IMPLEMENTASI MODEL LSTM PADA MIKROKONTROLER UNTUK PREDIKSI IKLIM MIKRO JAKARTA UTARA SECARA *REAL-TIME* MENGGUNAKAN *TINYML*

Kelvin Riyanto, I Gusti Ngurah Suryantara
Informatika, Universitas Bunda Mulia
Jl. Lodan Raya No. 2, Ancol, Pademangan, Jakarta Utara, DKI Jakarta
S32220066@student.ubm.ac.id

ABSTRAK

Penerapan Artificial Intelligence pada sisi pengguna (*Edge AI)* menjadi kebutuhan penting dalam ekosistem *Internet of Things* untuk mengurangi latensi, menjaga privasi data, dan menekan biaya komputasi berbasis cloud. Namun, keterbatasan sumber daya perangkat *edge* menjadi tantangan utama dalam implementasi model pembelajaran mesin yang kompleks. Penelitian ini bertujuan merancang dan mengimplementasikan sistem *Edge AI* berbasis ESP32-S3 untuk prediksi cuaca mikro secara real-time menggunakan pendekatan *Tiny Machine Learning*. Model prediksi dikembangkan menggunakan arsitektur *Long Short-Term Memory* (LSTM) dengan data historis cuaca Jakarta Utara (Sunda Kelapa) yang diperoleh dari Open-Meteo. Untuk menyesuaikan dengan keterbatasan perangkat, model dioptimalkan melalui *Post-Training Full Integer Quantization*. Sistem memanfaatkan sensor BME280 untuk mengakuisisi data suhu, kelembapan, dan tekanan udara selama 24 jam terakhir sebagai masukan prediksi 6 jam ke depan, dengan pengelolaan data menggunakan *circular buffer*. Hasil prediksi dikirimkan melalui protokol MQTT ke InfluxDB. Hasil pengujian menunjukkan bahwa model kuantisasi Int8 menghasilkan waktu inferensi rata-rata 48,1 ms dengan penggunaan memori sebesar 147,04 KB serta nilai MAE 0,074, yang relatif mendekati model Float32. Hasil ini menunjukkan bahwa ESP32-S3 layak digunakan sebagai platform *Edge AI* untuk peramalan cuaca lokal.

Kata kunci : *Edge AI, Tiny Machine Learning, ESP32-S3, Long Short-Term Memory*, Mikrokontroler

1. PENDAHULUAN

Perkembangan kecerdasan buatan dalam satu dekade terakhir menunjukkan lonjakan adopsi yang signifikan di sektor industri. Laporan AI Index 2025 dari Stanford mencatat bahwa 78% responden korporat telah mengintegrasikan AI dalam proses bisnis mereka, sebuah peningkatan drastis dibandingkan tahun 2017 yang hanya mencapai 20%. Tren ini diperkuat oleh penggunaan *generative-AI* yang melonjak dari 30% menjadi 70% pada tahun 2023 seiring dengan semakin terjangkau biaya teknologinya. Namun, mayoritas penerapan AI saat ini masih sangat bergantung pada cloud platform dengan sumber daya komputasi tinggi, yang juga memunculkan tantangan fundamental terkait latensi, biaya operasional, serta privasi data [1].

Ketergantungan pada cloud menjadi kendala kritis bagi sektor-sektor yang membutuhkan respons cepat dan bersifat lokal, seperti pergudangan dan agrikultur skala mikro. Sektor-sektor ini membutuhkan informasi lingkungan yang real-time dan presisi. Data cuaca makro yang disediakan oleh badan meteorologi atau satelit umumnya memiliki resolusi yang luas, sehingga sulit merepresentasikan kondisi iklim mikro yang sangat dipengaruhi oleh faktor lokal seperti bangunan, vegetasi, dan struktur penutup [2]. Akibatnya, keputusan operasional sering kali terdapat diskrepansi antara prediksi makro dengan kondisi mikro yang sebenarnya.

Untuk mengatasi kesenjangan ini, muncul paradigma *Tiny Machine Learning* (TinyML) yang memungkinkan model cerdas dijalankan secara lokal pada mikrokontroler berdaya rendah. Salah satu perangkat yang potensial adalah ESP32-S3 yang rilis pada tahun 2020, yang menawarkan akselerasi instruksi vektor (SIMD) untuk komputasi AI, meskipun memiliki keterbatasan memori internal sebesar 512KB [3]. Tantangan utamanya adalah bagaimana menjalankan algoritma prediksi deret waktu yang kompleks, seperti *Long Short-Term Memory* (LSTM), pada perangkat terbatas tersebut. Meskipun LSTM terbukti lebih unggul daripada *Recurrent Neural Network* (RNN) klasik dalam menangkap dependensi jangka panjang berkat mekanisme gating-nya [4] kebutuhan komputasi dan memorinya yang tinggi menjadikan implementasi langsung pada mikrokontroler sebagai tantangan teknis yang berat.

Maka dari itu, penelitian ini ditujukan untuk merekayasa sistem peramalan cuaca mikro berbasis *Edge AI* menggunakan model LSTM yang dioptimalkan melalui teknik kuantisasi bobot. Pendekatan ini diharapkan mampu menghasilkan sistem yang akurat secara prediktif, dan efisien dalam penggunaan memori dan latensi, sehingga layak diimplementasikan sebagai solusi mandiri pada mikrokontroler ESP32-S3 untuk kebutuhan pemantauan lingkungan yang bersifat lokal dan *real-time*.

