuk jamaah lansia dan kebutuhan khusus, diterapkan sesuai dengan Pasal 25 PMA No. 13 Tahun 2021 (Astuti et al., 2021). Calon jamaah yang memenuhi syarat prioritas akan mendapatkan posisi lebih awal dalam daftar keberangkatan. Setelah itu, hasil akhir berupa prediksi tahun keberangkatan disampaikan kepada calon jamaah untuk memberikan gambaran estimasi waktu keberangkatan.

Namun, sistem ini memiliki keterbatasan dalam hal fleksibilitas. SISKOHAT cenderung mengandalkan data statis dari pendaftaran awal dan kuota yang ditetapkan, sehingga kurang responsif terhadap perubahan mendadak seperti tambahan kuota atau kebijakan baru. Sebagai contoh, pada tahun 2024, Indonesia menerima tambahan kuota sebanyak 20.000 jamaah, yang membutuhkan alokasi ulang dalam waktu singkat. Proses penyesuaian ini sering kali memerlukan intervensi manual, yang dapat menyebabkan keterlambatan atau kesalahan pada proses prediksi.

Dengan demikian, meskipun SISKOHAT memberikan layanan administratif yang terstruktur, keterbatasannya dalam menyesuaikan perubahan dinamika kuota dan kebijakan menunjukkan kebutuhan akan sistem yang lebih fleksibel dan responsif untuk mendukung pengelolaan haji yang lebih efektif dan adaptif.

2.4 *Long Short Term Memory* (LSTM)

LSTM adalah sebuah jenis jaringan saraf tiruan yang dirancang khusus untuk bekerja dengan data yang berurutan, seperti data yang menunjukkan perubahan dari waktu ke waktu (Andhika et al., 2021). LSTM dikembangkan dari *Recurrent Neural Network* (RNN), yang merupakan jenis jaringan saraf tiruan yang juga bekerja dengan data berurutan. Namun, LSTM memiliki keunggulan yang signifikan dibandingkan dengan RNN konvensional karena kemampuannya untuk mengingat informasi penting dalam jangka waktu yang panjang dan mengatasi beberapa kelemahan yang ada pada RNN (Yang, 2022).

Dalam jaringan saraf biasa, ketika kita bekerja dengan data berurutan, informasi dari data sebelumnya bisa hilang atau menjadi kurang relevan seiring

dengan berjalannya waktu (Ehteram et al., 2024). Ini adalah salah satu kelemahan utama dari RNN, di mana model sering kali kesulitan untuk mengingat informasi penting dari langkah-langkah sebelumnya, terutama ketika data memiliki urutan yang panjang. LSTM dirancang untuk mengatasi masalah ini dengan menggunakan struktur khusus yang memungkinkan model untuk memilih informasi mana yang harus disimpan dalam memori untuk jangka waktu yang lebih lama, dan informasi mana yang bisa dilupakan.

Di dalam LSTM, terdapat tiga komponen utama yang dikenal sebagai gerbang, yang berfungsi untuk mengatur aliran informasi di setiap langkah waktu *forget gate* (gerbang lupa), *input gate* (gerbang masukan), dan *output gate* (gerbang keluaran) (Ran et al., 2019). *Forget gate* bertugas untuk memutuskan informasi mana yang sudah tidak relevan dan harus dihapus dari memori. Ini berarti LSTM dapat secara otomatis melupakan informasi yang tidak lagi dibutuhkan, menjaga agar memori tidak terlalu penuh dengan data yang tidak relevan. *Input gate* berfungsi untuk menentukan informasi baru apa yang perlu disimpan dalam memori (Sokooti et al., 2021). Dengan kata lain, gerbang ini memutuskan informasi baru yang penting dan menambahkannya ke dalam memori sel LSTM. *Output gate* bertugas untuk menentukan informasi mana yang akan digunakan sebagai *output*, serta mempengaruhi status memori yang akan diteruskan ke langkah berikutnya (Alshuwaier et al., 2022).

2.5 Implementasi LSTM Dalam Prediksi Keberangkatan Haji

Implementasi model LSTM dalam prediksi jadwal keberangkatan haji di Kemenag Kota Malang melibatkan beberapa tahapan yang krusial, dimulai dari

pengumpulan dan praproses data hingga integrasi model ke dalam SISKOHAT (Umpu Singa et al., 2022). Data historis yang mencakup informasi terkait jadwal keberangkatan haji, kuota yang diberikan setiap tahun, serta faktor-faktor lain yang mempengaruhi jadwal keberangkatan, digunakan sebagai *input* untuk melatih model LSTM (Kaddoura & Nassar, 2024). Proses mempersiapkan data melibatkan pembersihan data dari *noise* dan ketidakkonsistenan serta normalisasi data untuk memastikan bahwa data siap digunakan dalam model pembelajaran mesin (Gkikas & Tsiknakis, 2023).

Setelah data diproses, tahap berikutnya adalah merancang dan melatih model LSTM. Arsitektur LSTM yang digunakan dalam penelitian ini terdiri dari beberapa lapisan LSTM dengan jumlah unit yang bervariasi, diikuti oleh lapisan *fully connected* yang menghasilkan *output* prediksi jadwal keberangkatan. Model dilatih menggunakan data historis yang telah dibagi menjadi data latih dan data uji untuk memastikan model tidak hanya mampu memprediksi data yang pernah dilihat, tetapi juga dapat menggeneralisasi pola dari data baru (Iskandaryan et al., 2022).

Proses pelatihan dilakukan dengan menggunakan algoritma optimasi seperti *Adam optimizer* dan dua fungsi *loss*, yaitu MAE (*Mean Absolute Error*) dan RMSE (*Root Mean Squared Error*) (Chen et al., 2023). Penggunaan MAE berfungsi untuk mengukur rata-rata dari perbedaan *absolut* antara prediksi dan nilai aktual, memberikan gambaran seberapa besar kesalahan prediksi tanpa mengkuadratkan nilai *error*, sehingga lebih tahan terhadap *outlier*. Sementara itu, RMSE digunakan untuk mengukur kesalahan prediksi dengan mempertimbangkan kuadrat dari *error*, yang lebih sensitif terhadap kesalahan besar. Kombinasi penggunaan MAE dan

RMSE memungkinkan model untuk meminimalkan kesalahan prediksi secara efektif, baik untuk kesalahan kecil maupun besar (Freecenta et al., 2022).

2.6 Implementasi LSTM Dalam Prediksi Keberangkatan Haji

Mean Absolute Error (MAE) adalah metrik evaluasi yang banyak digunakan untuk menilai performa model prediksi karena sifatnya yang mudah dipahami dan memberikan informasi langsung tentang rata-rata kesalahan dalam unit asli data. Salah satu keunggulan MAE adalah tidak terlalu sensitif terhadap *outlier*, sehingga cocok digunakan pada dataset yang memiliki *noise* atau nilai ekstrem. Dalam penelitian, MAE sering digunakan untuk menilai performa model pada prediksi berbasis *time-series* karena memberikan hasil evaluasi yang stabil dan representatif, bahkan dalam kondisi data yang bervariasi (Shi et al., 2021).

Root Mean Square Error (RMSE) memberikan bobot yang lebih besar pada kesalahan besar karena sifat kuadratnya. Ini menjadikannya pilihan yang lebih baik ketika tujuan evaluasi adalah untuk menyoroti dan meminimalkan kesalahan yang signifikan. Dalam beberapa penelitian, RMSE dianggap lebih sesuai untuk mengevaluasi model yang beroperasi pada data dengan risiko tinggi jika terjadi kesalahan, karena sensitivitasnya terhadap nilai ekstrem memungkinkan deteksi anomali yang lebih baik (Kumar Dubey et al., 2021).

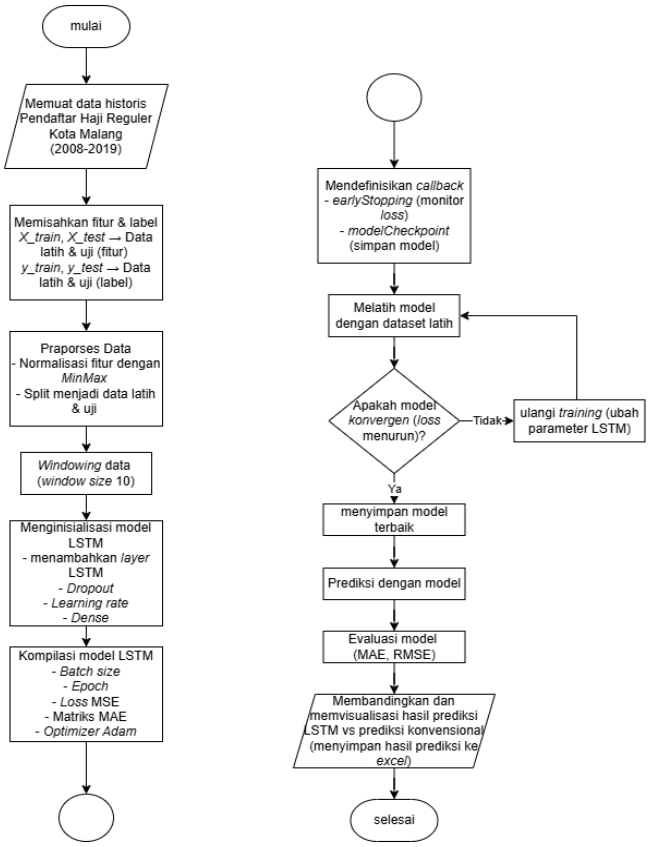
Kombinasi penggunaan MAE dan RMSE dapat memberikan wawasan yang lebih luas tentang performa model, mencakup kesalahan rata-rata keseluruhan MAE dan sensitivitas terhadap kesalahan besar RMSE. Pendekatan ini sering digunakan dalam berbagai penelitian untuk memastikan evaluasi yang menyeluruh terhadap keandalan dan akurasi model prediksi (Wu et al., 2021).

BAB III

DESAIN DAN IMPLEMENTASI

3.1 Desain Sistem

Desain sistem dalam penelitian ini bertujuan untuk mengembangkan dan mengevaluasi model LSTM (*Long Short Term Memory*) dalam memprediksi jadwal keberangkatan haji. Desain ini berfokus pada penerapan model LSTM dan membandingkan kinerjanya dengan metode konvensional yang digunakan oleh Sistem Komputerisasi Haji Terpadu (SISKOHAT). Penelitian ini tidak melibatkan integrasi langsung model ke dalam SISKOHAT, tetapi lebih kepada membandingkan performa kedua sistem secara terpisah.

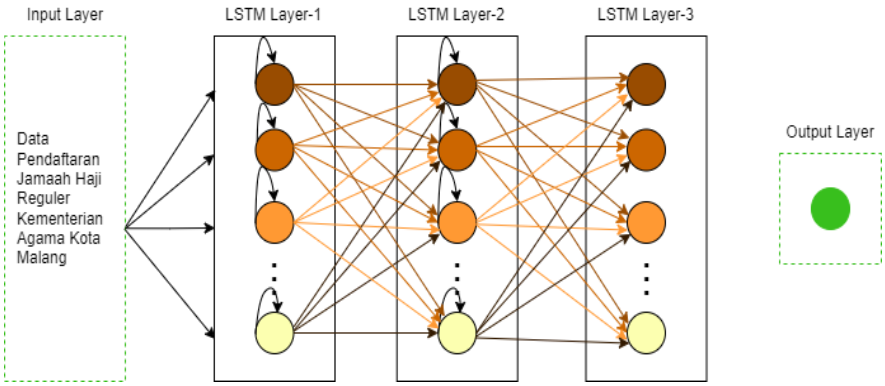


Gambar 3.1 *Flowchart* Penelitian dengan LSTM

Gambar 3.1 menunjukkan alur penelitian yang dimulai dari *Input* Data**,** di mana data historis calon jamaah haji dikumpulkan dari sistem SISKOHAT dengan variabel utama seperti Tahun Daftar, Bulan Daftar, Usia Daftar, Usia Berangkat, Masa Tunggu (Tahun), Kuota, Prediksi Konvensional, Tahun Berangkat Lunas dan Jangka Waktu Melunasi karena relevansi tinggi terhadap prediksi keberangkatan (Dzakiroh, 2025). Data kemudian dinormalisasi untuk menyamakan skala variabel, seperti Usia dan Kuota, guna mencegah bias pada model dan meningkatkan deteksi pola. Selanjutnya, dilakukan *windowing* untuk membentuk struktur data sekuensial, di mana setiap data latih disusun dalam bentuk jendela waktu, berukuran tetap (misalnya 10 baris) agar model dapat mengenali pola temporal. Proses ini memungkinkan LSTM mempelajari urutan dan tren data historis secara bertahap dari masa ke masa. Pada proses LSTM, model mengolah data sekuensial dengan mempertimbangkan pola temporal melalui tiga mekanisme utama *forget gate* (menghapus informasi tidak relevan, seperti Masa Tunggu tertentu yang tidak signifikan), *input gate* (menambahkan informasi baru, seperti hubungan antara Usia, Masa Tunggu dan Kuota), dan *output gate* (menghasilkan keluaran sementara untuk prediksi keberangkatan). Setelah melalui LSTM *layer*, hasil sementara di-*denormalisasi* ke skala aslinya agar interpretasi prediksi, seperti waktu tunggu dalam bulan atau tahun, dapat dimengerti dalam konteks nyata. Tahap akhir adalah *output*, di mana hasil prediksi akhir seperti estimasi waktu keberangkatan calon jamaah haji diperoleh dari *output layer* yang mengonversi informasi olahan menjadi bentuk yang sesuai dengan target penelitian.

3.1.1 Arsitektur Model LSTM

Model LSTM yang digunakan dalam penelitian ini dirancang khusus untuk menangani data sekuensial yang berkaitan dengan jadwal keberangkatan haji, dengan tujuan menghasilkan prediksi yang akurat berdasarkan data historis. Arsitektur model LSTM terdiri dari tiga lapisan utama, *input layer*, *hidden layers*, dan *output layer* (Zhao et al., 2022).



Gambar 3.2 Arsitektur LSTM

Input layer berfungsi menerima berbagai jenis data historis yang relevan, seperti jadwal keberangkatan haji, kuota tahunan, jumlah pendaftar, dan faktor eksternal lainnya yang memengaruhi prediksi (Adikara, 2023). Data yang dimasukkan ke dalam *input layer* kemudian diteruskan ke *hidden layers* untuk diproses lebih lanjut. *Hidden layers* merupakan inti dari model LSTM, terdiri dari unit-unit LSTM yang dirancang untuk memproses data sekuensial dengan lebih efektif (Rokhsat-Yazdi et al., 2020). Unit LSTM ini memiliki tiga komponen utama yang disebut sebagai gerbang, yaitu gerbang *input* (*input gate*), gerbang *output* (*output gate*), dan gerbang lupa (f*orget gate*) (Oruh & Viriri, 2022). Gerbang *input* bertugas mengatur informasi baru yang akan disimpan dalam sel memori LSTM (P.

Li et al., 2022). Gerbang ini memutuskan data mana dari *input* yang relevan dan perlu diintegrasikan ke dalam memori. Sebaliknya, gerbang lupa menentukan informasi mana yang tidak lagi relevan atau diperlukan, sehingga harus dihapus dari sel memori. Dengan cara ini, LSTM mampu membuang informasi yang tidak signifikan, menjaga agar memori tidak terlalu penuh dengan data yang tidak relevan. Gerbang *output*, di sisi lain, mengatur informasi mana yang akan digunakan sebagai *output* dari unit LSTM pada langkah waktu tertentu. Gerbang ini memastikan bahwa hanya informasi yang relevan dan penting yang diambil dari sel memori untuk mempengaruhi *output* dan langkah waktu berikutnya.

Setelah data diproses di *hidden layers*, maka diteruskan ke *output layer*. *Output layer* ini bertanggung jawab untuk menghasilkan prediksi akhir terkait jadwal keberangkatan haji. Prediksi ini didasarkan pada pola yang telah dipelajari dari data historis selama tahap pelatihan (Gauch et al., 2021). Dengan menggunakan arsitektur ini, model LSTM mampu menangkap dan mempelajari pola jangka panjang dalam data sekuensial, memungkinkan prediksi yang lebih akurat dan responsif terhadap perubahan data atau tren baru. Arsitektur ini sangat efektif dalam mengolah data yang memiliki urutan waktu, karena kemampuan unit LSTM untuk mengingat informasi penting dan melupakan yang tidak relevan, memberikan keunggulan signifikan dalam aplikasi prediktif seperti penjadwalan keberangkatan haji (Fuente et al., 2024).

3.2 Analisis Data

Data historis yang digunakan dalam penelitian ini mencakup rentang waktu dari tahun 2008 hingga 2019, dengan total 1132 entri dan 10 fitur atau atribut data

yang beragam. Untuk mempersiapkan data ini agar siap digunakan dalam pelatihan model LSTM, beberapa langkah analisis dan praproses perlu dilakukan terlebih dahulu (Sharma & Chariar, 2024). Langkah awal adalah pembersihan data, yang bertujuan untuk menghapus *noise* dan ketidakkonsistenan, termasuk entri yang tidak lengkap atau duplikat (Baduge et al., 2022). Setelah pembersihan, data perlu ditransformasi melalui proses normalisasi untuk menyelaraskan skala data numerik, seperti Usia dan Kuota, serta *encoding* untuk mengubah data kategorikal ke dalam format numerik yang dapat diterima oleh model pembelajaran mesin.

Selanjutnya, data yang telah diproses ini dibagi menjadi dua set, set pelatihan dan set pengujian, biasanya dengan rasio 80:20 (Huang et al., 2024). Pembagian ini penting untuk memastikan bahwa model LSTM dapat belajar dari data yang ada dan mampu melakukan generalisasi yang baik terhadap data baru yang belum pernah dilihat sebelumnya. Dalam proses pelatihan model LSTM, beberapa parameter kunci harus ditentukan untuk memastikan performa optimal (Vural et al., 2022).

Selain itu, pemilihan fungsi aktivasi yang tepat pada lapisan *output* juga penting. Untuk masalah ini, fungsi aktivasi '*tanh*' digunakan karena dapat menangani nilai positif dan negatif secara kontinu, yang cocok untuk regresi waktu (Saputra et al., 2022). Evaluasi model dilakukan menggunakan metrik seperti *Mean Absolute Error* (MAE) dan *Root Mean Square Error* (RMSE) untuk mengukur tingkat kesalahan prediksi (Sahu, 2022). Kedua metrik ini cocok untuk masalah prediksi waktu dengan nilai kontinu, karena mereka memberikan gambaran yang jelas tentang akurasi prediksi model (Masood et al., 2022). Dengan melakukan

praproses data yang tepat dan memilih parameter model yang optimal, model LSTM diharapkan mampu mempelajari pola dalam data sekuensial secara efektif dan menghasilkan prediksi yang lebih akurat dibandingkan metode konvensional yang digunakan dalam sistem SISKOHAT.

3.3 Implementasi Model dan Evaluasi

Bab ini menjelaskan proses implementasi model prediksi keberangkatan haji berbasis LSTM serta evaluasi kinerjanya. Implementasi model ini melibatkan beberapa tahapan, mulai dari persiapan data, pembangunan model, hingga pelatihan dan evaluasi hasil prediksi. Dalam tahap implementasi, penyesuaian parameter model juga dilakukan untuk mencapai performa terbaik. Evaluasi kinerja model dilakukan dengan menggunakan metrik yang sesuai, seperti MAE dan RMSE, untuk mengukur akurasi prediksi keberangkatan haji berdasarkan data yang tersedia.

3.3.1 Pengumpulan dan Praproses Data

Data historis yang digunakan untuk pelatihan model dikumpulkan dari sumber data SISKOHAT, yang mencakup berbagai informasi penting terkait calon jamaah haji. Beberapa parameter utama dalam data ini adalah Porsi, Usia Daftar, Usia Berangkat, Masa Tunggu (Tahun), Kuota, Bulan Daftar, Tahun Daftar, Prediksi Konvensional, Tahun Berangkat Lunas, dan Jangka Waktu Melunasi yang secara langsung memengaruhi hasil prediksi keberangkatan.

Tabel 3.1 Sampel Data

Porsi	Usia Daftar	Usia Berangkat	Masa Tunggu	Kuota	Bulan Daftar	Tahun Daftar	Prediksi Konvensional	Tahun Berangkat Lunas	Jangka Waktu Munasi
1300200690	15	31	13	1100	2	2008	2024	2024	16
1300188281	39	55	13	1100	1	2008	2024	2024	16
…….	…….	…….	…….	…….	…….	…….	…….	…….	…….
1300471410	19	32	13	1100	11	2010	2024	2024	14
1300472192	21	34	13	1100	11	2010	2024	2024	14
……	…….	…….	…….	…….	…….	…….	…….	…….	…….
1300586201	45	58	13	1100	12	2011	2024	2024	13
1300586197	46	59	13	1100	12	2011	2024	2024	13
……	…….	…….	…….	…….	…….	…….	…….	…….	…….
1300613190	44	57	13	1100	2	2012	2025	2024	12
1300595708	59	71	13	1100	1	2012	2025	2024	12
……	…….	…….	…….	…….	…….	…….	…….	…….	…….
1300820235	76	87	13	1100	12	2013	2026	2024	11
……	…….	…….	…….	…….	…….	…….	…….	…….	…….
1300869551	75	85	13	1100	6	2014	2027	2024	10

Porsi	Usia Daftar	Usia Berangkat	Masa Tunggu	Kuota	Bulan Daftar	Tahun Daftar	Prediksi Konvensional	Tahun Berangkat Lunas	Jangka Waktu Munasi
1300850387	74	85	13	1100	2	2014	2027	2024	10
……	…….	…….	…….	…….	…….	…….	…….	…….	…….
1301008134	78	87	13	1100	12	2015	2028	2024	9
……	…….	…….	…….	…….	…….	…….	…….	…….	…….
1301138856	84	91	13	1100	4	2017	2030	2024	7
1301185138	78	85	13	1100	9	2017	2030	2024	7
……	…….	…….	…….	…….	…….	…….	…….	…….	…….
1301325381	85	90	13	1100	10	2018	2031	2024	6
……	…….	…….	…….	…….	…….	…….	…….	…….	…….
1301385129	81	86	13	1100	4	2019	2032	2024	5

Parameter yang digunakan dalam penelitian ini memiliki peran yang saling berhubungan dalam menentukan jadwal keberangkatan jamaah haji. Tanggal Daftar menjadi dasar penentuan urutan antrean berdasarkan prinsip *first come, first served*, seperti yang diatur dalam sistem SISKOHAT (Hafiz & Nurdianty, 2024). Parameter ini dilengkapi dengan Nomor Porsi, yang berfungsi sebagai identifikasi unik setiap calon jamaah dan memantau status mereka dalam daftar tunggu secara transparan (Fredricka & Ihsan, 2020). Usia juga menjadi faktor penting, terutama dalam memberikan prioritas kepada jamaah lansia sesuai Pasal 25 PMA No. 13 Tahun 2021, yang menegaskan pentingnya perhatian khusus terhadap jamaah berusia lanjut (Astuti et al., 2021). Selain itu, Masa Tunggu, yaitu jarak waktu antara pendaftaran dan perkiraan keberangkatan, digunakan untuk mengevaluasi efisiensi pengelolaan haji, di mana rata-rata masa tunggu di Indonesia mencapai lebih dari 20 tahun (Farhan, 2020). Terakhir, Kuota, yang ditetapkan melalui kesepakatan *bilateral* antara pemerintah Indonesia dan Arab Saudi, menjadi variabel yang sangat memengaruhi panjangnya masa tunggu dan prioritas keberangkatan (Anwar, 2020). Hubungan antara kelima parameter ini memungkinkan prediksi keberangkatan menjadi lebih akurat dan sesuai dengan kebijakan pengelolaan haji yang ada.

Sebelum data ini dapat digunakan untuk melatih model, diperlukan serangkaian langkah praproses untuk memastikan kualitas dan konsistensinya. Langkah pertama dalam praproses data adalah pembersihan data, yang bertujuan untuk menghapus *noise* dan ketidak konsistenan yang mungkin ada, sehingga hanya informasi yang relevan dan akurat yang tersisa (Lee et al., 2021). Setelah pembersihan, langkah berikutnya adalah pemilihan fitur, setelah melakukan

pemilihan fitur selanjutnya dilakukan normalisasi data, yang menyelaraskan skala data agar sesuai untuk digunakan dalam model pembelajaran mesin, memastikan bahwa setiap fitur memiliki bobot yang seimbang dalam proses pelatihan. Terakhir, data yang telah diproses ini kemudian dibagi menjadi dua set, set latih dan set uji. Pembagian ini penting untuk memastikan bahwa model yang dilatih tidak hanya mampu mempelajari pola dari data yang telah dilihatnya (set latih), tetapi juga dapat melakukan generalisasi yang baik terhadap data baru (set uji), yang merupakan indikasi kinerja model yang efektif dalam situasi dunia nyata.

3.3.2 Pelatihan Model LSTM

Setelah data siap, model LSTM dilatih menggunakan dataset latih. Proses pelatihan melibatkan penggunaan algoritma optimasi seperti *Adam optimizer* dan dua fungsi *loss*, yaitu MAE dan RMSE (Zipporah & Christopher, 2023). Fungsi MAE digunakan untuk mengukur rata-rata kesalahan *absolut* antara prediksi dan nilai aktual, sedangkan RMSE mengukur kesalahan prediksi dengan mempertimbangkan kuadrat dari *error*, yang lebih sensitif terhadap kesalahan besar (Patil et al., 2022).

3.3.3 Evaluasi Kinerja Model LSTM dan Sistem SISKOHAT

Setelah proses pelatihan model LSTM selesai, langkah selanjutnya adalah melakukan evaluasi kinerja untuk menilai efektivitas model dalam memprediksi jadwal keberangkatan haji. Evaluasi ini dilakukan dengan membandingkan hasil prediksi yang dihasilkan oleh model LSTM dengan hasil prediksi dari sistem konvensional yang digunakan oleh SISKOHAT. Beberapa metrik evaluasi

digunakan untuk mengukur seberapa baik model LSTM berfungsi dibandingkan dengan metode konvensional. Metrik pertama yang digunakan adalah MAE dan RMSE, yang keduanya digunakan untuk mengukur tingkat kesalahan prediksi yang dibuat oleh model LSTM. MAE menghitung rata-rata kesalahan absolut antara nilai prediksi dan nilai aktual, sedangkan RMSE memberikan gambaran yang lebih sensitif terhadap kesalahan besar dengan memperhitungkan kuadrat dari perbedaan antara nilai prediksi dan nilai aktual (Tong et al., 2022).

3.4 Cara Kerja LSTM dalam Memprediksi Keberangkatan

Proses perhitungan dalam LSTM dalam penelitian ini dimulai dari tahapan praproses data. Pada tahap ini, data mentah yang digunakan harus dinormalisasi untuk memastikan setiap fitur dalam dataset memiliki skala yang seragam. Langkah ini penting karena model LSTM bekerja dengan data numerik dalam skala yang sama sehingga menghindari adanya fitur yang mendominasi pelatihan model hanya karena skala datanya lebih besar (Dropka et al., 2021). Misalnya, jika fitur usia jamaah haji berada dalam rentang 20 hingga 70 tahun, sementara fitur seperti kuota bersifat biner atau dalam skala yang lebih kecil, tanpa normalisasi, fitur usia akan memberikan bobot yang lebih besar dalam pelatihan, menyebabkan model lebih memperhatikan usia daripada faktor lain seperti masa tunggu. Normalisasi membantu menjaga keseimbangan antara fitur-fitur tersebut, sehingga model dapat belajar dari keseluruhan data dengan cara yang adil.

3.4.1 Normalisasi Data

Rumus normalisasi yang digunakan adalah sebagai berikut:

$$x_{norm} = \frac{x - min(x)}{max(x) - min(x)} \quad (3.1)$$

Di mana:
a. *x* adalah nilai asli dari fitur yang sedang dinormalisasi,
b. *min*(*x)* adalah nilai minimum dari fitur tersebut dalam dataset,
c. *max*(*x*) adalah nilai maksimum dari fitur tersebut,
d. x_{norm} adalah nilai hasil normalisasi (Y. Li et al., 2023).

Sebagai contoh, pada penelitian ini, jika usia jamaah haji adalah 50 tahun, dan rentang usia dalam dataset adalah 20 hingga 70 tahun, maka nilai usia dinormalisasi sebagai berikut:

$$Usia_{norm} = \frac{50-20}{70-20} = \frac{30}{50} = 0.6$$

Normalisasi membantu memastikan bahwa model tidak memberikan bobot berlebihan pada fitur usia dibandingkan fitur lain seperti masa tunggu atau kuota haji. Dengan nilai normalisasi 0.6, usia calon jamaah terletak di tengah rentang usia yang ada, sehingga fitur ini memiliki bobot yang proporsional dalam proses pelatihan model.

3.4.2 Menghapus Entri Kosong (*Null Values*)

Langkah selanjutnya adalah menghapus entri yang memiliki nilai kosong atau *null values*. Nilai kosong dapat terjadi karena data yang tidak lengkap atau kesalahan *input*, dan jika tidak dihapus, bisa menyebabkan kesalahan dalam perhitungan model.

Rumus yang digunakan adalah:

$$D_{final} = D_{cleaned} - D_{null} \quad (3.2)$$

Di mana:
a. $D_{cleaned}$ adalah dataset yang sudah dibersihkan dari duplikasi,
b. D_{null} adalah baris atau entri yang memiliki nilai kosong,
c. D_{final} adalah dataset akhir yang telah dibersihkan dari *null values* (Deloffre et al., 2023).

Sebagai contoh, jika ditemukan 100 entri dengan nilai kosong, maka dataset final yang digunakan untuk pelatihan model menjadi:

$$D_{final} = 1132 - 100 = 1032$$

Menghapus entri dengan nilai kosong memastikan bahwa model hanya dilatih menggunakan data yang lengkap dan valid. Dengan hanya menggunakan data yang lengkap, prediksi menjadi lebih stabil dan terpercaya.

3.4.3 Proses *Windowing*

Dalam pemodelan deret waktu, penting bagi model untuk memahami pola hubungan antar data berdasarkan urutan waktunya (Islam et al., 2024). Salah satu pendekatan yang digunakan adalah *windowing*, yaitu proses membagi data deret waktu menjadi urutan-urutan data dengan panjang tertentu secara bergeser. Teknik ini memungkinkan model LSTM untuk mempelajari dinamika data dari segmen waktu sebelumnya dalam proses pelatihan. Pada tahap ini, data hasil normalisasi dibentuk menjadi sekumpulan jendela, di mana setiap jendela terdiri atas sejumlah baris data yang berurutan sepanjang ukuran *window* yang ditentukan, yaitu W_d. Jendela tersebut dibentuk dengan cara menggeser satu baris ke depan secara berulang dari awal hingga akhir data. Proses pembentukan jendela ini secara matematis menyebabkan jumlah jendela yang dihasilkan lebih kecil dari jumlah

data awal, tergantung pada panjang *window* yang digunakan. Jumlah jendela J dapat dihitung dengan rumus berikut:

$$\mathrm{J} = T - W_d + 1 \tag{3.3}$$

Di mana:
a. J adalah jumlah jendela (*window*) yang terbentuk,
b. T adalah jumlah total data deret waktu,
c. W_d adalah *window* yang digunakan.

