Analisis Waktu Latensi Pada Sistem Peringatan Dini Banjir dan Tanah Longsor

Laura Gracella[1], Syabilla Anandhira Muzzaki[2], Shofiyah Zahidah[3], Hariyanto[4]

Instrumentasi dan Kontrol Industri, Teknik Elektro, Politeknik Negeri Jakarta, Jl. Prof. DR. G.A. Siwabessy, Depok, 16425, Indonesia
Politeknik Negeri Jakarta, Jl. Prof. DR. G.A. Siwabessy, Depok, 16425, Indonesia

E-mail: laura.gracella.te22@mhsw.pnj.ac.id

Abstrak

Keterlambatan informasi pada sistem peringatan dini bencana sering dipicu oleh tingginya latensi transmisi menuju *cloud*. Penelitian ini bertujuan menganalisis performa temporal arsitektur pemantauan banjir dan tanah longsor dengan memindahkan komputasi secara lokal (*on-device*). Sistem mengintegrasikan sensor fisik maket dengan Jetson Nano sebagai *gateway* utama. Alur kerja berjalan otonom melalui prapemrosesan, *buffering* data (48 *timestep* banjir; 60 *timestep* longsor), inferensi model LSTM berformat TFLite, hingga kalkulasi tunda waktu. Pengujian dilakukan melalui metode *stress testing* dengan menginjeksikan 181 data sintetis kontinu yang merepresentasikan status Aman, Siaga, dan Bahaya. Hasil menunjukkan efisiensi komputasi lokal yang tinggi, di mana total waktu pemrosesan internal Jetson Nano berada di bawah 4 ms, dengan tunda inferensi LSTM berkisar antara 1,71 ms hingga 2,42 ms berkat akselerasi GPU CUDA. Pada tahap distribusi, transmisi data ke *web dashboard* lokal mencatatkan latensi sub-milidetik (< 0,1 ms) via WebSocket, sedangkan tunda jaringan eksternal untuk notifikasi Telegram Bot terkendali di kisaran ~1 detik menggunakan logika pembatasan dinamis. Sistem ini membuktikan keandalan arsitektur komputasi lokal dalam memangkas latensi eksekusi secara signifikan .

Kata kunci: Peringatan Dini, Waktu Latensi, Jetson Nano, LSTM, Banjir, Tanah Longsor

Abstract

Delayed information delivery in disaster early warning systems is frequently caused by high data transmission latency to cloud servers. This study analyzes the temporal performance of a localized computing architecture for flood and landslide monitoring by shifting computational workloads on-device. The system integrates physical mockup sensors with a Jetson Nano as the primary gateway. The autonomous workflow comprises preprocessing, data buffering (48 timesteps for floods; 60 timesteps for landslides), TFLite LSTM model inference, and delay logging. Evaluation was conducted via stress testing by injecting 181 continuous synthetic data samples representing Safe, Warning, and Danger statuses. The results demonstrate highly efficient local computation; the Jetson Nano's total internal processing time remains under 4 ms, with LSTM inference delays ranging from 1.71 ms to 2.42 ms due to CUDA GPU acceleration. At the output stage, data transmission to the local web dashboard achieved sub-millisecond latency (< 0.1 ms) via WebSocket, while external network delay for Telegram Bot notifications was contained at approximately 1 second using dynamic throttling logic. This system validates the reliability of localized computing architectures in significantly mitigating execution latency. .

Keywords: Early Warning, Latency Time, Jetson Nano, LSTM, Flood, Landslide

1. Pendahuluan

Kondisi geografis dan klimatologis Indonesia menempatkan wilayah ini dalam zona kerentanan tinggi terhadap bencana hidrometeorologi dan geologi, khususnya banjir dan tanah longsor. Dinamika anomali cuaca yang semakin ekstrem seringkali memperpendek transisi waktu antara pemicu lingkungan dan titik kritis terjadinya bencana. Dalam skenario kedaruratan tersebut, kegagalan sistem mitigasi konvensional dalam memberikan informasi secara tepat waktu dapat berakibat fatal [1]. Oleh karena itu, pengembangan

Sistem Peringatan Dini atau *Early Warning System* (EWS) yang memiliki tingkat responsivitas tinggi menjadi sebuah urgensi absolut guna menyediakan *golden time* untuk proses evakuasi dan mitigasi kerugian bagi masyarakat di area terdampak [2].

