Penerapan IoT dan CNN dengan *Attention Layer* untuk Pendeteksian Penyakit pada Daun Jagung Berbasis Gambar Digital

Sutan Raya Dharma Namanika
Fakultas Informatika
Telkom University
Bandung, Indonesia
sutanrayadharmanaman@studen.telkomuniversity.ac.id

Aji Gautama Putrada
Fakultas Informatika
Telkom University
Bandung, Indonesia
ajigps@telkomuniversity.ac.id

Ikke Dian Oktaviani
Fakultas Informatika
Telkom University
Bandung, Indonesia
oktavianiid@telkomuniversity.ac.id

***Abstrak* — Penyakit pada daun jagung seperti Northern Leaf Blight, Common Rust, dan Gray Leaf Spot sering menyebabkan penurunan produktivitas panen secara signifikan. Identifikasi penyakit secara manual oleh petani di lapangan memiliki keterbatasan dalam aspek waktu, akurasi, dan subjektivitas pengamatan, sehingga rentan terjadi kesalahan diagnosis. Oleh karena itu, diperlukan sistem deteksi dini yang akurat dan efisien untuk meminimalisir risiko gagal panen. Penelitian ini mengembangkan sistem deteksi otomatis berbasis Deep Learning dengan arsitektur Convolutional Neural Network (CNN) MobileNetV3Large. Sistem dirancang menggunakan arsitektur Local Client-Server, di mana kamera smartphone difungsikan sebagai perangkat akuisisi citra yang praktis untuk mengirimkan data visual secara nirkabel ke server lokal, sedangkan proses klasifikasi berat dilakukan pada sisi server. Untuk mengoptimalkan performa model pada dataset yang tidak seimbang, penelitian ini menerapkan strategi Transfer Learning dengan penambahan teknik Class Weighting. Hasil eksperimen menunjukkan bahwa pelatihan model pada fase Transfer Learning (Base Model) mampu mencapai kinerja optimal tanpa memerlukan tahapan fine-tuning yang agresif, dengan perolehan akurasi validasi tertinggi sebesar 94,95%. Hasil ini membuktikan bahwa sistem yang dibangun mampu mendeteksi jenis penyakit daun jagung dengan presisi tinggi serta memiliki arsitektur komunikasi yang siap untuk diterapkan sebagai alat bantu diagnosis di lingkungan perkebunan.**

Kata kunci*— mobilenetv3, daun jagung, *convolutional neural network, transfer learning, class weighting, client-server.

I. PENDAHULUAN

Jagung merupakan salah satu komoditas pangan strategis di Indonesia selain padi. Kebutuhan jagung terus meningkat seiring dengan pertambahan jumlah penduduk dan berkembangnya industri pakan ternak serta pangan olahan [1]. Produktivitas tanaman jagung menjadi kunci utama dalam menjaga ketahanan pangan nasional dan stabilitas ekonomi petani. Namun, upaya peningkatan produksi ini sering kali menghadapi berbagai kendala di lapangan yang dapat menurunkan kualitas dan kuantitas panen.

Salah satu ancaman terbesar dalam budidaya jagung adalah serangan penyakit pada daun. Beberapa penyakit utama yang sering menyerang adalah hawar daun (*Northen Leaf Blight*), karat daun *(Common Rust*), dan bercak abu-abu (*Gray Leaf Spot*) [2]. Penyakit-penyakit ini bisa berbahaya bagi tanaman jagung, yang jika tidak ditangani dengan cepat, dapat menyebabkan kegagalan panen yang signifikan. Saat ini, metode identifikasi penyakit yang umum dilakukan oleh petani masih sangat bergantung pada pengamatan visual secara konvenional [2].

Ketergantungan pada pengamatan manual ini memunculkan beberapa masalah krusial. Pertama, identifikasi visual bersifat subjektif dan sangat bergantung pada pengalaman petani. Gejala awal penyakit seperti *Blight* dan *Gray Leaf Spot* sering kali memiliki kemiripan visual, sehingga rentan terjadi kesalahan diagnosis [3]. Kedua, keterbatasan jumlah pakar atau penyuluh pertanian di daerah terpencil menyebabkan penanganan penyakit sering terlambat. Ketiga, dampak dari kesalahan diagnosis ini adalah penggunaan pestisida yang tidak tepat sasaran, yang tidak hanya merugikan secara ekonomi tetapi juga berpotensi merusak lingkungan. Oleh karena itu, dibutuhkan suatu pendekatan teknologi yang mampu mengidentifikasi penyakit secara objektif, cepat, dan akurat [3].

Berbagai penelitian telah dilakukan untuk mengatasi masalah ini menggunakan pendekatan *Computer Vision* dan *Machine Learning*. Metode konvensional yang bergantung pada ekstraksi fitur manual (*hand-crafted features*) sering kali memiliki keterbatasan dalam mengenali variasi penyakit yang kompleks [2]. Perkembangan terbaru beralih ke *Deep Learning*, khususnya *Convolutional Neural Network* (CNN), yang terbukti handal dalam klasifikasi gambar. Sebagai contoh, penelitian oleh *Aditya et al.* [4] berhasil menerapkan arsitektur ResNet untuk tujuan ini. Akan tetapi, model-model tersebut memiliki struktur yang sangat dalam dan jumlah parameter yang masif. Hal ini berdampak pada tingginya beban komputasi serta waktu inferensi yang lama, sehingga menjadi kurang efisien jika diterapkan pada sistem yang menuntut respon cepat dan skalabilitas tinggi [2].