Sebagai contoh, jika jumlah data awal adalah 1000 baris dan ukuran *window* yang digunakan adalah 10, maka akan terbentuk:

$$\mathrm{J} = 1000 - 10 + 1 = 991$$

Jendela pertama memuat data dari baris ke-1 hingga ke-10, jendela kedua dari baris ke-2 hingga ke-11, dan seterusnya hingga jendela ke-991. Susunan data seperti ini memudahkan model untuk mempelajari urutan dan pola historis dengan lebih baik. Pendekatan ini telah digunakan secara luas dalam pemrosesan deret waktu dan terbukti efektif dalam meningkatkan akurasi klasifikasi serta deteksi anomali (Ghalyan et al., 2022).

3.4.4 Proses Perhitungan LSTM

Langkah berikutnya adalah menentukan parameter model, seperti jumlah unit dalam setiap lapisan LSTM, ukuran *batch*, laju pembelajaran, dan jumlah *epoch* untuk pelatihan (Bhandari et al., 2022). Penentuan parameter ini penting untuk memastikan bahwa model mampu menangkap pola yang relevan dari data sekuensial dan memprediksi hasil dengan akurat. Proses dalam model LSTM dimulai dengan data *input* yang diterima oleh sel LSTM dan diproses melalui beberapa gerbang yang berfungsi untuk mengatur aliran informasi. *Cell state*, yang

berfungsi sebagai wadah untuk menyimpan informasi sekuensial, pertama kali diinisialisasi dengan nilai-nilai tertentu.Setelah itu, *Forget gate* dan *input gate* dihitung untuk menentukan informasi yang harus dilupakan dan yang perlu diperbarui dalam *cell state*. Selama langkah ini, *cell state* diperbarui berdasarkan hasil dari kedua gerbang tersebut, yang memungkinkan model untuk mengingat informasi relevan dari langkah sebelumnya. Model LSTM memiliki beberapa langkah utama yang memproses data secara berurutan dan memperbarui informasi untuk menghasilkan *output* yang diprediksi. Berikut adalah penjelasan tentang rumus-rumus yang digunakan pada setiap langkah LSTM:

1. *Forget Gate* (Gerbang Lupa)

   Pada langkah pertama, *forget gate* memutuskan berapa banyak informasi dari *cell state* yang perlu dilupakan atau dipertahankan berdasarkan *input* saat ini dan *hidden state* pada waktu sebelumnya. Fungsi aktivasi *sigmoid* digunakan untuk mengontrol seberapa banyak informasi yang akan dilupakan.

$$y_{cj}(t) = \sigma\left(net_{cj}(t)\right) \tag{3.4}$$

Di mana:

a. $y_{cj}(t)$, *output* dari *forget gate* yang menentukan berapa banyak informasi dari *cell state* yang dilupakan.
b. σ, fungsi aktivasi *sigmoid*, yang menghasilkan nilai antara 0 dan 1.
c. $net_{cj}(t)$, kombinasi dari *hidden state* pada waktu $t-1$ dan *input* saat ini, yang dihitung menggunakan pembobotan dan bias.

Tabel 3.2 Sample Data Bobot dan Bias

Layer	Urutan	Nilai
Bobot	Top 5 Bobot	
	1	0.000000
	2	-0.000162
	3	-0.000281
	4	0.000018
	5	-0.000204
	Bottom 5 Bobot	
	29580	-0.084975

Layer	Urutan	Nilai
	29581	0.097671
	29582	0.079309
	29583	-0.088790
	29584	0.089066
Bias	Top 5 Biases	
	1	-0.056676
	2	0.055872
	3	0.017868
	4	0.041787
	5	-0.037051
	Bottom 5 Biases	
	103435	-0.659140
	103436	-0.205519
	103437	0.497439
	103438	0.071664
	103439	-0.154366

$$net_{cj}(t) = (Hidden\ state \text{ pada waktu } t-1) \times (Input \text{ saat ini}) + \text{Bias} \quad (3.5)$$

Dengan memasukkan nilai-nilai yang telah disebutkan, memperoleh:

$$net_{cj}(t) = (-0.000162) \times 0.05667568 + (-0.056676)$$

$$net_{cj}(t) = -0.00000918 + (-0.056676) = -0.05668518$$

Setelah menghitung *net* untuk *forget gate*, yang hasilnya adalah −0.05668518-0.05668518−0.05668518, langkah berikutnya adalah menghitung *output* dari *forget gate* menggunakan fungsi aktivasi *sigmoid*. Fungsi *sigmoid* digunakan untuk membatasi *output* antara 0 dan 1. Fungsi *sigmoid* didefinisikan sebagai:

$$\sigma(x) = \frac{1}{1+e^{-x}} \quad (3.6)$$

Dengan $x = -0.05668518$, dapat menghitung *output*nya:

$$y_{cj}(t) = \sigma(-0.05668518)$$

Sekarang, menghitung menggunakan rumus *sigmoid*:

$$y_{cj}(t) = \frac{1}{1+e^{-0.05668518}} = \frac{1}{1+1.0585} = \frac{1}{1+2.0585} = 0.4858$$

Output dari *forget gate* $y_{cj}(t)$, akan digunakan untuk memodifikasi *cell state* di langkah berikutnya. Nilai 0.4858 berarti bahwa sekitar 48.58% dari informasi yang ada dalam *cell state* akan dipertahankan, sementara sisanya (lebih dari 50%) akan dilupakan atau tidak dipertahankan. Nilai ini berada dalam rentang 0 dan 1, yang memungkinkan LSTM untuk memutuskan informasi mana yang perlu dilupakan dan mana yang harus dipertahankan berdasarkan data sebelumnya dan saat ini.

2. *Input Gate* (Gerbang *Input*)

   Input Gate berfungsi untuk memutuskan berapa banyak informasi baru yang akan disimpan dalam *cell state*. Informasi yang disimpan di dalam *cell state* ini akan dipertahankan untuk langkah-langkah berikutnya, yang akan berkontribusi dalam menghasilkan prediksi. Fungsi aktivasi *sigmoid* digunakan pada *Input Gate* untuk mengontrol berapa banyak informasi baru yang disimpan berdasarkan informasi dari *hidden state* pada waktu sebelumnya dan *input* yang diterima pada waktu t.

$$y_{inj}(t) = \sigma\left(net_{inj}(t)\right) \tag{3.7}$$

Di mana:
a. $y_{inj}(t)$, *output* dari *input gate* yang menentukan berapa banyak informasi baru yang akan disimpan dalam *cell state*.
b. σ, fungsi aktivasi *sigmoid*, yang menghasilkan nilai antara 0 dan 1.
c. $net_{inj}(t)$, kombinasi dari *hidden state* dan *input* saat ini.

Dengan menggunakan *sigmoid*, model dapat memutuskan seberapa banyak informasi baru yang relevan untuk disimpan dalam *cell state*, memungkinkan model untuk menyesuaikan pembelajaran berdasarkan data baru yang diterima pada setiap langkah waktu. Seperti halnya *forget gate*, *input gate* juga

menggunakan *hidden state* dan *input* saat ini. Pembobotan dan bias yang digunakan dalam perhitungan ini diambil dari nilai yang telah kita tentukan sebelumnya.

Dengan memasukkan nilai-nilai yang telah disebutkan, memperoleh:

$$net_{inj}(t) = (-0.000162) \times 0.05667568 + (-0.056676)$$

$$net_{inj}(t) = -0.00000918 + (-0.056676) = -0.05668518$$

Setelah mendapatkan nilai *net*, langkah berikutnya adalah menghitung *output* dari *input gate* menggunakan fungsi aktivasi *sigmoid*. Dengan $x = -0.05668518$, dapat menghitung *output*nya:

$$y_{inj}(t) = \sigma(-0.05668518)$$

Sekarang, menghitung menggunakan rumus *sigmoid*:

$$y_{inj}(t) = \frac{1}{1 + e^{-0.05668518}} = \frac{1}{1 + 1.0585} = \frac{1}{1 + 2.0585} = 0.4858$$

Jadi, *output* dari *input gate* adalah sekitar 0.4858. *Output* ini menunjukkan seberapa banyak informasi baru yang akan disimpan dalam *cell state*. Nilai ini berada dalam rentang 0 hingga 1, yang memungkinkan LSTM untuk memutuskan berapa banyak informasi baru yang relevan perlu disimpan. Dalam hal ini, sekitar 48.58% dari informasi baru akan disimpan dalam *cell state*, *Output* dari *input gate* ini kemudian digunakan untuk memperbarui *cell state* dan berlanjut ke perhitungan *hidden state* pada langkah waktu berikutnya.

3. Unit Memori dengan *tanh*

   Setelah *input gate*, informasi yang dipilih untuk disimpan akan diproses dalam unit memori atau *cell state*. Untuk memastikan informasi yang disimpan memiliki rentang yang stabil, *tanh* digunakan pada perhitungan *cell state*. Fungsi

tanh mengatur nilai memori dalam rentang [−1.1], yang memungkinkan model untuk menyimpan informasi secara lebih efisien dan terkontrol.

$$sc_j(t) = sc_j(t-1) + y_{inj}(t) \cdot tanh\left(net_{cj}(t)\right) - y_{cj}(t) \cdot tanh\left(sc_j(t)\right) \quad (3.8)$$

Di mana:
a. $sc_j(t)$, *cell state* yang mrnyimpan informasi relevan.
b. $tanh$, fungsi aktivasi *hyperbolic tanngent*, yang menghasilkan *output* [-1.1].
c. $y_{inj}(t)$, *output* dari *input gate*.
d. $net_{cj}(t)$, kombinasi *input* dan *hidden state* menghasilkan *candidate memory*.
e. $y_{cj}(t)$, *output* dari *forget gate* yang mengontrol banyaknya informasi yang dilupakan.

Fungsi *tanh* memastikan bahwa informasi dalam *cell state* tetap stabil dan dapat berfungsi dengan baik dalam jangka panjang, menghindari masalah seperti hilangnya gradien. *tanh* juga memastikan bahwa *cell state* dapat menampung berbagai informasi dalam rentang yang dapat ditangani oleh model. Setelah menghitung *output* dari *input gate* dan *forget gate*, langkah selanjutnya adalah memperbarui *cell state* menggunakan fungsi aktivasi *tanh* untuk memastikan nilai dalam *cell state* tetap stabil dan terkontrol. Dalam perhitungan ini, *output* dari *input gate* adalah $y_{inj}(t) = 0.4858$, dan *output* dari *forget gate* adalah $y_{cj}(t) = 0.4858$. Nilai net dari *forget gate* yang telah dihitung sebelumnya adalah $net_{cj}(t) = -0.05668518$. Asumsikan *cell state* pada waktu $t-1$ adalah 0 untuk memudahkan perhitungan awal. Untuk memperbarui *cell state*, langkah pertama adalah menghitung *tanh* dari nilai net yang dihasilkan oleh *forget gate*. Fungsi *tanh* digunakan untuk menjaga nilai tetap dalam rentang antara -1 dan 1, sehingga informasi yang tersimpan tetap stabil. Menghitung *tanh* dari $net_{cj}(t)$:

$$tanh(-0.05668518) = -0.056673$$

Setelah itu, *cell state* pada waktu *t* dihitung dengan mengalikan *output* dari *input gate* dengan *tanh* dari *net forget gate*, kemudian mengurangkan hasil dari *forget gate* yang dikalikan dengan *tanh* dari *cell state* sebelumnya. Karena *cell state* pada waktu $t-1$ diasumsikan 0, hanya bagian pertama yang berkontribusi:

$$cell\ state(t) = 0 + 0.4858 \times (-0.056673) = -0.027577$$

Dengan demikian, *cell state* pada waktu *t* setelah diperbarui adalah sekitar −0.027577. Pembaruan ini memungkinkan model untuk menyimpan informasi yang relevan dan melupakan informasi yang tidak diperlukan, menjaga agar *cell state* tetap dalam rentang yang dapat dikelola, serta mempersiapkan informasi untuk langkah berikutnya dalam proses LSTM.

4. Perhitungan Gradien dann Pembaruan Bobot dengan *tanh*

Selama proses pelatihan, model LSTM melakukan pembaruan bobot berdasarkan kesalahan yang dihitung pada langkah sebelumnya. Dalam hal ini, *tanh* digunakan pada gradien untuk memastikan pembaruan bobot dilakukan dengan baik tanpa menyebabkan gradien menghilang atau meledak.

$$\frac{\partial E(t)}{\partial wlm} = -\frac{\partial E(t)}{\partial y(t)} \cdot y(t-1) \qquad (3.9)$$

Di mana:

a. $\frac{\partial E(t)}{\partial wlm}$, pembaruan bobot berdasarkan gradien kesalahan.
b. $\frac{\partial E(t)}{\partial y(t)}$, gradien dari kesalahan yang dihitung.
c. $y(t-1)$, *output* dari *hidden state* sebelumnya saat menghitung pembaruan.

Dengan menggunakan *tanh* pada gradien, pembaruan bobot lebih stabil, sehingga proses pembelajaran bisa dilakukan secara efektif. *tanh* membantu

menjaga agar informasi yang diteruskan dari langkah waktu sebelumnya tetap relevan dan tidak hilang selama pelatihan.

Setelah *cell state* diperbarui, langkah berikutnya adalah menghitung gradien kesalahan dan pembaruan bobot menggunakan *tanh* untuk memastikan pembaruan bobot yang stabil selama pelatihan. Gradien kesalahan dihitung berdasarkan MSE (*Mean Squared Error*), yang merupakan perbedaan antara prediksi model dan nilai aktual. Sebagai contoh, gradien kesalahan untuk MSE dihitung dengan turunan fungsi MSE terhadap prediksi $y(t)$. Misalkan nilai prediksi pada waktu *t* adalah $y_{prediksi}(t) = 0.9950$, dan nilai aktual adalah $y_{aktual}(t) = 1.0$, Maka gradien kesalahan:

$$\frac{\partial E(t)}{\partial wlm} = -0.9950 - 1.0 = -0.0050$$

Setelah gradien kesalahan dihitung, pembaruan bobot dilakukan dengan mengalikan gradien kesalahan dengan *output hidden state* pada waktu $t-1$, yang dalam contoh ini adalah $y(t-1) = 0.4858$. Maka Pembaruan bobot,

$$\frac{\partial E(t)}{\partial wlm} = -(-0.0050) \cdot 0.4858 = 0.002429$$

Untuk memastikan pembaruan bobot yang stabil, *tanh* digunakan untuk menghitung gradien yang distabilkan. Dengan menghitung *tanh* dari gradien $\frac{\partial E(t)}{\partial wlm} = 0.002429$, memperoleh nilai:

$$\tanh(0.002429) = 0.002429$$

Pembaruan bobot akhir, yang telah distabilkan, menjadi $\Delta w = 0.002429$. Pembaruan bobot ini memastikan bahwa pembelajaran dapat dilakukan secara efektif tanpa menyebabkan hilangnya gradien atau meledaknya gradien selama

pelatihan. Dengan nilai *tanh* yang sangat kecil, pembaruan bobot tetap stabil dan memungkinkan model untuk terus belajar dengan baik.

5. *Output Gate* (Gerbang *Output*)

   Output gate menggunakan fungsi *sigmoid* untuk memutuskan berapa banyak informasi dari *cell state* yang akan diteruskan ke *hidden state* dan digunakan untuk prediksi. Dengan menggunakan *sigmoid*, *output gate* membatasi aliran informasi, yang memungkinkan model untuk fokus hanya pada informasi yang relevan dalam membuat prediksi.

$$y_{outj}(t) = \sigma\left(net_{outj}(t)\right) \tag{3.10}$$

Di mana:
a. $y_{outj}(t)$, *output* dari *Output Gate* yang mengontrol berapa banyak informasi dari *cell state* yan diteruskan ke *hidden state*.
b. σ, fungsi aktivasi *sigmoid*, yang membatasi *output* antara 0 dan 1.
c. $net_{outj}(t)$, kombinasi dari *hidden state* sebelumnya dan *input* saat ini, yang dihitung dengan pembobotan dan bias.

Semakin besar *output sigmoid* dari *output gate*, semakin banyak informasi yang diteruskan. Sebaliknya, semakin kecil *output*, semakin sedikit informasi yang diteruskan.

Setelah pembaruan bobot ditemukan, langkah selanjutnya adalah menghitung *output* dari *output gate* yang menggunakan fungsi aktivasi *sigmoid*. Pada langkah ini, *output gate* menghitung net yang merupakan kombinasi antara *hidden state* pada waktu $t-1$ dan *input* saat ini, yang kemudian ditambahkan dengan bias. Misalkan nilai-nilai yang digunakan dalam perhitungan ini adalah sebagai berikut: *hidden state* pada waktu $t-1$ yaitu $y(t-1) = 0.4858$, *input* saat ini adalah 0.05667568, dan bias adalah -0.056676. Maka dapat menghitung net untuk *output gate* seperti berikut:

$$net_{outj}(t) = (0.4858) \times 0.05667568 + (-0.056676)$$

$$net_{outj}(t) = 0.027577 + (-0.056676) = -0.029099$$

Dengan nilai net yang dihitung, maka memasukkannya ke dalam fungsi *sigmoid*:

$$y_{outj}(t) = \sigma(-0.029099)$$

Sekarang, menghitung menggunakan rumus *sigmoid*:

$$y_{outj}(t) = \frac{1}{1 + e^{-0.029099}} = \frac{1}{1 + 1.0295} = \frac{1}{2.0295} = 0.4934$$

Output ini menunjukkan seberapa banyak informasi dari *cell state* yang akan diteruskan ke *hidden state* dan akhirnya digunakan untuk prediksi. Nilai sekitar 0.4934 berarti bahwa sekitar 49.34% dari informasi dalam *cell state* akan diteruskan, sementara sisanya (lebih dari 50%) akan dipertahankan atau tidak diteruskan ke langkah waktu berikutnya.

6. Fungsi Aktivasi untuk *Cell Memory* dengan *tanh*

   Setelah pembaruan *cell state*, *output* dari *cell memory* dihitung untuk menentukan berapa banyak informasi yang diteruskan ke *hidden state*. Dalam hal ini, *tanh* digunakan untuk memastikan bahwa informasi yang dikeluarkan dari *cell state* tetap stabil dan terkontrol sebelum diteruskan ke langkah berikutnya.

$$y_{cj}(t) = y_{outj}(t) \cdot tanh\left(sc_j(t)\right) \quad (3.11)$$

Di mana:

a. $y_{cj}(t)$, *output* dari *memory cell* yang diteruskan ke *hidden state*.
b. $y_{outj}(t)$, *output* dari *output gate* yang mengontrol aliran informasi.
c. $tanh\left(sc_j(t)\right)$, *cell state* yang diproses dengan *tanh*.

Dengan menggunakan *tanh*, model dapat memastikan bahwa informasi yang diteruskan ke *hidden state* dari *cell state* lebih stabil dan memiliki rentang yang konsisten, yang penting untuk prediksi pada langkah waktu berikutnya.

Misalkan nilai *output* dari *output gate* pada waktu *t* $y_{outj}(t)$, yang telah dihitung sebelumnya adalah 0.4934, dan nilai *cell state* pada waktu *t*, $sc_j(t)$, adalah -0.027577 (dari perhitungan sebelumnya). Langkah pertama adalah memproses *cell state* dengan fungsi *tanh*. Fungsi *tanh* menghitung nilai dari *cell state* yang diproses, yang diberikan oleh:

$$tanh(-0.027577) = -0.027556$$

Selanjutnya, *output* dari *memory cell* dihitung dengan mengalikan hasil *output gate* dengan nilai yang sudah diproses menggunakan *tanh*:

$$y_{cj}(t) = 0.4934 \cdot (-0.027556) = -0.013586$$

Dengan demikian, *output* dari *memory cell*, yang diteruskan ke *hidden state*, adalah sekitar -0.013586. Nilai ini akan digunakan untuk langkah-langkah berikutnya dalam proses LSTM untuk menghasilkan prediksi pada langkah waktu selanjutnya.

7. *Hidden State*

Setelah *output gate* mengontrol berapa banyak informasi dari *cell state* yang diteruskan, hasilnya digunakan untuk menghasilkan *hidden state* yang akan diteruskan ke langkah waktu berikutnya dan digunakan untuk prediksi.

$$h_j(t) = y_{outj}(t) \cdot tanh\left(sc_j(t)\right) \quad (3.12)$$

Di mana:

a. $h_j(t)$, *hidden state* pada waktu *t*, yang digunakann untuk prediksi pada langkah waktu tersebut dan diteruskan ke langkah waktu berikutnya.

b. $y_{outj}(t)$, *output* dari *output gate* yang mengontrol aliran informasi dari *cell state* ke *hidden state*.
c. $tanh\left(sc_j(t)\right)$, *cell state* pada waktu *t* yang diproses dengan *tanh* untuk memastikan nilai stabil dan berada dalam rentang [−1,1].

Hidden state adalah informasi yang dihasilkan dari *cell state* yang telah diproses melalui *output gate* dan tanh untuk menjaga stabilitas. Ini digunakan untuk menghasilkan prediksi dan diteruskan ke langkah waktu berikutnya dalam LSTM.

Dari perhitungan sebelumnya, kita sudah mengetahui bahwa *output* dari *output gate*, $y_{outj}(t)$, adalah 0.4934, dan *output* dari *memory cell* yang telah diproses dengan *tanh* adalah -0.013586. Dengan menggunakan nilai-nilai tersebut, dapat menghitung *hidden state*:

$$h_j(t) = 0.4934 \cdot (-0.013586) = -0.006701$$

Jadi, *hidden state* pada waktu *t* adalah sekitar -0.006701. Nilai *hidden state* ini akan digunakan pada langkah waktu selanjutnya untuk prediksi dan juga untuk memperbarui *cell state* di waktu yang akan datang. Dengan cara ini, model LSTM terus memperbarui informasi yang relevan sepanjang urutan waktu dan membuat prediksi yang lebih akurat (Hochreiter & Schmidhuber, 1997).

3.4.5 Proses Prediksi dengan LSTM

Setelah proses pembaruan *hidden state* dan *cell state* selesai, langkah selanjutnya adalah melakukan prediksi dengan menggunakan nilai *hidden state* yang telah diperbarui. Pada LSTM, prediksi dihitung dengan menghubungkan *hidden state* yang telah dihitung dengan lapisan *output*, yang kemudian memberikan nilai hasil prediksi. Proses ini dapat dilakukan dengan menggunakan

fungsi aktivasi, seperti *sigmoid* atau *softmax*, tergantung pada jenis masalah yang dihadapi. Untuk regresi, biasanya digunakan fungsi aktivasi linear atau tidak ada fungsi aktivasi pada lapisan *output*, sedangkan untuk klasifikasi, biasanya digunakan fungsi *sigmoid* atau *softmax* (L. Lin et al., 2024).

Sebagai contoh, menggunakan *hidden state* pada waktu *t*, yang telah dihitung sebelumnya sebagai -0.006701, dan menghubungkannya dengan lapisan *output*. Misalkan, bobot untuk lapisan *output* adalah 0.6, dan biasnya adalah 0.1. Prediksi dihitung dengan rumus:

$$y_{prediksi}(t) = hidden\ state \times weight + \text{bias} \tag{3.13}$$

Dengan memasukkan nilai yang telah diketahui:

$$y_{prediksi}(t) = (-0.006701) \times 0.6 + 0.1$$

$$y_{prediksi}(t) = -0.0040206 + 0.1 = 0.0959794$$

Jadi, hasil prediksi pada waktu *t* adalah sekitar 0.0959794. Nilai ini merupakan hasil dari perhitungan yang menghubungkan *hidden state* yang telah diperbarui dengan lapisan *output*, memberikan prediksi berdasarkan informasi yang telah dipelajari oleh model. Prediksi ini kemudian dapat digunakan untuk membuat keputusan atau untuk tujuan lainnya, tergantung pada aplikasi model LSTM yang digunakan.

3.4.6 Mengembalikan Nilai Prediksi ke Bentuk Asli (*Denormalisasi*)

Setelah proses prediksi selesai dilakukan oleh model LSTM, nilai yang dihasilkan biasanya berada dalam bentuk yang telah dinormalisasi menggunakan teknik seperti *MinMaxScaler* atau *StandardScaler*. Namun, untuk mendapatkan nilai yang sesuai dengan skala asli dari data, hasil prediksi perlu melalui tahap

denormalisasi. Proses *denormalisasi* ini bertujuan untuk mengembalikan nilai prediksi ke dalam rentang atau skala yang sesuai dengan data asli. Proses *denormalisasi* dilakukan dengan menggunakan rumus kebalikan dari rumus normalisasi yang digunakan sebelumnya. Misalnya, jika data awal dinormalisasi menggunakan *MinMaxScaler*, rumus *denormalisasi* yang digunakan adalah sebagai berikut:

$$\text{Nilai } Denormalisasi = (\text{Prediksi Normalisasi} \times (Max \text{ Asli} - Min \text{ Asli})) + Min \text{ Asli} \quad (3.14)$$

Sebagai contoh, misalkan hasil prediksi dari model LSTM adalah 0.0959794 (nilai normalisasi). Data asli memiliki *Max* Asli sebesar 100 dan *Min* Asli sebesar 0. Maka, untuk mengembalikan nilai prediksi ke bentuk aslinya, dilakukan perhitungan *denormalisasi* sebagai berikut:

$$\text{Nilai } Denormalisasi = (0.0959794 \times (100 - 0)) + 0$$

$$\text{Nilai } Denormalisasi = 0.0959794 \times 100 = 9.59794$$

Dengan demikian, nilai prediksi yang telah dinormalisasi 0.0959794 dikembalikan ke bentuk asli dengan hasil 9.59794. Proses ini memungkinkan prediksi yang dihasilkan oleh model untuk kembali ke rentang data asli, sehingga hasilnya dapat digunakan untuk analisis lebih lanjut atau pengambilan keputusan.

3.4.7 Menginterpresentasikan Hasil Prediksi

Setelah dilakukan *denormalisasi* terhadap hasil prediksi, langkah selanjutnya adalah menginterpretasikan hasil prediksi yang diperoleh, yaitu tahun jangka waktu melunasi. Misalnya, hasil prediksi setelah *denormalisasi* adalah 9.59794. Sebagai langkah pertama, nilai desimal tersebut perlu dibulatkan agar lebih mudah dipahami dan digunakan. Dalam hal ini, angka desimal setelah koma

kurang dari 0.5, sehingga nilai prediksi dibulatkan ke bawah menjadi 10. Proses pembulatan ini dilakukan untuk mempermudah interpretasi hasil tanpa mengurangi validitasnya. Dalam prediksi temporal, data sering dibulatkan ke unit waktu terdekat, mengikuti aturan statistik di mana angka desimal lebih besar dari 0,5 dibulatkan ke atas dan yang kurang dari 0,5 dibulatkan ke bawah (Sharma & Chariar, 2024).

Setelah pembulatan, langkah berikutnya adalah mengonversi hasil prediksi ke tahun keberangkatan yang sesuai dengan standar penghitungan konvensional. Untuk memudahkan pembacaan oleh orang awam, tahun keberangkatan dihitung dengan menjumlahkan masa tunggu dengan tahun daftar. Sebagai contoh, jika jangka waktu melunasi yang telah diprediksi adalah 9.59794 atau jika dibulatkan menjadi 10 tahun, dan tahun daftar adalah 2014, maka tahun keberangkatan yang diprediksi adalah:

$$\text{Tahun Keberangkatan} = \text{Tahun Daftar} + \text{Jangka Waktu Melunasi} \quad (3.15)$$

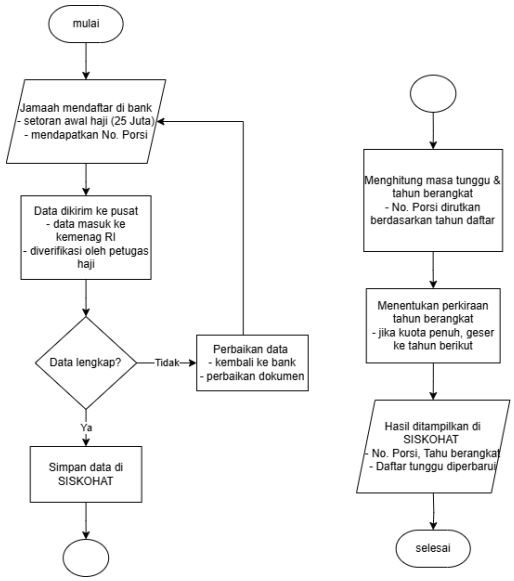
$$\text{Tahun Keberangkatan} = 2014 + 10 = 2024$$

Dengan demikian, hasil akhir yang dapat dipahami oleh pengguna adalah tahun keberangkatan yang diprediksi, yaitu 2024. Tahun ini lebih mudah diinterpretasikan dan digunakan dalam konteks prediksi waktu keberangkatan.

3.4.8 Perhitungan Konvensional SISKOHAT

Sistem SISKOHAT dirancang untuk mengelola data pendaftaran calon jamaah haji di Indonesia. Perhitungan tahun keberangkatan dalam sistem ini didasarkan pada nomor porsi, kuota tahunan, dan masa tunggu rata-rata. Nomor porsi diberikan kepada calon jamaah setelah mereka melakukan setoran awal di

bank penerima setoran haji. Selanjutnya, data dikirim ke pusat, diverifikasi oleh Kemenag, dan diproses dalam sistem untuk menentukan estimasi keberangkatan.



Gambar 3.3 *Flowchart* SISKOHAT menentukan Estimasi Keberangkatan Haji

Dalam Gambar 3.3 menunjukkan alur SISKOHAT, dimana tahun keberangkatan calon jamaah haji dihitung berdasarkan beberapa faktor utama. Tahun Daftar adalah tahun ketika seorang calon jamaah melakukan pendaftaran resmi dengan mendapatkan nomor porsi setelah menyetorkan dana awal haji. Nomor porsi ini menentukan urutan antrean keberangkatan berdasarkan prinsip *first come, first served* (Mohd et al., 2024).