Seiring dengan evolusi teknologi industri, penerapan *Internet of Things* (IoT) telah mendisrupsi metode pemantauan kebencanaan menuju otomatisasi yang terdistribusi dan masif. Pemantauan lingkungan yang komprehensif menuntut arsitektur *multi-sensor* untuk mengkuantifikasi kompleksitas parameter fisik secara konkuren [3]. Guna memodelkan risiko banjir secara presisi, diperlukan instrumen akuisisi untuk mengukur laju presipitasi melalui sensor *tipping bucket* dan dinamika elevasi muka air menggunakan sensor ultrasonik JSN-SR04T [4]. Secara paralel, kerentanan lereng terhadap tanah longsor diobservasi melalui tingkat saturasi air pada tanah menggunakan *DFRobot Soil Moisture Sensor V2* serta anomali pergeseran kemiringan bidang spasial melalui sensor *accelerometer* dan *gyroscope* MPU6050 [5]. Integrasi keempat sensor ini pada mikrokontroler seperti ESP32 menghasilkan aliran data multivariat berbasis deret waktu (*time-series*) yang memiliki korelasi kausalitas tingkat tinggi.

Untuk mentransformasi aliran data mentah tersebut menjadi keputusan prediktif, implementasi kecerdasan buatan (*Artificial Intelligence*) menjadi suatu keharusan teknis. Pemodelan berbasis *Machine Learning*, khususnya algoritma *Long Short-Term Memory* (LSTM), secara empiris terbukti superior dalam menangani data *time-series* [6]. Arsitektur jaringan saraf berulang pada LSTM mampu mengidentifikasi dependensi jangka panjang dan mereduksi masalah *vanishing gradient*, sehingga model dapat mempelajari pola non-linear—seperti probabilitas longsor akibat akumulasi kelembapan pasca-hujan persisten—dan menghasilkan prediksi bencana dengan tingkat akurasi tinggi sebelum nilai ambang batas (*threshold*) kritis terlampaui.

Meskipun model LSTM menawarkan keunggulan analitik yang signifikan, implementasi praktisnya menghadapi hambatan fundamental pada arsitektur komputasi dan komunikasi data. Secara konvensional, EWS berbasis IoT beroperasi pada paradigma komputasi awan (*Cloud Computing*). Dalam skema ini, *node* sensor secara kontinu mengirimkan data melalui jaringan internet menuju *server cloud* terpusat untuk dieksekusi oleh model AI. Namun, pendekatan tersentralisasi ini memunculkan permasalahan latensi (*latency*) yang kritis [3]. Ketergantungan penuh pada *bandwidth* jaringan seluler atau internet publik membuat sistem sangat rentan terhadap fluktuasi *delay* transmisi [7]. Pada saat terjadi cuaca ekstrem, infrastruktur telekomunikasi kerap mengalami degradasi performa atau pemutusan koneksi (*downtime*). Keterlambatan waktu respon (*response time*) dalam proses transmisi, komputasi di *cloud*, hingga pengiriman kembali sinyal aktuasi dapat mendegradasi, bahkan menggagalkan fungsi sistem peringatan dini yang menuntut kapabilitas pemrosesan waktu-nyata (*real-time processing*) [8].

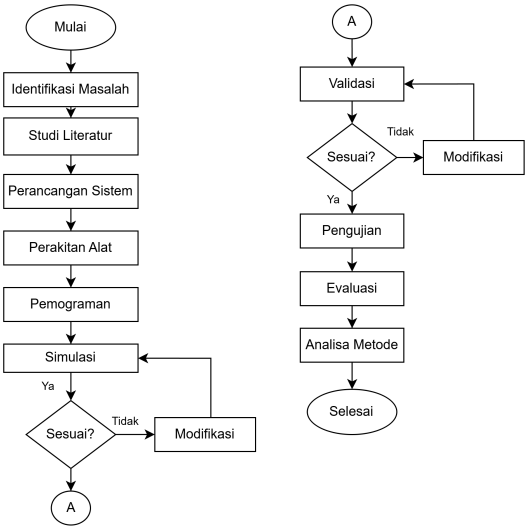
Sebagai solusi strategis untuk mereduksi hambatan latensi tersebut, migrasi beban komputasi menuju lapisan tepi jaringan melalui arsitektur *Edge Computing* menjadi paradigma baru yang sangat menjanjikan [8][9]. Melalui arsitektur ini, beban inferensi model LSTM tidak lagi diserahkan kepada *cloud*, melainkan dieksekusi secara lokal pada perangkat *Edge Server* (seperti *Single Board Computer* Jetson Nano) yang berada dalam kedekatan fisik dengan *node* sensor. Pendekatan ini mendesentralisasi kekuatan komputasi, secara drastis memangkas waktu propagasi transmisi data, meminimalkan konsumsi *bandwidth*, dan memastikan sistem prediksi tetap dapat beroperasi secara otonom meskipun konektivitas internet ke *server* global terputus.