Oleh karena itu, MobileNetV3 dipilih karena arsitekturnya yang ringan (*lightweight*) menjamin efisiensi komputasi yang unggul. Berbeda dengan model konvensional yang membebani sistem dengan jumlah parameter yang masif

sehingga menghasilkan ukuran model yang besar [5], MobileNetV3 menawarkan waktu inferensi yang lebih cepat sehingga dapat meminimalkan waktu tunggu pengiriman hasil klasifikasi [6]. Keunggulan ini diperkuat oleh mekanisme *attention* yang terbukti mampu menekan gangguan latar belakang (*background interference*) dan memfokuskan model pada fitur penyakit [5]. Penelitian oleh *Maximilliano et al.* [7] melakukan klasifikasi penyakit daun jagung dengan MobileNetV3-Small, mencapai akurasi 99,5%.

Di sisi lain, proses pengumpulan data gambar daun jagung menjadi tantangan tersendiri. Untuk itu, pemanfaatan kamera untuk akuisisi data gambar digital dari lapangan menjadi pendekatan yang sangat relevan. Integrasi kamera dengan sistem deteksi penyakit berbasis *deep learning* dapat menjadi solusi dalam pengembangan sistem deteksi penyakit tanaman yang cepat, efisien, dan dapat diimplementasikan langsung di lapangan. Penelitian oleh *Choudary et al.* [8] menggunakan modul IoT ESP32-CAM karena kemampuannya dalam menangkap dan mentransmisikan gambar daun tanaman secara nirkabel ke penyimpanan *cloud*. Berbeda dengan penelitian sebelumnya, penelitian ini menggunakan kamera *smartphone* untuk menangkap gambar daun jagung di lapangan, lalu mengirim gambar daun jagung ke *server* lokal untuk diproses.

Özbilge et al. [9] mengembangkan model *Compact* CNN untuk klasifikasi penyakit daun tomat. Model ini dirancang untuk meminimalkan beban komputasi karena hanya terdiri dari enam lapisan. Penelitian ini menggunakan dataset *PlantVilaage* yang terdiri dari 9 kelas penyakit dan 1 kelas sehat dengan input ukuran gambar 64x64 piksel, model ini berhasil mencapai akurasi sebesar. Hasil eksperimen penelitian mereka menunjukkan bahwa model *Compact CNN* mampu mengungguli akurasi model *pre-trained* populer seperti VCG16, ResNet50, dan MobileNetV2 pada dataset tersebut, dengan parameter yang lebih kecil. Relevansi penelitian ini dengan penelitian penulis adalah penggunaan model ringan untuk sistem deteksi penyakit tanaman. Namun penelitian sebelumnya membangun arsitektur dari awal, sedangkan penulis menggunakan *Transfer Learning* pada arsitektur MobileNetV3. Naresh V. Et al. [10] meneliti efektivitas mekanisme *attention* dalam dteksi penyakit tanaman cabai. Penelitian ini mengusulkan arsitektur *Squeeze and Excitation Densely Connected CNN* (SEDCNN) yang mengintegrasikan blok *Squeeze-and-Excitation* (SE) ke dalam jaringan DenseNet. Modul SE ini berfungsi untuk mempelajari ulang fitur secara adaptif, sehingga model dapat memberikan perhatian lebih pada fitur penyakit dan menekan fitur lain yang kurang penting. Hasil eksperimen menunjukkan bahwa integrasi blok SE berhasil meningkatkan akurasi klasifikasi hingga 98,86% dengan augmentasi. Meskipun demikian, penelitian ini menerapkan arsitektur DenseNet yang memiliki kompleksitas komputasi tinggi, sedangkan penulis memanfaatkan arsitektur MobileNetV3 yang sudah ada modul SE bawaan. Dari permasalahan dan tinjauan studi terdahulu tersebut, penelitian ini bertujuan untuk mencapai dua sasaran utama, yaitu:

1. Menganalisis performa model *Deep Learning* dengan arsitektur MobileNetV3 dalam mendeteksi penyakit daun jagung berdasarkan parameter evaluasi *accuracy*, *precision*, *recall*, dan *F1-Score*.
2. Merancang sistem deteksi penyakit berbasis arsitektur *Local Client-Server* yang memanfaatkan kamera *smartphone* sebagai perangkat akuisisi gambar dan media pengiriman data nirkabel ke *server* lokal.

Sistematika penulisan artikel ini selanjutnya disusun sebagai berikut: Bagian II membahas penelitian-penelitian terdahulu yang relevan dengan topik ini. Bagian III menguraikan metodologi penelitian serta perancangan sistem yang diusulkan. Bagian IV mendiskusikan hasil eksperimen performa model dan pengujian fungsionalitas sistem. Terakhir, Bagian V menyajikan kesimpulan yang diperoleh dari penelitian ini.

II. KAJIAN TEORI

A. *Machine Learning dan Deep learning*

Machine Learning (ML) adalah kemampuan sebuah program komputer yang memungkinkan suatu program komputer memilki performa yang meningkat seiring pengalaman terhadap suatu tugas tertentu. Tujuan ML adalah mengotomatiskan proses pembangunan model untuk tugas-tugas kognitif seperti deteksi objek maupun terjemahan bahasa alami. Proses ini dilakukan dengan menerapkan algoritma yang secara iteratif belajar dari data pelatihan, sehingga memungkinkan mesin dapat menemukan pola kompleks tanpa perlu diprogram secara eksplisit [11].