2. TINJAUAN PUSTAKA

2.1. Mikrokontroler

Mikrokontroler atau sering disingkat dengan MCU (*Microcontroller)*, adalah sebuah sirkuit terintegrasi yang umumya digunakan untuk sebuah pengaplikasian spesifik dan didesain untuk melakukan sebuah tugas tertentu seperti sebuah produk atau aplikasi yang harus di kontrol secara otomatis dalam sebuah situasi tertentu, seperti perangkat elektronik, alat berat, kendaraan otomatis, hingga pada komputer. *Microcontroller* dapat dianggap sebagai sebuah komputer yang memiliki sebuah masukkan dan keluaran yang dapat diprogram, dengan 1 atau lebih inti CPU, dan memiliki sebuah RAM, *Flash*, dan ROM dengan kapasitas yang terbatas [6]. Mikrokontroler yang akan digunakan adalah Xiao Seed ESP32-S3 dengan SRAM 512KB, PSRAM 8MB, dan *external flash* 8MB.

2.2. *Tiny Machine Learning*

Tiny Machine Learning adalah bidang pembelajaran mesin yang memungkinkan AI berjalan pada perangkat tertanam dengan sumber daya terbatas di tepi jaringan. Tujuan utamanya adalah mengalihkan pembelajaran mesin dari sistem berdaya tinggi ke klien berdaya rendah, melakukan analitik pada perangkat dalam rentang daya *milliwatt* atau lebih rendah. Paradigma ini dibangun di atas interaksi tiga elemen kunci: perangkat keras, perangkat lunak, dan algoritma [7].

2.3. *Mean Absolute Error*

Mean Absolute Error umumnya disingkat MAE adalah salah satu fungsi *loss* yang dapat digunakan untuk mengukur keakuratan model dalam tugas prediksi. Nilai MAE menunjukkan rata-rata kesalahan yang absolut antara hasil prediksi dengan nilai riil. MAE di notasikan sebagai rumus berikut:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \quad (1)$$

2.4. *Min-Max Normalization*

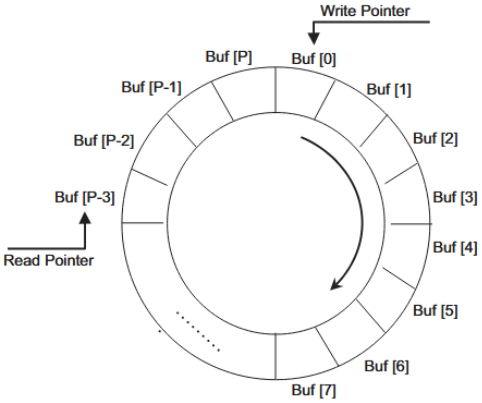
Normalisasi merupakan proses yang dilakukan pada fase praproses data dengan menykalakan kembali nilai-nilai numerik data sehingga membuat pemrosesan lebih mudah dan efisiensi secara memori dan daya pemrosesan [8]. *Min-Max Normalization* adalah salah satu dari teknik normalisasi yang menyekalakan data di antara 0 hingga 1. Pendekatan *Min-Max Normalization* menjamin seluruh nilai yang ada di dalam *dataset* berjalan dalam jarak yang konsisten dan memberikan skala perbandingan dan analisis yang sama. *Min-Max Normalization* dapat dirumuskan sebagai berikut:

$$X' = \frac{(x - x_{min})}{(x_{max} - x\_min)} \quad (2)$$

Min-max scaling menormalisasikan seluruh dengan perlakuan yang proporsional, dan dengan jarak yang sudah ditentukan, sehingga memungkinkan pembelajaran dan generalisasi yang lebih baik oleh model LSTM. Hal ini sangat penting untuk data *time-series* dengan rentang atribut yang bervariasi [9].

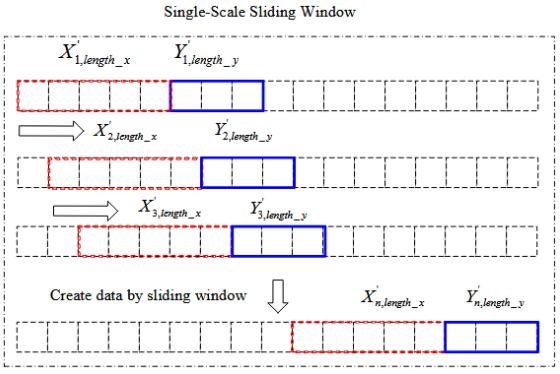
2.5. *Circular Buffer*

Circular buffer, atau sering disebut juga *ring buffer*, adalah sebuah struktur data penyangga yang bekerja dengan prinsip *First-In-First-Out* (FIFO). Artinya, data yang pertama kali masuk akan menjadi data pertama yang dikeluarkan. *Circular buffer* umumnya berbentuk sebuah *array* berukuran tetap yang bersifat melingkar. Ketika penulisan sudah mencapai akhir *array*, *pointer* penulisan akan otomatis kembali ke posisi awal *array*, sehingga data baru akan menimpa data lama. Pola menumpuk ini memungkinkan *circular buffer* untuk selalu menyimpan sekumpulan sampel terbaru tanpa harus memindahkan atau menggeser data secara manual. Ini menghasilkan, hanya satu posisi memori yang perlu diperbarui setiap kali sampel baru masuk. *Circular buffer* digunakan secara luas dalam pemrosesan sinyal digital. Pada digital *filter* seperti FIR dan IIR, merupakan sistem yang membutuhkan akses cepat ke beberapa sampel terbaru dari sinyal masukan dan terkadang keluaran. *Circular buffer* memungkinkan penyimpanan dan pembaruan sampel - sampel tersebut secara efisien saat data terus mengalir masuk [10].