Pemrosesan *real-time* EWS pada arsitektur *Edge Computing* menuntut analisis teknis yang komprehensif terhadap waktu respon dari hulu ke hilir (*end-to-end response time*). Kinerja sistem ini bergantung pada efisiensi rantai komunikasi berjenjang: dimulai dari akuisisi data sensor oleh ESP32, transmisi *payload* dengan beban *overhead* rendah melalui protokol MQTT (*Message Queuing Telemetry Transport*), eksekusi komputasi LSTM pada lingkungan *Edge Server*, visualisasi data dinamis ke *Web Dashboard* via protokol *WebSocket*, hingga diseminasi peringatan melalui *Telegram Bot API*. Menganalisis performa sistem pada topologi yang kompleks ini, terutama dengan keterbatasan sumber daya pada perangkat komputasi tepi, membutuhkan pengujian simulasi berbeban tinggi menggunakan data sintetis (*dummy data*). Pembedahan dan analisis mendalam terhadap komponen metrik performansi ini sangat esensial untuk memetakan secara fisis sejauh mana reduksi latensi dapat dicapai pada tiap lapisan arsitektur hibrida tersebut. Berdasarkan urgensi teknis dan fungsional tersebut, maka penelitian ini dilakukan dengan judul "Analisis Waktu Latensi Pada Sistem Peringatan Dini Banjir dan Tanah Longsor".

2. Metode Penelitian

2.1 Alur Penelitian

Tahapan pelaksanaan penelitian ini dilakukan secara sistematis dan terstruktur guna menjamin validitas serta akurasi analisis data yang diperoleh. Seluruh rangkaian aktivitas riset, mulai dari perumusan masalah, perancangan sistem fisik dan kecerdasan buatan, hingga tahapan pengujian performa.



Gambar 1. Metodologi penelitian

Penelitian ini dilakukan melalui tahapan sistematis guna menjamin validitas dan akurasi sistem peringatan dini berbasis *Edge AI*. Alur penelitian diawali dengan tahap perancangan sistem dan pemrograman perangkat, di mana arsitektur model *Long Short-Term Memory* (LSTM) diintegrasikan untuk menangani klasifikasi multi-kelas (Aman, Siaga, Bahaya). Untuk mengoptimalkan kinerja pada Jetson Nano, model dikembangkan menggunakan *framework* TensorFlow/Keras dan dikonversi ke format TensorFlow Lite (TFLite). Dalam implementasinya, dirancang dua model terpisah sesuai karakteristik fisik bencana: model banjir menggunakan jendela temporal 48 *timestep* dengan variabel elevasi air (Lt) dan curah hujan (Rt), sementara model tanah longsor menggunakan jendela temporal 60 *timestep* dengan variabel kelembapan tanah (Mt) dan sudut kemiringan lereng (Gt).

Setelah perancangan, penelitian berlanjut ke tahap simulasi dengan menginjeksikan data multivariat *real-time* ke dalam maket. Pada fase ini, data sensor dikirimkan secara sekuensial untuk membentuk matriks *buffer* temporal, yang memungkinkan dilakukan modifikasi ulang jika ditemukan ketidaksesuaian pembacaan fisik atau kegagalan sinkronisasi data. Tahap validasi kemudian dilakukan menggunakan 19.000 sampel data riil dengan rasio pembagian kronologis 80:20 (15.200 sampel latih dan 3.800 sampel uji). Proses ini bertujuan untuk mengukur performa model melalui metrik *accuracy, precision, recall,* serta *F1-score*. Apabila model telah memenuhi ambang batas keandalan, alur penelitian diakhiri dengan pengujian beban menyeluruh, evaluasi performa sistem, serta analisis mendalam mengenai efisiensi waktu respons *real-time* untuk memastikan kesiapan operasional sistem.

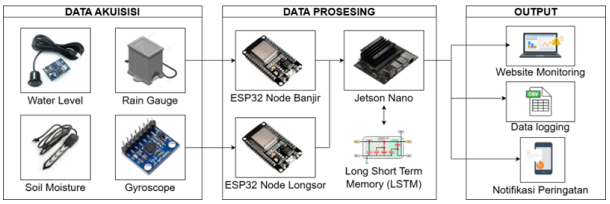
2.2 Metode Pengujian Sistem

Pengujian performa arsitektur sistem dilakukan menggunakan metode injeksi data sintetis (dummy) secara luring (offline) sebanyak 181 sampel data kontinu. Penggunaan data dummy ini dirancang khusus untuk melakukan stress testing pada infrastruktur perangkat keras guna mengukur konsistensi waktu tanggap sistem tanpa harus melakukan simulasi fisik berulang pada maket. Struktur payload dan nilai parameter pada data dummy ini dibuat identik dengan data sensor asli, mencakup fluktuasi nilai variabel kelembapan tanah, curah hujan, tinggi muka air, dan sudut kemiringan lereng yang merepresentasikan transisi kondisi dari status Aman, Siaga, hingga Bahaya.

Dalam skenario pengujian ini, data dummy diinjeksikan secara berurutan ke dalam sistem untuk memicu seluruh alur komputasi pada Jetson Nano. Proses pengujian ini mengevaluasi kesiapan arsitektur dalam menangani beban kerja secara simultan, mulai dari kecepatan pengisian matriks buffer sesuai timestep (48 untuk banjir dan 60 untuk longsor), kecepatan komputasi inferensi model LSTM, hingga efisiensi pemrosesan paralel saat memperbarui komponen luaran (WebSocket dashboard, log .csv, dan API Telegram). Data tunda waktu yang dihasilkan kemudian diakumulasikan untuk dianalisis nilai rata-rata (mean) dan stabilitas latensinya.