B. *Convolutional Neural Network*

Convolutional Neural Network (CNN) termasuk dalam arsitektur *deep learning* yang dirancang untuk tugas-tugas identifikasi, klasifikasi, deteksi, serta segmentasi objek dalam gambar. CNN dapat mempelajari fitur secara langsung tanpa perlu proses ekstraksi fitur manual [12].

C. *Arsitektur MobileNetV3*

MobileNetV3 adalah pengembangan terbaru dari keluarga arsitektur MobileNet yang dirancang khusus untuk berjalan secara efisien pada perangkat *mobile* dengan sumber daya komputasi terbatas. Arsitektur ini bertujuan untuk mencapai keseimbangan terbaik antara latensi (kecepatan) dan akurasi dibandingkan dengan model CNN lainnya. Struktur MobileNetV3 dibangun menggunakan kombinasi teknik pencarian arsitektur otomatis, yaitu *Platform-Aware Network Architecture Search* (NAS) dan algoritma NetAdapt. Pendekatan ini memungkinkan model untuk menyesuaikan struktur jaringannya secara spesifik terhadap karakteristik perangkat keras ponsel (*mobile phone* CPUs), sehingga menghasilkan performa klasifikasi gambar yang optimal dengan beban komputasi yang minimal [13].

Salah satu komponen yang digunakan dalam MobileNetV3 untuk meningkatkan akurasi adalah modul *Squeeze-and-Excitation* (SE). SE block dirancang untuk memperkuat kemampuan representasi jaringan saraf dengan memodifikasi bobot setiap filter pada lapisan konvolusi secara adaptif selama pelatihan, sehingga meningkatkan sensitivitas jaringan terhadap fitur penting. Tahap *squeeze* melakukan kompresi setiap channel fitur menjadi satu nilai menggunakan *global pooling*. Tahap *excitation* menggunakan dua lapisan *fully connected* (FC) dengan ReLU dan Sigmoid untuk menentukan seberapa penting tiap *channel* fitur, kemudian menyesuaikan bobotnya secara dinamis [10].

D. ***Transfer Learning***

Transfer learning adalah teknik menggunakan model yang sudah dilatih sebelumnya dan dimodifikasi untuk menyelesaikan suatu masalah. Cara ini dapat memudahkan penyelesaian masalah karena tidak perlu membuat model dari awal [14].

E. ***Fine-Tuning***

Fine-Tuning merupakan strategi lanjutan di mana model dilatih kembali untuk menyesuaikan bobotnya terhadap dataset baru. Proses ini memungkinkan model menjadi lebih sensitif terhadap fitur-fitur spesifik, seperti pola visual penyakit yang memiliki kemiripan tinggi. Pendekatan ini terbukti menghasilkan akurasi yang lebih baik dibandingkan metode standar, sebagaimana ditunjukkan dalam optimalisasi *epoch* pada klasifikasi penyakit anggur [15].

F. ***Class Weighting***

Teknik ini bekerja dengan memberikan bobot penalti yang berbeda pada fungsi kerugian (*loss function*) selama proses pelatihan. Kelas dengan jumlah sampel sedikit akan diberikan bobot yang lebih besar, sehingga kesalahan prediksi pada kelas tersebut akan menghasilkan nilai loss yang lebih tinggi. Hal ini memaksa model untuk "memberi perhatian lebih" pada kelas minoritas agar meminimalkan *total error* secara seimbang [16].

G. ***Attention Layer***

Attention Layer adalah sebuah mekanisme dalam *deep learning* yang terinspirasi dari mekanisme manusia dalam memperhatikan bagian penting dari suatu informasi. Mekanisme attention *layer* dapat memperhatikan bagian penting dari informasi agar model dapat lebih fokus pada aspek-aspek yang relevan. Mekanisme ini bekerja dengan cara menghitung skor perhatian dan memilih representasi tersembunyi (*hidden states*) yang penting, untuk meningkatkan performa model dalam memahami konteks data [17].

H. ***Squeeze and Excitation (SE)***

SE *block* dirancang untuk memperkuat kemampuan representasi jaringan saraf dengan memodifikasi bobot setiap *filter* pada lapisan konvolusi secara adaptif selama pelatihan, sehingga meningkatkan sensitivitas jaringan terhadap fitur penting. Tahap *squeeze* melakukan kompresi setiap *channel* fitur menjadi satu nilai menggunakan *global pooling*. Tahap *excitation* menggunakan dua lapisan *fully connected* (FC) dengan ReLU dan Sigmoid untuk menentukan seberapa penting tiap channel fitur, kemudian menyesuaikan bobotnya secara dinamis [10].

I. ***Internet of Things***

IoT merupakan konsep keterhubungan perangkat fisik melalui jaringan internet untuk pengendalian dan akses data sensor jarak jauh. IoT dapat menjadi objek pintar karena mampu berkomunikasi melalui jaringan internet dan data yang diambil dapat digabung dengan informasi dari sumber lain [18].

J. ***Arsitektur Client-Server***

Interaksi *Client-Server* terjadi pada lapisan aplikasi, di mana aplikasi pengguna seperti *web browser* beroperasi. Pada lapisan ini, request dibuat oleh *client* dan ditujukan kepada *web server* untuk diproses lebih lanjut. Lapisan ini bertanggung jawab menangani protokol tingkat tinggi serta isu representasi data, yang memungkinkan pengguna berinteraksi langsung dengan sistem komunikasi [19].