Gambar 1. Struktur *Circular Buffer*

2.6. *Sliding Window*



Gambar 2. Struktur *Sliding Window*

Metode *sliding window* adalah metode praproses untuk menyusun data deret waktu berdasarkan rentang waktu tertentu sehingga dapat diformulasikan sebagai masalah klasifikasi atau regresi dengan sifat pembelajaran terawasi. Pada deret waktu yang dinyatakan sebagai $y = f(x)$, metode *sliding window* merekonstruksi data dengan memanfaatkan nilai keluaran pada waktu sebelumnya sebagai fitur untuk memprediksi keluaran pada deret waktu kemudian. Misalnya, pada ukuran jendela sebesar 1, nilai keluaran pada waktu t dipengaruhi oleh nilai keluaran pada waktu $t - 1$ [11].

2.7. *Long Short-Term Memory*

Long Short-Term Memory atau LSTM merupakan bentuk modifikasi dari *Recurrent Neural Network* (RNN) yang mampu mengatasi kelemahan inheren daripada algoritma RNN itu sendiri, yaitu *exploding gradient* dan *vanishing gradient*. LSTM memiliki kemampuan untuk untuk mempelajari dependensi jangka panjang dengan lebih baik ketimbang RNN. Keunggulan LSTM adalah cocok dalam melakukan prediksi *time-series* dan ketahanannya dalam menangani data yang besar dan non-linier [12].

Perbedaan arsitektur LSTM dan RNN terdapat pada lapisan rantainya. LSTM sendiri memiliki sistem gerbang, yakni *forget gate*, *input gate*, masukkan *modulation gate* dan *output gate*. Masing-masing unitnya memiliki *Internal Cell State* yang berfungsi untuk menyimpan informasi pilihan dari unit sebelumnya [12].

2.8. *Post-Training Quantization*

Post-Training Quantization adalah teknik dalam mengoptimalkan model *deep learning* setelah proses pelatihan selesai. Bertujuan untuk mengubah *floating point* pada bobot dan masukkan, menjadi sebuah bentuk *Integer* atau bilangan bulat yang lebih ringkas dan hemat sumber daya. Terdapat beberapa teknik yang digunakan, yaitu *Full Integer Quantization*, yang seluruh parameter model diubah menjadi bilangan bulat terdekat, dengan tipe data *signed integer 8 bit*, dengan jarak [-128, 127], kemudian ada *Dynamic Range Quantization* merupakan teknik kuantisasi yang mengonversi model menjadi bilangan bulat dengan skala yang dinamis dan adaptif sesuai pada nilai parameter dalam data pelatihan, *Float16 Quantization* merupakan teknik kuantisasi, yang masih menggunakan *Floating Point*, tetapi dengan presisi yang lebih rendah, yaitu 16 *bit* [13].

2.9. Studi Literatur

Penerapan TinyML pada perangkat mikrokontroler dengan sumber daya terbatas telah menunjukkan potensi besar dalam memindahkan beban komputasi dari *cloud* ke *edge*. Penelitian [13] menganalisis performa *TensorFlow Lite* pada ESP32, menyoroti bahwa metode kuantisasi *Full Integer* merupakan pendekatan paling optimal untuk menjalankan model AI pada perangkat keras tersebut karena efisiensinya terhadap memori. Temuan mengenai kapabilitas mikrokontroler ini diperkuat oleh studi [16] yang mengembangkan sistem klasifikasi suara kendaraan berbasis TinyML, membuktikan bahwa model sederhana yang diimplementasikan pada mikrokontroler mampu mencapai akurasi tinggi dan efisiensi yang baik dalam skenario *real-time*.

Dalam domain peramalan deret waktu, penggunaan *Deep Learning* pada perangkat *edge* menjadi fokus utama untuk efisiensi energi dan kemandirian sistem, studi [17] membandingkan berbagai model untuk prediksi energi surya dan menyimpulkan bahwa *Unidirectional Long Short-Term Memory* menawarkan sebuah performa antara akurasi prediksi dan efisiensi komputasi bagi perangkat mikrokontroler. Lebih lanjut, penelitian [18] mengevaluasi kerangka kerja TinyML *end-to-end* di bidang pertanian cerdas dengan model *Convolutional Neural network*, yang menekankan pentingnya analisis *trade-off* antara kompleksitas model, kecepatan inferensi, dan *memory footprint*. Berbeda dengan penelitian-penelitian sebelumnya yang umumnya berfokus pada klasifikasi atau prediksi satu variabel, penelitian ini mengembangkan sistem prediksi cuaca mikro yang bersifat *multivariate* dan *multi-step* secara *real-time* menggunakan ESP32-S3.

3. METODE PENELITIAN

3.1. Akuisisi Data

Pada tahap implementasi akuisisi data, data akan diakuisisi melalui situs Open-Meteo. Peneliti mengambil *dataset* dengan rentang waktu 1 Agustus 2021 hingga 31 Agustus 2025 dengan lokasi bertepat di Jakarta Utara. *Dataset* ini total memiliki 37248 baris, dengan interval 1 jam. Data yang diakuisisi memiliki fitur waktu, temperatur, kelembapan relatif, dan tekanan permukaan.

3.2. Pra-Proses Data

Dalam proses ini, data akan dilakukan *scaling* menggunakan *min-max scaler*, yang menghasilkan data dengan rentang 0 hingga 1. Data yang telah dinormalisasi akan diuabh dalam dengan menjadi bentuk target dan juga data lampau menggunakan *sliding window*, yang kemudian akan di potong menjadi 2 bagian, yaitu data latih, dan validasi dengan rasio 0.8 untuk data latih, dan 0.2 untuk validasi. Data-data tersebut, kemudian dicacah menjadi sebuah runtutan data menggunakan *sliding-window technique*, dengan data yang terus menerus bergeser, dengan setiap jendela berisikan data yang memiliki 24 data historis yang dilihat oleh model, dan 6 data historis yang harus diprediksi oleh model.