2.3 Perancangan Sistem

Secara konseptual, sistem peringatan dini berbasis IoT ini dirancang menggunakan arsitektur empat lapis (four-layer architecture) untuk mengisolasi fungsi setiap komponen secara modular. Keempat lapisan tersebut meliputi lapisan persepsi, jaringan, komputasi tepi, dan aplikasi. Pada lapisan persepsi, mikrokontroler ESP32 berfungsi sebagai node pengumpul data parameter fisik dari maket. Data tersebut kemudian ditransmisikan melalui lapisan jaringan menuju Jetson Nano yang berada di lapisan komputasi tepi (edge computing). Berbeda dengan sistem konvensional berbasis cloud, Jetson Nano bertindak sebagai server lokal yang langsung melakukan preprocessing dan inferensi model LSTM secara real-time. Terakhir, pada lapisan aplikasi, hasil prediksi disebarkan secara simultan ke web dashboard untuk visualisasi data dan ke API Telegram untuk mengirimkan notifikasi peringatan dini.



Gambar 2. Blok Diagram Alur Sistem

Diagram blok pada Gambar 2 menggambarkan arsitektur aliran data sistem yang terbagi menjadi tiga tahapan utama. Proses diawali pada tahap Data Akuisisi, di mana empat sensor (*Soil Moisture*, *Rain Gauge*, *Water Level*, dan *Gyroscope*) mengumpulkan parameter lingkungan dari maket secara bersamaan. Data multivariat tersebut kemudian diteruskan ke tahap Data Processing untuk diproses secara lokal pada *edge gateway* Jetson Nano, yang mengeksekusi inferensi model *Deep Learning* LSTM untuk menghasilkan prediksi status kebencanaan. Pada tahap akhir (Output), hasil keputusan tersebut didiseminasikan secara *real-time* melalui tiga jalur simultan: visualisasi dinamis pada *Website Monitoring*, pengiriman notifikasi peringatan dini ke Telegram pengguna, dan pembukuan og data ke dalam penyimpanan lokal berformat .csv

Untuk meminimalkan tunda waktu (*delay*) pemrosesan lokal, *web dashboard* dan *edge AI* (model LSTM) dijalankan secara bersamaan pada perangkat yang sama. Konfigurasi ini memungkinkan proses komunikasi data antar-komponen internal di dalam Jetson Nano diisolasi melalui jaringan *Localhost Loopback Interface* (127.0.0.1) menggunakan protokol WebSocket. Karena pertukaran data berjalan langsung di dalam memori RAM, transmisi data lokal terbebas dari hambatan fisik jaringan luar seperti kabel LAN atau Wi-Fi, sehingga mampu mengoptimalkan kecepatan waktu tanggap sistem pada level mikrodetik

Gambar 3. Web Dashboard

Gambar 3 merupakan implementasi antarmuka *Web Dashboard* pada lapisan aplikasi yang berfungsi sebagai pusat visualisasi data terintegrasi. Halaman utama ini memuat empat indikator numerik yang menampilkan nilai parameter terkini secara *real-time* dari tiap sensor (*Rain Gauge*, *Water Level*, *Soil Moisture*, dan *Gyroscope*). Selain itu, terdapat grafik untuk memetakan tren fluktuasi data fisis, serta dua panel indikator status utama hasil inferensi model *Edge AI* yang menampilkan tingkat risiko kedaruratan.

Gambar 4. Telegram Bot

Saat model LSTM memprediksi status "Bahaya", Jetson Nano mengirimkan paket data terenkripsi via HTTPS POST ke *API Gateway Server* Telegram. Melalui mekanisme ini, layanan *Pantau Bencana Bot* (Gambar 4) menyebarkan notifikasi darurat secara otonom ke ponsel pengguna menggunakan logika pembatasan dinamis, di mana pesan hanya dikirim jika terjadi transisi tingkat risiko bencana. Kemudian secara simultan, Jetson Nano juga melakukan operasi I/O untuk membukukan seluruh data sensor dan hasil prediksi ke dalam penyimpanan lokal berformat .csv. File ini disusun terstruktur berdasarkan baris waktu (*timestamp*) sebagai basis data lokal yang ringan dan efisien untuk pengarsipan.