Selain itu, faktor kuota haji per tahun memengaruhi kecepatan antrean berjalan. Kuota haji adalah jumlah jamaah yang bisa diberangkatkan setiap tahunnya, yang ditentukan oleh kebijakan pemerintah Indonesia dan Arab Saudi. Jika kuota tahunan besar, maka antrean lebih cepat bergerak, sementara jika kuota kecil, masa tunggu menjadi lebih lama. Terdapat juga faktor prioritas khusus, seperti jamaah lansia. Kelompok ini bisa mendapatkan prioritas keberangkatan

lebih awal dibandingkan jamaah lainnya, meskipun mereka mendaftar lebih belakangan.

Dalam hal ini, jika seorang calon jamaah mendaftar pada tahun 2011, sistem akan langsung menghitung prediksi tahun keberangkatan mereka sesuai dengan masa tunggu rata-rata yang merupakan estimasi waktu yang dibutuhkan oleh seorang calon jamaah haji dari saat pendaftaran hingga keberangkatan, berdasarkan ketersediaan kuota tahunan dan jumlah total pendaftar. Perhitungan ini digunakan untuk memberikan perkiraan tahun keberangkatan jamaah secara lebih sistematis dan transparan. Prediksi keberangkatan dihitung menggunakan rumus:

$$T_{berangkat} = T_{daftar} + masa\ tunggu\ rata - rata \tag{3.13}$$

Misalnya, seorang jamaah mendaftar pada 2011 dan masa tunggunya adalah 13 tahun, maka prediksi keberangkatannya adalah:

$$T_{berangkat} = 2011 + 13$$

$$T_{berangkat} = 2024$$

Sesuai dengan tabel sampel data (Tabel 3.1), kolom Masa Tunggu menunjukkan nilai aktual, yaitu durasi waktu tunggu yang dihitung untuk setiap individu berdasarkan prioritas dan kebijakan penjadwalan. Dalam perhitungan konvensional SISKOHAT, digunakan masa tunggu rata-rata karena nilai ini merepresentasikan generalisasi dari pola masa tunggu historis secara nasional. Masa tunggu rata-rata diinisialisasikan untuk mempermudah estimasi waktu keberangkatan, mengingat variabel-variabel seperti kuota tahunan dan kebijakan lokal dapat menyebabkan variasi nilai masa tunggu aktual. Dengan asumsi stabilitas

kuota dan pola penjadwalan, nilai rata-rata ini memberikan pendekatan prediktif yang lebih praktis.

Hasil ini menunjukkan bahwa calon jamaah yang mendaftar pada tahun 2011 diperkirakan akan berangkat pada tahun 2024. Tahun 2024 adalah hasil prediksi langsung yang ditampilkan oleh sistem SISKOHAT kepada jamaah haji. Dalam kasus ini, jika jamaah mendaftar pada usia 69 tahun pada 2011, maka mereka diperkirakan akan berusia 82 tahun pada tahun keberangkatan 2024.

Namun, jika jamaah tersebut masuk dalam kategori lansia prioritas, yang diatur dalam Pasal 25 PMA No 13 Tahun 2021, maka mereka dapat diprioritaskan untuk keberangkatan lebih cepat (Astuti et al., 2021). Jamaah yang berusia 65 tahun atau lebih memenuhi syarat untuk kuota prioritas ini (Farhan, 2020). Misalnya, jika jamaah tersebut mencapai usia 69 tahun pada 2011, mereka dapat masuk dalam kuota lansia prioritas yang terdiri atas 5% dari total kuota tahunan. Dalam hal ini, nomor porsi mereka akan dimajukan dalam antrean, meskipun prediksi awal mereka adalah 2024. Hal ini juga mencerminkan bagaimana kuota prioritas lansia dapat memengaruhi keberangkatan jamaah dalam rentang waktu yang lebih cepat dibandingkan prediksi reguler.

Perhitungan ini dapat memberikan prediksi, namun tetap bersifat estimasi. Keberangkatan akhir calon jamaah dipengaruhi oleh kebijakan kuota, situasi global, atau perubahan kebijakan lainnya. Sistem seperti SISKOHAT perlu terus diperbarui dengan data terbaru untuk memastikan bahwa hasil prediksi tetap valid dan dapat digunakan sebagai acuan oleh calon jamaah haji dan pihak terkait di Kemenag.

3.4.9 Menghitung MAE dan RMSE

Setelah melakukan perhitungan untuk *hidden state* dan *prediksi*, langkah selanjutnya adalah menghitung metrik evaluasi model, seperti MAE dan RMSE. Kedua metrik ini digunakan untuk mengukur seberapa baik model memprediksi nilai yang sesuai dengan data aktual (Si & Mart, 2020).

a. MAE

Mengukur rata-rata selisih absolut antara nilai yang diprediksi dan nilai aktual. Rumus untuk menghitung MAE adalah sebagai berikut:

$$\frac{1}{n}\sum_{i=1}^{n} | y_{aktual}(i) - y_{prediksi}(i) | \quad (3.14)$$

Di mana:
a. $y_{aktual}(i)$, adalah nilai aktual untuk data ke-i.
b.$y_{prediksi}(i)$, adalah nilai yang diprediksi oleh model untuk data ke-i.
c.n, adalah jumlah data yang digunakan untuk evaluasi.

Contoh Perhitungan MAE:

Misalkan hasil prediksi dari model LSTM adalah sebagai berikut:

$$y_{prediksi}(1) = 2024, y_{prediksi}(2) = 2025, y_{prediksi}(3) = 2023,$$

Sedangkan nilai aktual (target) yang diketahui adalah:

$$y_{aktual}(1) = 2024, y_{aktual}(2) = 2024, y_{aktual}(3) = 2023,$$

Maka, MAE dapat dihitung sebagai berikut:

$$MAE = \frac{1}{3}(| 2024 - 2024 | + | 2025 - 2024 | + | 2023 - 2023 |)$$

$$MAE = \frac{1}{3}(0 + 1 + 0) = 31 = 0.33$$

Jadi, MAE untuk model ini adalah 0.33.

b. RMSE

RMSE mengukur akar kuadrat dari rata-rata kuadrat selisih antara nilai yang diprediksi dan nilai aktual. RMSE memberikan penalti yang lebih besar untuk kesalahan yang lebih besar dibandingkan MAE, karena kesalahan yang lebih besar dikuadratkan. Rumus untuk menghitung RMSE adalah sebagai berikut:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_{aktual}(i) - y_{prediksi}(i))^2} \quad (3.15)$$

Contoh Perhitungan RMSE dengan nilai yang sama seperti contoh MAE, maka RMSE dihitung sebagai berikut:

$$RMSE = \sqrt{\frac{1}{3}((2024 - 2024)^2 + (2025 - 2024)^2 + (2023 - 2023)^2)}$$

$$RMSE = \sqrt{\frac{1}{3}(0^2 + 1^2 + 0^2)} = \sqrt{\frac{1}{3}(1)} = \sqrt{0.33} = 0.577$$

Jadi, RMSE untuk model ini adalah sekitar 0.577.

BAB IV

HASIL DAN PEMBAHASAN

Bab ini menyajikan hasil penelitian yang diperoleh dari eksperimen yang telah dilakukan. Setiap tahapan dalam pemrosesan data, pembangunan model, serta evaluasi performa dijelaskan secara rinci untuk memastikan bahwa model yang dikembangkan memiliki tingkat akurasi yang optimal. Analisis dilakukan berdasarkan metrik evaluasi yang telah ditentukan sebelumnya.

4.1 *Preprocessing* Data

Pada langkah awal dilakukan beberapa tahap untuk mempersiapkan dataset sebelum digunakan dalam pelatihan model *Long Short Term Memory* (LSTM). Tahap pertama yaitu *preprocessing* yang meliputi pembersihan data, pemilihan fitur yang relevan, serta konversi format data agar sesuai dengan kebutuhan model. Selanjutnya, data numerik dinormalisasi untuk memastikan skala yang konsisten antar fitur. Selain itu, langkah-langkah seperti penghilangan duplikasi data dan transformasi kolom dilakukan untuk meningkatkan kualitas data dan memastikan data siap diproses oleh model *machine learning*. Semua tahapan ini bertujuan untuk memaksimalkan akurasi model dan meminimalkan kemungkinan kesalahan dalam prediksi (Bischl et al., 2024).

4.1.1 Pembersihan Data

Pembersihan data merupakan tahap awal yang sangat penting dalam analisis data (Sudarsono et al., 2021). Pada tahap ini melakukan pengecekan untuk

mengidentifikasi *missing value* atau data yang tidak lengkap, serta *outlier* yang dapat mempengaruhi hasil analisis. *Missing value* dapat terjadi karena berbagai alasan, seperti kesalahan input data atau kegagalan dalam proses pengumpulan data (Ningrum et al., 2021). Selain itu, dilakukan pengecekan format data seperti pada *pseudocode* dibawah ini untuk memastikan bahwa setiap kolom memiliki tipe data yang sesuai.

```
// 1. Cek Missing Values
SET missing_values = hitung jumlah nilai null di setiap kolom
pada df
RENAME kolom menjadi ["Kolom", "Jumlah Missing"]


// 2. Cek Format Data
SET data_types = ambil tipe data dari setiap kolom pada df
RENAME kolom menjadi ["Kolom", "Tipe Data"]
```

Berikut hasil dari tahapan pembersihan data pada Tabel 4.1 dalam melakukan pengecekan *missing value*.

Tabel 4.1 Hasil dari *missing value*

No	Kolom	Jumlah *Missing*
0	Porsi	0
1	Usia Daftar	0
2	Usia Berangkat	0
3	Masa Tunggu	0
4	Kuota	0
5	Bulan Daftar	0
6	Tahun Daftar	0
7	Prediksi Konvensional	0
8	Tahun Lunas Berangkat	0
9	Jangka Waktu Melunasi	0

Berikut hasil dari tahapan pembersihan data pada Tabel 4.2 dalam melakukan pengecekan format data.

Tabel 4.2 Hasil pengecekan format data

No	Kolom	Tipe Data
0	Porsi	Int64
1	Usia Daftar	Int64
2	Usia Berangkat	Int64
3	Masa Tunggu	Int64
4	Kuota	Int64
5	Bulan Daftar	Int64
6	Tahun Daftar	Int64
7	Prediksi Konvensional	Int64
8	Tahun Lunas Berangkat	Int64
9	Jangka Waktu Melunasi	Int64

4.1.2 Pemilihan Fitur

Pada tahap pemilihan fitur dalam proses *preprocessing*, merupakan tahap penting dalam analisis data yang bertujuan untuk mengurangi dimensi data dan meningkatkan kinerja model. Dalam proses ini, kita memilih fitur-fitur yang paling relevan dengan target yang ingin diprediksi. Dengan menggunakan kode *features* = ["Usia Daftar", "Usia Berangkat", "Masa Tunggu (Tahun)", "Kuota", "Bulan Daftar", "Tahun Daftar", "Prediksi Konvensional", "Tahun Lunas Berangkat"], memilih fitur-fitur yang dianggap paling berpengaruh terhadap variabel target, yaitu "Jangka Waktu Melunasi". Pemilihan fitur ini dilakukan berdasarkan pengetahuan domain dan analisis korelasi antara fitur-fitur tersebut dengan target. Dengan demikian, dengan *pseudocode* dibawah ini digunakan agar dapat memastikan bahwa model yang dibangun menggunakan fitur-fitur yang paling relevan, sehingga hasil prediksi menjadi lebih akurat dan reliabel.

```
// 1. Tentukan nama-nama fitur yang digunakan
SET features = ["Usia Daftar", "Usia Berangkat", "Masa Tunggu
(Tahun)",
                "Kuota", "Bulan Daftar", "Tahun Daftar",
```

```
                "Prediksi Konvensional", "Tahun Lunas
Berangkat"]

// 2. Tentukan kolom target (label)
SET target = "Jangka Waktu Melunasi"

// 3. Ambil data fitur dari dataframe
SET X = ambil kolom features dari df

// 4. Ambil data target dari dataframe
SET y = ambil kolom target dari df
```

Kode di atas memilih kolom-kolom tertentu sebagai fitur (X) dan target (Y). Setelah itu, data ditampilkan untuk memverifikasi bahwa pemilihan fitur telah dilakukan dengan benar. Proses ini memastikan bahwa hanya kolom-kolom yang relevan digunakan dalam pelatihan model, sehingga meningkatkan efisiensi dan akurasi prediksi.

Tabel 4.3 Pemilihan Fitur X

Usia Daftar	**Usia Berangkat**	**Masa Tunggu**	**Kuota**	**Bulan Daftar**	**Tahun Daftar**	**Prediksi Konvensional**	**Tahun Lunas Berangkat**
46	59	13	1100	1	2011	2024	2024
39	52	13	1100	5	2011	2024	2024
…	…	…	…	…	…	…	…
82	89	13	1100	12	2017	2030	2024
78	87	13	1100	12	2015	2028	2024

Pada tabel hasil pemilihan fitur di atas, terlihat bahwa dataset telah diproses untuk memilih kolom-kolom berikut sebagai fitur (X): "Usia Daftar", "Usia Berangkat", "Masa Tunggu (Tahun)", "Kuota", "Bulan Daftar", "Tahun Daftar", "Prediksi Konvensional", dan "Tahun Lunas Berangkat". Kolom "Jangka Waktu Melunasi" dipilih sebagai target (Y), yaitu variabel yang akan diprediksi oleh model. Tabel ini menampilkan beberapa contoh data urutan awal dan akhir dari dataset setelah pemilihan fitur dilakukan, memberikan gambaran bahwa dataset telah disiapkan dengan baik untuk proses pelatihan model.

4.1.3 Normalisasi Data

Normalisasi data merupakan tahap penting dalam *preprocessing* yang bertujuan untuk mengubah skala data menjadi rentang yang seragam. Dengan demikian, semua fitur memiliki kesempatan yang sama untuk berkontribusi dalam proses pelatihan model, sehingga mengurangi risiko bahwa fitur dengan skala besar mendominasi fitur lainnya. Sebelum melakukan normalisasi perlu mengetahui nilai terendah (minimum) dan tertinggi (maksimum) dari dataset yang digunakan. Nilai *Min* dan *Max* ini menjadi acuan dalam proses *Min-Max Scaling*, yaitu teknik normalisasi yang mengubah data menjadi rentang 0 hingga 1 (Roni Merdiansah et al., 2024).

Tabel 4.4 Nilai minimum dan maksimum setiap fitur sebelum Normalisasi

Fitur	Nilai Minimum	Nilai Maksimum
Usia Daftar	11	91
Usia Berangkat	0	96
Masa Tunggu (Tahun)	13	13
Kuota	0	1100
Bulan Daftar	1	12
Tahun Daftar	2008	2019
Prediksi Konvensional	2021	2032
Tahun Lunas Berangkat	0	2025

Sistem memperoleh nilai *Min* dan *Max* melalui proses *fit* pada objek *MinMaxScaler* dari *library scikit-learn*. Saat fungsi `fit()` dijalankan, *scaler* membaca seluruh data yang ada pada kolom fitur (variabel *input*) dan target (variabel *output*) untuk mencari nilai terkecil dan terbesar dari masing-masing kolom. Nilai *Min* dan *Max* pada *pseudocode* dibawah ini kemudian disimpan dalam atribut internal *scaler*, yaitu `scaler.data_min_` dan `scaler.data_max_`.

```
// 1. Buat objek MinMaxScaler
SET scaler = MinMaxScaler()
```

```
// 2. Lakukan transformasi fitur X agar nilainya berada dalam
rentang [0,1]
SET X_scaled = scaler.fit_transform(X)

// 3. Ubah hasil scaling menjadi DataFrame baru dengan kolom
sama seperti fitur awal
SET X_scaled_df = buat DataFrame dari X_scaled dengan nama kolom
= features
```

Proses ini tidak hanya memperbaiki kinerja model tetapi juga memudahkan interpretasi hasil. Dengan normalisasi, kita dapat memastikan bahwa model yang dibangun lebih stabil dan akurat dalam memprediksi variabel target.

Tabel 4.5 Nilai minimum dan maksimum setiap fitur setelah Normalisasi

Fitur	Nilai Minimum	Nilai Maksimum
Usia Daftar	0.0	1.0
Usia Berangkat	0.0	1.0
Masa Tunggu (Tahun)	0.0	0.0 *(konstan)*
Kuota	0.0	1.0
Bulan Daftar	0.0	1.0
Tahun Daftar	0.0	1.0
Prediksi Konvensional	0.0	1.0
Tahun Lunas Berangkat	0.0	1.0

Dalam *pseudocode* diatas, *MinMaxScaler* digunakan untuk mengubah nilai-nilai pada kolom "Usia Daftar", "Usia Berangkat", "Masa Tunggu (Tahun)", "Kuota", "Bulan Daftar", "Tahun Daftar", "Prediksi Konvensional", dan "Tahun Lunas Berangkat" menjadi rentang antara 0 dan 1. Metode `fit_transform(X)` dari `MinMaxScaler` di *Scikit-learn* adalah kombinasi dari dua langkah penting dalam pra-pemrosesan data `fit()` dan `transform()`. Langkah `fit()` menghitung nilai minimum dan maksimum dari setiap fitur dalam dataset `X`, yang kemudian disimpan sebagai parameter internal `data_min_` dan `data_max_`. Setelah parameter ini ditentukan, langkah `transform()` mengaplikasikan rumus normalisasi sesuai yang dijelaskan pada rumus 3.1 pada bab III.

Tabel 4.6 Hasil Normalisasi

Usia Daftar	Usia Berangkat	Masa Tunggu	Kuota	Bulan Daftar	Tahun Daftar	Prediksi Konvensional	Tahun Lunas Berangkat
0.4375	0.614583	0.0	1.0	0.000000	0.272727	0.272727	0.999506
0.3500	0.541667	0.0	1.0	0.363636	0.272727	0.272727	0.999506
……	……	……	……	……	……	……	……
0.9750	0.979167	0.0	1.0	0.909091	1.000000	1.000000	0.999506
0.9250	0.937500	0.0	1.0	0.909091	1.000000	1.000000	0.999506

Hasil tabel 4.6 menunjukkan perubahan nilai pada kolom. Setiap nilai pada kolom "Usia Daftar", "Usia Berangkat", "Masa Tunggu (Tahun)", "Kuota", "Bulan Daftar", "Tahun Daftar", "Prediksi Konvensional", dan "Tahun Lunas Berangkat" telah diubah menjadi rentang antara 0 dan 1. Hal ini terlihat dari nilai-nilai yang muncul, seperti 0.4375 untuk "Usia Daftar" dihasilkan dari proses normalisasi terhadap nilai asli 46.

$$Usia_{norm} = \frac{46 - 11}{91 - 11} = \frac{35}{80} = 0.4375$$

Dengan menggunakan `fit_transform()`, proses ini berlangsung secara otomatis berdasarkan nilai minimum dan maksimum yang telah dihitung sebelumnya melalui metode `fit()`. Proses ini membantu memastikan bahwa semua fitur memiliki pengaruh yang sama dalam model, sehingga meningkatkan kinerja dan stabilitas model. Dengan demikian, data menjadi lebih siap untuk digunakan dalam pelatihan model prediksi.

4.2 Pembagian Data untuk LSTM

Pada tahapan ini dataset dibagi menjadi dua bagian, data latih yang digunakan untuk melatih model dan data uji yang digunakan untuk mengevaluasi kinerja model. Pembagian data dilakukan seperti pada *pseudocode* dibawah ini di mana 80% data digunakan sebagai data latih dan 20% sisanya sebagai data uji. Pembagian dataset menjadi data latih dan data uji merupakan langkah penting dalam mempersiapkan data untuk pelatihan model LSTM. Dataset yang terdiri dari 1132 data dibagi menjadi dua bagian, data latih (*training* data) dan data uji (*testing* data).

```
// 1. Hitung jumlah total data
SET jumlah_data = panjang dari X_scaled_df

// 2. Hitung batas index untuk 80% data (data training)
SET batas_index = 80% dari jumlah_data

// 3. Ambil data training dari baris 0 sampai batas_index
SET X_train = baris ke-0 sampai batas_index dari X_scaled_df
RESET index X_train

SET y_train = baris ke-0 sampai batas_index dari y
RESET index y_train

// 4. Ambil data testing dari baris batas_index sampai akhir
SET X_test = baris dari batas_index sampai akhir dari
X_scaled_df
RESET index X_test

SET y_test = baris dari batas_index sampai akhir dari y
RESET index y_test
```

Tabel 4.7 Hasil Pembagian Dataset

Keterangan	Jumlah
Jumlah total data	1132
Jumlah data latih (80%)	905
Jumlah data uji (20%)	227

Data latih memungkinkan model untuk mempelajari pola dan hubungan dalam data, sedangkan data uji memberikan gambaran tentang seberapa baik model dapat melakukan prediksi pada data yang belum pernah dilihat sebelumnya.

Pembagian ini penting untuk menghindari masalah *overfitting*, di mana model hanya bekerja dengan baik pada data yang sudah dikenal tetapi gagal memberikan hasil yang akurat pada data baru (Fatunnisa & Marcos, 2024).

4.3 Hasil *Windowing*

Setelah data dinormalisasi, proses *windowing* diterapkan untuk menyusun ulang data menjadi format sekuensial yang sesuai dengan kebutuhan model LSTM. *Windowing* dilakukan dengan panjang jendela sebesar 10, artinya setiap satu baris data hasil *windowing* disusun dari 10 baris data sebelumnya. Proses ini menghasilkan struktur data tiga dimensi, di mana setiap data memiliki bentuk (jumlah *window*) baris, (panjang urutan waktu) kolom waktu, dan (jumlah fitur) kolom variabel.

```
// 1. Definisikan fungsi create_windows dengan parameter data
dan window_size
FUNCTION create_windows(data, window_size):

// 2. Buat list berisi potongan data sepanjang window_size
RETURN array dari:
UNTUK i dari 0 hingga panjang data - window_size + 1:
AMBIL data dari indeks i sampai i + window_size

// 3. Tetapkan ukuran jendela (window) yang digunakan
SET window_size = 10
```

Fungsi pada *pseudocode* diatas membentuk susunan *window* dari data normalisasi dengan cara mengambil 10 baris berturut-turut untuk setiap *window*. *Window* pertama terbentuk dari baris ke-1 hingga ke-10, *window* kedua dari baris ke-2 hingga ke-11, dan seterusnya, dengan langkah geser satu baris. Dari proses ini, diperoleh 896 sampel *window* untuk data pelatihan (*X_train*) dan 218 sampel *window* untuk data pengujian (*X_test*).

Tabel 4.8 Hasil *Windowing* – Data *Training*

No	Usia Daftar_t-10	Usia Berangkat_t-10	Kuota_t-10	Tahun Daftar_t-10	Tahun Lunas_t-1	Target
1	0.4375	0.4861	1	0.2727	0.5	0.7273
2	0.3500	0.3889	1	0.2727	0.5	0.7273
3	0.5500	0.6111	1	0.2727	0.5	0.7273
4	0.1625	0.1944	1	0.2727	0.5	0.7273
5	0.2375	0.2778	1	0.2727	0.5	0.7273

Tabel 4.8 menampilkan lima contoh hasil *windowing* dari *X_train*, terdiri dari tiga baris pertama dan dua baris terakhir. Setiap baris mewakili satu jendela yang menyusun data historis dari 10 waktu sebelumnya untuk masing-masing fitur yang digunakan dalam model.

Tabel 4.9 Hasil *Windowing* – Data *Testing*

No	Usia Daftar_t-10	Usia Berangkat_t-10	Kuota_t-10	Tahun Daftar_t-10	Tahun Lunas_t-1	Target
1	0.2000	0.2222	1	0.2727	0.5	0.7273
2	0.1875	0.2083	1	0.2727	0.5	0.7273
3	0.1250	0.1389	1	0.2727	0.5	0.7273
4	0.2625	0.2917	1	0.2727	0.5	0.7273
5	0.1750	0.1944	1	0.2727	0.5	0.7273

Tabel 4.9 menampilkan lima data awal dari hasil *windowing X_test*. Struktur dan fitur yang digunakan sama dengan data pelatihan, dan digunakan sebagai *input* evaluasi model saat pengujian. Untuk memperjelas bagaimana proses *windowing* bekerja terhadap satu fitur, Tabel 4.10 menyajikan contoh visualisasi hasil *windowing* untuk fitur *Usia Daftar* dari satu baris hasil *windowing* di data pelatihan. Dari tabel 4.10 terlihat bahwa nilai usia calon jamaah terekam dari waktu ke-10 hingga ke-1 sebelum target diprediksi.

Tabel 4.10 *Windowing* pada Fitur Usia Daftar (*X_train*)

Urutan Waktu	Nilai Normalisasi
Usia Daftar_t-10	0.4375
Usia Daftar_t-9	0.3500
Usia Daftar_t-8	0.5500
Usia Daftar_t-7	0.5625
Usia Daftar_t-6	0.3625

Dari susunan nilai pada Tabel 4.10, terlihat bahwa setiap *window* menyimpan riwayat 10 waktu ke belakang untuk satu fitur secara berurutan. Setiap nilai diambil langsung dari data asli yang telah dinormalisasi dan disusun sesuai urutan barisnya. Misalnya, lima nilai dalam *window* tersebut berasal dari baris ke-1 hingga ke-5 pada data awal. *Window* kedua akan mengambil baris ke-2 hingga ke-6, dan seterusnya secara bergeser satu langkah. Karena model membutuhkan 10 riwayat waktu sebelumnya untuk membentuk satu *input*, maka data awal sebanyak 905 baris akan menghasilkan 896 jendela pada data pelatihan. Secara matematis, hal ini dihitung dengan mengurangkan panjang *window* dari jumlah total data, kemudian menambahkan 1.

$$\mathrm{J} = 905 - 10 + 1 = 896$$

Proses ini menciptakan susunan *window* yang saling beririsan (*overlapping*), di mana setiap baris *window* memerlukan 10 baris historis, sehingga 9 baris pertama belum memenuhi syarat untuk membentuk satu *window*. Penurunan jumlah baris inilah yang menjadi konsekuensi dari pembentukan *window* berurutan, dan juga berlaku untuk data testing dengan pola yang sama (Wang et al., 2022).

Sebelum membangun model LSTM, data yang telah dihasilkan dari proses *windowing* perlu diubah ke dalam format yang sesuai dengan *input* model LSTM. Pasa *pseudocode* dibawah ini proses ini dilakukan dengan *mereshape* data agar dapat diterima oleh model LSTM yang membutuhkan data dalam bentuk tiga dimensi yaitu, jumlah *window*, panjang urutan waktu, jumlah fitur. Dengan `window_size` = 10, dan `num_features` yang dihitung berdasarkan jumlah fitur

pada data pelatihan, data *X_train* dan *X_test* diubah bentuknya menggunakan fungsi *reshape*.

```
// 1. Definisikan fungsi create_windows dengan parameter data
dan window_size
FUNCTION create_windows(data, window_size):

// 2. Buat list berisi potongan data sepanjang window_size
RETURN array dari:
UNTUK i dari 0 hingga panjang data - window_size + 1:
AMBIL data dari indeks i sampai i + window_size

// 3. Tetapkan ukuran jendela (window) yang digunakan
SET window_size = 10
```

Perubahan bentuk data ini memungkinkan setiap sampel terdiri dari `window_size` langkah waktu yang berurutan, dengan `num_features` sebagai jumlah kolom fitur yang diolah. Setelah dilakukan *reshape*, data akan siap digunakan sebagai *input* untuk model LSTM.

4.4 Membangun dan Melatih Model LSTM

Pembangunan dan pelatihan model LSTM merupakan tahap penting dalam memprediksi tahun keberangkatan jamaah haji. Pada tahap ini, model dilatih menggunakan data yang telah diproses dan dipersiapkan melalui *windowing*, untuk mengenali pola historis yang relevan. Arsitektur model LSTM dirancang untuk menangani data sekuensial dengan beberapa lapisan LSTM yang memungkinkan model memahami hubungan temporal dalam data.

4.4.1 Arsitektur Model LSTM

Arsitektur model LSTM yang dibangun untuk memprediksi tahun berangkat jamaah haji terdiri dari beberapa lapisan LSTM yang dilengkapi dengan teknik-

teknik untuk mengurangi *overfitting*. Pada arsitektur ini, tiga lapisan LSTM digunakan dengan konfigurasi sebagai berikut:

1. Lapisan LSTM Pertama, Lapisan ini memiliki 128 unit LSTM dan menggunakan `return_sequences = True` untuk memungkinkan output dari setiap *time-step* diproses lebih lanjut oleh lapisan berikutnya (Ramadhan & Nathasia, 2025). Regularisasi L2 diterapkan untuk membatasi kompleksitas model dan mencegah *overfitting*. `input_shape=input_shape` di sini berfungsi untuk menetapkan dimensi input yang akan diterima oleh lapisan pertama LSTM. Dimensi *input* ini didapat dari data pelatihan, yaitu ukuran dari urutan waktu dan jumlah fitur yang diberikan ke model. Lapisan ini juga dilengkapi dengan *dropout* dan *batch normalization* untuk mengurangi *overfitting* (Hanafiah et al., 2023).