3.3. Pelatihan Model

Model kemudian akan dilatih selama 50 *epochs*, dengan menggunakan *Early Stopper* sebagai fungsi *callback*, dengan *Keras Adam* sebagai *optimizer*, dan

MAE sebagai fungsi *loss*. Arsitektur model yang akan dilatih adalah sebagai berikut:

Tabel 1. Arsitektur Model

Lapisan	*Neuron/Shape*	Fungsi Aktivasi
Input()	(1,24,3)	
LSTM1	32	*tanh*
LSTM2	64	*tanh*
Dense1	6*64	*linear*
Reshape()	(6,64)	
LSTM3	32	*tanh*
Flatten		
Dense2	6*32	*relu*
Reshape	(6,32)	
Dense3	32	*relu*
Dense4	3	*relu*
Reshape	(6,3)	

3.4. Kuantisasi

Untuk mengoptimalkan model agar dapat berjalan di ESP32-S3, teknik *Post-Training Full Integer Quantization* akan diterapkan menggunakan Tensorflow Lite. Teknik ini mengkonversi semua parameter model (bobot dan aktivasi) dari *floating-point* 32-bit menjadi *integer* 8-bit. Proses ini secara signifikan mengurangi ukuran model dan jejak memori, serta mempercepat komputasi pada perangkat keras yang mendukungnya, meskipun berpotensi menurunkan performa model, dan bahkan menjadi fatal jika tidak dilakukan dengan baik.

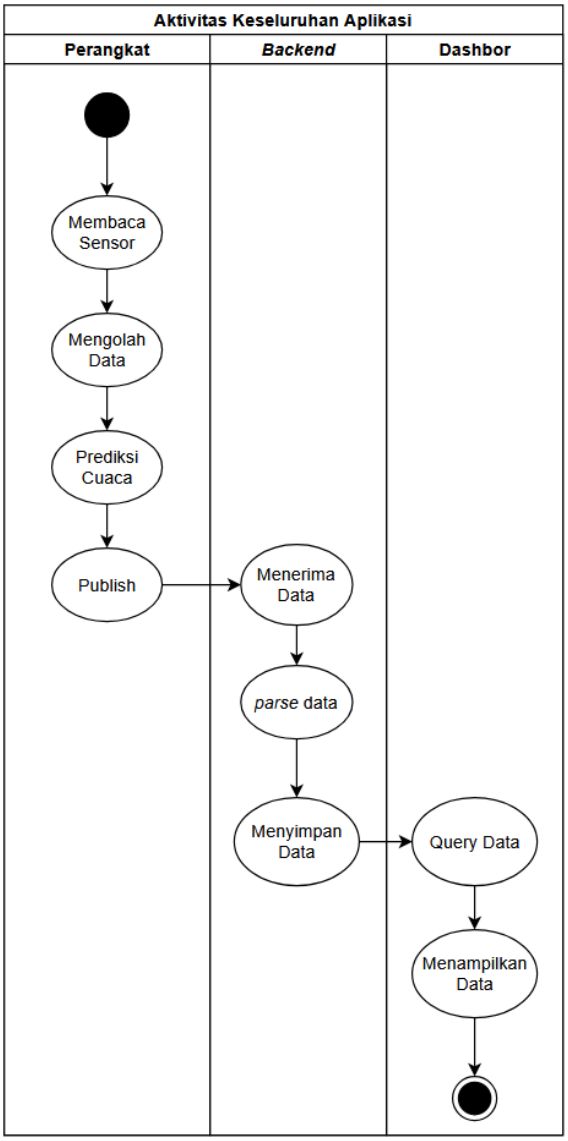
3.5. Requirement Analysis

Berasarkan tujuan dan batasan perangkat *edge* di penelitian ini, didapat *requirement* sebagai berikut:

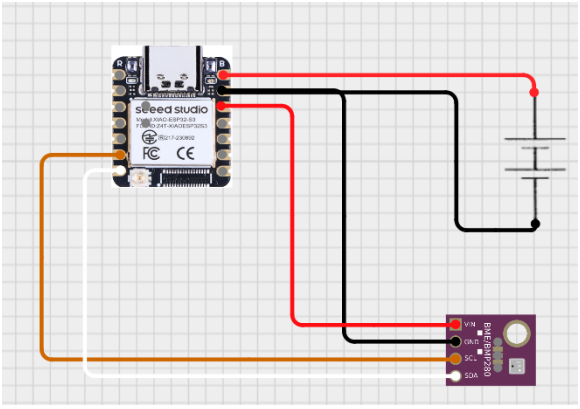
a. Perangkat dapat melakukan inferensi,
b. Perangkat dapat mengirim data inferennsi dan sensor,
c. Data dapat dipresentasikan dalam sebuah dasbor.

3.6. Design

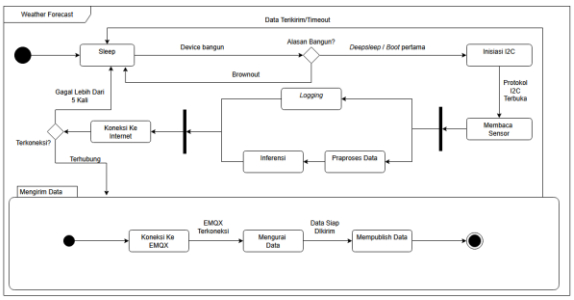
Perancangan sistem terdiri dari desain perangkat keras dan perangkat lunak. Secara perangkat keras, ESP32-S3 dihubungkan dengan BME280 melalui jalur I2C dan ditenagai baterai, seperti terlihat pada Rangkaian Sistem. Untuk logika perangkat lunak, State Machine Diagram digunakan untuk mengatur siklus kerja perangkat mulai dari inisialisasi, pembacaan sensor, inferensi, hingga transmisi data dan kembali ke mode *Deep Sleep* untuk efisiensi daya.