2.4 Akuisisi Data Sensor

Proses akuisisi data dilakukan dengan mengumpulkan parameter lingkungan dari maket fisik sebagai sumber data latih model LSTM, serta menyiapkan data sintetis (dummy) untuk pengujian infrastruktur. Pada skenario banjir, parameter presipitasi diukur melalui sensor Tipping Bucket dan dinamika elevasi muka air diukur menggunakan sensor ultrasonik JSN-SR04T sebagai indikator utama kerentanan banjir [4]. Sementara itu, pada skenario tanah longsor, pengukuran memanfaatkan sensor Soil Moisture untuk saturasi air dan sensor Gyroscope untuk mendeteksi pergeseran lereng, di mana integrasi parameter tersebut secara multivariat terbukti krusial dalam mendeteksi sinyal peringatan dini tanah longsor dengan akurasi tinggi [12].

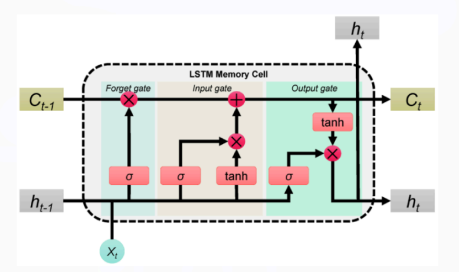
Dari hasil pemantauan sensor-sensor tersebut, diperoleh total data riil dari simulasi maket sebanyak 19.000 sampel deret waktu yang terdiri dari 9.000 sampel banjir dan 10.000 sampel tanah longsor. Seluruh data riil ini dibagi menggunakan rasio kronologis 80:20, menghasilkan 15.200 sampel sebagai data latih dan 3.800 sampel sebagai data uji validasi model. Di sisi lain, untuk pengujian performa arsitektur sistem dan analisis latensi pada Jetson Nano, digunakan 181 sampel data dummy kontinu dengan struktur payload yang dirancang identik dengan data asli. Penggunaan 181 data dummy ini difungsikan khusus sebagai stress testing lokal untuk mengukur kecepatan waktu tanggap dan konsistensi pemrosesan tanpa harus mengulang simulasi fisis pada maket.

2.5 Kriteria Ambang Batas Kebencanaan

Kriteria ambang batas (threshold) status Aman, Siaga, dan Bahaya ditetapkan secara empiris melalui kalibrasi dan eksperimen langsung yang disesuaikan secara presisi dengan dimensi dan karakteristik fisik maket, dengan rincian sebagai berikut:

1. Bencana Banjir: Status Aman berada pada JSN 0 cm dan tipping 0 mm. Status Siaga dipicu saat tinggi air mencapai 5 cm dan curah hujan 10 mm. Status Bahaya ditetapkan ketika tinggi air menembus ≥8 cm dengan akumulasi curah hujan ≥20 mm.
2. Tanah Longsor: Status Aman berada pada kelembapan tanah ≤40% dan kemiringan lereng 0°. Status Siaga tercapai saat kelembapan tanah meningkat hingga 80% dengan kemiringan ≥ 4°. Status Bahaya aktif saat kelembapan tanah ≥80% diikuti pergeseran lereng ekstrem di rentang ≥ 11°.Nilai ambang batas khusus maket ini digunakan sebagai basis pelabelan data pada model LSTM serta pemicu otomatisasi notifikasi Telegram dan pembaruan data pada web dashboard.

2.6 Model *Long Short-Term Memory* (LSTM)



Gambar 5 Arsitektur LSTM

Deep learning menggunakan jaringan saraf tiruan berlapis banyak untuk memodelkan hubungan nonlinier yang kompleks. Untuk data runtun waktu (*time series*), *Long Short-Term Memory* (LSTM) diterapkan sebagai arsitektur hasil pengembangan dari *Recurrent Neural Network* (RNN). Berdasarkan Gambar 5, LSTM mampu menangkap ketergantungan temporal jangka pendek maupun panjang menggunakan *memory cell* yang dikendalikan oleh mekanisme tiga gerbang utama, yaitu:.

- **Forget Gate (ft):** Berfungsi mengevaluasi memori sebelumnya (Ct-1) untuk menentukan informasi yang harus dipertahankan atau dihapus, berdasarkan input saat ini (xt) dan *hidden state* sebelumnya (hight-1), sesuai dengan Persamaan (2.1).

$$ft = \boldsymbol{\sigma}(Wf.[ht-1, xt] + bf)$$

- **Input Gate (it):** Berfungsi menyeleksi informasi baru yang akan masuk ke memori melalui kombinasi fungsi sigmoid (vektor seleksi) dan fungsi tanh (kandidat nilai memori baru), sesuai Persamaan (2.2).

$$it = \boldsymbol{\sigma}(Wi.[ht-1, xt] + bi)$$

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

- **Pembaruan Status Sel (Ct):** Proses krusial untuk memperbarui status memori utama dengan mengombinasikan data historis lama yang lolos sensor dan kandidat input baru, sesuai Persamaan (2.3).

$$C_t = f_t \odot C_{t-1} + i_t \odot \hat{C}_t$$

- **Output Gate (ot):** Berperan menentukan porsi dari memori sel saat ini yang akan diteruskan sebagai keluaran (*hidden state*), sesuai Persamaan (2.4).