2. Lapisan LSTM Kedua, Lapisan kedua memiliki 64 unit LSTM dan juga menggunakan `return_sequences = True` untuk meneruskan output urutan ke lapisan ketiga (Ramadhan & Nathasia, 2025). Pengurangan jumlah unit ini bertujuan untuk menyaring informasi yang relevan dan mengurangi kompleksitas model.
3. Lapisan LSTM Ketiga, Lapisan kedua memiliki 32 unit LSTM dan juga menggunakan `return_sequences = False`, yang berarti hanya *output* terakhir yang digunakan sebagai input untuk lapisan *Dense* (Ramadhan & Nathasia, 2025).
4. Lapisan *Dense*, terdapat lapisan `Dense(16)` yang berfungsi untuk mengubah hasil dari lapisan LSTM sebelumnya menjadi representasi yang lebih kompleks

sebelum menghasilkan output akhir. 16 neuron di lapisan ini memberikan model kemampuan untuk belajar representasi lebih kaya dari data yang diterima dari lapisan LSTM. Fungsi aktivasi *tanh* digunakan di lapisan ini untuk memberikan kemampuan model dalam memproses nilai yang beragam, dengan rentang *output* antara -1 hingga 1 (Essai Ali et al., 2022). Lapisan *output* terdiri dari satu neuron yang digunakan untuk memprediksi tahun keberangkatan (nilai kontinu). Lapisan *output* `Dense(1)` ini tanpa fungsi aktivasi atau menggunakan aktivasi linear secara *default*, agar model dapat menghasilkan *output* prediksi yang tepat tanpa batasan.

```
// 1. Definisikan fungsi untuk membangun model LSTM dengan
parameter input_shape, dropout, l2_reg, dan learning_rate
FUNCTION build_lstm_model(input_shape, dropout=0.2,
l2_reg=0.001, learning_rate=0.001):

    // 2. Bangun model dengan urutan lapisan LSTM, Dropout,
BatchNormalization, dan Dense
    model = Sequential([
        LSTM(128, return_sequences=True, activation='tanh',
kernel_regularizer=l2(l2_reg), input_shape=input_shape),
        Dropout(dropout),
        BatchNormalization(),

        LSTM(64, return_sequences=True, activation='tanh',
kernel_regularizer=l2(l2_reg)),
        Dropout(dropout),

        LSTM(32, return_sequences=False, activation='tanh',
kernel_regularizer=l2(l2_reg)),
        Dropout(dropout),

        Dense(16, activation='tanh',
kernel_regularizer=l2(l2_reg)),
        Dense(1)
    ])

    model.compile(optimizer=Adam(learning_rate=learning_rate),
                  loss='mse',
                  metrics=['mae', custom_accuracy])

    RETURN model
```

Pada *pseudocode* diatas, proses pembangunan model LSTM dimulai dengan mendefinisikan lapisan-lapisan yang akan membentuk model. Model LSTM terdiri dari tiga lapisan LSTM yang masing-masing dilengkapi dengan teknik untuk mengurangi overfitting, seperti *Dropout* dan *Batch Normalization*. Selain itu, regularisasi L2 diterapkan pada setiap lapisan LSTM untuk membatasi kompleksitas model. Fungsi `compile` digunakan untuk mengonfigurasi model dengan menggunakan *optimizer Adam*, yang mengatur pembelajaran model, dan *loss function Mean Squared Error* (MSE) untuk mengukur seberapa akurat prediksi model terhadap nilai target.

```
SET input_shape = (X_train.shape[1], X_train.shape[2])
model = build_lstm_model(input_shape)
```

Pada *pseudocode* diatas, menunjukkan bagaimana parameter `input_shape` diatur berdasarkan data pelatihan. Nilai `input_shape` ini adalah dimensi data input yang diperlukan untuk lapisan pertama LSTM. Selanjutnya, fungsi `build_lstm_model` dipanggil untuk membuat model berdasarkan parameter yang telah ditentukan.

```
SET model_save_path = path_to_save_directory +
"/best_lstm_model_with_accuracy.h5"
SET callbacks = [
    EarlyStopping(monitor='loss', patience=10,
restore_best_weights=True),
    ModelCheckpoint(model_save_path, monitor='loss',
save_best_only=True, verbose=1)
]
```

Pada *pseudocode* diatas, digunakan *callbacks* untuk mengontrol jalannya pelatihan. *EarlyStopping* memastikan pelatihan berhenti ketika model tidak lagi menunjukkan peningkatan yang signifikan, dengan `patience=10` yang berarti

model akan menunggu selama 10 iterasi tanpa perbaikan sebelum pelatihan dihentikan. *ModelCheckpoint* menyimpan model terbaik yang diperoleh selama pelatihan untuk memastikan model yang optimal disimpan.

Tabel 4.11 Model Skuensial LSTM

Layer (type)	Output Shape	Param #
Lstm (LSTM)	(None, 10, 128)	70,144
Dropout (Dropout)	(None, 10, 128)	0
Batch_normalization (BatchNormalization)	(None, 10, 128)	512
lstm_1 (LSTM)	(None, 10, 64)	49,408
dropout_1 (Dropout)	(None, 10, 64)	0
lstm_2 (LSTM)	(None, 32)	12,416
dropout_2 (Dropout)	(None, 32)	0
dense (Dense)	(None, 16)	528
Dense_1 (Dense)	(None, 1)	17

Dalam layer LSTM, *output* biasanya memiliki tiga dimensi yaitu *Batch Size* (dalam hal ini *None*, yang berarti tidak terbatas), *Timestep* (jumlah langkah waktu yang mewakili urutan data, misalnya 10 *timestep*), dan *Unit output* (jumlah unit neuron pada setiap *timestep*, misalnya 128 unit). LSTM pertama ini memiliki *output* dengan bentuk (*None*, 10, 128), yang berarti 10 *timestep* dan 128 unit *output* per *timestep*. Ketika menggunakan *Batch Normalization*, dua parameter ditambahkan untuk setiap unit *output* di setiap *timestep* yaitu *Gamma* (untuk skala) dan *Beta* (untuuk pergeseran) (Peerthum & Stamp, 2023). Dengan demikian, *Batch Normalization* memiliki dua parameter untuk setiap unit *output* di setiap *timestep*. Untuk menghitung jumlah parameter pada LSTM pertama, rumus yang digunakan adalah:

$$Jumlah\ Parameter = 4\ ((\text{Jumlah Input} + \text{Jumlah unit})\ .\ \text{Jumlah unit} + \text{Jumlah unit}) \quad (4.1)$$

Dalam hal ini, jumlah *input* adalah 10 dan jumlah unit adalah 128. Sehingga perhitungannya menjadi:

$$4((10 + 128) \times 128 + 128 = 70.144\ parameter$$

Selanjutnya, *layer dropout* tidak memiliki parameter yang dapat dilatih, sehingga jumlah parameter untuk *layer* ini adalah 0. *Layer Batch Normalization* ditambahkan setelah LSTM pertama, dengan 128 unit *output* dan 10 *timestep*. Pada *layer* ini, 2 parameter (*Gamma* dan *Beta*) diterapkan untuk setiap unit *output* pada setiap *timestep*, sehingga perhitungannya adalah:

$$Jumlah\ parameter\ per\ timestep = 2 \times 128 = 256\ \ parameter\ per\ timestep$$

Karena ada 10 *timestep*, maka jumlah total parameter *Batch Normalization* untuk seluruh *timestep* adalah:

$$256 \times 2 = 512\ parameter$$

Untuk LSTM kedua yang memiliki *output* dengan 64 unit, jumlah parameter dihitung menggunakan rumus yang sama dengan LSTM pertama, namun kali ini dengan jumlah unit 64. Perhitungannya menjadi:

$$4((128 + 64) \times 64 + 64) = 49408\ parameter$$

Pada LSTM ketiga, yang memiliki 32 unit *output*, perhitungannya adalah:

$$4((64 + 32) \times 32 + 32) = 12416\ parameter$$

Layer dropout setelah setiap LSTM tidak memiliki parameter yang dapat dilatih, sehingga jumlah parameter untuk masing-masing adalah 0. Setelah itu, *layer Dense* pertama yang menghubungkan 32 unit dengan 16 *output* menghasilkan:

$$(32 + 1)16 = 12416\ parameter$$

Sedangkan *Dense* kedua untuk menghasilkan *output* dengan 1 unit menghitung parameter sebagai berikut:

$$(32 + 1)16 = 12416\ parameter$$

LSTM pertama dengan *input size* 8 dan *output size* 128 menghasilkan 70,144 parameter, LSTM kedua dengan *input size* 128 dan *output size* 64 menghasilkan 49,408 parameter, dan LSTM ketiga dengan *input size* 64 dan *output size* 32 menghasilkan 12,416 parameter. *Layer Dropout* tidak memiliki parameter yang dapat dilatih, *sementara Batch Normalization* menambah 512 parameter untuk mengatur skala dan pergeseran pada setiap unit *output*. *Layer Dense* pertama dengan *input size* 32 dan *output size* 16 menambah 528 parameter, dan *Dense* kedua dengan *input size* 16 dan *output size* 1 menghasilkan 17 parameter. Dengan demikian, jumlah total parameter dalam model ini adalah 133,025, terdiri dari 132,769 parameter yang dapat dilatih dan 256 parameter yang tidak dapat dilatih.

4.4.2 Konfigurasi *Hyperparameter*

Konfigurasi hyperparameter merupakan langkah krusial dalam memastikan kinerja model LSTM yang optimal. Untuk *hyperparameter tuning* pada model LSTM, terdapat dua parameter penting yang sering dieksplorasi dalam eksperimen yaitu *epochs* dan *batch size*. *Epochs* antara 100 hingga 300 dan *batch size* 32 atau 16 umum digunakan pada dataset kecil hingga menengah (Ayu et al., 2024)(Santosa et al., 2024). Nilai *dropout* = 0.2 digunakan untuk mengurangi *overfitting* dengan menonaktifkan 20% neuron pada setiap *batch*. L2 *regularization* dengan `l2_reg = 0.001` membantu mencegah *overfitting* dengan menambahkan penalti pada bobot besar. *Learning rate* = 0.001 adalah nilai standar yang sering digunakan dalam optimisasi Adam untuk mencapai konvergensi yang stabil (Peerthum & Stamp, 2023). Selanjutnya, model LSTM dikompilasi menggunakan *TensorFlow* dengan *optimizer Adam* dan fungsi *loss* MSE. *Optimizer* Adam dipilih karena mampu

menyesuaikan laju pembelajaran secara adaptif, sedangkan fungsi *loss* MSE digunakan untuk mengukur kesalahan prediksi model (Pipin et al., 2023).

4.4.3 Proses Pelatihan Model

Proses pada *pseudocode* dibawah ini, pelatihan model dimulai dengan mencatat waktu *start time*. Lalu, LSTM *layer* pertama dimodifikasi dengan `return_state = True` untuk mendapatkan *hidden state* dan *cell state*. Model kemudian dilatih menggunakan data *X_train* dan *y_train* dengan jumlah *epochs* antara 100 hingga 300 dengan *batch size* anatara 16 dan 32. Setelah pelatihan selesai, waktu end time dicatat, dan total waktu pelatihan dihitung. Selanjutnya, jumlah *epoch* yang digunakan diambil berdasarkan panjang history *loss* yang tercatat dalam pelatihan.

```
// 1. Mencatat waktu mulai pelatihan
START time = current time

// 2. Modifikasi LSTM layer pertama untuk mendapatkan hidden
state dan cell state
SET model.layers[0].return_state = True

// 3. Melatih model dengan data training
CALL model.fit() WITH parameters:
    - X_train
    - y_train
    - epochs = 100    // Ubah jumlah epoch sesuai dengan
skenario
    - batch_size = 16  // Ubah batch size sesuai dengan skenario
    - callbacks = callbacks
    - verbose = 1

// 4. Mencatat waktu selesai pelatihan
END time = current time

// 5. Menghitung total waktu pelatihan
SET training_time = END time - START time

// 6. Mengambil jumlah epoch yang digunakan selama pelatihan
SET actual_epochs = LENGTH of history.history['loss']
```

Setelah proses pelatihan selesai, langkah selanjutnya adalah melakukan pengujian skenario untuk mengevaluasi kinerja model dengan parameter yang telah disesuaikan, seperti *epochs* dan *batch size*. Pengujian ini akan memberikan wawasan lebih lanjut tentang efektivitas model dalam menangani data dan pengaruh variasi hyperparameter terhadap performa model. Model terbaik disimpan dalam format `.h5` untuk memudahkan pemanggilan kembali jika diperlukan.

4.5 Pengujian Model LSTM

Pada bab ini, dilakukan pengujian terhadap model LSTM yang telah dilatih sebelumnya untuk mengevaluasi performa model berdasarkan berbagai skenario. Pengujian ini bertujuan untuk mengukur efektivitas dan kemampuan model dalam menangani data dengan berbagai konfigurasi *hyperparameter*, seperti variasi pada epochs dan *batch size*, sebanyak 6 skenario pengujian. Setiap skenario dirancang untuk menilai pengaruh perubahan konfigurasi terhadap kinerja model, termasuk akurasi, kecepatan pelatihan, dan generalisasi model pada data yang digunakan. Dari *pseudocode* dibawah, hasil dari pengujian ini diharapkan dapat memberikan wawasan tentang bagaimana berbagai faktor mempengaruhi performa model LSTM dalam konteks aplikasi yang sedang dianalisis.

```
// 1. Prediksi pada data uji
PREDICT y_pred USING model.predict(X_test)

// 2. Menghitung MAE dan RMSE
CALCULATE mae = mean_absolute_error(y_test, y_pred)
CALCULATE rmse = sqrt(mean_squared_error(y_test, y_pred))

// 3. Menghitung Accuracy ±1 Tahun jika kolom "Tahun Daftar" ada
dalam data uji
tahun_col = FIND columns in test_df that contain "Tahun Daftar"
IF tahun_col IS FOUND:
    tahun_daftar = test_df[tahun_col[-1]].values
    y_pred_tahun = flatten(y_pred) + tahun_daftar
    y_true_tahun = y_test + tahun_daftar
    toleransi = 1
    akurasi_tahun = mean(abs(y_pred_tahun - y_true_tahun) <=
toleransi)
    PRINT "Akurasi Prediksi Tahun Keberangkatan (±{toleransi}
tahun): {akurasi_tahun:.2%}"
ELSE:
    akurasi_tahun = None
    PRINT "Kolom Tahun Daftar_t-1 tidak ditemukan."

// 4. Menampilkan evaluasi model pada data uji
PRINT "Evaluasi Model pada Data Uji:"
PRINT "MAE: {mae:.4f}"
PRINT "RMSE: {rmse:.4f}"

// 5. Menyimpan hasil prediksi pada file Excel
COPY test_df TO hasil_df
ADD "Prediksi_LSTM" column to hasil_df with values of y_pred
IF tahun_col IS FOUND:
    ADD "Prediksi_LSTM (Tahun)" column to hasil_df with values
of "Prediksi_LSTM" + tahun_daftar

DISPLAY the first 20 rows of hasil_df

// 6. Menyimpan hasil prediksi ke file Excel
SET output_path = save_dir +
"/hasil_prediksi_lstm_terpisah.xlsx"
SAVE hasil_df TO output_path
PRINT "Hasil prediksi disimpan di: {output_path}"

// 7. Menyimpan model pelatihan
SAVE model TO model_save_path
PRINT "Model disimpan di: {model_save_path}"
```

Model yang telah dilatih digunakan untuk memprediksi hasil berdasarkan data uji *X_test*, dan hasil prediksi disimpan dalam variabel *y_pred*. Setelah itu, dilakukan perhitungan MAE dan RMSE antara nilai prediksi dan nilai aktual untuk

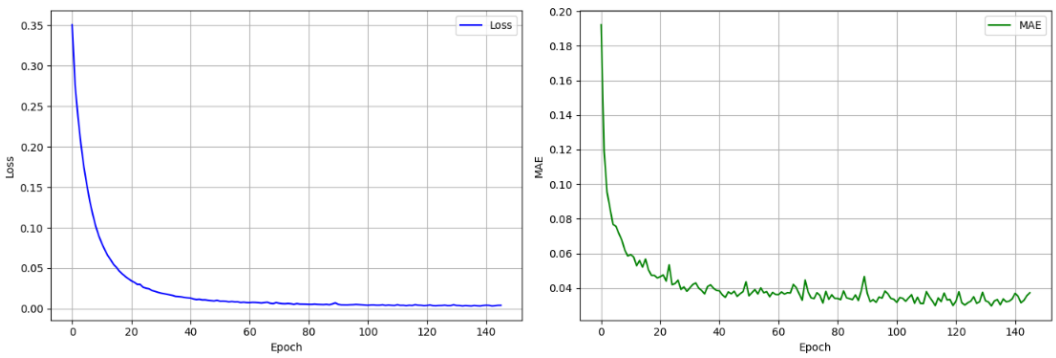
mengukur seberapa besar kesalahan prediksi. Jika terdapat kolom Tahun Daftar, dilakukan evaluasi akurasi prediksi tahun keberangkatan dengan rentang kesalahan ±1 tahun. Hasil evaluasi MAE dan RMSE dicetak untuk menilai performa model. Selanjutnya, hasil prediksi disalin ke dalam dataframe `hasil_df` dan disimpan dalam file *Excel* `hasil_prediksi_lstm_terpisah.xlsx`, termasuk prediksi model yang dikombinasikan dengan tahun pendaftaran jika ada. Model yang telah dilatih dan dievaluasi juga disimpan di direktori yang telah ditentukan dengan `best_lstm_model_with_accuracy.h5`.

4.5.1 Skenario Pengujian 1

Pada pengujian skenario 1 ini dengan *pseudocode* dibawah, model dilatih dengan *300 epochs* dan *batch size 32* untuk mengevaluasi performanya. Selama pelatihan, dilakukan penyimpanan model terbaik setiap kali ada perbaikan pada fungsi loss, yang diukur dengan *custom_accuracy* dan MAE. Proses pelatihan memanfaatkan callback untuk menyimpan model dengan akurasi terbaik, serta menggunakan fungsi `verbose=1` untuk memantau kemajuan pelatihan pada setiap *epoch*.

```
// 1. Mencatat waktu mulai pelatihan
START time = current time

// 2. Menambahkan return_state pada LSTM pertama
SET model.layers[0].return_state = True

// 3. Melatih model dengan data training
CALL model.fit() WITH parameters:
    - X_train
    - y_train
    - epochs = 300    // Pengaturan untuk skenario
    - batch_size = 32  // Pengaturan untuk skenario
    - callbacks = callbacks
    - verbose = 1

// 4. Mencatat waktu selesai pelatihan
```

```
END time = current time

// 5. Menghitung total waktu pelatihan
SET training_time = END time - START time

// 6. Menampilkan jumlah epoch yang digunakan
SET actual_epochs = LENGTH of history.history['loss']
```



Gambar 4.1 Grafik Evaluasi *Training Loss* dan MAE Skenario 1

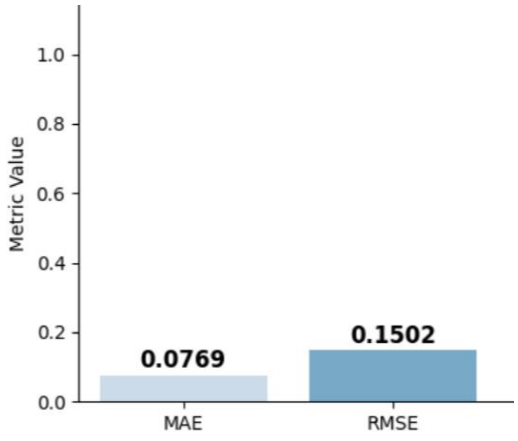
Pada Gambar 4.1, yang berwarna biru merupakan grafik *Training Loss*, grafik ini menunjukkan penurunan signifikan pada nilai *loss* model seiring berjalannya waktu pelatihan. Pada awalnya, pada epoch pertama, nilai *loss* dimulai dengan angka 0.35, yang menurun secara drastis setelah beberapa *epoch* pertama. Hingga *epoch* ke-100, nilai *loss* telah turun menjadi 0.00445, dan terus berlanjut menurun hingga mencapai 0.00386 pada *epoch* ke-114. Pelatihan hanya berlangsung hingga *epoch* ke-114 karena *early stopping* yang diaktifkan atau model mencapai konvergensi lebih cepat dari yang diperkirakan. *Early stopping* adalah metode untuk menghentikan pelatihan saat model tidak lagi menunjukkan peningkatan yang signifikan pada metrik evaluasi (seperti *loss*). Pada kasus ini, meskipun jumlah *epoch* yang ditargetkan adalah 300, model sudah mencapai titik dimana *loss* tidak lagi berkurang mengindikasikan bahwa model mampu

meminimalkan kesalahan memperbaiki prediksi seiring dengan bertambahnya *epoch*. Sedangkan yang berwarna hijau merpakan grafik *Training* MAE menunjukkan perubahan nilai MAE selama proses pelatihan. Grafik ini menggambarkan penurunan signifikan pada MAE yang dimulai dengan nilai yang tinggi pada awal pelatihan (sekitar 0.18) dan menurun tajam hingga hampir 0.04 pada *epoch* ke-100. Setelah *epoch* ke-100, MAE mengalami fluktuasi kecil namun tetap berada pada nilai yang sangat rendah, mengindikasikan kestabilan dalam prediksi model yang terus meningkat selama pelatihan.

Tabel 4.12 Sample Hasil Prediksi LSTM Skenario 1

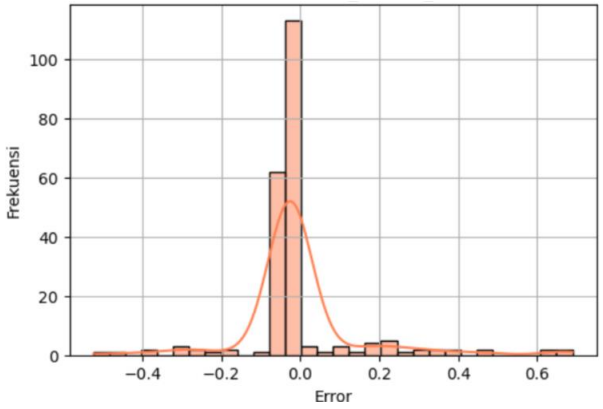
Target (*Normalized*)	**Prediksi LSTM**	**Prediksi (Tahun)**	**Tahun Aktual**	***Error***	***Error* (Tahun)**
0.727273	0.725008	0.997735	0.5	0.002265	-0.497735
0.727273	0.726967	0.999694	0.5	0.000306	-0.499694
0.727273	0.728066	1.000793	0.5	-0.000793	-0.500793
0.727273	0.729616	1.002343	0.5	-0.002343	-0.502343
0.727273	0.730364	1.003091	0.5	-0.003091	-0.503091

Tabel 4.12 hasil prediksi LSTM masih sangat dekat dengan target aktual, dengan kesalahan prediksi (*error*) dalam satuan tahun yang sangat kecil, kurang dari ±0.51 tahun. Selisih ini tetap masuk dalam rentang akurasi ±1 tahun, sehingga tetap dikategorikan sebagai prediksi yang akurat.



Gambar 4.2 Grafik Evaluasi Model Pengujian Skenario 1

Pada Gambar 4.2 menunjukkan garfik evaluasi akhir model (*Testing* Data), dapat dilihat hasil evaluasi yang menggambarkan nilai MAE dan RMSE untuk data uji. Nilai MAE sebesar 0.0769 dan RMSE sebesar 0.1502 menunjukkan kesalahan model yang kecil.



Gambar 4.3 Grafik Distribusi *Error* Skenario 1

Gambar 4.3 grafik distribusi *error* menunjukkan distribusi selisih antara prediksi model dan nilai aktual. Sebagian besar kesalahan (*error*) terkonsentrasi di sekitar 0, yang menunjukkan bahwa model menghasilkan prediksi yang sangat mendekati nilai aktual. Puncak yang tajam pada nilai 0 menunjukkan bahwa sebagian besar prediksi model cukup akurat, sementara distribusi yang lebih lebar di sekitar nilai 0,2 hingga -0.2 menunjukkan adanya beberapa prediksi yang sedikit meleset.

4.5.2 Skenario Pengujian 2

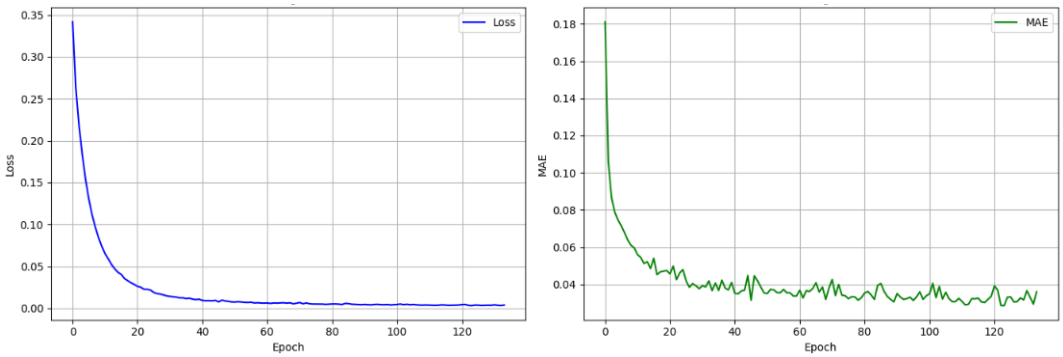
Pada skenario pengujian 2 ini dengan *pseudocode* dibawah, dilakukan proses pelatihan model LSTM menggunakan parameter utama yaitu jumlah *epoch* sebanyak 200 dan ukuran *batch* (*batch size*) sebesar 32. Proses pelatihan dimulai

dengan mencatat waktu awal, lalu model dilatih dengan data *X_train* dan *y_train* yang sudah dipersiapkan sebelumnya.

```
// 1. Mencatat waktu mulai pelatihan
START time = current time

// 2. Menambahkan return_state pada LSTM pertama
SET model.layers[0].return_state = True

// 3. Melatih model dengan data training
CALL model.fit() WITH parameters:
    - X_train
    - y_train
    - epochs = 200    // Pengaturan untuk skenario
    - batch_size = 32  // Pengaturan untuk skenario
    - callbacks = callbacks
    - verbose = 1

// 4. Mencatat waktu selesai pelatihan
END time = current time

// 5. Menghitung total waktu pelatihan
SET training_time = END time - START time

// 6. Menampilkan jumlah epoch yang digunakan
SET actual_epochs = LENGTH of history.history['loss']
```

Selama pelatihan berlangsung, digunakan *callback* `ModelCheckpoint` untuk menyimpan bobot terbaik berdasarkan nilai *loss* terkecil yang dicapai pada setiap *epoch*. Meskipun ditargetkan untuk melatih hingga 200 *epoch*, pelatihan berhenti pada *epoch* ke-134 karena model tidak lagi mengalami perbaikan signifikan pada metrik evaluasi, menandakan bahwa model telah mencapai kondisi optimal lebih awal. Total waktu pelatihan yang dibutuhkan adalah 82.08 detik, yang menunjukkan efisiensi proses training dalam konteks model *deep learning* berskala sedang. Secara teknis, batch size 32 berarti model memperbarui bobotnya setelah melihat 32 sampel data. Ini memberikan keseimbangan antara akurasi dan kecepatan komputasi karena tidak terlalu kecil seperti *stochastic* (*batch size* = 1) maupun terlalu besar seperti *full-batch*. Sementara itu, *epoch* adalah iterasi penuh

terhadap seluruh dataset. Dari hasil pelatihan yang tercatat, terlihat bahwa nilai *loss* menurun tajam dari *epoch* pertama ke *epoch* 10, yaitu dari 0.4188 ke sekitar 0.0759, yang menunjukkan proses pembelajaran awal berlangsung sangat efektif. Penurunan *loss* secara konsisten hingga ke angka 0.00330 pada *epoch* ke-124, kemudian stagnan atau tidak mengalami perbaikan signifikan hingga *epoch* ke-134. Artinya, model telah belajar optimal pada titik tersebut, dan pelatihan bisa dihentikan untuk mencegah *overfitting*.



Gambar 4.4 Grafik Evaluasi *Training Loss* dan MAE Skenario 2

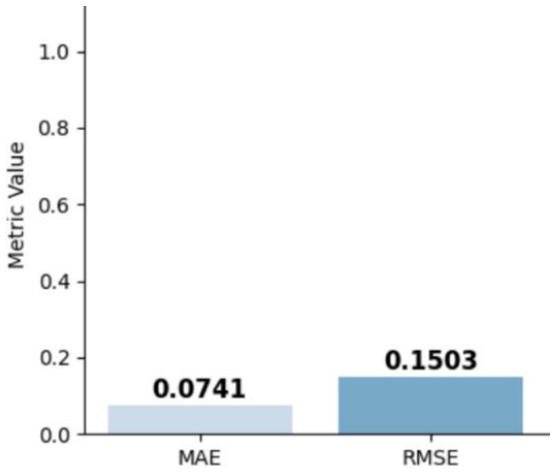
Pada Gambar 4.4 sebelah kiri (berwarna biru) menunjukkan grafik penurunan nilai *loss* yang sangat tajam di awal pelatihan, dari sekitar 0.35 menjadi kurang dari 0.01 setelah lebih dari 100 *epoch*, yang mengindikasikan model semakin mampu meminimalkan kesalahan prediksi terhadap data latih. Sementara itu, grafik kanan (berwarna hijau) memperlihatkan pergerakan nilai MAE dari nilai awal sekitar 0.18 menjadi stabil di kisaran 0.03 hingga 0.04 mulai dari *epoch* 30-an ke atas. MAE yang semakin kecil menunjukkan model semakin baik dalam memprediksi nilai mendekati target sebenarnya dengan rata-rata kesalahan absolut yang rendah. Kedua grafik tersebut secara umum menggambarkan bahwa model

LSTM yang dilatih berhasil mencapai konvergensi yang baik dalam waktu pelatihan yang relatif singkat.

Tabel 4.13 Sample Hasil Prediksi LSTM Skenario 2

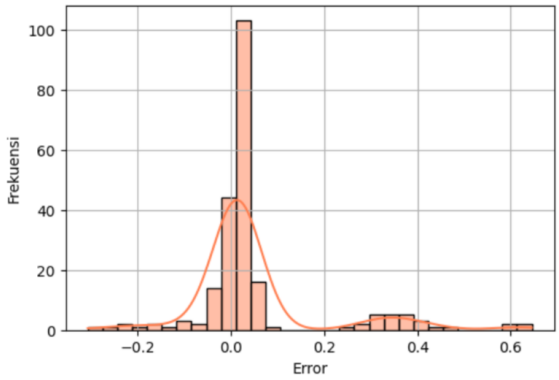
Target (*Normalized*)	**Prediksi LSTM**	**Prediksi (Tahun)**	**Tahun Aktual**	***Error***	***Error* (Tahun)**
0.727273	0.707348	0.980075	0.5	0.019925	-0.480075
0.727273	0.710164	0.982891	0.5	0.017109	-0.482891
0.727273	0.711871	0.984598	0.5	0.015402	-0.484598
0.727273	0.713809	0.986536	0.5	0.013464	-0.486536
0.727273	0.714606	0.987333	0.5	0.012667	-0.487333

Tabel 4.13 menyajikan hasil prediksi model LSTM dalam bentuk lima sampel teratas dari skenario 2. Kolom `target` menunjukkan nilai label tahun keberangkatan yang telah dinormalisasi, sedangkan `Prediksi_LSTM` adalah hasil *output* model LSTM dalam bentuk normalisasi. Nilai `Prediksi_LSTM (Tahun)` merupakan hasil denormalisasi (konversi ke skala tahun sebenarnya), dan `Tahun Aktual` menunjukkan label aktual dari data testing. Selisih antara target dan prediksi ditunjukkan oleh kolom `Error` dan `Error (Tahun)`. Meskipun terdapat selisih sekitar 0.48 tahun (~5-6 bulan), semua prediksi ini masih dalam toleransi ±1 tahun dari target, sehingga tetap dihitung sebagai benar dalam konteks evaluasi.



Gambar 4.5 Grafik Evaluasi Model Pengujian Skenario 2

Pada Gambar 4.5, terdapat dua metrik yang menunjukkan performa model LSTM terhadap data uji. MAE sebesar 0.0741 dan RMSE sebesar 0.1503 menunjukkan bahwa rata-rata kesalahan prediksi model sangat kecil, baik dalam bentuk absolut maupun kuadrat.