Gambar 3. *Activity Diagram*



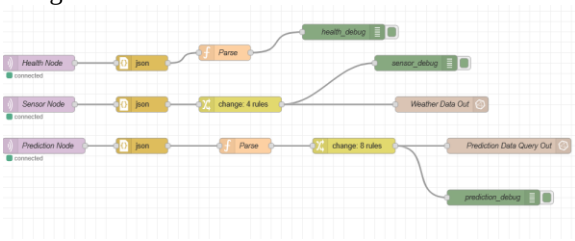
Gambar 4. Rangkaian Sistem



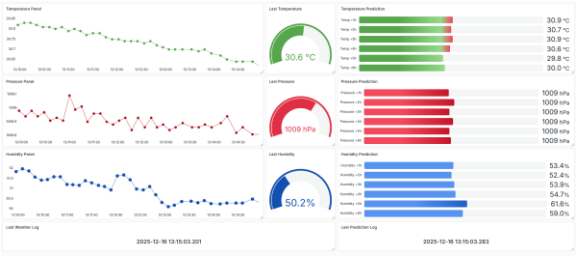
Gambar 5. *State Machine Diagram*

3.7. *Development*

Tahap pengembangan berfokus pada pembuatan firmware menggunakan bahasa C++. Proses ini meliputi integrasi pustaka *TensorFlow Lite for Microcontrollers* untuk memuat model yang telah dikuantisasi, *circular buffer* sebagai struktur data, serta pengembangan driver untuk pembacaan sensor dan modul komunikasi WiFi/MQTT yang berfungsi unutk mengirimkan data berupa hasil prediksi, nilai bacaan sensor, memori yang digunakan oleh model, dan juga lama inferensi. Kemudian adapun *backend* yang menggunakan NodeRED dan juga Grafana sebagai dasbor.



Gambar 6. *Backend* NodeRED



Gambar 7. Tampilan Dasbor Grafana

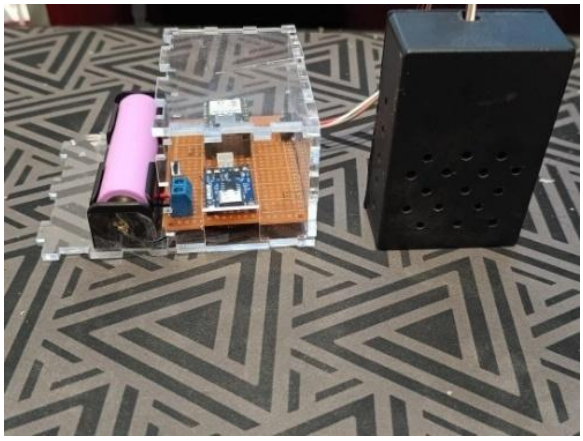
3.8. *Testing*

Pada tahap ini, perangkat yang telah di hasilkan pada masa implementasi akan dilakukan testing. Testing yang akan dilakukan menggunakan metode *black-box testing* yang pengujian perangkat secara langsung tanpa penguji terekspos pada detil implementasi, logika, dan struktur kode. Basis testing ini akan melihat pada *requirement* yang lebih rinci yang telah di tentukan, masukkan, dan juga ekspektasi keluaran dari fungsi tersebut

.

3.9. *Implementation*

Perangkat yang ditempatkan pada titik pengujian yang telah ditentukan menggunakan pelindung berventilasi. Hal ini untuk memastikan sensor terlindung dari paparan langsung air hujan namun tetap mendapatkan sirkulasi udara yang cukup untuk pembacaan suhu dan kelembapan yang akurat. Keseluruhannya direalisasikan sesuai pada rangkaian yang ada pada Gambar 5.



Gambar 8. Implementasi Desain Rangkaian

3.10. Maintenance

Selama perangkat IoT tengah berjalan, akan dilakukan pemantauan terhadap performa dari IoT tersebut. Jika diperlukan untuk kembali ke tahap awal, untuk memperbaiki hal yang diperlukan, maka akan dilakukan iterasi lanjutan. Pemantauan dapat menggunakan NodeRED, melalui *debug node*.

4. HASIL DAN PEMBAHASAN

4.1. *Blackbox Testing*

Dalam penelitian ini, akan digunakan pengujian *blackbox* yang berguna untuk memastikan sistem dapat berjalan sesuai dengan kebutuhan pengguna tanpa pengguna mengetahui proses internal sistem dengan merancang scenario pengujian dengan masukan dan keluarannya. Hasil testing dapat dilihat sebagaimana yag terdapat di Tabel 2.