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

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

Secara konkret, arsitektur jaringan saraf tiruan yang dirancang dalam penelitian ini terdiri dari tiga lapisan utama yang disusun secara sekuensial untuk menghasilkan klasifikasi multi-kelas berdasarkan mekanisme tersebut. Lapisan pertama adalah lstm_layer (LSTM) dengan kapasitas 64 *hidden units* yang menghasilkan dimensi keluaran (None, 64) dan memiliki 17.152 parameter terlatih. Lapisan ini dikonfigurasi dengan parameter return_sequences=False guna mengekstrak vektor fitur temporal tunggal dari akhir urutan sekuensial.

Selanjutnya, lapisan regularisasi dropout_layer diterapkan dengan dimensi keluaran tetap (None, 64) tanpa menambahkan parameter baru. Lapisan *dropout* ini berfungsi untuk mencegah terjadinya kegagalan generalisasi (*overfitting*) selama proses pelatihan model dengan cara menonaktifkan sebagian neuron secara acak. Sebagai lapisan terakhir, output_layer (*Dense Layer*) yang bersifat terhubung penuh (*fully connected*) digunakan dengan dimensi keluaran (None, 3) dan melibatkan 195 parameter. Lapisan densitas ini mentransformasikan fitur-fitur abstrak dari blok LSTM

menjadi nilai probabilitas untuk menentukan keputusan klasifikasi tiga status kebencanaan, yaitu Aman, Siaga, dan Bahaya.

3. Hasil dan Pembahasan

3.1 Cara Kerja Alat

Sistem beroperasi otonom dari hulu ke hilir pada Jetson Nano sebelum memasuki tahap pengujian performa. Proses diawali saat sensor JSN-SR04T, Tipping Bucket, Soil Moisture, dan Gyroscope pada maket mentransmisikan data fisik lingkungan ke Jetson Nano secara bersamaan. Di dalam Jetson Nano, data mentah melewati tahap prapemrosesan (preprocessing) untuk normalisasi format, lalu masuk ke dalam proses buffering data guna mengumpulkan deret waktu yang sesuai dengan kebutuhan timestep model, yaitu 48 timestep untuk skenario banjir dan 60 timestep untuk tanah longsor. Setelah matriks buffer terpenuhi, data tersebut diumpankan ke komponen AI untuk inferensi model LSTM berformat TFLite hingga menghasilkan prediksi status kebencanaan. Pada tahap pascapemrosesan (post-processing), sistem menerjemahkan hasil prediksi menjadi aksi luaran sekaligus menghitung kalkulasi tunda waktu (delay) pemrosesan di setiap titik alur data, yang langsung dibukukan ke penyimpanan lokal berformat .csv bersama log data sensor. Secara simultan, Jetson Nano mendistribusikan data ke Website Monitoring via WebSocket dan mengeksekusi request HTTPS POST ke API Gateway Server Telegram untuk mengirimkan notifikasi darurat menggunakan logika pembatasan dinamis saat status bahaya dipicu. Setelah seluruh alur terintegrasi, sistem dievaluasi melalui dua metode pengujian utama: pengujian keandalan model prediksi LSTM untuk validasi akurasi, dan pengujian analisis latensi untuk mengukur kecepatan respon infrastruktur perangkat keras.

3.2 Uji Waktu Latensi

Alur perekaman data mentah pada lapisan persepsi dari kondisi aman, siaga, hingga bahaya didokumentasikan secara berkala pada berkas log sensor banjir dan longsor. Tren perubahan parameter lingkungan yang dikirimkan oleh masing-masing perangkat *node* sensor disajikan pada Tabel 1 dan Tabel 2.

Tabel 1. Data Hasil Akuisisi Sensor pada Node Banjir

No	Waktu	Paket	Curah Hujan	Level Air	Status
1	15:21	1	0,1	0	Aman
2	15:21	2	0,2	0	Aman
3	15:21	3	0,3	0	Aman
..	..	..	..	..	..
101	15:38	100	10,2	4,2	Siaga
102	15:38	101	10,3	4,3	Siaga
103	15:38	102	10,4	4,4	Siaga
..	..	..	..	..	..
179	15:51	178	41,4	9,5	Bahaya
180	15:51	179	42	9,6	Bahaya
181	15:51	180	42,5	9,6	Bahaya

Mekanisme sensor Tipping Bucket mencatat laju presipitasi melalui frekuensi jungkitan akibat volume air buatan. Air hujan buatan ini mengalir ke bak penampungan maket sehingga memicu peningkatan tinggi muka air secara linier yang dideteksi oleh sensor ultrasonik JSN-SR04T. Hubungan kausalitas kedua variabel ini terlihat pada Tabel 1, di mana kenaikan akumulasi curah hujan dari 10,2 mm ke ≥41,4 mm secara linier berbanding lurus dengan melonjaknya elevasi air dari 4,2 cm hingga mencapai batas kritis ≥9,5 cm.