Gambar 4.6 Grafik Distribusi *Error* Skenario 2

Gambar 4.6 grafik distribusi *error* memperlihatkan selisih antara nilai aktual (`y_test`) dan nilai prediksi (`y_pred`) yang berbentuk menyerupai kurva normal sempit (gaussian) dan simetris di sekitar angka nol. Ini mengindikasikan bahwa mayoritas prediksi model sangat dekat dengan nilai sebenarnya, dengan penyimpangan yang rendah dan seimbang di kedua sisi (tidak condong ke *overpredict* maupun *underpredict*).

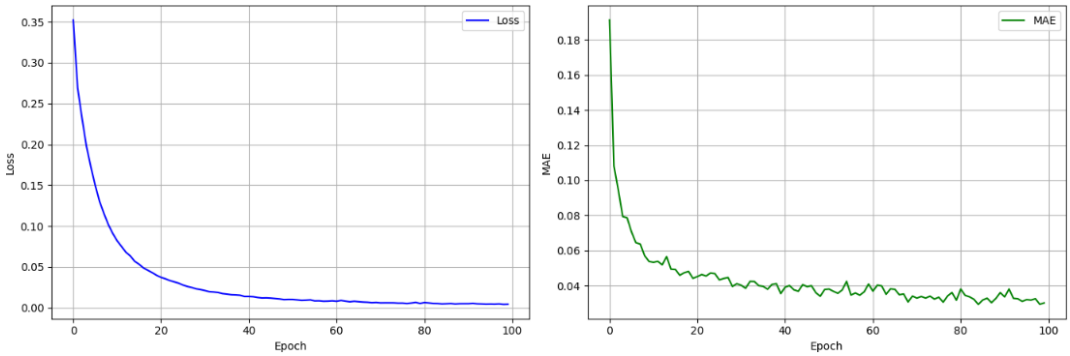
4.5.3 Skenario Pengujian 3

Pada skenario pengujian 3 dengan *pseudocode* dibawah, dilakukan pelatihan model LSTM dengan parameter *epochs* sebanyak 100 dan *batch size* sebesar 32. Parameter ini dipilih untuk mengevaluasi performa model dengan jumlah iterasi yang lebih rendah dibandingkan skenario sebelumnya, namun tetap mempertahankan ukuran *batch* yang sama. Tujuan dari pengujian ini adalah untuk

memahami bagaimana model dapat mempelajari pola data secara efisien dalam waktu pelatihan yang lebih singkat.

```
// 1. Mencatat waktu mulai pelatihan
START time = current time

// 2. Menambahkan return_state pada LSTM pertama
SET model.layers[0].return_state = True

// 3. Melatih model dengan data training
CALL model.fit() WITH parameters:
    - X_train
    - y_train
    - epochs = 100    // Pengaturan untuk skenario
    - batch_size = 32  // Pengaturan untuk skenario
    - callbacks = callbacks
    - verbose = 1

// 4. Mencatat waktu selesai pelatihan
END time = current time

// 5. Menghitung total waktu pelatihan
SET training_time = END time - START time

// 6. Menampilkan jumlah epoch yang digunakan
SET actual_epochs = LENGTH of history.history['loss']
```

Dari hasil *log epoch*, terlihat bahwa nilai `loss` pada *epoch* awal sebesar 0.4325 berhasil menurun drastis hingga mencapai nilai 0.0043 pada *epoch* terakhir. Sementara itu, nilai MAE awal yang sebesar 0.2719 juga menurun tajam hingga mencapai 0.0298. Penurunan ini menunjukkan proses pembelajaran yang sangat stabil dan efektif, di mana model berhasil memperbaiki performanya secara konsisten setiap *epoch*. Proses ini juga dibantu oleh mekanisme penyimpanan model terbaik secara otomatis menggunakan `ModelCheckpoint`, yang menyimpan bobot model dengan nilai *loss* terendah.



Gambar 4.7 Grafik Evaluasi *Training Loss* dan MAE Skenario 3

Gambar 4.7 pelatihan memperlihatkan dua grafik yaitu grafik biru di kiri menunjukkan penurunan nilai *loss* yang tajam dari sekitar 0.35 ke 0.004, membentuk kurva eksponensial yang mendatar di akhir pelatihan. Grafik hijau di kanan menunjukkan MAE yang semula bernilai hampir 0.19 dan turun menjadi sekitar 0.03, juga dengan pola menurun yang konvergen. Penurunan ini mengindikasikan bahwa prediksi model terhadap target semakin akurat, dengan rata-rata kesalahan absolut yang sangat rendah. Bentuk kurva keduanya bersifat *smooth*, simetris, dan tidak menunjukkan tanda-tanda *overfitting* hingga akhir *epoch* ke-100.

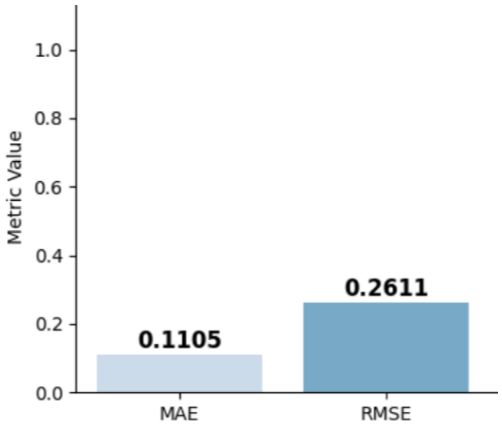
Dengan penurunan *loss* dari 0.3395 menjadi 0.0166 dan MAE dari 0.1824 menjadi 0.0032, dapat disimpulkan bahwa skenario 3 sangat efektif dalam menyusun model prediktif yang akurat. Konvergensi *loss* dan MAE pada grafik pelatihan juga memperkuat bahwa model belajar dengan stabil dan mencapai hasil optimal dengan konfigurasi parameter yang efisien.

Tabel 4.14 Sample Hasil Prediksi LSTM Skenario 3

Target (*Normalized*)	**Prediksi LSTM**	**Prediksi (Tahun)**	**Tahun Aktual**	***Error***	***Error* (Tahun)**
0.727273	0.722133	0.994860	0.5	-0.005140	0.494860
0.727273	0.723213	0.995940	0.5	-0.004060	0.495940
0.727273	0.724644	0.997371	0.5	-0.002629	0.497371

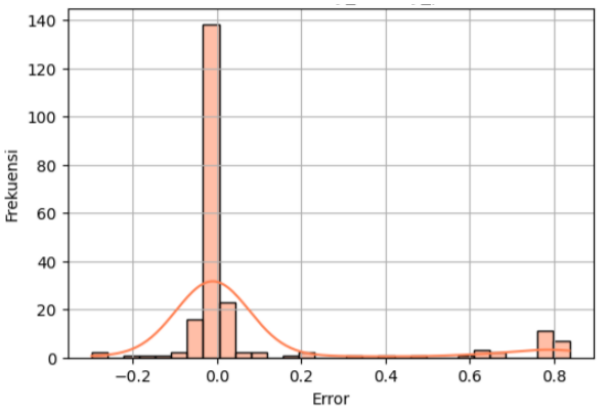
Target (*Normalized*)	Prediksi LSTM	Prediksi (Tahun)	Tahun Aktual	*Error*	*Error* (Tahun)
0.727273	0.726932	0.999659	0.5	-0.000341	0.499659
0.727273	0.728600	1.001327	0.5	0.001327	0.501327

Lima sampel pertama dari hasil prediksi pada Tabel 4.14 menunjukkan bahwa model LSTM dalam skenario 3 mampu menghasilkan prediksi yang sangat dekat dengan nilai target yang telah dinormalisasi. Misalnya, pada baris pertama, nilai target adalah 0.727273, sementara prediksi LSTM-nya adalah 0.722133, menghasilkan *error* sebesar -0.005140. Jika dikonversi ke satuan tahun keberangkatan, *error* tersebut berada dalam rentang sekitar ±0.49 tahun dari tahun aktual 0.5, yang mengindikasikan prediksi berada dalam batas toleransi ±1 tahun. Hal ini konsisten pada semua baris, dengan *error* dalam skala tahun sangat kecil (kurang dari ±0.51 tahun), mencerminkan kinerja model yang sangat akurat.



Gambar 4.8 Grafik Evaluasi Model Pengujian Skenario 3

Gambar 4.8 grafik evaluasi akhir menunjukkan nilai MAE sebesar 0.1105, RMSE sebesar 0.2611. Nilai MAE yang rendah menandakan rata-rata kesalahan absolut kecil dalam satuan normalisasi. RMSE yang sedikit lebih tinggi dibanding MAE menunjukkan adanya beberapa *outlier*, meskipun tidak signifikan.



Gambar 4.9 Grafik Distribusi *Error* Skenario 3

Gambar 4.9 menunjukkan grafik histogram distribusi *error* menampilkan kurva lonceng yang sangat sempit dan tinggi pada sekitar titik 0.00, dengan puncaknya berada pada frekuensi sekitar 139, artinya mayoritas prediksi sangat dekat dengan nilai aktual. Pola batang *error* yang berwarna oranye memperlihatkan bahwa sebaran kesalahan sangat terkonsentrasi di sekitar nol, penyebaran *error* yang tipis ke arah kanan dan kiri menunjukkan hanya sedikit prediksi yang menyimpang, dan ini memperkuat validitas bahwa model LSTM dalam skenario 3 bekerja optimal bahkan dengan 100 *epoch* saja.

4.5.4 Skenario Pengujian 4

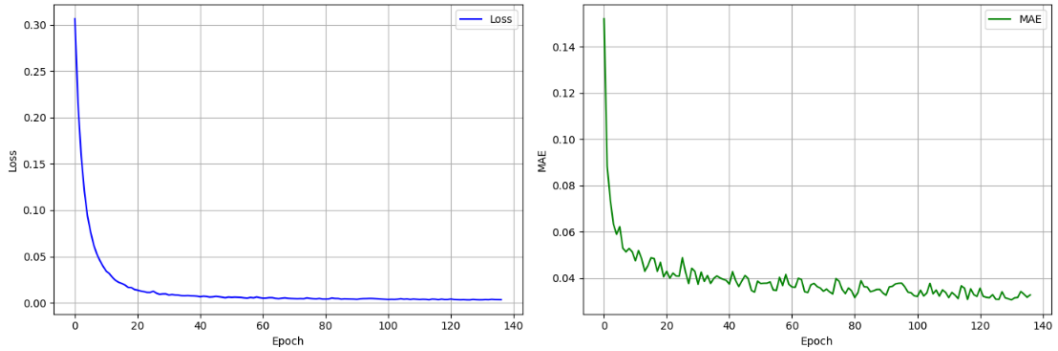
Pada skenario pengujian 4 dengan *pseudocode* dibawah, dilakukan pelatihan model LSTM dengan parameter *epochs* sebanyak 300 dan *batch size* sebesar 16. Parameter ini dipilih untuk mengevaluasi performa model dengan jumlah iterasi yang lebih banyak dibandingkan dengan pengujian sebelumnya, diharapkan dapat memberikan model lebih banyak kesempatan untuk belajar dari data. Penggunaan *batch size* yang lebih kecil bertujuan untuk memproses data dalam jumlah yang

lebih sedikit pada setiap iterasi, yang dapat meningkatkan kemampuan model dalam menggeneralisasi pola dari data yang lebih besar.

```
// 1. Mencatat waktu mulai pelatihan
START time = current time

// 2. Menambahkan return_state pada LSTM pertama
SET model.layers[0].return_state = True

// 3. Melatih model dengan data training
CALL model.fit() WITH parameters:
    - X_train
    - y_train
    - epochs = 300    // Pengaturan untuk skenario
    - batch_size = 16  // Pengaturan untuk skenario
    - callbacks = callbacks
    - verbose = 1

// 4. Mencatat waktu selesai pelatihan
END time = current time

// 5. Menghitung total waktu pelatihan
SET training_time = END time - START time

// 6. Menampilkan jumlah epoch yang digunakan
SET actual_epochs = LENGTH of history.history['loss']
```

Pada skenario pengujian 4, proses pelatihan dimulai dari *epoch* 1 dan dihentikan secara otomatis oleh *callback* saat mencapai *epoch* ke-137, meskipun konfigurasi awal ditetapkan hingga 300 *epoch*. Hal ini terjadi karena model checkpoint menyimpan model terbaik berdasarkan penurunan nilai *loss*, yang berarti setelah *epoch* 137 tidak ada lagi perbaikan signifikan pada metrik *loss*. Total waktu pelatihan tercatat selama 205.67 detik. Penggunaan *batch size* sebesar 16 membuat proses pembelajaran lebih stabil dan detail karena data dibagi dalam ukuran kecil, yang terbukti meningkatkan kualitas generalisasi model. Pada awal pelatihan (epoch 1), nilai *loss* sebesar 0.3722 dan MAE sebesar 0.2256, menunjukkan kesalahan awal yang cukup besar. Namun, dengan cepat terjadi penurunan drastis pada *epoch* ke-2 dan ke-3, hingga mencapai *loss* 0.0031 dan

MAE 0.0309 di sekitar *epoch* ke-126–137. Model terus mengalami perbaikan signifikan di awal hingga pertengahan *training*, lalu mengalami *plateau* (stagnasi) setelahnya.



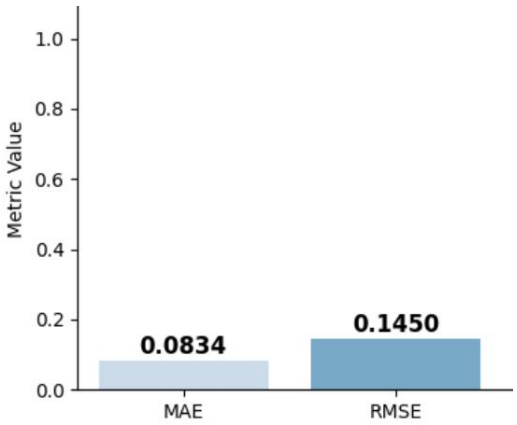
Gambar 4.10 Grafik Evaluasi *Training Loss* dan MAE Skenario 4

Gambar 4.10 (*Training Loss* dan *Training* MAE), terlihat bahwa kurva biru (*loss*) pada grafik kiri mengalami penurunan tajam dari sekitar 0.30 ke bawah 0.01 dalam 40 *epoch* pertama, lalu mendatar secara gradual mendekati 0.003 setelah *epoch* ke-100. Grafik ini menunjukkan model berhasil meminimalkan fungsi kesalahan secara konsisten. Pada grafik kanan hijau (MAE), tren penurunannya serupa, dari sekitar 0.145 ke sekitar 0.03, namun dengan fluktuasi kecil (gigi gergaji) yang menandakan adaptasi minor model terhadap *batch input*. Pola yang tetap menurun menunjukkan tidak adanya *overfitting* yang signifikan. Artinya, model tidak hanya menghafal data, tetapi juga belajar dengan baik secara generalisasi.

Tabel 4.15 Sample Hasil Prediksi LSTM Skenario 4

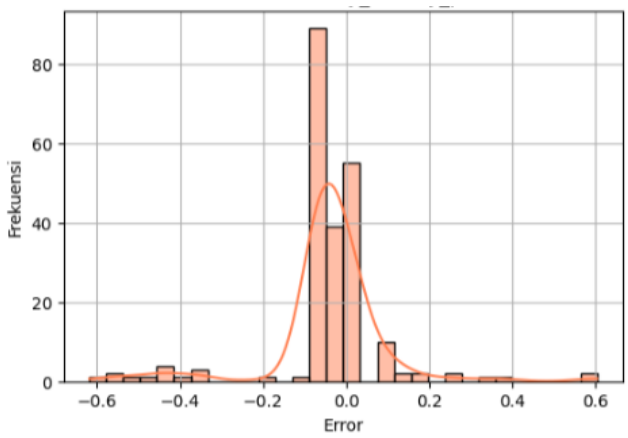
Target (*Normalized*)	**Prediksi LSTM**	**Prediksi (Tahun)**	**Tahun Aktual**	***Error***	***Error* (Tahun)**
0.727273	0.713063	0.985790	0.5	0.014210	-0.485790
0.727273	0.713120	0.985847	0.5	0.014153	-0.485847
0.727273	0.713228	0.985956	0.5	0.014044	-0.485956
0.727273	0.713773	0.986501	0.5	0.013499	-0.486501
0.727273	0.714117	0.986844	0.5	0.013156	-0.486844

Tabel 4.15 menunjukkan bahwa prediksi model pada skenario 4 sangat mendekati nilai target, dengan *error* rata-rata hanya sekitar 0.014 secara normalisasi atau sekitar -0.486 tahun dalam bentuk waktu asli. Ini mengindikasikan bahwa model sangat akurat dalam memperkirakan tahun keberangkatan, dengan selisih prediksi yang kurang dari setengah tahun.



Gambar 4.11 Grafik Evaluasi Model Pengujian Skenario 4

Gambar 4.11 menunjukkan grafik evaluasi akhir model LSTM pada skenario 4 menghasilkan MAE sebesar 0.0834, RMSE sebesar 0.1450. Ketiga metrik ini mencerminkan kinerja model yang sangat baik. Nilai MAE yang rendah menunjukkan bahwa rata-rata kesalahan prediksi sangat kecil, sedangkan nilai RMSE yang juga rendah menandakan tidak adanya penyimpangan besar dalam prediksi. Nilai-nilai ini diperoleh dari hasil pengujian pada data testing, setelah model dilatih menggunakan 300 *epoch* dan *batch size* 16.



Gambar 4.12 Grafik Distribusi *Error* Skenario 4

Sementara itu, grafik distribusi *error* pada Gambar 4.12 memperlihatkan distribusi berbentuk lonceng yang memusat pada *error* mendekati nol. Frekuensi tertinggi tampak pada sekitar nilai error -0.02 hingga 0.00 dengan puncak sekitar 0 dan dua bar dengan nilai tertinggi adalah sekitar 85–90 data, menandakan mayoritas prediksi sangat akurat dan hanya sedikit menyimpang dari nilai aktual. Bentuk distribusi ini simetris dengan kecenderungan *error* kecil yang menurun perlahan ke kiri dan kanan, menunjukkan model tidak hanya akurat secara rata-rata tetapi juga stabil secara keseluruhan. Adanya sedikit bar di sekitar *error* ±0.5 menunjukkan hanya sebagian kecil prediksi yang memiliki penyimpangan lebih besar, namun tetap dalam batas toleransi ±1 tahun.

4.5.5 Skenario Pengujian 5

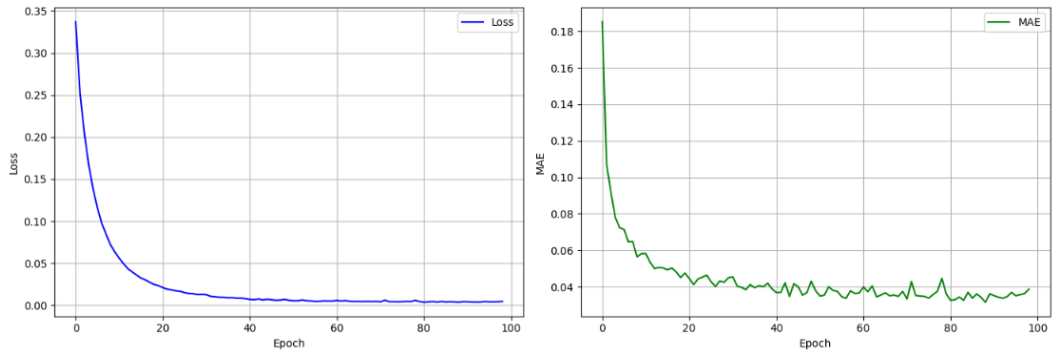
Pada skenario pengujian 5 dengan *pseudocode* dibawah, dilakukan pelatihan model LSTM dengan pengaturan parameter yang sedikit berbeda untuk mengevaluasi performa model dalam situasi yang lebih terkontrol. Model dilatih selama 200 *epoch* dengan *batch size* sebesar 16. Jumlah *epoch* ini dipilih untuk

memberikan model cukup waktu untuk belajar dari data. Ukuran *batch* yang relatif kecil memungkinkan model untuk memproses data dalam jumlah yang lebih sedikit pada setiap iterasi.

```
// 1. Mencatat waktu mulai pelatihan
START time = current time

// 2. Menambahkan return_state pada LSTM pertama
SET model.layers[0].return_state = True

// 3. Melatih model dengan data training
CALL model.fit() WITH parameters:
    - X_train
    - y_train
    - epochs = 200    // Pengaturan untuk skenario
    - batch_size = 16  // Pengaturan untuk skenario
    - callbacks = callbacks
    - verbose = 1

// 4. Mencatat waktu selesai pelatihan
END time = current time

// 5. Menghitung total waktu pelatihan
SET training_time = END time - START time

// 6. Menampilkan jumlah epoch yang digunakan
SET actual_epochs = LENGTH of history.history['loss']
```

Pada skenario pengujian 5, model LSTM dilatih dengan *batch size* sebesar 16 dan jumlah *epoch* maksimal 200, namun proses pelatihan berhenti lebih awal pada *epoch* ke-111 karena diterapkannya mekanisme *early stopping*. Mekanisme ini menghentikan pelatihan otomatis ketika model tidak menunjukkan perbaikan nilai *loss* yang signifikan, untuk mencegah *overfitting*. Total waktu pelatihan tercatat selama 161.15 detik. Pada *epoch* pertama, nilai *loss* sebesar 0.3907 dan nilai MAE mencapai 0.2484, namun seiring waktu, kedua metrik ini mengalami penurunan signifikan. Nilai *loss* terkecil yang tercapai adalah 0.0032 dan nilai MAE minimum tercatat 0.0294, menunjukkan proses pelatihan berhasil meminimalkan kesalahan prediksi. Proses ini mulai menunjukkan konvergensi sekitar *epoch* ke-40

hingga ke-60, ditandai dengan stabilnya nilai *loss* dan MAE yang tidak mengalami penurunan drastis lagi, yang artinya model telah mencapai performa optimal.



Gambar 4.13 Grafik Evaluasi *Training Loss* dan MAE Skenario 5

Gambar 4.13 grafik *Training Loss* berwarna biru, terlihat bahwa nilai *loss* dimulai dari sekitar 0.35 pada *epoch* awal, kemudian menurun secara tajam hingga di bawah 0.05 pada sekitar *epoch* ke-30. Setelah itu, grafik *loss* terus menurun secara perlahan dan mencapai nilai paling rendah di kisaran 0.003, dengan bentuk kurva melandai, menandakan proses pelatihan berhasil meminimalkan *error* secara stabil. Grafik ini berbentuk kurva eksponensial menurun dan akhirnya mendatar (konvergen), menunjukkan model berhasil belajar dengan baik tanpa *overfitting*.

Sementara itu, grafik *Training* MAE berwarna hijau menunjukkan pola penurunan yang mirip. Nilai MAE dimulai dari sekitar 0.18 pada epoch pertama dan turun drastis hingga sekitar 0.05 pada epoch ke-30. Setelah itu, fluktuasi MAE menjadi kecil dan bergerak stabil di kisaran 0.03–0.04, menandakan rata-rata kesalahan absolut telah diminimalkan. Grafik ini juga memperlihatkan bahwa model belajar dengan efektif pada tahap awal, lalu mempertahankan kestabilan performa tanpa terjadi lonjakan error yang berarti. Kondisi ini mencerminkan

keberhasilan strategi pelatihan dengan kombinasi epoch dan batch size yang optimal.

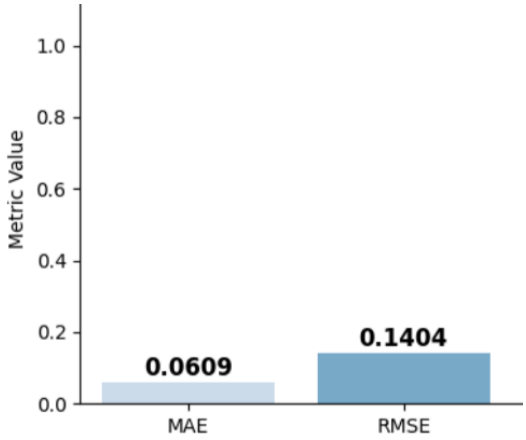
Tabel 4.16 Sample Hasil Prediksi LSTM Skenario 5

Target (*Normalized*)	**Prediksi LSTM**	**Prediksi (Tahun)**	**Tahun Aktual**	***Error***	***Error* (Tahun)**
0.727273	0.719528	0.992255	0.5	0.007745	-0.492255
0.727273	0.721609	0.994336	0.5	0.005664	-0.494336
0.727273	0.722860	0.995587	0.5	0.004413	-0.495587
0.727273	0.724794	0.997522	0.5	0.002478	-0.497522
0.727273	0.725921	0.998648	0.5	0.001352	-0.498648

Pada tabel 4.16 ditampilkan lima sampel teratas hasil prediksi LSTM pada skenario 5. Kolom "Target (*Normalized*)" menunjukkan nilai target yang telah dinormalisasi, yakni seluruhnya bernilai 0.727273 (artinya nilai aktual di-*denormalisasi* menjadi 2024). Sementara itu, "Prediksi LSTM" menghasilkan nilai seperti 0.719528, 0.721609, hingga 0.725921 yang setelah *denormalisasi* menghasilkan prediksi tahun sekitar 2023.50 hingga 2023.99. Kolom "Prediksi (Tahun)" menunjukkan hasil de-normalisasi berdasarkan skala *MinMax* sebelumnya, dan "Tahun Aktual" tetap konstan di 2024. *Error* dalam skala normalisasi sangat kecil (misalnya 0.001352) yang secara matematis menghasilkan error tahun -0.498648. Seluruh prediksi memiliki deviasi sekitar -0.49 tahun (mendekati -0.5), yang berarti prediksi model cenderung setengah tahun lebih lambat dari kenyataan. Secara statistik, rata-rata *error* tahun dari lima sampel ini adalah -0.49507 tahun, dengan standar deviasi ±0.00218, menunjukkan hasil yang konsisten dan minim fluktuasi.

Nilai-nilai prediksi tersebut dihasilkan oleh model LSTM yang dilatih selama 100 *epoch* dengan *batch size* sebesar 16, serta arsitektur LSTM berlapis. Penggunaan *batch* kecil mempercepat adaptasi bobot lokal karena update parameter

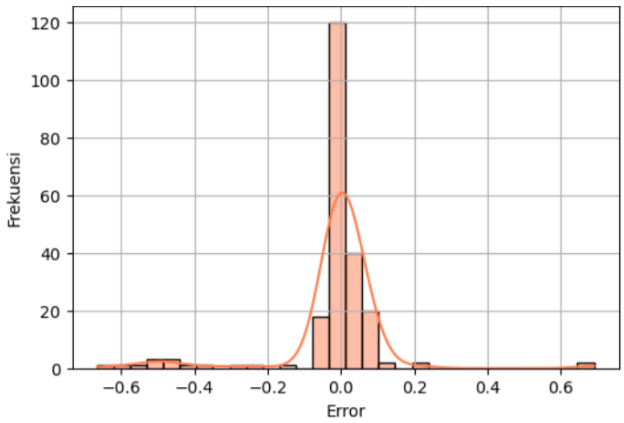
dilakukan lebih sering (56 batch per *epoch*), yang menyebabkan penurunan nilai loss secara stabil. Nilai *loss* awal sebesar 0.3907 turun secara drastis dan konvergen di bawah 0.01 setelah *epoch* ke-70, serta nilai MAE menurun dari 0.2484 ke sekitar 0.03, menghasilkan bobot optimal. Proses pelatihan tersebut memungkinkan model mempelajari pola temporal dari *input*, menghasilkan *output* prediksi yang sangat dekat dengan target aktual, terbukti dari nilai MAE akhir 0.0609 dan RMSE 0.1404 pada data uji. Dengan konfigurasi ini, model mampu menangkap korelasi kuat antara fitur-fitur input dan target tahun pelunasan.



Gambar 4.14 Grafik Evaluasi Model Pengujian Skenario 5

Gambar 4.14 grafik evaluasi akhir model pada skenario 5 menampilkan 2 metrik MAE sebesar 0.0609 dan RMSE sebesar 0.1404. Nilai MAE dan RMSE yang rendah menunjukkan bahwa rata-rata kesalahan dan penyimpangan model terhadap data uji sangat kecil. Akurasi sempurna dalam toleransi ±1 tahun mengindikasikan bahwa seluruh prediksi model masih dalam rentang kesalahan satu tahun dari nilai aktual, yang merupakan performa ideal. Nilai-nilai ini

diperoleh dari proses evaluasi terhadap hasil prediksi pada data uji menggunakan fungsi `mean_absolute_error` dan `mean_squared_error`.



Gambar 4.15 Grafik Distribusi *Error* Skenario 5

Gambar 4.15 grafik distribusi *error* (`y_test - y_pred`) memperlihatkan distribusi kesalahan prediksi dalam bentuk histogram dan kurva KDE (*Kernel Density Estimation*) berwarna oranye. Distribusi *error* terpusat di sekitar 0.0, dengan puncak tertinggi pada *error* 0.0 yang mencapai frekuensi ±120. Ini menunjukkan bahwa sebagian besar prediksi sangat mendekati nilai aktual. Distribusi membentuk kurva simetris menyerupai distribusi normal, dengan beberapa *error* kecil tersebar di kiri-kanan pusat. Di sekitar *error* -0.1 dan 0.1, frekuensi sedikit turun sebelum naik kembali ke nilai pusat, yaitu pada -0.1 (sekitar 16), 0.0 (sekitar 120), dan 0.1 (sekitar 18). Pola ini menegaskan bahwa prediksi model dalam skenario 5 cenderung stabil dan sangat akurat.