Tabel 2. Skenario Pengujian

Skenario	Masukan	Keluaran	Hasil
Koneksi WiFi Putus	Mematikan *router* atau *hotspot* saat ESP32 sedang menyala dan mencoba mengirim data	Sistem mendeteksi kegagalan koneksi setelah *timeout* tertentu, lalu masuk ke mode *Deep Sleep* untuk menghemat baterai dan mencoba lagi di siklus berikutnya.	Berhasil
Putusnya Koneksi MQTT Broker	WiFi terhubung, tetapi koneksi ke EMQX terputus.	Sistem tidak tersangkut menunggu koneksi. Sistem harus *timeout*, lalu kembali tidur.	Berhasil

Skenario	Masukan	Keluaran	Hasil
Kegagalan Sensor BME280	Mencabut kabel SDA/SCL BME280 saat perangkat menyala.	Sistem mendeteksi sensor tidak terbaca (nilai *NaN/Null*). dan mengirim nilai '0' ke Dasbor.	Berhasil
Data Ekstrem Atas	Masukkan memiliki nilai pada salah satu fitur di atas dari nilai yang pernah dilihat model	Model LSTM tetap berjalan dan memberikan prediksi meskipun tidak akurat. Nilai tidak menyebabkan *overflow* pada variabel *INT8*.	Berhasil
Data Ekstrem Bawah	Masukkan memiliki nilai pada salah satu fitur di bawah dari nilai yang pernah dilihat model.	Model LSTM tetap berjalan dan memberikan prediksi meskipun tidak akurat. Nilai tidak menyebabkan *underflow* pada variabel *INT8*.	Berhasil
Transisi *Circular Buffer*	Menjalankan alat hingga *buffer* ke-24, dan masuk ke data ke-25.	*Pointer buffer* kembali ke indeks 0 (awal) dengan mulus. Data lama tertimpa data baru tanpa merusak urutan *windowing* untuk inferensi.	Berhasil
Stabilitas Jangka Panjang	Menjalankan perangkat selama 24 jam penuh (24x siklus inferensi).	Tidak terjadi *brownout*, dan inferensi terus berjalan.	Berhasil
Pengiriman Data	Data akan dikirim setiap jam, ke MQTT kemudian NodeRed dan terakhir InfluxDB.	Data yang dikirim, dapat dilihat dari debug log pada NodeRed dan juga dasbor Grafana.	Berhasil
Koneksi WiFi Putus	Mematikan *router* atau *hotspot* saat ESP32 sedang menyala dan mencoba mengirim data	Sistem mendeteksi kegagalan koneksi setelah *timeout* tertentu, lalu masuk ke mode *Deep Sleep* untuk menghemat baterai dan mencoba lagi di siklus berikutnya.	Berhasil

4.2. Hasil Pengujian

Setelah sistem dinyatakan berfungsi dengan baik berdasarkan pengujian *blackbox*, tahap selanjutnya adalah mengukur kinerja metode prediksi menggunakan metrik *Mean Absolute Error*. Nilai MAE digunakan untuk melihat seberapa besar rata-rata kesalahan absolut antara nilai hasil prediksi dan nilai aktual. Dengan data sampel 6 hari pengujian yang dilakukan dari tanggal 27 Oktober hingga tanggal 1 November, didapatkan hasil metrik yang telah di *scaling* menggunakan *min-max scaling* sebagai berikut:

Tabel 3. MAE Model *Int8*

n+ jam	MAE Temperatur	MAE Kelembapan	MAE Tekanan
1	0.041	0.038	0.073
2	0.062	0.058	0.081
3	0.069	0.068	0.092
4	0.072	0.077	0.094
5	0.076	0.086	0.094
6	0.075	0.086	0.097

Tabel 4. Keseluruhan Skor MAE Per-jam

n+ jam	Skor MAE
1	0.051
2	0.067
3	0.077
4	0.081
5	0.085
6	0.086

Tabel 5. Keseluruhan Skor MAE Per-fitur

Fitur	*Mean Absolute Error*
Temperatur	0.068
Kelembapan	0.068
Tekanan	0.087
Keseluruhan	0.074

Dengan mengakumulasi seluruh skor rata-rata per jam, dan kemudian di bagi dengan jumlah jam, maka didapat MAE model sebesar: 0.074. Kemudian dilihat dari model *Float32* yang merupakan tolok ukur penelitian ini, dilakukan pengujian model dengan data yang telah didapat, dengan demikian didapatkan MAE sebesar: 0.071. Rincian dari performa yang didapatkan pada model *Float32*, dapat dilihat pada tabel sebagai berikut:

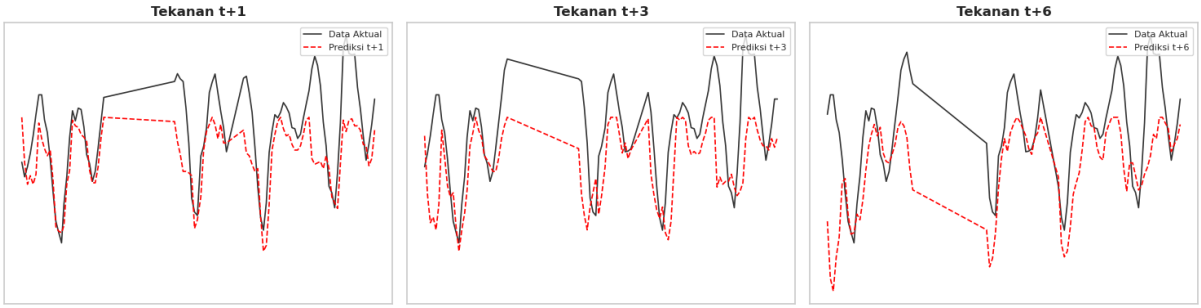
Tabel 6. Performa Model *Float32*

Fitur	*Mean Absolute Error*
Temperatur	0.06013992
Kelembapan	0.07517223
Tekanan	0.06869814
Keseluruhan	0.068