K. ***Protokol HTTP***

Hypetext Transfer Protocol (HTTP) merupakan protokol komunikasi yang mendukung pertukaran data seperti gambar, video, dan JSON melalui REST API. HTTP mengirim data antara *client* dan *server* melalui pesan terstruktur yang terbagi menjadi *request* dan *response*. *Request* merupakan permintaan dari *client* ke *server*, contoh: GET, POST, PUT. *Response* merupakan jawaban dari *server* ke *client* [20].

L. ***REST API***

REST didefinisikan sebagai gaya arsitektur perangkat lunak yang dirancang untuk menyederhanakan interaksi antara *client* dan *server* melalui protokol HTTP. Dalam arsitektur ini, setiap entitas data dianggap sebagai resource yang dapat diakses melalui identitas unik berupa URI. Interaksi pertukaran data dilakukan menggunakan *verb* HTTP standar (seperti GET dan POST), yang menjamin bahwa proses pengambilan maupun pengiriman data dapat berjalan secara independen tanpa dibatasi oleh perbedaan lingkungan eksekusi antar perangkat [21].

M. ***JSON (JavaScript Object Notation)***

JSON didefinisikan sebagai format teks ringan yang dirancang untuk kebutuhan pertukaran data antar platform. JSON menawarkan struktur yang jauh lebih ringkas dibandingkan XML, sehingga sangat efektif untuk mengurangi beban lalu lintas data pada jaringan. Sifatnya yang independen dari bahasa pemrograman dan bebas skema menjadikan JSON komponen vital dalam arsitektur aplikasi terdistribusi masa kini [22].

N. ***Confusion Matrix***

Confusion matrix adalah evaluasi yang digunakan untuk memahami pola kesalahan pada model deteksi. Cara ini dapat digunakan untuk melihat hubungan antara label dan prediksi [23]. Tabel 1 di bawah dapat membantu memahami konsep *confusion matrix* yang digunakan untuk evaluasi performa model deteksi.

TABEL 1
(CONFUSION MATRIX)

Kelas		Kelas sebenarnya	
		True (1)	*False* (0)
Kelas Prediksi	Positif (1)	TP	FP
	Negatif (0)	FN	TN

1. *True Positive* (TP): Kondisi di mana model berhasil mengklasifikasi suatu sampel sebagai kelas k dan memang benar sampel tersebut adalah kelas k. Prediksi positif yang benar.
2. *False Positive* (FP): Kondisi di mana model mengklasifikasi suatu sampel sebagai kelas k, tetapi sebenarnya sampel bukan kelas k. Prediksi positif yang salah.
3. *True Negative* (TN): Kondisi di mana model mengklasifikasi suatu sampel bukan sebagai kelas k, dan memprediksi ke kelas lain yang juga bukan kelas k. Prediksi negatif yang benar
4. *False Negative* (FN): Kondisi di mana model mengklasifikasi suatu sampel sebagai kelas lain, tetapi sampel sebenarnya merupakan kelas k. Prediksi negatif yang salah.

O. ***Evaluation Metrics***

Evaluation Metrics adalah evaluasi kuantitatif yang digunakan untuk mengukur seberapa baik model dalam melakukan prediksi [23] [24].

1. *Precision*: proporsi prediksi positif yang benar (TP) dari seluruh prediksi positif.

$$Precision = \frac{TP}{\mathrm{TP} + \mathrm{FP}} \quad (1)$$

2. *Recall*: adalah proporsi prediksi positif yang benar (TP) dari seluruh kasus positif yang sebenarnya.

$$Recall = \frac{TP}{\mathrm{TP} + \mathrm{FN}} \quad (2)$$

3. *F1-Score*: adalah rata-rata dari precision dan recall

$$\mathrm{F1} = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (3)$$

4. Accuracy: adalah proporsi keseluruhan prediksi yang benar.

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \times 100\% \quad (3)$$

P. ***Dataset***

Dataset yang digunakan pada penelitian ini adalah *Corn or Maize Leaf Disease Dataset* dari sumber terbuka di *platform* Kaggle [25]. Dataset ini terdiri dari 4 kategori kelas, yaitu: *Blight*, *Common Rust*, *Gray Leaf Spot*, dan *Healthy*. Di mana *Blight, Common Rust*, dan *Gray Leaf Spot* merupakan penyakit yang umum ditemukan pada daun jagung, dan *Healthy* adalah daun yang sehat.

Gambar di bawah merupakan beberapa contoh gambar *dataset* daun jagung yang terserang penyakit *blight* pada gambar 3.1, *common rust* pada gambar 3.2, *gray leaf spot* pada gambar 3.3, dan daun yang sehat pada gambar 3.4. Gambar-gambar ini dapat membantu memberikan gambaran seperti apa perbedaan ketiga daun yang terkena penyakit dan daun yang sehat.