Tabel 2. Data Hasil Akuisisi Sensor pada Node Longsor

No	Waktu	Paket	Kelembapan tanah	gyroscope	Status
1	12:45	1	15,1	1,02	Aman
2	12:45	2	15,3	1,02	Aman
3	12:45	3	15,6	1,04	Aman
..	..	..	..	..	..
101	12:53	100	56,3	7,86	Siaga
102	12:53	101	56,8	8,06	Siaga
103	12:53	102	57,3	8,26	Siaga
..	..	..	..	..	..
179	13:00	178	87,4	71,2	Bahaya
180	13:00	179	87,6	75,9	Bahaya
181	13:00	180	87,9	78,4	Bahaya

Berdasarkan Tabel 2, pengujian maket membuktikan kausalitas antara kejenuhan air dan instabilitas lereng. Pada status Aman, kelembapan tanah yang rendah (15,1%–15,6%) menjaga kohesi partikel tetap kuat sehingga sudut lereng stabil (1,02°–1,04°). Saat simulasi hujan meningkatkan kelembapan ke 56,3%–57,3%, air mengisi rongga pori yang menambah beban massa lereng sekaligus menurunkan kekuatan geser (*shear strength*) tanah. Akibatnya, terjadi pergerakan awal

(rayapan) yang terekam giroskop pada sudut 7,86°–8,26° (Siaga). Fase kritis tercapai saat tanah menyentuh saturasi absolut (87,4%–87,9%), di mana air bertindak sebagai lubrikasi bidang gelincir [1] yang menghancurkan daya ikat tanah dan memicu longsor masif, terkonfirmasi oleh lonjakan sudut giroskop hingga 71,2°–78,4° (Bahaya).

3.2. Hasil Analisis

Setelah data parameter lingkungan berhasil diakuisisi, tahapan berikutnya adalah menganalisis unjuk kerja pemrosesan data dan efisiensi pengiriman informasi darurat. Analisis kuantitatif terhadap komputasi perangkat *edge gateway* Jetson Nano dalam mengeksekusi arsitektur LSTM.

Tabel 3. Distribusi Rata-Rata Waktu Komputasi Internal

Node	Timestep	Rata-rata preproses	Rata-rata inferensi	Rata-rata pascaproses
Banjir	48	0,19852	2,42701	0,05617
Longsor	60	0,22546	1,71694	0,05216

Berdasarkan Tabel 3, efisiensi komputasi lokal pada Jetson Nano terbukti sangat tinggi di seluruh tahapan pemrosesan model LSTM. Tahap prapemrosesan data mentah diselesaikan secara instan dalam waktu rata-rata 0,198 ms (banjir) dan 0,225 ms (longsor). Tahap inferensi model LSTM berformat TFLite menjadi titik konsumen waktu terbesar, yaitu rata-rata 2,427 ms pada 48 *timestep* banjir dan 1,716 ms pada 60 *timestep* longsor. Kecepatan sub-milidetik ini tercapai karena arsitektur model dieksekusi langsung di atas core GPU NVIDIA Jetson melalui akselerasi CUDA. Selanjutnya, tahap pascapemrosesan untuk translasi keputusan dan kalkulasi tunda waktu diselesaikan paling cepat di kisaran 0,052 ms hingga 0,056 ms. Secara akumulatif, total waktu komputasi internal lokal berada jauh di bawah 4 ms, membuktikan keandalan pemrosesan *intelligence* lokal dalam memangkas waktu tunda eksekusi.

Tabel 4. Karakteristik Rata-Rata Latensi Transmisi

node	status	Rata-rata latensi web	Latensi telegram
banjir	aman	0,066567	0,0
	Siaga	0,0656	972,4523
	bahaya	0,063815	1028,912
longsor	aman	0,082505	0,0
	Siaga	0,07525	996,2586
	bahaya	0,079173	1004,8115

Tabel 4 menunjukkan karakteristik kontras antara latensi transmisi lokal dan eksternal pada sistem. Pembaruan data pada *web dashboard* mencatatkan latensi ekstrim rendah di bawah 0,1 ms (0,063 ms–0,082 ms) pada seluruh status karena dasbor diletakkan secara paralel pada satu mesin fisik dengan Jetson Nano (*co-located*). Jalur ini memanfaatkan *Localhost Loopback Interface* via protokol WebSocket, sehingga transfer data diproses langsung di dalam memori RAM tanpa melalui sirkuit fisik kartu jaringan (NIC). Sebaliknya, latensi Telegram Bot bernilai 0,0 ms saat status Aman karena efisiensi logika pembatasan dinamis, namun melonjak hingga kisaran 972,45 ms–1028,91 ms (~1 detik) ketika mendeteksi status Siaga dan Bahaya. Lonjakan ini terjadi karena paket data HTTPS POST harus menembus jaringan internet publik menuju server *cloud* global Telegram sebelum didistribusikan ke ponsel pengguna.