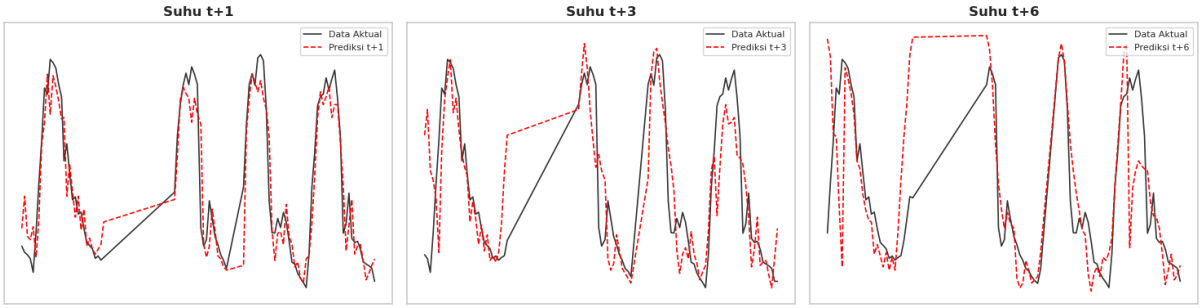
Berdasarkan komparasi data pada Tabel 5 dan Tabel 6, penerapan teknik *Full Integer Quantization* berhasil mempertahankan kinerja model dengan degradasi performa secara minim. MAE secara garis besar pada model tercatat sebesar 0.074, ini merupakan peningkatan *error* dibanding pada *Float32* yang memiliki MAE sebesar 0.068. Selisih performa yang tipis ini menunjukkan proses kalibrasi dengan *representative dataset* telah berjalan dengan baik, mengakibatkan *trade-off* dengan degradasi akurasi demi efisiensi memori dan kecepatan inferensi yang layak untuk implementasi pada perangkat terbatas.

Analisis mendalam pada setiap fitur menunjukkan performa yang konsisten. Pada fitur Temperatur dan Kelembapan, model yang dikuantisasi menunjukkan kinerja yang tidak jauh berbeda dari model *Float32*, ini dapat dilihat dari skor MAE yang hanya selisihnya marginal dengan versi *Float32*. Tetapi, pada fitur tekanan udara, meskipun tercatat MAE sebesar 0.087 yang memang lebih tinggi jika dibandingkan pada model *Float32* dengan MAE 0.068. Meskipun begitu model tetap mampu mengikuti pola data aktual. Selisih ini diasumsikan merupakan konsekuensi dari penurunan resolusi bit pada data tekanan yang memiliki variansi nilai sangat sempit, namun tidak sampai menyebabkan distorsi prediksi yang signifikan.

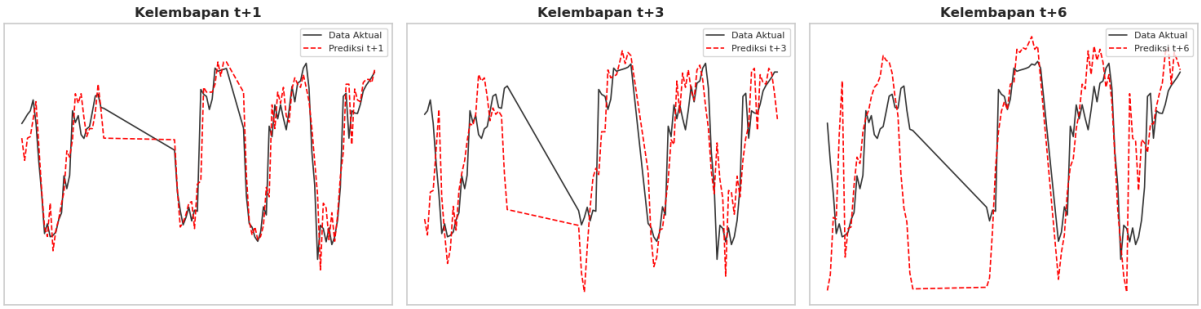
Fitting dari model *int8* dapat direpresentasikan dengan beberapa *plot* yang dapat diperhatikan pada gambar 9 hingga gambar 11.



Gambar 9. Validasi Tekanan Udara



Gambar 10. Validasi Temperatur Udara



Gambar 11. Validasi Kelembapan Udara

Kemudian, adapun hasil evaluasi kinerja model pada perangkat ESP32-S3, yang diukur melalui waktu dan juga sumber daya memori yang dibutuhkan untuk menjalankan model.

Tabel 7. Penggunaann Sumber Daya

Sumber Daya	Nilai
Rata-Rata Waktu Inferensi	48.1ms
Penggunaan Memori	147.04 KB

Pada Tabel 7, penggunaan memori tercatat menggunakan sebesar 147.04 KB dengan waktu inferensi rata-rata yaitu 48.1ms. Angka ini membuktikan bahwa tujuan utama optimalisasi *TinyML* telah terpenuhi, sebuah model LSTM dalam format *Float32* yang memerlukan sumber daya lebih besar berhasil dikompresi untuk berjalan dengan memori yang kecil, dan juga inferensi yang dilakukan terdapat pada rata-rata 0.04 detik per inferensi.

5. KESIMPULAN DAN SARAN

Penelitian ini berhasil mengimplementasikan sistem prediksi iklim mikro *real-time* berbasis TinyML pada mikrokontroler ESP32-S3 yang terintegrasi dengan sensor BME280 dan protokol komunikasi MQTT. Hasil evaluasi kinerja menunjukkan bahwa penerapan teknik *Post-Training Full Integer Quantization* pada model LSTM terbukti efektif dalam menyeimbangkan efisiensi komputasi dan akurasi, dengan rata-rata waktu inferensi 48.1 ms dan konsumsi memori sebesar 147.04 KB. Secara akurasi, model terkuantisasi mampu mempertahankan nilai *Mean Absolute Error* ebesar 0.074, yang hanya memiliki selisih marginal dibandingkan model *Float32* (0.068). Untuk pengembangan selanjutnya, disarankan penggunaan protokol komunikasi jarak jauh berdaya rendah seperti LoRaWAN serta penambahan parameter input cuaca lain, seperti curah hujan dan kecepatan angin, guna meningkatkan cakupan area implementasi dan reliabilitas prediksi.