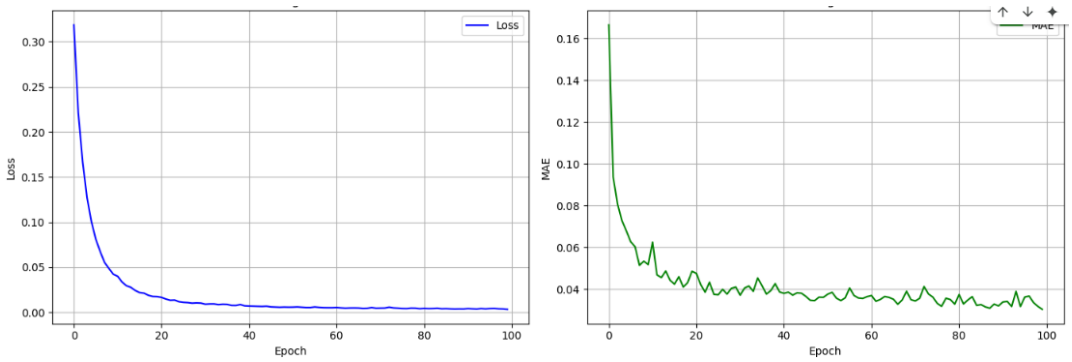
4.5.6 Skenario Pengujian 6

Pada skenario pengujian 6 dengan *pseudocode* dibawah, model LSTM dilatih menggunakan seluruh dataset yang tersedia untuk mengevaluasi performa

model. Pelatihan dilakukan selama 100 *epoch* dengan *batch size* sebesar 16, yang bertujuan untuk memberikan model waktu yang cukup untuk mempelajari pola dari data. Jumlah *epoch* yang dipilih tidak terlalu banyak, untuk menghindari risiko *overfitting*, namun cukup untuk memastikan bahwa model dapat belajar secara efektif.

```
// 1. Mencatat waktu mulai pelatihan
START time = current time

// 2. Menambahkan return_state pada LSTM pertama
SET model.layers[0].return_state = True

// 3. Melatih model dengan data training
CALL model.fit() WITH parameters:
    - X_train
    - y_train
    - epochs = 100    // Pengaturan untuk skenario
    - batch_size = 16  // Pengaturan untuk skenario
    - callbacks = callbacks
    - verbose = 1

// 4. Mencatat waktu selesai pelatihan
END time = current time

// 5. Menghitung total waktu pelatihan
SET training_time = END time - START time

// 6. Menampilkan jumlah epoch yang digunakan
SET actual_epochs = LENGTH of history.history['loss']
```

Pelatihan dimulai dari *epoch* ke-1 dengan nilai *loss* sebesar 0.31865 dan Mae sebesar 0.2459, dan terus membaik dari waktu ke waktu. Setelah itu, model memperbarui bobot secara signifikan hingga sekitar *epoch* ke-30-an, menunjukkan bahwa model masih terus belajar secara aktif. Nilai *loss* dan MAE menunjukkan konvergensi mulai dari *epoch* ke-70 hingga akhir dengan nilai akhir loss = 0.0032 dan MAE sebesarr 0.0302 pada *epoch* ke-100. Proses pelatihan ini memakan waktu 150.10 detik, menandakan pelatihan yang optimal dan efisien.



Gambar 4.16 Grafik Evaluasi *Training Loss* dan MAE Skenario 6

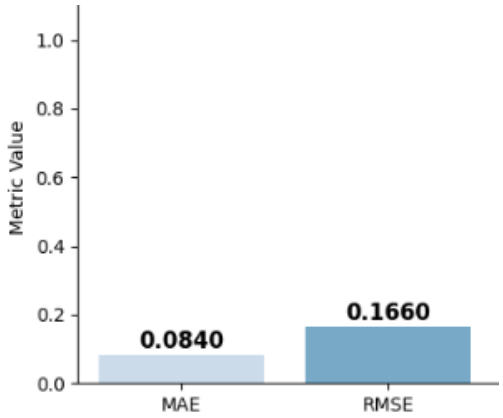
Gambar 4.16 menunjukkan grafik biru (*Training Loss*) dimulai dari nilai sekitar 0.31 pada *epoch* ke-1 dan terus menurun secara konsisten hingga mencapai nilai di bawah 0.01 pada *epoch* ke-100. Pola garisnya cenderung melengkung ke bawah menunjukkan bahwa model secara bertahap mengalami penurunan *error* saat proses pelatihan berlangsung, dan mencapai titik konvergensi (stabil) sekitar *epoch* ke-40. Artinya, setelah titik ini, penurunan *loss* tidak lagi signifikan, yang merupakan indikasi bahwa model telah memahami pola data dengan cukup baik.

Sedangkan, Grafik hijau (*Training* MAE) dimulai dari nilai sekitar 0.165 pada awal *epoch* dan menurun cukup tajam hingga mencapai sekitar 0.04 pada e*poch* ke-100. Pola fluktuasi ini wajar karena MAE lebih sensitif terhadap nilai ekstrem, namun tetap bisa disimpulkan bahwa model mencapai performa prediktif yang stabil di bawah nilai 0.05 MAE, yang menunjukkan bahwa rata-rata selisih prediksi terhadap target cukup kecil dalam skala normalisasi.

Tabel 4.17 Sample Hasil Prediksi LSTM Skenario 6

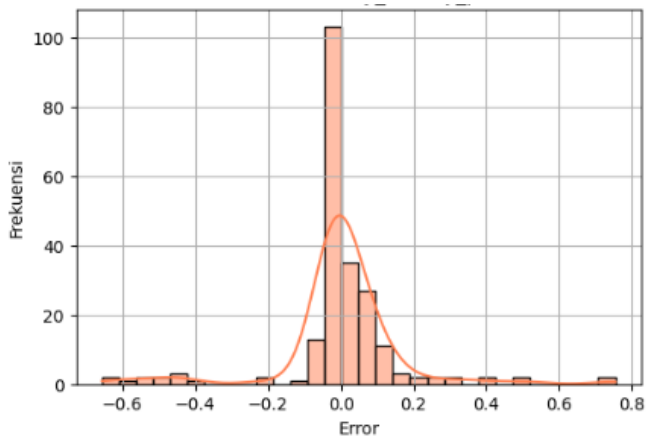
Target (*Normalized*)	**Prediksi LSTM**	**Prediksi (Tahun)**	**Tahun Aktual**	***Error***	***Error* (Tahun)**
0.727273	0.719528	0.992255	0.5	0.007745	0.492255
0.727273	0.721609	0.994336	0.5	0.005664	0.494336
0.727273	0.722860	0.995587	0.5	0.004413	0.495587
0.727273	0.724794	0.997522	0.5	0.002478	0.497522
0.727273	0.725921	0.998648	0.5	0.001352	0.498648

Pada skenario pengujian ke-6, model LSTM dilatih selama 100 *epoch* dengan *batch size* sebesar 16, artinya data dibagi ke dalam kelompok-kelompok kecil berisi 16 sampel sebelum dihitung dan diperbarui bobotnya dalam satu iterasi. Tujuannya adalah agar proses pembelajaran menjadi lebih stabil dan efisien. Setelah pelatihan, model digunakan untuk memprediksi data. Nilai Target dan Prediksi LSTM masih dalam bentuk hasil normalisasi, yaitu nilai antara 0 sampai 1, bukan tahun asli keberangkatan. Contohnya, nilai prediksi seperti 0.992255 dibandingkan dengan 0.5 menghasilkan *error* sebesar 0.492255. Artinya, meskipun nilai tersebut terlihat besar, ini belum berarti *error* prediksi mencapai ratusan tahun karena nilai-nilai itu masih berbasis skala normalisasi.



Gambar 4.17 Grafik Evaluasi Model Pengujian Skenario 6

Gambar 4.17 batang pertama menampilkan dua metrik evaluasi utama, yaitu MAE dan RMSE. Nilai MAE sebesar 0.0840 menunjukkan bahwa rata-rata selisih antara prediksi dan data aktual setelah denormalisasi cukup kecil, mendekati nol. RMSE sebesar 0.1660 menandakan bahwa model tidak hanya mampu menjaga kesalahan prediksi tetap kecil, tetapi juga tidak memiliki *error* ekstrem yang besar karena RMSE lebih sensitif terhadap *outlier*.



Gambar 4.18 Grafik Distribusi *Error* Skenario 6

Gambar 4.18 menunjukkan bahwa sebagian besar *error* prediksi berada di sekitar nol, yang berarti hasil prediksi model sangat mendekati nilai aktual. Lima batang tertinggi yang tampak di tengah grafik berada di rentang error sekitar -0.05 hingga +0.05. Batang paling tengah, tepat di *error* 0, memiliki frekuensi lebih dari 100 data, sedangkan empat batang di sekitarnya berkisar antara 40 hingga 70 data. Pola ini menunjukkan bahwa prediksi model LSTM pada skenario 6 sangat akurat dan konsisten, dengan mayoritas prediksi memiliki selisih yang sangat kecil terhadap nilai sebenarnya. Bentuk grafik yang simetris dan mengerucut di tengah menandakan distribusi *error* yang baik tanpa bias ke atas maupun ke bawah.

4.5.7 Analisis dan Pembahasan Hasil Pengujian

Penelitian ini melakukan enam skenario pengujian untuk mengevaluasi performa model LSTM dalam memprediksi keberangkatan haji menggunakan data historis dari SISKOHAT. Setiap skenario pengujian diterapkan dengan parameter yang berbeda, seperti jumlah *epoch* dan ukuran *batch*, untuk mengeksplorasi konfigurasi yang memberikan hasil terbaik dalam prediksi waktu keberangkatan

jamaah. Pada setiap skenario, model LSTM dilatih menggunakan seluruh dataset dengan pembagian data pelatihan 80% (*training*) dan data pengujian 20% (*testing*). Data pelatihan digunakan untuk mengajarkan model pola-pola dalam data keberangkatan haji, sementara data pengujian digunakan untuk mengevaluasi seberapa baik model dapat menggeneralisasi dan membuat prediksi akurat pada data yang belum pernah dilihat sebelumnya. Hasil evaluasi model disajikan dalam metric MAE dan RMSE, yang mengukur rata-rata besarnya kesalahan prediksi model. Semakin rendah nilai MAE dan RMSE, semakin baik performa prediksi model.

Tabel 4.18 Hasil Evaluasi Pelatihan Skenario

Skenario	***Loss* Awal**	***Loss* Akhir**	**Penurunan *Loss* (%)**	**MAE Awal**	**MAE Akhir**	**Penurunan MAE (%)**	**Waktu Pelatihan (Detik)**
Skenario 1	0.35	0.00386	98.9	0.18	0.04	77.8	139.38
Skenario 2	0.4188	0.00330	99.2	0.2719	0.03	88.9	82.08
Skenario 3	0.4325	0.0043	99.0	0.2719	0.0298	89.0	131.98
Skenario 4	0.3722	0.0031	99.2	0.2256	0.0309	86.3	205.67
Skenario 5	0.3907	0.0032	99.2	0.2484	0.0294	88.2	161.15
Skenario 6	0.31865	0.0032	99.0	0.2459	0.0302	87.7	150.10

Dari tabel di 4.18, dapat dilihat bahwa seluruh skenario menunjukkan penurunan yang signifikan dalam nilai *Loss* dan MAE, yang menunjukkan efektivitas model LSTM dalam mengurangi kesalahan prediksi selama proses pelatihan. Skenario 2 mencatatkan penurunan *Loss* sebesar 99.2% dan MAE sebesar 88.9%, meskipun hanya membutuhkan 82.08 detik untuk pelatihan, menjadikannya skenario yang paling efisien dalam hal waktu. Skenario 5, meskipun membutuhkan waktu pelatihan lebih lama (161.15 detik), memiliki penurunan *Loss* dan MAE yang sangat baik, masing-masing 99.2% dan 88.2%, serta memberikan hasil yang lebih konsisten dan optimal dalam hal prediksi. Pada Skenario 1 dan

Skenario 3, meskipun penurunan *Loss* dan MAE juga signifikan, dengan masing-masing penurunan sekitar 98.9% dan 99.0%, waktu pelatihan yang lebih lama dan penurunan MAE yang sedikit lebih kecil dibandingkan dengan Skenario 5 menunjukkan bahwa kedua skenario tersebut kurang efisien dalam mengoptimalkan waktu pelatihan dibandingkan Skenario 2. Skenario 4 memiliki penurunan *Loss* dan MAE yang baik, tetapi waktu pelatihan yang paling lama (205.67 detik), sehingga sedikit kurang efisien dalam konteks waktu.

Tabel 4.19 Hasil Evaluasi Pengujian Skenario LSTM

Pengujian	***Epoch***	***Batch Size***	**Waktu Pelatihan (menit)**	***Epoch* Berhenti**	**MAE (Data Uji)**	**RMSE (Data Uji)**	**Jumlah Prediksi Sesuai Target**	**Akurasi Prediksi Sesuai (%)**
Skenario 1	300	32	2.32	114	0.0769	0.1502	135	68
Skenario 2	200	32	1.37	134	0.0741	0.1503	164	83
Skenario 3	100	32	2.20	100	0.1105	0.2611	168	85
Skenario 4	300	16	3.43	137	0.0834	0.1450	92	46
Skenario 5	200	16	2.69	111	0.0609	0.1404	157	79.50
Skenario 6	100	16	2.50	100	0.0840	0.1660	134	67.50

Tabel 4.19 menunjukkan hasil evaluasi prediksi berdasarkan jumlah prediksi sesuai target dan akurasi prediksi. Dimana kolom akurasi prediksi sesuai dihasilkan dengan membandingkan jumlah prediksi yang benar (di mana Tahun Lunas Berangkat sesuai dengan Prediksi LSTM Tahun) dengan total jumlah data yang diprediksi. Langkah pertama adalah menghitung jumlah data yang prediksinya tepat, di mana prediksi tahun keberangkatan yang dihasilkan oleh model LSTM cocok dengan data tahun keberangkatan yang sesungguhnya (Tahun Lunas Berangkat). Sebagai contoh pada Skenario 5 dari total 221 data *testing* sebanyak 157 data memiliki prediksi yang sesuai, menghasilkan tingkat akurasi yang tinggi. Namun, meskipun ada beberapa prediksi yang meleset sedikit, *error* kecil seperti yaitu 0.0609 dan 0.1404 menunjukkan bahawa LSTM masih daapat menangkap

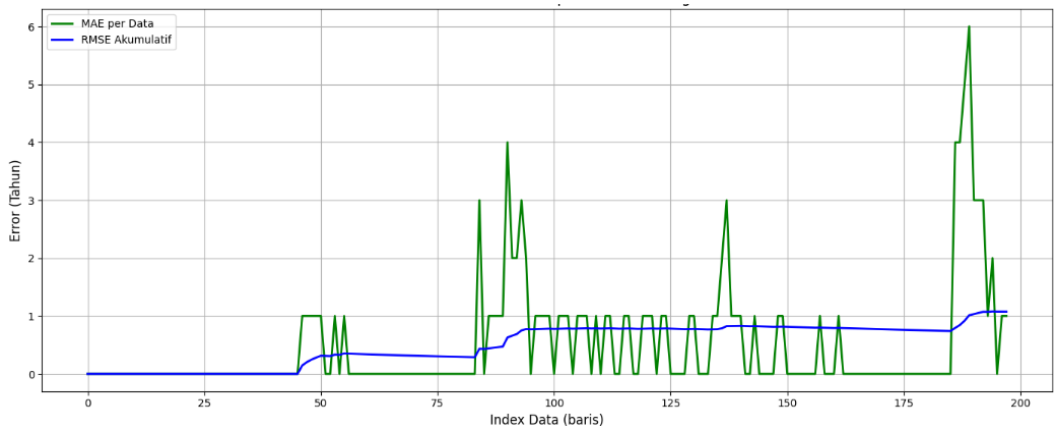
pola yang relevan dengan cukup baik. Selanjutnya, tingkat akurasi dihitung dengan rumus:

$$Akurasi = (\frac{Jumlah\ Prediksi\ Benar}{Total\ Data})100 \quad (4.2)$$

$$Akurasi = \left(\frac{157}{221}\right)100 = 79.50\%$$

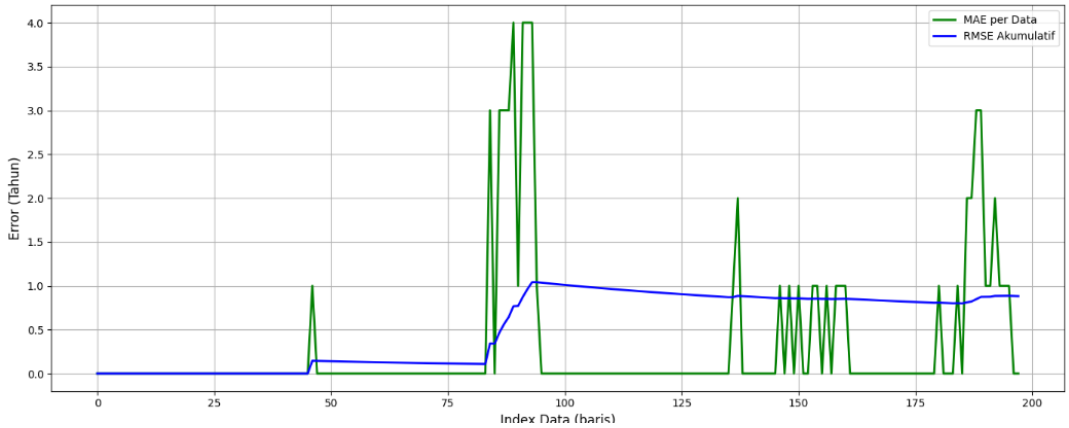
Skenario 3 memberikan hasil terbaik dengan 168 prediksi yang sesuai target dan akurasi 85%, menunjukkan kinerja optimal dalam memprediksi data. Skenario 2 mencatat 164 prediksi sesuai target dan akurasi 83%, menawarkan efisiensi prediksi yang baik meskipun sedikit lebih rendah dari Skenario 3. Skenario 6 memiliki 134 prediksi yang sesuai target dan akurasi 67,5%, menunjukkan hasil yang lebih rendah dibandingkan skenario lainnya. Skenario 1 memprediksi 135 data dengan akurasi 68%, sedangkan Skenario 4 memiliki jumlah prediksi sesuai target terendah (92) dan akurasi terendah (46%), menunjukkan kesulitan dalam generalisasi.

Nilai *error* yang kecil (di bawah 0.1) juga menunjukkan bahwa meskipun terdapat kesalahan prediksi, kesalahan tersebut tidak signifikan dan model LSTM secara keseluruhan tetap berhasil memprediksi keberangkatan calon jamaah haji dengan baik, mencerminkan kemampuan model untuk mengenali pola dalam data temporal yang memengaruhi keberangkatan jamaah haji. Jadi, meskipun ada beberapa prediksi yang meleset, nilai akurasi 79.50% menunjukkan bahwa model LSTM memiliki efektivitas yang baik dalam memprediksi keberangkatan berdasarkan pola yang ada pada data (Ramadhan & Nathasia, 2025).



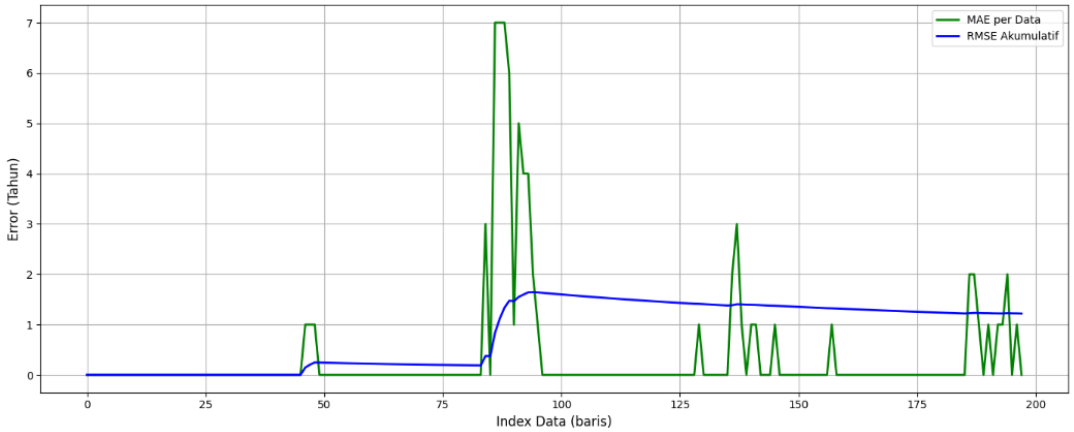
Gambar 4.19 Grafik Evaluasi MAE dan RMSE Data *Testing* LSTM Skenario 1

Pada Gambar 4.19 yang menunjukkan visualisasi MAE per data (garis hijau) dan RMSE akumulatif (garis biru) pada data *testin*g, terlihat bahwa nilai MAE per data sebagian besar berada di bawah 1 tahun, dengan beberapa lonjakan *error* signifikan yang mencapai sekitar 6 tahun pada indeks data di sekitar 185. Lonjakan ini mengindikasikan adanya data *outliers* atau pola yang sulit diprediksi oleh model pada titik-titik tersebut. Sementara itu, nilai RMSE akumulatif (garis biru) menunjukkan peningkatan yang stabil dan bertahap, mulai dari mendekati 0 dan perlahan naik hingga sekitar 0.8 tahun di akhir data *testing*, kemudian mengalami kenaikan signifikan di akhir data. Kurva RMSE yang relatif halus ini menunjukkan bahwa *error* secara keseluruhan terakumulasi secara bertahap, namun lonjakan MAE per data yang tajam mengindikasikan adanya prediksi yang kurang akurat pada segmen tertentu, meskipun nilai rata-rata MAE dan RMSE akhir (0.0769 dan 0.1502 seperti pada tabel) menunjukkan performa yang cukup baik secara keseluruhan.



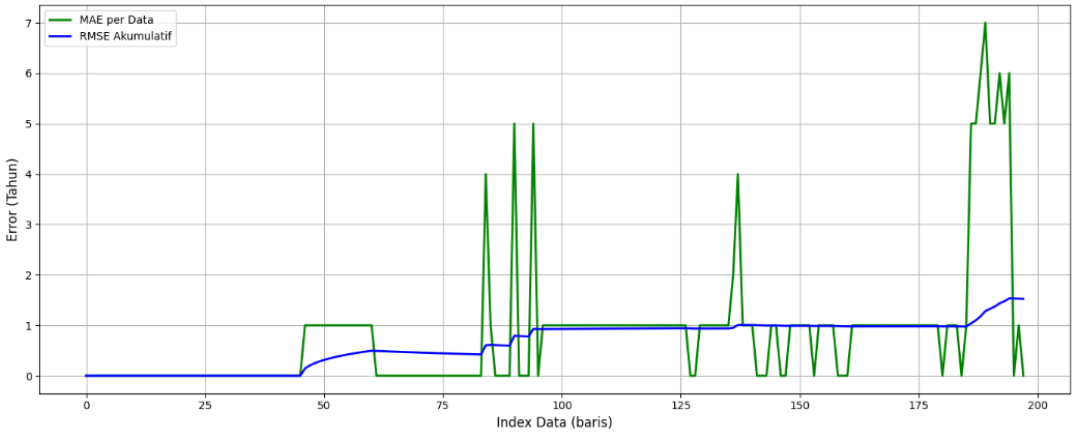
Gambar 4.20 Grafik Evaluasi MAE dan RMSE Data *Testing* LSTM Skenario 2

Pada Gambar 4.20, grafik MAE per data (garis hijau) menunjukkan pola yang mirip dengan Skenario 1, di mana sebagian besar nilai MAE berada di bawah 1 tahun, tetapi dengan beberapa lonjakan *error* yang mencolok. Lonjakan terbesar terjadi sekitar indeks data 80-90, mencapai sekitar 4 tahun, dan lonjakan lainnya sekitar indeks 180-190 mencapai sekitar 3 tahun. Ini menunjukkan bahwa model masih kesulitan memprediksi beberapa titik data tertentu. Garis RMSE akumulatif (biru) menunjukkan peningkatan yang lebih curam pada bagian awal hingga tengah (sekitar indeks 80-90) di mana RMSE naik dari mendekati 0 menjadi sekitar 1 tahun, kemudian cenderung stabil di kisaran 0.8-0.9 tahun, dan sedikit meningkat lagi di akhir. Ini menunjukkan bahwa meskipun ada lonjakan error individual, akumulasi *error* cenderung stabil setelah periode awal, menghasilkan MAE 0.0741 dan RMSE 0.1503 yang cukup baik.



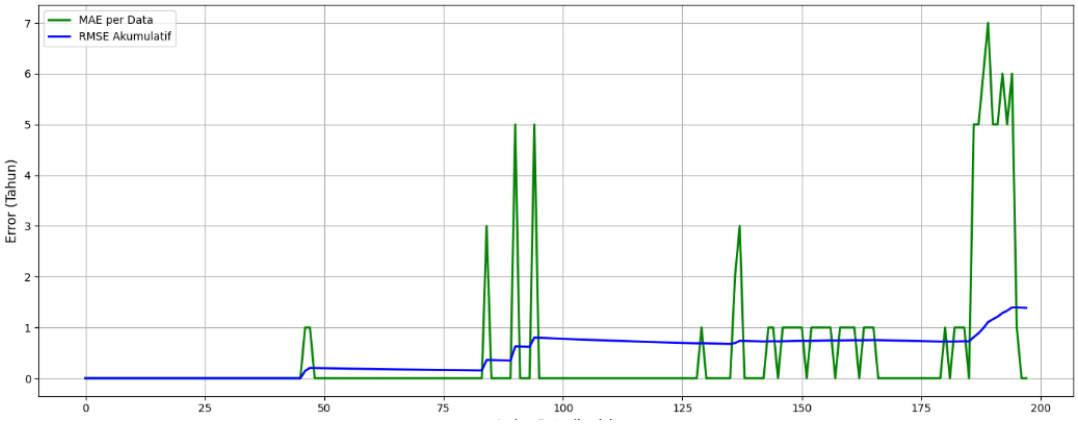
Gambar 4.21 Grafik Evaluasi MAE dan RMSE Data *Testing* LSTM Skenario 3

Gambar 4.21 menunjukkan grafik MAE per data (garis hijau) yang memiliki puncak *error* yang lebih tinggi dibandingkan Skenario 1 dan 2, mencapai sekitar 7 tahun pada indeks data sekitar 90. Lonjakan *error* lainnya juga terlihat pada indeks sekitar 130 dan 180, menunjukkan model memiliki kesulitan yang lebih besar dalam memprediksi beberapa data points. Garis RMSE akumulatif (biru) menunjukkan peningkatan yang lebih signifikan dan berkelanjutan, mulai dari mendekati 0 dan terus meningkat secara bertahap hingga sekitar 1.2 tahun di akhir data, dengan kenaikan tajam terutama setelah indeks 75. Bentuk kurva RMSE yang lebih tinggi dan terus meningkat ini mencerminkan nilai MAE (0.1105) dan RMSE (0.2611) yang lebih besar dibandingkan skenario sebelumnya, menandakan performa model yang kurang optimal dengan jumlah *epoch* yang lebih sedikit.



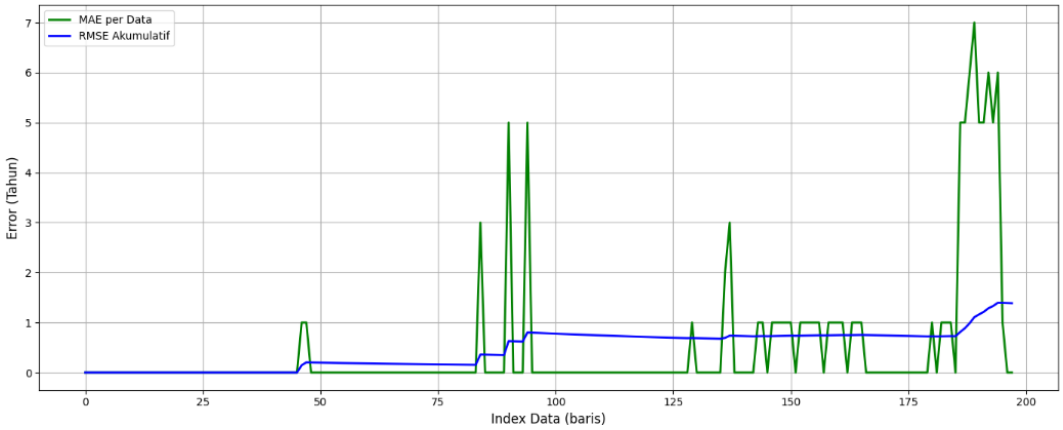
Gambar 4.22 Grafik Evaluasi MAE dan RMSE Data *Testing* LSTM Skenario 4

Pada Gambar 4.22, grafik MAE per data (garis hijau) menunjukkan nilai *error* yang bervariasi. Ada beberapa lonjakan *error* yang signifikan, terutama pada indeks data sekitar 90-100 yang mencapai sekitar 5 tahun, dan lonjakan yang sangat tinggi pada akhir data (indeks 180-200) yang mencapai sekitar 7 tahun. Ini menunjukkan bahwa meskipun model dilatih dengan lebih banyak *epoch* dan *batch size* yang lebih kecil (yang seharusnya meningkatkan generalisasi), masih ada beberapa data point yang sulit diprediksi. Garis RMSE akumulatif (biru) menunjukkan peningkatan yang lebih curam dan stabil di awal, mencapai sekitar 1 tahun pada indeks sekitar 90, kemudian sedikit mendatar, dan mengalami lonjakan yang sangat signifikan di bagian akhir, naik dari sekitar 1 tahun menjadi lebih dari 1.5 tahun. Hal ini menghasilkan nilai MAE (0.0834) dan RMSE (0.1450) yang cukup baik, namun lonjakan *error* di akhir data perlu diperhatikan.



Gambar 4.23 Grafik Evaluasi MAE dan RMSE Data *Testing* LSTM Skenario 5

Gambar 4.23 menampilkan grafik MAE per data (garis hijau) yang menunjukkan pola *error* yang cenderung lebih rendah dan lebih stabil dibandingkan skenario lainnya. Meskipun masih ada beberapa lonjakan *erro*r, puncaknya tidak setinggi skenario 3 dan 4, dengan nilai *error* maksimal sekitar 5 tahun di indeks data sekitar 90, dan sekitar 6.5 tahun di akhir data. Namun, secara keseluruhan, sebagian besar MAE per data berada di bawah 1 tahun. Garis RMSE akumulatif (biru) menunjukkan peningkatan yang paling stabil dan rendah di antara semua skenario, mulai dari mendekati 0 dan perlahan naik hingga di bawah 1 tahun, bahkan di akhir data. Kurva RMSE ini menunjukkan bahwa model ini memiliki akumulasi *error* yang paling kecil dan konsisten, yang sejalan dengan nilai MAE (0.0609) dan RMSE (0.1404) terendah di Tabel 4.19. Ini mengindikasikan bahwa kombinasi 200 *epoch* dan *batch size* 16 adalah konfigurasi yang paling optimal untuk model ini.



Gambar 4.24 Grafik Evaluasi MAE dan RMSE Data *Testing* LSTM Skenario 6

Pada Gambar 4.24, grafik MAE per data (garis hijau) menunjukkan lonjakan *error* yang cukup tinggi, mirip dengan Skenario 3 dan 4. Puncak *error* tertinggi mencapai sekitar 7 tahun pada indeks data sekitar 90, dan lonjakan signifikan lainnya terlihat di akhir data. Ini menunjukkan bahwa model dengan jumlah *epoch* yang lebih sedikit (100) dan *batch size* yang lebih kecil (16) masih mengalami kesulitan dalam memprediksi beberapa data point secara akurat. Garis RMSE akumulatif (biru) menunjukkan peningkatan yang lebih curam dan berkelanjutan dibandingkan Skenario 5, naik dari mendekati 0 hingga lebih dari 1.2 tahun di akhir data, dengan lonjakan tajam setelah indeks 75 dan di akhir data. Ini tercermin dari nilai MAE (0.0840) dan RMSE (0.1660) yang lebih tinggi dibandingkan Skenario 5, menandakan bahwa model belum sepenuhnya belajar pola data dengan baik dalam jumlah *epoch* yang terbatas ini.