4. Kesimpulan

Berdasarkan hasil perancangan dan pengujian sistem menggunakan 181 data injeksi sintetis pada maket, dapat disimpulkan bahwa arsitektur sistem peringatan dini banjir dan tanah longsor ini berhasil beroperasi secara otonom dengan performa temporal yang sangat efisien. Hubungan kausalitas fisis parameter lingkungan berhasil diklasifikasikan secara konsisten oleh model LSTM ke dalam status Aman, Siaga, dan Bahaya berdasarkan kriteria ambang batas fisik maket. Dari aspek performa, komputasi internal lokal pada Jetson Nano menunjukkan efisiensi tinggi dengan total waktu pemrosesan di bawah 4 ms, di mana inferensi model LSTM berformat TFLite hanya membutuhkan waktu 1,71 ms–2,42 ms berkat akselerasi GPU CUDA. Untuk distribusi luaran, transmisi data lokal ke *web dashboard* berjalan instan di bawah 0,1 ms memanfaatkan jaringan *Localhost Loopback* via WebSocket, sementara tunda jaringan eksternal pada Telegram Bot terkendali di kisaran ~1 detik yang diefisiensikan melalui logika pembatasan notifikasi dinamis.

Daftar Acuan

[1] R. Mulyandari, "Analisis kesiapsiagaan, kerentanan dan ketahanan masyarakat dalam menghadapi bencana alam (studi kasus di Desa Tegaltirto)," *Venus: Jurnal Publikasi Rumpun Ilmu Teknik*, vol. 3, no. 1, pp. 206–219, 2025.

[2] P. N. A. Reyhan, E. L. B. Purba, and L. Marlina, "Analisis Kesiapan BPBD Kota Binjai dalam Penerapan Kecerdasan Buatan untuk Sistem Peringatan Dini Bencana Banjir," *Bridge: Jurnal Publikasi Sistem Informasi dan Telekomunikasi*, vol. 3, no. 3, pp. 117-127, Agustus 2025.

[3] A. S. S. Ariyanto, D. N. Triwibowo, I. A. Ashari, and R. C. S. Haryono, "Implementasi dan Evaluasi Kinerja Sistem IoT Multi-Sensor Berbasis ESP32 untuk Pemantauan dan Peringatan Dini Lingkungan secara Real-Time," *JSAI : Journal Scientific and Applied Informatics*, vol. 9, no. 1, pp. 118-125, Januari 2026.

[4] T. D. Hendrawati, F. A. Wicaksana, I. Kumaran, and M. D. Abdillah, "Sistem Peringatan Dini Banjir Berbasis Internet of Things (IoT) dengan Integrasi Multi-Sensor dan Logika Fuzzy," *JTERA (Jurnal Teknologi Rekayasa)*, vol. 10, no. 2, pp. 131-140, Desember 2025.

[5] Satriyo, A. Haddad, Ramli, and S. A. Salim, "Implementasi Wireless Sensor Network pada Pendeteksi Tanah Longsor Berbasis Internet of Things (IoT) dan LoRa," *JIRE (Jurnal Informatika & Rekayasa Elektronika)*, vol. 8, no. 1, pp. 1-10, April 2025.

[6] A. Yoani, Sediono, M. F. F. Mardianto, and E. Pusporani, "Prediksi Jumlah Kejadian Banjir Bulanan di Indonesia Berdasarkan Analisis Long Short Term Memory," *G-Tech: Jurnal Teknologi Terapan*, vol. 7, no. 4, pp. 1663-1672, Oktober 2023.

[7] K. Kavya Sree, "An adaptive AI-driven multi-hazard prediction and early alert framework for real-time emergency response using sensor fusion and deep learning models," *SciTePress*, 2026.

[8] Y. Qomarullah, U. Ristian, and Kasliono, "Perbandingan QoS Protokol HTTP Dan MQTT Dalam Merespon Sistem Pendeteksi Kebakaran Menggunakan Notifikasi Bot Telegram," *Coding: Jurnal Komputer dan Aplikasi*, vol. 13, no. 3, pp. 216-223, 2025.

[9] A. Bourechak, O. Zedadra, M. N. Kouahla, A. Guerrieri, H. Seridi, and G. Fortino, "At the confluence of artificial intelligence and edge computing in IoT-based applications: A review and new perspectives," *Sensors*, vol. 23, no. 3, p. 1639, 2023.

[10] H. N. Saha *et al.*, "Internet of Things (IoT) architecture, concepts, technologies, challenges, and future directions," in *IEEE 8th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, 2017, pp. 1-7.

[11] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[12] D. Zhang, K. Wei, Y. Yao, J. Yang, G. Zheng, and Q. Li, "Capture and prediction of rainfall-induced landslide warning signals using an attention-based temporal convolutional neural network and entropy weight methods," *Sensors*, vol. 22, no. 16, p. 6240, 2022.