GAMBAR 1
(*Blight*)

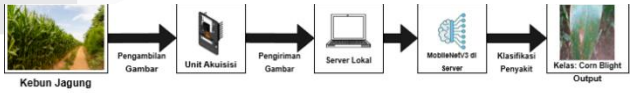
GAMBAR 2
(*Common Rust*)

GAMBAR 3
(*Gray Leaf Spot*)

GAMBAR 4
(*Healthy*)

III. METODE

A. Arsitektur Sistem Deteksi



GAMBAR 5
(Alur Sistem Deteksi)

Sistem deteksi penyakit daun jagung dalam penelitian ini dirancang dengan mengadopsi model arsitektur *Local Client-Server*. Pemilihan arsitektur ini bertujuan untuk meminimalkan *latency* antara proses pengambilan gambar dan penerimaan hasil prediksi. Dalam skema ini, sistem bekerja dengan mekanisme berbasis *request*. Artinya, sistem dioptimalkan untuk memberikan respon diagnosis yang cepat (*responsive*) segera setelah pengguna melakukan pengambilan gambar secara manual. Diagram alur pada

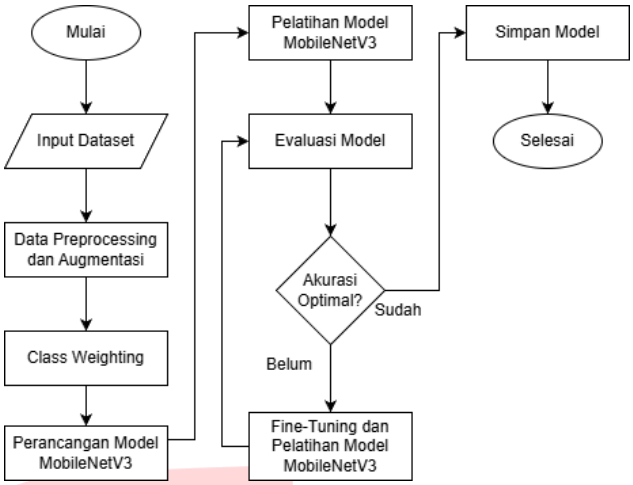
Gambar 5 memvisualisasikan arsitektur sistem yang terdiri dari tiga komponen utama:

1. Unit Akuisisi (*Client Node*): Menggunakan kamera yang berfungsi sebagai alat akuisisi gambar. Unit ini bertugas menangkap gambar daun jagung untuk dikirim ke server melalui jaringan Wi-Fi.
2. Jalur Transmisi Data (*Communication Layer*): Menggunakan infrastruktur jaringan Wi-Fi lokal (*Local Area Network*) sebagai media penghubung. Metode transmisi menggunakan protokol HTTP POST, di mana data dikirim langsung ke alamat IP *server* tanpa melalui perantara internet publik, sehingga meminimalkan latensi pengiriman.
3. Unit Pemrosesan Pusat (*Local Server*): Berupa perangkat laptop yang menjalankan layanan *backend*. Unit ini bertugas melakukan *preprocessing* gambar, menjalankan MobileNetV3 untuk deteksi, serta memberikan hasil deteksi secara visual kepada pengguna melalui *dashboard monitoring*.

B. Perancangan Model *Deep Learning*

Pengembangan model diawali dengan penyiapan *dataset* sebagai basis pengetahuan utama bagi sistem. Dataset penyakit pada tanaman jagung yang digunakan dalam penelitian ini mencakup empat kelas klasifikasi kondisi daun jagung. Sebelum diproses oleh arsitektur jaringan saraf, seluruh gambar dalam dataset diseragamkan dimensinya menjadi 224 x 224 piksel untuk memenuhi standar *input* model MobileNetV3, serta dibagi menjadi data *training*, *validation*, dan *testing* guna menjamin objektivitas evaluasi kinerja model.

Subbab ini menguraikan tahapan perancangan dan pembangunan model *Deep Learning* yang berfungsi sebagai inti pemrosesan dalam sistem deteksi penyakit. Arsitektur yang dipilih dalam penelitian ini adalah MobileNetV3. Gambar 6 berikut menggambarkan alur pelatihan model *Deep Learning*. Proses dimulai dari *preprocessing dataset* dan augmentasi data. Pelatihan dilakukan menggunakan strategi *Transfer Learning* dengan *Fine-Tuning*. Pada tahap ini, 50 *layer* terakhir dari model dibuka kuncinya (*unfreeze*) dan dilatih ulang untuk meningkatkan akurasi pada fitur spesifik penyakit daun jagung. *Transfer Learning* bertujuan untuk melatih model yang sudah dilatih sebelumnya dari dataset ImageNet agar dapat mengenali fitur penyakit daun jagung secara presisi tanpa memerlukan waktu pelatihan yang berlebihan. Model yang disimpan adalah model dengan akurasi terbaik.



GAMBAR 5
(*Flowchart* Perancangan Model *Deep Learning*)