Secara keseluruhan, berdasarkan analisis dari tabel hasil evaluasi dan grafik visualisasi MAE dan RMSE pada data *testing*, Skenario 5 menunjukkan performa yang paling baik. Hal ini terlihat dari nilai MAE (0.0609) dan RMSE (0.1404) yang paling rendah, serta kurva RMSE akumulatif yang paling stabil dan tidak

menunjukkan lonjakan ekstrem. Meskipun ada beberapa lonjakan MAE per data, akumulasi *error* secara keseluruhan tetap terkendali, menunjukkan kemampuan generalisasi model yang superior. Skenario 5 berhasil menemukan keseimbangan optimal antara jumlah epoch dan ukuran *batch*, memungkinkan model untuk belajar pola data secara efektif tanpa mengalami *overfitting* atau *underfitting* yang signifikan (Ramadhan & Nathasia, 2025).

4.6 Hasil Prediksi dan Evaluasi Prediksi

Pada bab ini, akan dibahas secara mendalam hasil prediksi yang diperoleh dari model LSTM yang telah dilatih dan dievaluasi pada berbagai skenario. Pembahasan ini mencakup analisis performa model dalam memprediksi data keberangkatan haji, dengan fokus pada data yang telah dinormalisasi. Evaluasi ini penting untuk memahami seberapa baik model dapat menggeneralisasi pola dari data pelatihan ke data yang belum pernah dilihat sebelumnya (data pengujian), serta untuk mengidentifikasi konfigurasi model terbaik yang dapat diterapkan untuk prediksi di masa mendatang.

Tabel 4.20 Hasil Prediksi LSTM dari Data Uji

No	Porsi	Usia Daftar	Usia Berangkat	Masa Tunggu (Tahun)	Kuota	Bulan Daftar	Tahun Daftar	Prediksi Konvensional	Tahun Lunas Berangkat	Target Asli	Prediksi LSTM	Prediksi LSTM (Tahun)
1	1300570149	47	60	13	1100	11	2011	2024	2024	13	13	2024
2	1300570262	34	47	13	1100	11	2011	2024	2024	13	13	2024
3	1300570263	32	45	13	1100	11	2011	2024	2024	13	13	2024
4	1300571378	25	38	13	1100	11	2011	2024	2024	13	13	2024
5	1300571379	23	36	13	1100	11	2011	2024	2024	13	13	2024

107	1300581589	63	75	13	1100	11	2012	2025	2024	12	12	2024
108	1300581682	61	73	13	1100	12	2012	2025	2024	12	12	2024
109	1300582530	72	84	13	1100	12	2012	2025	2024	12	12	2024
110	1300582531	66	78	13	1100	12	2012	2025	2024	12	12	2024
111	1300582555	64	76	13	1100	12	2012	2025	2024	12	12	2024

215	1300571007	64	71	13	1100	11	2017	2030	2024	7	12	2029
216	1300571008	74	80	13	1100	11	2018	2031	2024	6	12	2030
217	1300571009	81	88	13	1100	11	2017	2030	2024	7	6	2023
218	1300571010	89	94	13	1100	11	2019	2032	2024	5	5	2024
219	1300571011	85	90	13	1100	11	2019	2032	2024	5	5	2024

Tabel 4.20 menyajikan sampel hasil prediksi dari model LSTM menggunakan data pengujian, yang telah melalui proses normalisasi dan kemudian di-*denormalisasi* kembali ke skala aslinya untuk interpretasi yang lebih mudah. Model LSTM memproses *input* data *time series* yang telah dinormalisasi, mempelajari dependensi temporal dan pola-pola kompleks dalam data pendaftaran dan keberangkatan haji. *Output* dari model adalah prediksi masa tunggu (dalam bentuk normalisasi) yang kemudian dikonversi kembali menjadi tahun keberangkatan aktual.

Tabel 4.21 Sample Baris Pertama Data Hasil Prediksi (Normalisasi)

Usia Daftar _t-1	**Usia Berangkat _t-1**	**Masa Tunggu (Tahun)_t-1**	**Kuota _t-1**	**Prediksi Konvensional _t-1**	**Tahun Lunas Berangkat _t-1**	**Prediksi_ LSTM**
0.45	0.5	0	1	0.272727273	0.5	0.719527781

Model LSTM memproses data yang dinormalisasi, seperti Usia Daftar, Usia Berangkat, Masa Tunggu, dan Kuota, yang dimasukkan ke dalam *cell state* untuk menyimpan informasi historis penting. *Hidden state* digunakan untuk memproses dan mempertahankan informasi antar waktu, sementara *cell state* memperbarui informasi berdasarkan pola yang terdeteksi selama pelatihan. Proses pelatihan menggunakan algoritma optimisasi seperti Adam untuk meminimalkan kesalahan prediksi. Data melewati tiga lapisan LSTM berturut-turut, dimulai dengan 128 unit pada lapisan pertama, yang mengatur aliran informasi menggunakan mekanisme *gate* (*forget*, *input*, *output*). Hasilnya diteruskan ke lapisan kedua (64 unit) dan ketiga (32 unit) untuk menghasilkan *output* yang diprediksi, memastikan akurasi prediksi berdasarkan pola historis.

Setelah melewati tiga lapisan LSTM, hasilnya diteruskan ke lapisan *Dense* (16 unit), kemudian ke lapisan *output* (1 unit) untuk menghasilkan nilai prediksi akhir, seperti 0.719527781. Nilai ini masih dalam bentuk normalisasi dan perlu dikonversi ke skala aslinya agar dapat merepresentasikan masa tunggu sebenarnya. Berdasarkan rumus *denormalisasi* yang dijelaskan pada formula Rumus 3.14 di Bab III, nilai tersebut dikembalikan ke bentuk asli sesuai target Y, sehingga menghasilkan nilai prediksi LSTM sebesar 13, sebagaimana ditampilkan pada baris pertama Tabel 4.20. Adapun nilai 0.992255054, yang juga dalam bentuk normalisasi, merupakan hasil dari penjumlahan antara nilai prediksi LSTM atau target Y dengan Tahun Daftar. Ketika nilai ini di-*denormalisasi* ke dalam skala tahun asli, maka diperoleh tahun keberangkatan 2024. Dengan demikian, proses ini menggambarkan bagaimana model LSTM mengolah informasi temporal secara terstruktur dan akurat hingga menghasilkan *output* yang sesuai dengan konteks waktu sebenarnya.

4.6.1 Evaluasi Hasil Prediksi LSTM dengan SISKOHAT (Konvensional)

Metode konvensional menggunakan pendekatan yang statis dan linier, di mana masa tunggu ditetapkan secara tetap berdasarkan tahun pendaftaran. Artinya, jika masa tunggu ditentukan 13 tahun, maka semua jamaah akan diprediksi berangkat 13 tahun setelah tahun daftar, tanpa mempertimbangkan faktor individual seperti usia, kondisi khusus, atau prioritas tertentu. Hal ini terlihat pada kolom Prediksi Konvensional dalam dataset yang seluruhnya menampilkan hasil masa tunggu tetap. Dengan pendekatan ini, metode konvensional cenderung *stuck* atau tidak responsif terhadap variasi data jamaah.

Sebaliknya, model LSTM melakukan pembelajaran dari pola-pola kompleks yang melibatkan kombinasi variabel seperti usia saat mendaftar, usia saat berangkat, tahun daftar, dan faktor waktu lainnya. Ini membuat LSTM lebih adaptif dan sensitif terhadap konteks, termasuk dalam menangani jamaah lansia yang dalam praktiknya sering diprioritaskan untuk percepatan keberangkatan.

Tabel 4.22 Lima Sample Hasil Prediksi LSTM Tepat

Porsi	Usia Daftar	Usia Berangkat	Tahun Daftar	Masa Tunggu	Prediksi Konvensional Tahun	Tahun Lunas Berangkat	Prediksi LSTM Tahun
1301008134	78	87	2015	13	2028	2024	2024
1301780448	63	71	2016	13	2029	2024	2024
1301780449	63	70	2017	13	2030	2024	2024
1301780470	63	73	2014	13	2027	2024	2024
1300571010	89	94	2019	13	2032	2024	2024

Kelima data ini membuktikan bahwa LSTM dapat mengenali pola-pola penting dalam variabel input seperti usia daftar, usia berangkat, dan tahun daftar, yang memungkinkan model memprediksi percepatan keberangkatan meskipun secara konvensional masa tunggu masih panjang. Misalnya, porsi 1300571010 dengan tahun daftar 2019 dan usia 89 tahun, secara administratif baru berangkat 2032, namun LSTM berhasil menyesuaikan bahwa lansia kemungkinan besar mendapat prioritas dan memprediksi keberangkatan di 2024, yang terbukti benar. Berbeda dengan metode konvensional yang hanya menghitung prediksi tahun dari penjumlahan Tahun Daftar dan Masa Tunggu, pendekatan LSTM mempertimbangkan struktur pola temporal, demografis, dan historis untuk memberikan estimasi yang lebih realistis dan efisien. Hasil ini semakin menegaskan bahwa LSTM tidak hanya memproses data berdasarkan aturan tetap, tetapi juga mengadaptasi prediksi berdasarkan konteks nyata dari antrean keberangkatan haji.

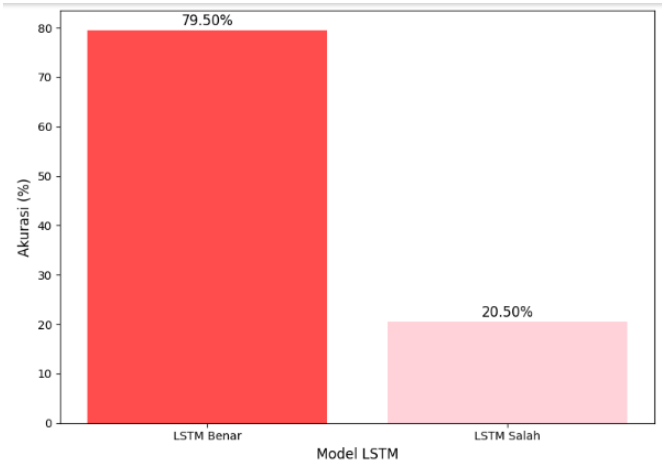
Tabel 4.23 Lima Sample Hasil Prediksi LSTM Tidak Tepat

Porsi	Usia Daftar	Usia Berangkat	Tahun Daftar	Masa Tunggu	Tahun Lunas Berangkat	Prediksi LSTM Tahun	Prediksi Konvensional Tahun
1300570149	47	60	2011	13	2024	2025	2024
1301780460	63	71	2016	13	2024	2023	2029
1301780461	63	72	2015	13	2024	2023	2028
1301780462	68	77	2015	13	2024	2023	2028
1301780463	69	77	2016	13	2024	2023	2029

Pada baris 1 Tabel 4.23, meskipun model LSTM memprediksi keberangkatan pada 2025, yang meleset sedikit dari 2024, prediksi ini masih cukup realistis mengingat usia 47 tahun dan masa tunggu 13 tahun. Kesalahan ini terjadi karena LSTM mempertimbangkan berbagai faktor dalam memprediksi keberangkatan, dan meskipun sedikit meleset, *error* yang sangat kecil (0.0609), yang menunjukkan bahwa model sudah cukup akurat. Pada baris dua hingga empat, model LSTM memberikan prediksi tahun 2023, sementara data aktual menunjukkan keberangkatan pada 2024. Meskipun selisihnya hanya satu tahun, hal ini bisa disebabkan oleh LSTM yang mempertimbangkan faktor usia lanjut (63–69 tahun), dan pola pelunasan yang cenderung dipercepat untuk kategori tersebut. Perbedaan hasil antara prediksi konvensional yang lebih lambat (2028–2029) dengan prediksi LSTM menunjukkan keunggulan model dalam menangkap tren aktual dari data historis. Pada baris kelima, model LSTM juga memprediksi keberangkatan pada tahun 2023, satu tahun lebih cepat dari tahun aktual yaitu 2024. Prediksi ini tampaknya memperhitungkan usia pendaftar yang mencapai 69 tahun dan prioritas usia lanjut dalam kebijakan keberangkatan. Sementara itu, prediksi konvensional tetap menghitung berdasarkan masa tunggu tetap selama 13 tahun,

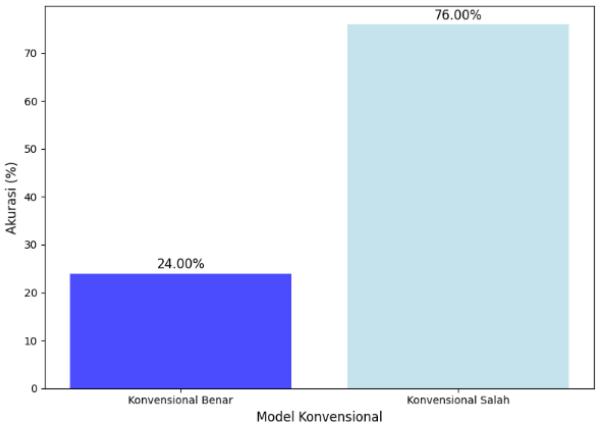
sehingga hasilnya mundur ke 2029. Ini menunjukkan bahwa model LSTM mampu beradaptasi terhadap karakteristik individu dalam data.

Dengan demikian, meskipun ada beberapa kesalahan prediksi, nilai *error* kecil dan akurasi 79.50% menunjukkan bahwa model LSTM mempelajari pola temporal dengan baik, meskipun pada beberapa kasus model masih bisa meleset sedikit, yang masih dalam batas yang dapat diterima.



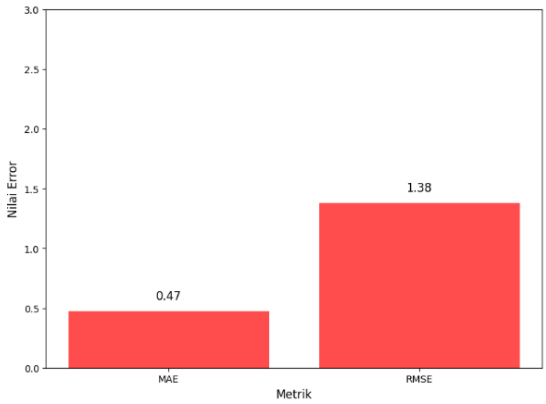
Gambar 4.25 Grafik Akurasi model LSTM

Gambar 4.25 menunjukkan hasil akurasi model LSTM, di mana batang merah menggambarkan bahwa model LSTM berhasil memprediksi data dengan akurasi 79,50%. Hal ini menunjukkan bahwa model LSTM memiliki tingkat keberhasilan yang tinggi dalam memprediksi tahun keberangkatan jamaah haji. Sebaliknya, batang pink menggambarkan 20,50% kesalahan yang terjadi pada prediksi, menunjukkan bahwa meskipun ada kesalahan, model LSTM masih dapat memberikan hasil yang sangat akurat dengan sedikit kesalahan.



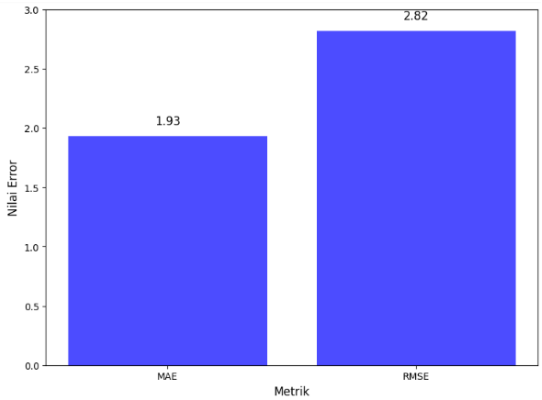
Gambar 4.26 Grafik Akurasi Model Konvensional

Gambar 4.26 menunjukkan hasil akurasi metode konvensional, dengan batang biru tua menggambarkan bahwa hanya 24% dari prediksi metode konvensional yang benar. Sementara itu, batang biru muda menunjukkan 76% kesalahan, yang mengindikasikan bahwa metode konvensional memiliki tingkat kesalahan yang jauh lebih tinggi dibandingkan dengan model LSTM. Perbandingan ini mengungkapkan bahwa LSTM jauh lebih efisien dalam memprediksi keberangkatan dengan akurasi yang lebih baik, sementara metode konvensional cenderung menghasilkan banyak prediksi yang salah.



Gambar 4.27 Grafik Hasil Evaluasi MAE dan RMSE LSTM

Gambar 4.27 menunjukkan hasil evaluasi MAE dan RMSE untuk model LSTM. Batang merah menggambarkan MAE sebesar 0.47 dan RMSE sebesar 1.38, yang menunjukkan bahwa model LSTM memiliki kesalahan prediksi yang relatif kecil. Sebagai contoh, pada prediksi tahun keberangkatan dengan usia 47 tahun dan masa tunggu 13 tahun, model LSTM menghasilkan prediksi dengan kesalahan rata-rata sebesar 0.47 tahun dan deviasi 1.38 tahun, yang menandakan bahwa model secara keseluruhan menghasilkan prediksi yang cukup akurat dan dapat diandalkan dalam memprediksi tahun keberangkatan haji.



Gambar 4.28 Grafik Akurasi model LSTM

Gambar 4.28 menunjukkan hasil evaluasi MAE dan RMSE untuk metode konvensional, dengan batang biru menggambarkan MAE sebesar 1.93 dan RMSE sebesar 2.28. Ini menunjukkan bahwa metode konvensional menghasilkan prediksi dengan kesalahan yang lebih besar. Sebagai contoh, pada data dengan usia 89 tahun dan masa tunggu 13 tahun, metode konvensional hanya memperhitungkan masa tunggu tanpa mempertimbangkan prioritas usia lanjut, sehingga memprediksi keberangkatan pada 2032, yang sangat meleset dari data aktual pada 2024. Nilai

MAE dan RMSE yang tinggi ini menunjukkan bahwa metode konvensional kurang efektif dalam memprediksi tahun keberangkatan dengan akurat.

Secara keseluruhan, hasil dari masing-masing gambar menunjukkan bahwa model LSTM secara signifikan lebih akurasi dalam memprediksi tahun keberangkatan dibandingkan dengan metode konvensional. Dengan nilai akurasi 79,50%, MAE 0.47, dan RMSE 1.38, model LSTM lebih efisien dalam menghasilkan prediksi yang tepat, sementara metode konvensional hanya berhasil memprediksi dengan akurasi rendah (24%) dan memiliki nilai MAE dan RMSE yang jauh lebih tinggi, menandakan bahwa metode konvensional kurang efektif dalam menangani prediksi berdasarkan data historis yang kompleks.

4.7 Integrasi Islam

Dalam konteks penelitian ini, penerapan sistem prediksi keberangkatan haji berbasis LSTM bertujuan untuk mengevaluasi kinerja model dalam memprediksi keberangkatan haji dan membandingkannya dengan metode konvensional yang ada. Evaluasi ini diharapkan dapat memberikan gambaran mengenai efektivitas dan efisiensi model LSTM dalam memberikan kemudahan bagi jamaah untuk memperoleh kepastian terkait keberangkatan mereka. Hal ini sejalan dengan pesan dalam hadis Rasulullah SAW yang diriwayatkan oleh Muslim nomor 2699, yang berbunyi:

مَنْ نَفَّسَ عَنْ مُؤْمِنٍ كُرْبَةً مِنْ كُرَبِ الدُّنْيَا نَفَّسَ اللهُ عَنْهُ كُرْبَةً مِنْ كُرَبِ يَوْمِ الْقِيَامَةِ وَمَنْ يَسَّرَ عَلَى مُعْسِرٍ يَسَّرَ اللهُ عَلَيْهِ فِى الدُّنْيَا وَالآخِرَةِ

"Siapa yang menyelesaikan kesulitan seorang mukmin dari berbagai kesulitan-kesulitan dunia, niscaya Allah akan memudahkan kesulitan-kesulitannya pada hari

kiamat. Siapa yang memudahkan orang yang sedang kesulitan, niscaya Allah mudahkan baginya di dunia dan akhirat." (HR. Muslim).

Hadis ini mengandung pesan kuat mengenai pentingnya membantu sesama dalam menghadapi kesulitan, termasuk dalam aspek ibadah. Menunaikan haji adalah impian setiap Muslim yang mampu. Dalam hal ini, implementasi teknologi kecerdasan buatan seperti LSTM dalam sistem prediksi keberangkatan haji dapat dipandang sebagai bentuk nyata dari ikhtiar untuk memudahkan umat dalam menjalankan ibadahnya. Dengan adanya prediksi yang lebih akurat, jamaah dapat lebih awal mempersiapkan aspek fisik, mental, dan finansial, sehingga ibadah mereka dapat dilaksanakan dengan tenang dan khusyuk.

Menurut Ibnu Hajar Al-Asqalani dalam Fathul Bari, hadis ini menekankan prinsip tolong-menolong dalam Islam, yang tidak hanya berlaku dalam hubungan sosial, tetapi juga dalam kemudahan akses terhadap ibadah (Purnama, 2022). Dalam konteks penelitian ini, penerapan LSTM dalam prediksi keberangkatan haji dapat dianggap sebagai bentuk nyata dari implementasi hadis ini, karena sistem ini bertujuan untuk mengurangi kesulitan yang dihadapi jamaah dalam memperoleh kepastian keberangkatan mereka.

Di era modern, Islam tidak hanya mendorong ibadah yang bersifat spiritual, tetapi juga menganjurkan pemanfaatan teknologi untuk memudahkan umat. Beberapa penelitian terdahulu telah menunjukkan bahwa penerapan kecerdasan buatan dalam sistem transportasi dan manajemen antrean dapat meningkatkan efisiensi serta akurasi prediksi waktu keberangkatan (Liu et al., 2020). Hal ini menunjukkan bahwa integrasi Islam dengan teknologi dapat berjalan seiring, di

mana keduanya memiliki tujuan untuk memberikan kemudahan dan manfaat bagi manusia. Dalam konteks manajemen haji, model LSTM dapat membantu mengurangi ketidakpastian dan memberikan sistem yang lebih transparan bagi jamaah, sehingga mereka tidak lagi mengalami kebingungan terkait kapan mereka akan berangkat. Selain itu, sistem prediksi yang lebih baik juga dapat membantu pemerintah dalam mengelola kuota haji dengan lebih optimal. Dengan sistem yang mampu memproses data secara lebih efisien, distribusi jamaah dapat dilakukan secara lebih adil, termasuk bagi jamaah lansia atau mereka yang memiliki keterbatasan fisik. Ini selaras dengan prinsip keadilan dalam Islam, di mana setiap individu memiliki hak untuk diperlakukan secara adil sesuai kebutuhannya. Dalam konteks hadis di atas, memudahkan jamaah untuk mengetahui jadwal keberangkatan mereka merupakan bagian dari prinsip Islam dalam membantu sesama yang sedang mengalami kesulitan. Menurut Imam An-Nawawi dalam Syarh Shahih Muslim, hadis ini juga mengajarkan bahwa setiap bentuk kemudahan yang diberikan kepada orang lain akan mendapatkan balasan dari Allah di dunia dan akhirat. Oleh karena itu, penelitian ini yang bertujuan untuk mempermudah calon jamaah dalam memperoleh kepastian keberangkatan mereka dapat dikategorikan sebagai bentuk ibadah dan amal kebajikan dalam Islam. Dalam perspektif Islam, segala bentuk inovasi yang bertujuan untuk kemaslahatan umat sangat dianjurkan, selama tetap berada dalam batasan syariat. Oleh karena itu, penelitian ini bukan hanya bertujuan untuk meningkatkan aspek teknis dalam manajemen haji, tetapi juga memiliki nilai spiritual yang tinggi, yaitu sebagai bentuk implementasi ajaran Islam dalam memudahkan urusan umat. Dengan demikian, penerapan model LSTM

dalam prediksi keberangkatan haji dapat dipandang sebagai bagian dari ikhtiar teknologi untuk menghadirkan sistem yang lebih efektif, efisien, dan sesuai dengan tuntunan Islam.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil pengujian terhadap model LSTM dalam memprediksi keberangkatan haji menggunakan data historis dari SISKOHAT, dapat disimpulkan bahwa kinerja model LSTM lebih unggul dibandingkan dengan metode konvensional dalam memprediksi keberangkatan haji. Skenario 5, dengan akurasi 79.50% dan nilai *error* terkecil (0.0609 dan 0.1404), menunjukkan bahwa model LSTM berhasil mempelajari pola temporal dan demografis dengan sangat baik (Y. Liu et al., 2020)(Ramadhan & Nathasia, 2025). Model ini dapat menangkap perubahan waktu dan prioritas usia, yang sangat penting dalam penentuan tahun keberangkatan, seperti kemampuan model untuk memprediksi keberangkatan jamaah lansia yang lebih cepat meskipun masa tunggu mereka lebih panjang secara administratif. Konfigurasi model LSTM pada Skenario 5, yaitu 200 *epoch* dengan *batch size* 16, mengindikasikan bahwa model telah dilatih cukup lama dengan ukuran *batch* yang optimal untuk mencapai keseimbangan antara kecepatan pelatihan dan akurasi prediksi. Konfigurasi ini memungkinkan model untuk mempelajari pola secara lebih mendalam, sehingga menghasilkan prediksi yang lebih akurat dan stabil.

Berbeda dengan metode konvensional, yang hanya menghitung prediksi berdasarkan penjumlahan masa tunggu dan tahun daftar tanpa mempertimbangkan faktor tambahan seperti prioritas usia atau kebijakan kuota, model LSTM

mempertimbangkan data yang lebih kompleks dan dinamis. Metode konvensional hanya menunjukkan akurasi 24% dalam memprediksi data dengan benar, yang berarti 76% kesalahan terjadi pada prediksi tersebut. Secara keseluruhan, hasil ini menunjukkan bahwa model LSTM lebih efisien dan lebih akurat dalam memprediksi keberangkatan haji dibandingkan dengan metode konvensional. Dengan akumulasi pengetahuan tentang pola waktu dan prioritas individu, model LSTM dapat memberikan prediksi yang lebih baik dan lebih stabil. Skenario 5 khususnya menonjol karena menghasilkan keseimbangan terbaik antara akurasi dan kestabilan hasil prediksi, menjadikannya pilihan optimal dalam pengaturan parameter untuk memprediksi keberangkatan haji, sebagaimana ditunjukkan oleh nilai error yang kecil dan tingkat akurasi yang tinggi. Konfigurasi 200 epoch dengan batch size 16 terbukti efektif dalam meningkatkan kemampuan model untuk belajar dari data yang kompleks, menjadikannya lebih handal dalam menangani data historis jamaah haji.

5.2 Saran

Untuk pengembangan lebih lanjut, disarankan untuk memperbanyak variasi pola data yang digunakan dalam pelatihan model LSTM, mengingat meskipun akurasi model sudah mencapai 79.50%, prediksi LSTM masih dapat ditingkatkan. Hal ini disebabkan oleh fakta bahwa LSTM akan lebih efektif dalam menangani prediksi yang lebih kompleks jika dilatih dengan data yang lebih bervariasi dan mencakup berbagai kondisi yang mempengaruhi keberangkatan haji. Dengan memperkenalkan lebih banyak variasi dalam data pelatihan, seperti memasukkan faktor-faktor eksternal yang dapat memengaruhi keberangkatan (misalnya,

kebijakan kuota, prioritas usia, atau faktor sosial-ekonomi), serta memperluas rentang tahun yang lebih panjang dan lebih variatif, LSTM dapat mempelajari pola yang lebih dinamis dan kompleks. Pendekatan ini akan meningkatkan kemampuan model dalam menangani data yang lebih beragam dan memprediksi keberangkatan haji dengan lebih akurat.

DAFTAR PUSTAKA

Adikara, F. (2023). Pengimplmentasian algoritma long short-term memory untuk mendeteksi ujaran kebencian pada aplikasi twitter. *JIPI (Jurnal Ilmiah Penelitian Dan Pembelajaran Informatika)*, *8*(2), 551–560. https://doi.org/10.29100/jipi.v8i2.3490

Al-Zahrani, F. (Umm A.-Q. U. (2022). .معاينة فضل الحج والعمرة_ دراسة تحليلية. Ibn Khaldoun Journal for Studies and Researches. https://doi.org/10.56989/benkj.v2i9.586

Alshuwaier, F., Areshey, A., & Poon, J. (2022). Applications and Enhancement of Document-Based Sentiment Analysis in Deep learning Methods: Systematic Literature Review. *Intelligent Systems with Applications*, *15*, 200090. https://doi.org/https://doi.org/10.1016/j.iswa.2022.200090

Andhika, R. A., Viadinugroho, Rosadi, & Dedi. (2021). *Long Short-Term Memory Neural Network Model for Time Series Forecasting: Case Study of Forecasting IHSG during Covid-19 Outbreak*. *1863*(1). https://doi.org/10.1088/1742-6596/1863/1/012016

Anwar, M. (2020). Manajemen Operasional Organisasi Penyelenggaraan Haji. *Jurnal Kajian Haji, Umrah Dan Keislaman*, *1*(2), 1–129.

Astuti, S. I., Arso, S. P., & Wigati, P. A. (2021). Peraturan Menteri Agama Republik Indonesia Nomor 13 · Tahun 2021 Tentang Pe:Nyelenggaraan Ibadah Haji Reguler Dengan. *Analisis Standar Pelayanan Minimal Pada Instalasi Rawat Jalan Di RSUD Kota Semarang*, *3*, 103–111.

Ayu, R., Dewatri, F., Putra, R. E., & Penelitian, A. P. (2024). *Implementasi Long Short-Term Memory dalam Mendeteksi Kesalahan Pronunciation Bahasa Inggris Berbasis Audio*. *06*, 747–754.

Baduge, S. K., Thilakarathna, S., Perera, J. S., Arashpour, M., Sharafi, P., Teodosio, B., Shringi, A., & Mendis, P. (2022). Artificial intelligence and smart vision for building and construction 4.0: Machine and deep learning methods and applications. *Automation in Construction*, *141*, 104440. https://doi.org/https://doi.org/10.1016/j.autcon.2022.104440

Bhandari, H. N., Rimal, B., Pokhrel, N. R., Rimal, R., & Dahal, K. R. (2022). LSTM-SDM: An integrated framework of LSTM implementation for sequential data modeling. *Software Impacts*, *14*, 100396. https://doi.org/10.1016/j.simpa.2022.100396

Bischl, B., Sonabend, R., Kotthoff, L., & Lang, M. (2024). Applied Machine Learning Using mlr3 in R. In *Applied Machine Learning Using mlr3 in R*. https://doi.org/10.1201/9781003402848

Chen, Gou, Zhang, Tian, & Li. (2023). Research on the Application of CEEMD-

LSTM-LSSVM Coupled Model in Regional Precipitation Prediction. *Water*, *15*(8), 1465. https://doi.org/10.3390/w15081465

Datta, P., & Faroughi, S. A. (2023). A multihead LSTM technique for prognostic prediction of soil moisture. *Geoderma*, *433*(January), 116452. https://doi.org/10.1016/j.geoderma.2023.116452

Deloffre, J., Jardani, A., Vu, M. T., & Massei, N. (2023). Use of long short-term memory network (LSTM) in the reconstruction of missing water level data in the River Seine. *Hydrological Sciences Journal-Journal Des Sciences Hydrologiques*, 1–19. https://doi.org/10.1080/02626667.2023.2221791

Dropka, N., Ecklebe, S., & Holena, M. (2021). *Real Time Predictions of VGF-GaAs Growth Dynamics by LSTM Neural Networks*. *11*(2). https://doi.org/10.3390/CRYST11020138

Dzakiroh, M. A. (2025). *Implementasi LSTM dalam Prediksi Keberangkatan Haji di Kementerian Agama Kota Malang*. *7*(1), 58–70.