DAFTAR PUSTAKA

[1] Nestor Maslej, "AI Index 2025: State of AI in 10 Charts | Stanford HAI." Accessed: Jul. 26, 2025. [Online]. Available: https://hai.stanford.edu/news/ai-index-2025-state-of-ai-in-10-charts

[2] Y. Pan, T. Morimoto, and T. Ichinose, "Combining Multi-Source Satellite Data with a Microclimate Model to Analyze the Microclimate of an Urban Park," *climate*, vol. 12, no. 197, 2024, doi: 10.3390/cli12120197.

[3] Espressif, "ESP32-S3 Series Datasheet Version 2.0," 2025.

[4] B. Ghojogh and A. Ghodsi ALIGHODSI, "Recurrent Neural Networks and Long Short-Term Memory Networks: Tutorial and Survey," Waterloo, Apr. 2023.

[5] D. Y. L. A. Rhasyid, B. A. Pramudita, and Istiqomah, "Sistem Pemantauan Cuaca Berdasarkan Kecepatan Angin, Suhu dan Kelembaban Udara Berbasis Internet of Things," *e-Proceeding of Engineering*, vol. 10, no. 4, Aug. 2023, Accessed: Sep. 09, 2025. [Online]. Available: https://openlibrarypublications.telkomuniversity.ac.id/index.php/engineering/article/view/20764

[6] "Exploring Various Applications of Micro Controller," *Electrical and Automation Engineering*, vol. 1, no. 1, pp. 47–53, May 2022, doi: 10.46632/eae/1/1/8.

[7] P. P. Ray, "A review on TinyML: State-of-the-art and prospects," *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 4, pp. 1595–1623, Apr. 2022, doi: 10.1016/j.jksuci.2021.11.019.

[8] I. Permana and F. Nur Salisah, "Pengaruh Normalisasi Data Terhadap Performa Hasil Klasifikasi Algoritma Backpropagation," *Indonesian Journal of Informatic Research and Software Engineering*, vol. 2, no. 1, pp. 67–72, Mar. 2022.

[9] A. Pranolo *et al.*, "Enhanced Multivariate Time Series Analysis Using LSTM: A Comparative Study of Min-Max and Z-Score Normalization Techniques," *ILKOM Jurnal Ilmiah*, vol. 16, no. 2, pp. 210–220, Aug. 2024, doi: 10.33096/ilkom.v16i2.2333.210-220.

[10] A. R. Hamza and M. A. Hussein, "An Innovative Embedded Processor-Based Signal Phase Shifter Algorithm," 2024, doi: 10.14500/aro.11358.

[11] N. M. Norwawi, "Sliding window time series forecasting with multilayer perceptron and multiregression of COVID-19 outbreak in Malaysia," *Data Science for COVID-19*, p. 547, Jan. 2021, doi: 10.1016/B978-0-12-824536-1.00025-3.

[12] R. Y. Rafael and F. Adikara, "Pengimplmentasian Algoritma Long-Short Term Memory Untuk Mendeteksi Ujaran Kebenncian Pada Aplikasi Twitter," *JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika)*, vol. 8, no. 2, pp. 551–560, May 2023, doi: 10.29100/jipi.v8i2.3490.

[13] M. Giri, W. Ngulandoro, S. Rizqika Akbar, and B. H. Prasetio, "Analisis Performa TensorFlow Lite untuk IoT dengan ESP32 DEVKIT-C (Studi Kasus: Pengenalan Gambar Sampah di Sungai)," *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*, vol. 7, no. 7, pp. 3342–3347, Jul. 2023, [Online]. Available: http://j-ptiik.ub.ac.id

[14] J. Andry and M. Stefanus, "Pengembangan Aplikasi E-learning Berbasis Web Menggunakan Model Waterfall Pada SMK Strada 2 Jakarta," *JURNAL FASILKOM*, vol. 10, no. 1, pp. 1–10, Apr. 2020, doi: 10.37859/jf.v10i1.1878.

[15] M. I. Pangestu, J. A. Ginting, I. G. N. Suryantara, and M. Marvelino, "Optimasi Pertanian Di Bekasi Utara: Prediksi Curah Hujan Dan Rekomendasi Tanaman Dengan Menggunakan Model Regresi Linier," *Jurnal Algoritma, Logika dan Komputasi*, vol. 7, no. 2, Feb. 2025, doi: 10.30813/j-alu.v2i2.8081.

[16] H. Kusumah, N. Nabbillah, and S. Audina, "Klasifikasi Kendaraan Berbasis Suara Di Lalu Lintas: Implementasi TinyML," *CERITA: Creative Education of Research in Information Technology and Artificial informatics*, vol. 9, no. 1, Feb. 2023.

[17] A. M. Hayajneh, F. Alasali, A. Salama, and W. Holderbaum, "Intelligent Solar Forecasts: Modern Machine Learning Models and TinyML Role for Improved Solar Energy Yield Predictions," *IEEE Access*, vol. 12, pp. 10846–10864, 2024, doi: 10.1109/ACCESS.2024.3354703.

[18] A. M. Hayajneh, S. Batayneh, E. Alzoubi, and M. Alwedyan, "TinyML Olive Fruit Variety Classification by Means of Convolutional Neural Networks on IoT Edge Devices," *AgriEngineering*, vol. 5, no. 4, pp. 2266–2283, Dec. 2023, doi: 10.3390/agriengineering5040139