1. *Input Dataset*: Tahap ini diawali dengan memuat *dataset* gambar daun jagung dan membaginya menjadi empat kelas (*Blight*, *Common Rust*, *Gray Leaf Spot*, *Healthy*).
2. *Data Preprocessing* dan Augmentasi: Pada tahap ini, semua gambar akan diubah ukurannya menjadi 224x224 piksel sesuai spesifikasi input MobileNetV3, serta penerapan augmentasi data (*rotate*, *zoom*, *flip*) untuk memperkaya variasi data untuk pelatihan model.
3. *Class Weighting*: Sebelum pelatihan dimulai, tiap kelas akan diberi bobot yang berbeda untuk mengatasi ketimpangan jumlah data, sehingga model lebih sensitif terhadap kelas minoritas
4. Perancangan Model MobileNetV3: Memanggil arsitektur MobileNetV3Large yang telah dilatih pada ImageNet sebagai base model. Bagian klasifikasi (*head*) diganti dengan lapisan baru: *Global Average Pooling*, *Dropout* (0.2) untuk mencegah *overfitting*, dan *Dense Layer* dengan fungsi aktivasi *Softmax* untuk empat kelas output
5. Pelatihan Model: Pada tahap ini, base model dibekukan agar bobot aslinya tidak berubah. Pelatihan hanya dilakukan pada lapisan klasifikasi baru selama 20 *epoch* menggunakan *optimizer* Adam (*Learning Rate* 0.001). Tujuannya agar model dapat mengenali mengenali pola dasar penyakit daun jagung
6. *Fine-Tuning*: Pada tahap ini akan dilakukan optimasi dengan *unfreeze* 50 *layer* terakhir dari *base model*, lalu Model dilatih kembali selama 10 *epoch* tambahan dengan *Learning Rate* yang lebih kecil (0,00001). Langkah ini bertujuan meningkatkan sensitivitas model terhadap fitur-fitur penyakit yang memiliki kemiripan visual (seperti *Blight* dan *Gray Leaf Spot*).
7. Evaluasi Model: Setelah pelatihan, evaluasi dilakukan untuk menilai kinerja model dengan metrik *accuracy*, *precision*, *recall*, dan *F1-score*. Analisis lebih dalam dilakukan menggunakan *Confusion Matrix* untuk memastikan prediksi antar kelas sudah cukup akurat.

8. Simpan Model: Pada tahap ini, model terbaik akan disimpan dan siap untuk digunakan untuk digunakan pada *server*.

C. *Perancangan Unit Akuisisi Gambar*

Perancangan unit akuisisi gambar bertujuan untuk menyediakan unit yang bertugas menangkap data visual daun jagung. Dalam arsitektur *Client-Server* yang dibangun, unit ini berfungsi sebagai *Client Node* yang beroperasi untuk mengakuisisi dan mentransmisikan gambar ke *server* melalui protokol komunikasi nirkabel.

D. *Spesifikasi Minimum Perangkat Client*

Mengingat tujuan penelitian untuk memvalidasi keandalan protokol komunikasi data dan akurasi model Deep Learning, sistem dirancang agar dapat menerima input dari berbagai jenis perangkat client. Tabel 2 memaparkan spesifikasi minimum yang dapat digunakan untuk pengujian.

TABEL 2
(SPESIFIKASI MINIMUM PERANGKAT CLIENT)

Parameter	Spesifikasi	Keterangan
Konektivitas	Wi-Fi 802.11 b/g/n	Memastikan perangkat dapat terhubung ke jaringan lokal yang sama dengan *server*.
Resolusi Kamera	Resolusi 2 Megapixel	Resolusi yang cukup untuk menangkap detail pola penyakit pada permukaan daun.
Protokol	HTTP/1.1 Client	Kemampuan mengirim HTTP POST *Request* untuk transmisi data.
Format *Payload*	Multipart/Form-Data	Format standar data gambar (JPEG/PNG) agar terbaca oleh *endpoint server*.
Kompatibilitas	*Browser* / REST *Client App*	Antarmuka untuk menjalankan fungsi pengiriman data oleh pengguna.

E. *Perancangan Server dan Dashboard*

Perancangan sisi *server* dan antarmuka pengguna bertujuan untuk menjembatani komunikasi antara *client* dengan model deteksi. Sistem ini dibangun menggunakan arsitektur *Client-Server* berbasis protokol HTTP REST API, yang memungkinkan pertukaran data secara asinkron dan efisien.

F. Perancangan *Backend Server*

Layanan *backend* dibangun menggunakan bahasa pemrograman Python dengan *micro framework* Flask. Pemilihan Flask didasarkan pada arsitekturnya yang ringan dan kemudahan integrasi dengan *library Deep Learning* seperti TensorFlow/Keras. *Backend server* dirancang untuk menangani dua tugas utama: Memuat Model, dan Manajemen API *Endpoint.*

1. *Input Dataset*: Tahap ini diawali dengan memuat *dataset* gambar daun jagung dan membaginya menjadi empat kelas (*Blight*, *Common Rust*, *Gray Leaf Spot*, *Healthy*).
2. Manajemen API *Endpoint*:
   a. *Endpoint* /*upload* (POST): Titik akses ini dirancang untuk menerima data gambar dari perangkat akuisisi. Pada sisi *server*, gambar yang diterima akan disimpan sementara, lalu melalui tahap *preprocessing* sebelum diproses oleh model. Hasil prediksi kemudian disimpan dalam variabel global sistem.
   b. *Endpoint* /*status* (GET): Titik akses ini berfungsi sebagai penyedia data untuk antarmuka pengguna. *Endpoint* ini mengembalikan status terakhir sistem yang meliputi label prediksi, dan tingkat keyakinan (*confidence score*) dalam format JSON.