Ehteram, M., Afshari Nia, M., Panahi, F., & Farrokhi, A. (2024). Read-First LSTM model: A new variant of long short term memory neural network for predicting solar radiation data. *Energy Conversion and Management*, *305*, 118267. https://doi.org/https://doi.org/10.1016/j.enconman.2024.118267

Elsheikh, A. H., Saba, A. I., Elaziz, M. A., Lu, S., Shanmugan, S., Muthuramalingam, T., Kumar, R., Mosleh, A. O., Essa, F. A., & Shehabeldeen, T. A. (2021). Deep learning-based forecasting model for COVID-19 outbreak in Saudi Arabia. *Process Safety and Environmental Protection*, *149*, 223–233. https://doi.org/https://doi.org/10.1016/j.psep.2020.10.048

Essai Ali, M. H., Abdel-Raman, A. B., & Badry, E. A. (2022). Developing Novel Activation Functions Based Deep Learning LSTM for Classification. *IEEE Access*, *10*(February), 97259–97275. https://doi.org/10.1109/ACCESS.2022.3205774

Fahmi, H. I. (2022). Pengaruh Penerapan Sistem Informasi Dan Komputerisasi Haji Terpadu (SISKOHAT) Terhadap Akses Layanan Haji Dalam Mewujudkan Kualitas Pelayanan Haji Di Kabupaten Garut. *Jurnal Publik*, *15*(2), 10–22. https://doi.org/10.52434/jp.v15i2.55

Farhan, N. (2020). Problematika Waiting List Dalam Penyelenggaraan Ibadah Haji Di Indonesia. *Jurnal Studi Agama Dan Masyarakat*, *12*(1), 57–80. https://doi.org/10.23971/jsam.v12i1.469

Fatunnisa, A., & Marcos, H. (2024). Prediksi Kelulusan Tepat Waktu Siswa SMK Teknik Komputer Menggunakan Algoritma Random Forest. *Jurnal Manajemen Informatika (JAMIKA)*, *14*(1), 101–111. https://doi.org/10.34010/jamika.v14i1.12114

Fredricka, J., & Ihsan, M. F. (2020). Analisis Sistem Keberangkatan Calon Jamaah

Haji Menggunakan Metode Weighted Product. *Jurnal Media Infotama*, *16*(2), 108–114. https://doi.org/10.37676/jmi.v16i2.1148

Freecenta, H. F., Puspaningrum, E. Y., & Maulan, H. (2022). Prediksi Curah Hujan Di Kab. Malang Menggunakan LSTM (Long Short Term Memory). *Jurnal Informatika Dan Sistem Informasi*, *3*(1), 51–55. https://doi.org/10.33005/jifosi.v3i1.448

Fuente, L. D. La, Ehsani, M. R., Gupta, H. V, & Condon, L. E. (2024). Toward interpretable LSTM-based modeling of hydrological systems. *Hydrology and Earth System Sciences*, *28*(4), 945–971. https://doi.org/10.5194/hess-28-945-2024

Gauch, M., Gauch, M., Kratzert, F., Klotz, D., Nearing, G., Nearing, G., Lin, J., & Hochreiter, S. (2021). Rainfall–runoff prediction at multiple timescales with a single Long Short-Term Memory network. *Hydrology and Earth System Sciences*, *25*(4), 2045–2062. https://doi.org/10.5194/HESS-25-2045-2021

Ge, Z., Yang, L., Li, J., Chen, Y., & Xu, Y. (2024). Bus Schedule Time Prediction Based on LSTM-SVR Model. *Mathematics*, *12*(22), 1–15. https://doi.org/10.3390/math12223589

Ghalyan, I. F., Ghalyan, N. F., & Ray, A. (2022). Optimal Window-Symbolic Time Series Analysis for Pattern Classification and Anomaly Detection. *IEEE Transactions on Industrial Informatics*, *18*(4), 2614–2621. https://doi.org/10.1109/TII.2021.3089199

Gkikas, S., & Tsiknakis, M. (2023). Automatic assessment of pain based on deep learning methods: A systematic review. *Computer Methods and Programs in Biomedicine*, *231*, 107365. https://doi.org/https://doi.org/10.1016/j.cmpb.2023.107365

Hafiz, A., & Nurdianty, M. S. (2024). Efektivitas penggunaan sistem informasi dan komputerisasi haji terpadu (siskohat) dalam pelayanan pendaftaran haji khusus pada kementerian agama provinsi dki jakarta. *Jurnal Manajemen Dakwah /Jurnal Manajemen Dakwah*, *12*(2). https://doi.org/10.15408/jmd.v12i2.40895

Hanafiah, A., Arta, Y., Nasution, H. O., & Lestari, Y. D. (2023). Penerapan Metode Recurrent Neural Network dengan Pendekatan Long Short-Term Memory (LSTM) Untuk Prediksi Harga Saham. *Bulletin of Computer Science Research*, *4*(1), 27–33. https://doi.org/10.47065/bulletincsr.v4i1.321

Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, *9*(8), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

Huang, D., Liu, J., Tian, Q., & Weng, H. T. (2024). *Training-set-free two-stage deep learning for spectroscopic data de-noising*. https://doi.org/10.48550/arxiv.2402.18830

Huda, Q., & Haeba, I. D. (2021). Hajj, Istitā'ah, and Waiting List Regulation in

Indonesia. *Al-'Adalah*, *18*(2), 193–212. https://doi.org/10.24042/adalah.v18i2.9903

Iskandaryan, D., Ramos, F., & Trilles, S. (2022). Comparison of Nitrogen Dioxide Predictions During a Pandemic and Non-pandemic Scenario in the City of Madrid using a Convolutional LSTM Network. *International Journal of Computational Intelligence and Applications*, *21*(02), 2250014:1-2250014:11. https://doi.org/10.1142/s1469026822500146

Islam, T., Hafiz, M. S., Jim, J. R., Kabir, M. M., & Mridha, M. F. (2024). A systematic review of deep learning data augmentation in medical imaging: Recent advances and future research directions. *Healthcare Analytics*, *5*, 100340. https://doi.org/https://doi.org/10.1016/j.health.2024.100340

Jasmi, K. A. (2023). *Pensyariatan Haji dalam Islam : Surah al-Baqarah (2 : 196-203). January 2020*, 196–203.

Jin, Y., Li, H., Dou, Q., Chen, H., Qin, J., Fu, C.-W., & Heng, P.-A. (2020). Multi-task recurrent convolutional network with correlation loss for surgical video analysis. *Medical Image Analysis*, *59*, 101572. https://doi.org/https://doi.org/10.1016/j.media.2019.101572

Kaddoura, S., & Nassar, R. (2024). EnhancedBERT: A feature-rich ensemble model for Arabic word sense disambiguation with statistical analysis and optimized data collection. *Journal of King Saud University - Computer and Information Sciences*, *36*(1), 101911. https://doi.org/https://doi.org/10.1016/j.jksuci.2023.101911

Khan, E. A., & Shambour, M. K. (2023). An optimized solution for the transportation scheduling of pilgrims in Hajj using harmony search algorithm. *Journal of Engineering Research (Kuwait)*, *11*(2), 100038. https://doi.org/10.1016/j.jer.2023.100038

Kiramy, R. Al, Permana, I., & Marsal, A. (2024). *Comparison of RNN and LSTM Algorithm Performance in Predicting the Number of Umrah Pilgrims at PT . Hajar Aswad Perbandingan Performa Algoritma RNN dan LSTM dalam Prediksi Jumlah Jamaah Umrah pada PT . Hajar Aswad*. *4*(October), 1224–1234.

Kumar Dubey, A., Kumar, A., García-Díaz, V., Kumar Sharma, A., & Kanhaiya, K. (2021). Study and analysis of SARIMA and LSTM in forecasting time series data. *Sustainable Energy Technologies and Assessments*, *47*(August 2020), 101474. https://doi.org/10.1016/j.seta.2021.101474

Lee, G. Y., Alzamil, L., Doskenov, B., & Termehchy, A. (2021). A Survey on Data Cleaning Methods for Improved Machine Learning Model Performance. In *arXiv: Databases*. https://arxiv.org/pdf/2109.07127.pdf

Li, J., Wu, G., Zhang, Y., & Shi, W. (2024). Optimizing flood predictions by integrating LSTM and physical-based models with mixed historical and

simulated data. *Heliyon*, *10*(13), e33669. https://doi.org/10.1016/j.heliyon.2024.e33669

Li, P., Zhang, J., & Krebs, P. (2022). Prediction of Flow Based on a CNN-LSTM Combined Deep Learning Approach. *Water*, *14*(6), 993. https://doi.org/10.3390/w14060993

Li, Y., Zhu, Z., Liu, Z., Zhou, Z., & Guo, Y. (2023). Quantitative trading strategy and portfolio optimization analysis based on LSTM model. *Highlights in Business, Economics and Management*, *14*, 219–226. https://doi.org/10.54097/hbem.v14i.9194

Lin, C. Y., & Lobo Marques, J. A. (2024). Stock market prediction using artificial intelligence: A systematic review of systematic reviews. *Social Sciences & Humanities Open*, *9*, 100864. https://doi.org/https://doi.org/10.1016/j.ssaho.2024.100864

Lin, L., Tan, Y., Zeng, G., Zhou, M., & Zhou, W. (2024). *Research on Prediction Model Based on Improved LSTM*. https://doi.org/10.1109/CISCE62493.2024.10653120

Liu, H., Xu, H., Yan, Y., Cai, Z., Sun, T., & Li, W. (2020). Bus Arrival Time Prediction Based on LSTM and Spatial-Temporal Feature Vector. *IEEE Access*, *8*, 11917–11929. https://doi.org/10.1109/ACCESS.2020.2965094

Liu, Y., Duan, J., & Meng, J. (2020). Difference Attention Based Error Correction LSTM Model for Time Series Prediction. *Journal of Physics: Conference Series*, *1550*(3). https://doi.org/10.1088/1742-6596/1550/3/032121

Masood, Z., Gantassi, R., Ardiansyah, & Choi, Y. (2022). A Multi-Step Time-Series Clustering-Based Seq2Seq LSTM Learning for a Single Household Electricity Load Forecasting. *Energies*, *15*(7), 2623. https://doi.org/10.3390/en15072623

Mohd, S., Syed, J., & Musa, L. (2024). *Istiṭāʿah in Hajj : An Analysis on Malaysia Hajj Queuing System*. *21*(2), 209–232.

Nandavita, A. Y., & Islahuddin, A. N. (2022). Pengaruh Antrian Haji Terhadap Minat Masyarakat Melaksanakan Ibadah Haji Di Kota Metro. *Multazam : Jurnal Manajemen Haji Dan Umrah*, *1*(2), 99. https://doi.org/10.32332/multazam.v1i2.5374

Ningrum, A. A., Syarif, I., Gunawan, A. I., Satriyanto, E., & Muchtar, R. (2021). Algoritma Deep Learning-LSTM untuk Memprediksi Umur Transformator. *Jurnal Teknologi Informasi Dan Ilmu Komputer*, *8*(3), 539–548. https://doi.org/10.25126/jtiik.2021834587

Patil, P. A., Rajpure, A., Thool, A., Gheware, S., & Moolya, A. (2022). Stock Market Prediction using LSTM. *International Journal of Advanced Research in Science, Communication and Technology*, 483–486. https://doi.org/10.48175/ijarsct-3336

Peerthum, Y., & Stamp, M. (2023). An empirical analysis of the shift and scale parameters in BatchNorm. *Information Sciences*, *637*, 1–24. https://doi.org/10.1016/j.ins.2023.118951

Pipin, S. J., Purba, R., & Kurniawan, H. (2023). Prediksi Saham Menggunakan Recurrent Neural Network (RNN-LSTM) dengan Optimasi Adaptive Moment Estimation. *Journal of Computer System and Informatics (JoSYC)*, *4*(4), 806–815. https://doi.org/10.47065/josyc.v4i4.4014

Prameswari, E., Afifudin, & Suyeno. (2021). Implementasi Siskohat Dalam Pelayanan Haji Studi Pada Kantor Kementerian Agama Kota Malang. *Jurnal Respon Publik*, *15*(7), 13–20.

Purnama, Y. (2022). *Penjelasan 45 Hadits Yang Banyak Disalah-Pahami*. 1–331.

Ramadhan, F. A., & Nathasia, N. D. (2025). *Prediksi Harga Dan Kinerja Aset Bitcoin Menggunakan Algoritma Long Short-Term Memory*. *15*(1), 68–76.

Ran, X., Shan, Z., Fang, Y., & Lin, C. (2019). An LSTM-Based Method with Attention Mechanism for Travel Time Prediction. *Sensors*, *19*(4). https://doi.org/10.3390/S19040861

Rokhsat-Yazdi, E., Rahnamayan, S., Amirinia, H., & Ahmed, S. (2020). *Optimizing LSTM Based Network For Forecasting Stock Market*. 1–7. https://doi.org/10.1109/CEC48606.2020.9185545

Roni Merdiansah, Khofifah Wulandari, Mentari Hasibuan, & Yuyun Umaidah. (2024). Perbandingan Kinerja Model RNN, LSTM, dan BLSTM dalam Memprediksi Jumlah Gempa Bulanan di Indonesia. *Jurnal Penelitian Rumpun Ilmu Teknik*, *3*(1), 262–277. https://doi.org/10.55606/juprit.v3i1.3466

Sahu, B. (2022). Seminal Stacked Long Short-Term Memory (SS-LSTM) Model for Forecasting Particulate Matter (PM2.5 and PM10). *Atmosphere*, *13*(10), 1726. https://doi.org/10.3390/atmos13101726

Salihoglu. (2023). Enhancing next destination prediction An application of LSTM using real-world airline data. *Nucl. Phys.*, *13*(1), 104–116.

Santosa, M. P. B., Ginantra, N. L. W. S. R., Iswara, I. B. A. I., & Putra, D. M. D. U. (2024). Analysis Cryptocurrency Prediction Price Using Recurrent Neural Network (RNN) Gate Recurrent Unit (GRU) Long Short-Term Memory (LSTM). *2024 Ninth International Conference on Informatics and Computing (ICIC)*, 1–5. https://doi.org/10.1109/ICIC64337.2024.10957408

Saputra, A. W., Wibawa, A. P., Pujianto, U., Utama, A. B. P., & Nafalski, A. (2022). LSTM-based Multivariate Time-Series Analysis: A Case of Journal Visitors Forecasting. *Ilkom Jurnal Ilmiah*, *14*(1), 57–62. https://doi.org/10.33096/ilkom.v14i1.1106.57-62

Shan, S., Li, C., Wang, Y., Fang, S., Zhang, K., & Wei, H. (2024). A deep learning model for multi-modal spatio-temporal irradiance forecast. *Expert Systems*

with Applications, *244*, 122925. https://doi.org/https://doi.org/10.1016/j.eswa.2023.122925

Sharma, C. M., & Chariar, V. M. (2024). Diagnosis of mental disorders using machine learning: Literature review and bibliometric mapping from 2012 to 2023. *Heliyon*, *10*(12), e32548. https://doi.org/https://doi.org/10.1016/j.heliyon.2024.e32548

Shi, Z., Xu, M., & Pan, Q. (2021). 4-D Flight Trajectory Prediction with Constrained LSTM Network. *IEEE Transactions on Intelligent Transportation Systems*, *22*(11), 7242–7255. https://doi.org/10.1109/TITS.2020.3004807

Si, P., & Mart, H. (2020). *Métricas de evaluación de los modelos*. 12–14.

Sokooti, H., Yousefi, S., Elmahdy, M. S., Lelieveldt, B. P. F., & Staring, M. (2021). Hierarchical Prediction of Registration Misalignment Using a Convolutional LSTM: Application to Chest CT Scans. *IEEE Access*, *9*, 62008–62020. https://doi.org/10.1109/ACCESS.2021.3074124

Sudarsono, B. G., Leo, M. I., Santoso, A., & Hendrawan, F. (2021). Analisis Data Mining Data Netflix Menggunakan Aplikasi Rapid Miner. *JBASE - Journal of Business and Audit Information Systems*, *4*(1), 13–21. https://doi.org/10.30813/jbase.v4i1.2729

Suzami, A., Hudaya, C., & Rodianto. (2021). Penerapan Sistem Komputerisasi Haji Terpadu (Siskohat) Terhadap Peningkatan Layanan Haji Pada Kantor Kementerian Agama Kabupaten Sumbawa. *Jurnal TAMBORA*, *5*(2), 97–104. https://doi.org/10.36761/jt.v5i2.1131

Tong, X., Wang, J., Li, C., Zhang, Z., Wu, T., Wang, H., & Wang, Y. (2022). LS-LSTM-AE: Power load forecasting via Long-Short series features and LSTM-Autoencoder. *Energy Reports*, *8*, 596–603. https://doi.org/10.1016/j.egyr.2021.11.172

Umpu Singa, H. A., Saleh, M., & Annur, A. F. (2022). Efektivitas Siskohat Dalam Pelayanan Pendaftaran Ibadah Haji Di Kantor Kementerian Agama Kota Metro. *Multazam : Jurnal Manajemen Haji Dan Umrah*, *2*(1), 37. https://doi.org/10.32332/multazam.v2i1.5240

Vural, N., Ilhan, F., & Yilmaz, S. (2022). Achieving Online Regression Performance of LSTMs With Simple RNNs. *IEEE Transactions on Neural Networks and Learning Systems*, *33*(12), 7632–7643. https://doi.org/10.1109/tnnls.2021.3086029

Wang, C., Wang, X., Jing, X., Yokoi, H., Huang, W., Zhu, M., Chen, S., & Li, G. (2022). Towards high-accuracy classifying attention-deficit/hyperactivity disorders using CNN-LSTM model. *Journal of Neural Engineering*, *19*(4), 46015. https://doi.org/10.1088/1741-2552/ac7f5d

Wardana, M. H. E., & Suhartini, D. (2023). Analisis Aspek Akuntabilitas

Pengelolaan Dana Haji Melalui Aplikasi SISKOHAT. *Jurnal Ilmiah Universitas Batanghari Jambi*, *23*(1), 24. https://doi.org/10.33087/jiubj.v23i1.2580

Wu, J., Du, B., Wu, Q., Shen, J., Zhou, L., Cai, C., Zhai, Y., Wei, W., & Zhou, Q. (2021). A Hybrid LSTM-CPS Approach for Long-Term Prediction of Train Delays in Multivariate Time Series. *Future Transportation*, *1*(3), 765–776. https://doi.org/10.3390/futuretransp1030042

Yang, Y. (2022). Application of LSTM Neural Network Technology Embedded in English Intelligent Translation. *Computational Intelligence and Neuroscience*, *2022*, 1–9. https://doi.org/10.1155/2022/1085577

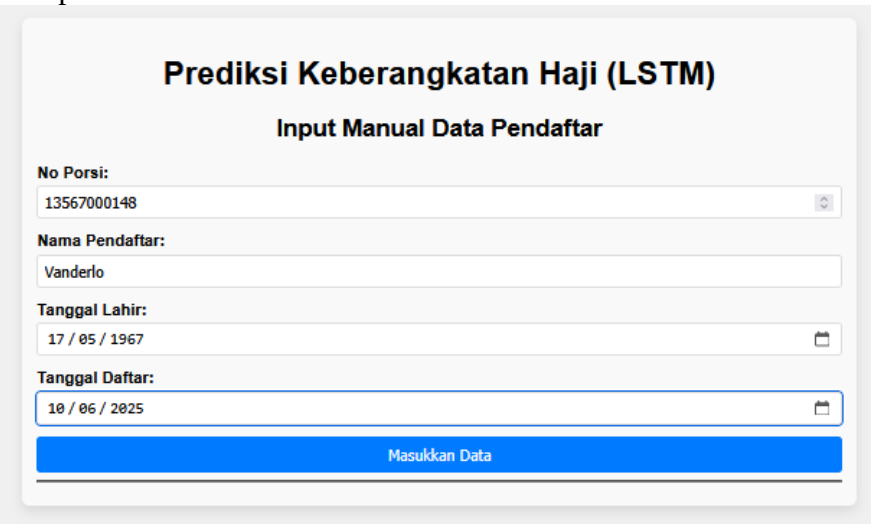
Zhao, Y., Zhang, X., Shang, Z., & Cao, Z. (2022). DA-LSTM-VAE: Dual-Stage Attention-Based LSTM-VAE for KPI Anomaly Detection. *Entropy*, *24*(11), 1613. https://doi.org/10.3390/e24111613

Zipporah, B., & Christopher, D. (2023). Residual U-Net Architecture for Retinal Layers in OCT Images with Choroidal Neovascularization. *SSRG International Journal of Electrical and Electronics Engineering*, *10*(5), 205–212. https://doi.org/10.14445/23488379/ijeee-v10i5p119

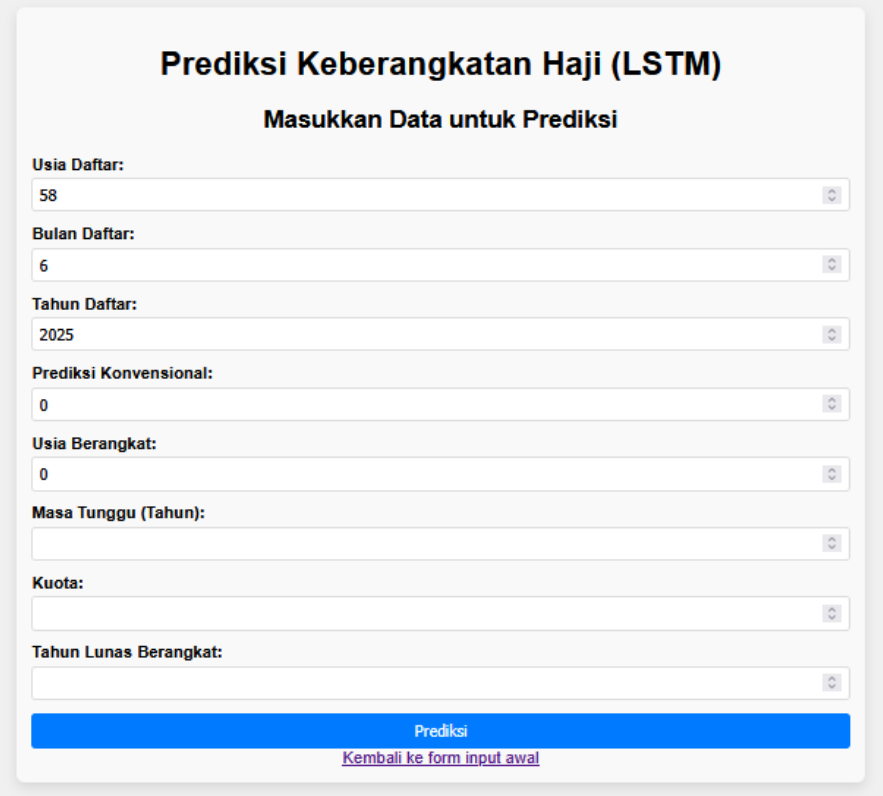
LAMPIRAN

Lampiran 1

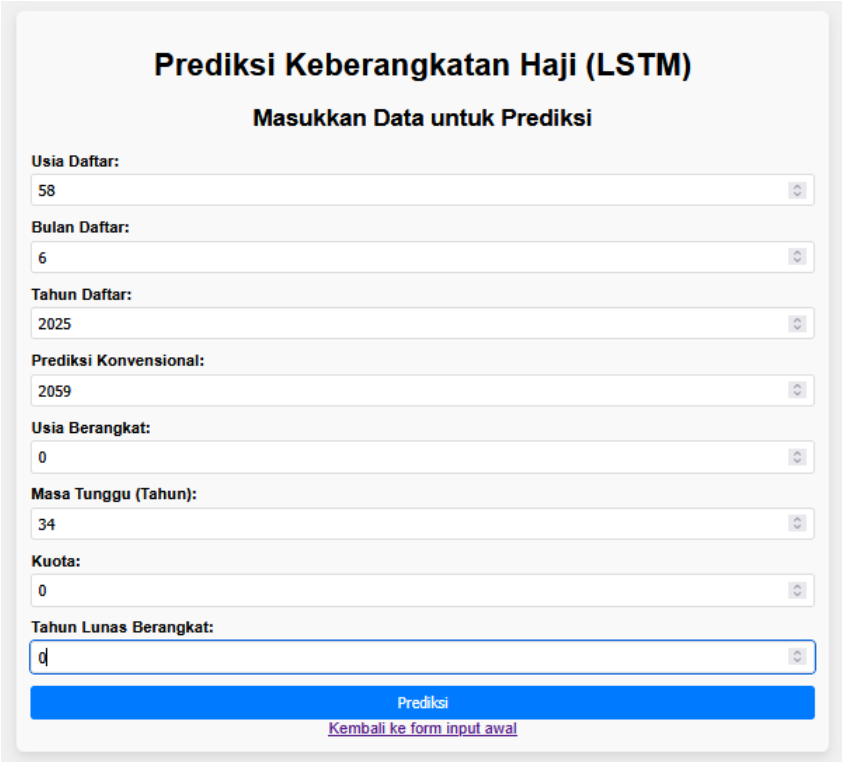
Tampilan awal



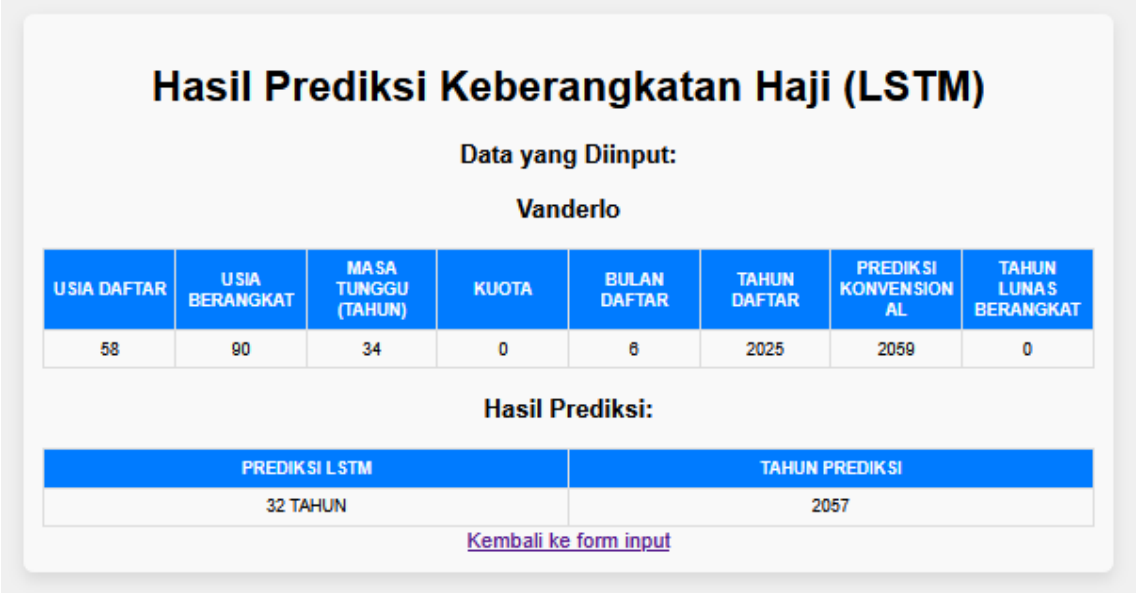
Tampilan setelah memasukkan data manual, maka usia daftar, bulan daftar, tahun daftar terisi secara otomatis



Tampilan Setalah memberi nilai pada form masa tunggu maka form prediksi konvensional terisi otomatis



Hasil Prediksi Menggunakan model LSTM dimana usia berangkat juga diprediksi



Lampiran 2

Surat Izin Penelitian

KEMENTERIAN AGAMA REPUBLIK INDONESIA
KANTOR KEMENTERIAN AGAMA KOTA MALANG
Jalan Raden Panji Suroso Nomor 2 Malang 65126
Telepon (0341) 491605
Website: kemenag.kotamalang.*go.id* ; *E-mail*: kotamalang@kemenag.go.id

Nomor : B- 2170/Kk.13.25/1/HM.00/8/2024 27 Agustus 2024
Sifat : Biasa
Lampiran : -
Hal : Izin Penelitian

Yth. Dekan Fakultas Sains dan Teknologi
UIN MALIKI Malang

Menindaklanjuti surat dari Dekan Fakultas Sains dan Teknologi UIN MALIKI Malang Nomor : B-97.O/FST.01/TL.00/08/2024 tanggal 23 Agustus 2024, perihal Permohonan Izin Penelitian, dengan ini kami sampaikan bahwa pada dasarnya menyetujui/tidak keberatan memberikan ijin kepada:

Nama : Mutiara Aprillia Dzakiroh
NIM : 210605110032
Program Studi : Teknik Informatika
Judul : Implementasi LSTM Untuk Prediksi Jadwal Pemberangkatan Haji dan Pengujian Usability SISKOHAT di Kemenag Kota Malang
Jangka Waktu : 12 agustus 2024 s/d 12 september 2024

mengadakan penelitian yang dilaksanakan di Kantor Kementerian Agama Kota Malang dengan ketentuan sebagai berikut:

1. Selama kegiatan penelitian mentaati tata tertib yang berlaku.
2. Setelah selesai kegiatan penelitian memberikan laporan secara tertulis kepada Kepala Kantor Kemenag Kota Malang

Untuk diketahui, seluruh layanan kementerian agama kota malang tanpa biaya dan seluruh pegawai kementerian agama kota malang tidak menerima gratifikasi. Salam Integritas!

Demikian atas perhatiannya disampaikan terima kasih.

a/n Kepala,
Kasubbag Tata usaha

Nurul Istiqomah

Dokumen ini telah ditanda tangani secara elektronik.
Token : 9cKqT8

Lampiran 3

Data

https://data.mendeley.com/datasets/4r9v52g82w/1