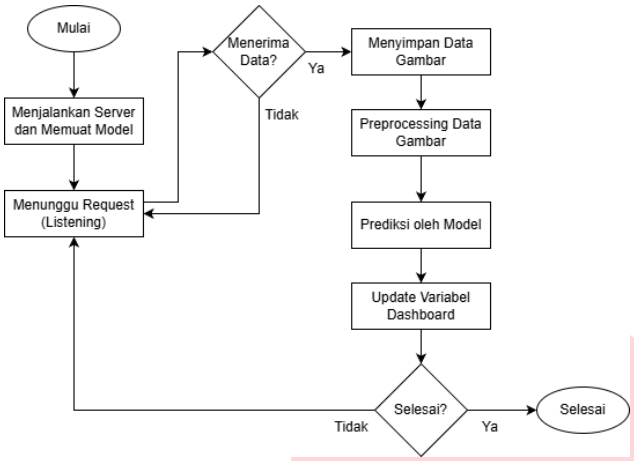
G. Perancangan *Dashboard Monitoring*

Dashboard Monitoring dirancang sebagai aplikasi *web* satu halaman yang responsif. Tujuannya adalah menampilkan hasil deteksi penyakit daun jagung secara *real-time* kepada pengguna tanpa memerlukan interaksi manual (seperti refresh halaman). Mekanisme pembaruan data menggunakan teknik AJAX Polling (*Asynchronous* JavaScript and XML). Berikut adalah logika perancangannya:

1. Struktur Halaman (HTML): Halaman *web* disusun untuk menampilkan komponen utama berupa gambar daun terkini, panel status penyakit (Sehat/Sakit), dan tingkat keyakinan model.
2. Logika Pembaruan Asinkron (JavaScript): Sebuah skrip berjalan di latar belakang yang secara berkala mengirimkan permintaan HTTP GET ke *endpoint* /status pada *server*.
3. Manipulasi DOM Dinamis: Ketika skrip mendeteksi adanya perubahan data (berdasarkan perbandingan timestamp data baru dan lama), skrip akan memperbarui elemen *Document Object Model* (DOM) secara otomatis. Hal ini memungkinkan gambar dan teks pada *dashboard* berubah secara instan begitu data baru diterima dari perangkat akuisisi

H. Flowchart Logika Program *Server*

Flowchart logika program pada sisi *server* dimulai dari inisialisasi model hingga penanganan permintaan klien. *Flowchart* berikut menggambarkan bagaimana *server* memproses permintaan.



GAMBAR 6
(*Flowchart Server*)

1. Menjalankan *Server* dan Memuat Model: Memulai program *server* dan memuat model MobileNetV3 pada *server*.
2. *Listening*: Pada tahap ini, server memasuki mode *standby*, dimana *server* menunggu *request* masuk pada *port* yang sudah ditentukan. Jika *server* menerima data, *server* akan melakukan valisadi data yang diterima dan menyimpannya sementara di folder.
3. *Preprocessing* dan Prediksi: Gambar mentah yang diterima dari klien akan diubah ukurannya secara otomatis menjadi dimensi 224x224 piksel. Proses ini memastikan bahwa input yang masuk ke model MobileNetV3 selalu seragam dan sesuai dengan dimensi yang dibutuhkan saat pelatihan.
4. *Update* Variabel: Setelah prediksi selesai dilakukan, hasil prediksi disimpan dalam variabel global agar bisa ditampilkan pada *dashboard*.

IV. HASIL DAN PEMBAHASAN

A. Lingkungan dan Parameter Pengujian Model

Eksperimen pelatihan dan pengujian model dilakukan pada platform komputasi awan Google Colab dengan dukungan perangkat keras GPU NVIDIA T4 Tensor Core. Kerangka kerja (*framework*) utama yang digunakan adalah TensorFlow dan Keras. Parameter konfigurasi data dan *hyperparameter* pelatihan ditetapkan untuk memastikan hasil eksperimen konsisten. Tabel 3 menyajikan rincian konfigurasi.

TABEL 3
(KONFIGURASI PELATIHAN)

Parameter	Konfigurasi / Nilai	Keterangan
Arsitektur Model	MobileNetV3Large	*Pre-trained model* pada ImageNet
Input Shape	224x224x3	Resolusi standar input MobileNetV3Large
Rasio Pembagian Data	80 : 10:10	80% untuk *Training* 10% untuk *Validation* 10% untuk *Test*
Batch Size	32	Jumlah sampel per propagasi
Optimizer	Adam	Algoritma optimasi adaptif
Loss Function	*Categorical Crossentropy*	Untuk klasifikasi multi kelas
Class Weights	*Blight*: 0,91 *Common_Rust*: 0,80 *Gray_Leaf_Spot*: 1,82 *Healthy*: 0,90	Pemberian bobot pada tiap kelas untuk mengatasi data yang tidak seimbang

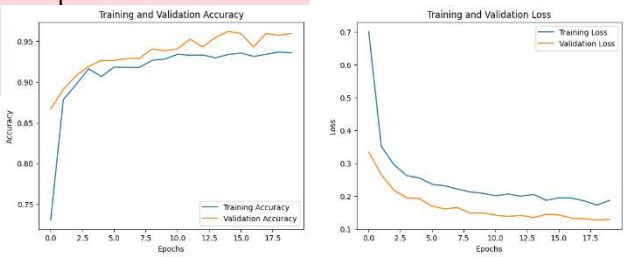
Pelatihan dilakukan dalam dua fase, yaitu: *Base Model Training*, dan *Fine-Tuning*.

1. *Base Model Training*: Pada tahap ini, sebagian model dibekukan agar tidak mengubah bobot saat *back propagation* pada *base model* yang sudah terlatih sebelumnya di ImageNet. Bagian yang tetap aktif pada tahap ini adalah pada lapisan klasifikasi (*head*) dengan bobot yang diinisialisasi secara acak.
2. *Fine-Tuning*: Tahap *fine-tuning* bertujuan untuk spesialisasi fitur. Pola Fitur yang sudah dipelajari MobileNetV3 dari ImageNet masih bersifat generik. Pada tahap ini, 50 lapisan terakhir dibuka (*unfreeze*) dan melatihnya dengan *learning rate* 0,00001 dan dilatih lagi selama 15 *epoch*, jadi model berkesempatan untuk mengubah filter konvolusinya agar lebih sensitif terhadap karakteristik penyakit daun jagung tanpa melupakan fitur dasar yang telah dipelajari sebelumnya.

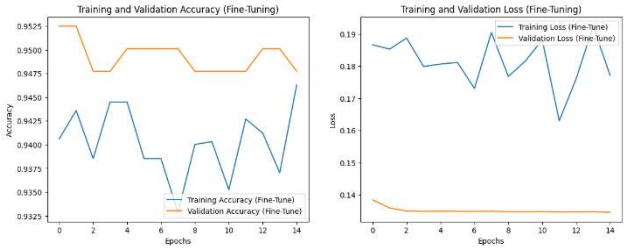
Skenario Pengujian Fungsional Sistem: Pengujian fungsional bertujuan untuk memvalidasi arsitektur *Client-Server*. Pengujian ini dilakukan dengan menggunakan kamera *smartphone* untuk melakukan akuisisi dan pengiriman gambar ke *server lokal* melalui protokol HTTP POST. Indikator keberhasilan pengujian adalah diterimanya respon status HTTP 200 dan hasil prediksi oleh model berhasil ditampilkan di *dashboard.*

B. Hasil Pelatihan Model

Proses pelatihan model MobileNetV3Large berlangsung sebanyak 2 fase, dimana fase pertama (*base model*) dilakukan selama 20 *epoch*, dan fase kedua (*fine-tuning*) selama 15 *epoch*. Gambar 4.1 dan Gambar 4.2 menunjukkan grafik pergerakan nilai akurasi dan *loss* selama proses pelatihan.



GAMBAR 7
(Grafik Nilai Akurasi dan *Loss* Fase 1)

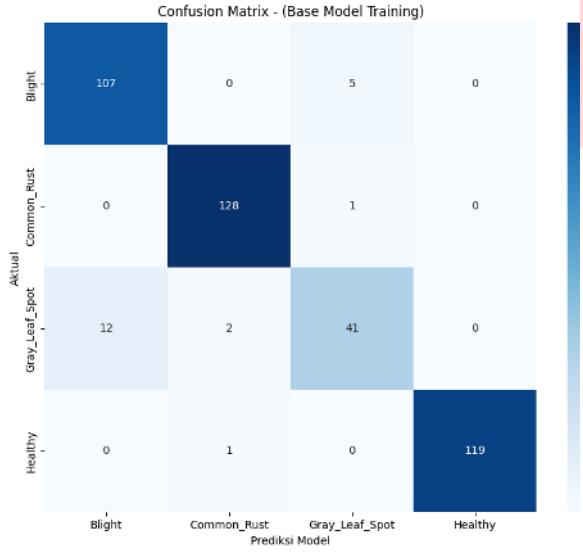


GAMBAR 8
(Grafik Nilai Akurasi dan *Loss* Fase 2)

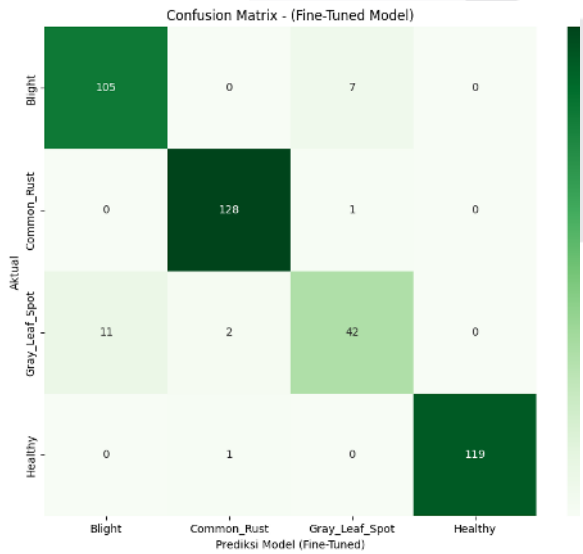
1. Analisis Pergerakan Grafik *Accuracy*: Grafik akurasi di atas memperlihatkan pergerakan akurasi model terhadap data latih, di mana garis biru adalah *Training Accuracy* dan garis oranye adalah *Validation Accuracy*. Pada tahap awal (*Base Model*), kenaikan kurva langsung melonjak tinggi dan bergerak stabil. Hal ini menunjukkan bahwa bobot pada *pre-trained* model dari ImageNet sangat efektif dalam melakukan ekstraksi fitur dasar, sehingga *classifier head* dapat beradaptasi dengan cepat. Namun, setelah bagian konvolusi dibuka (*unfreeze*) pada *fase fine-tuning*, grafik tidak mengalami peningkatan yang signifikan dan cenderung stagnan. Kondisi ini mengindikasikan bahwa model telah mencapai titik optimalnya pada fase pertama, sehingga pelatihan lanjutan tidak lagi memberikan kontribusi terhadap generalisasi fitur.
2. Analisis Pergerakan Grafik *Loss*: Grafik loss di atas memperlihatkan pergerakan nilai kesalahan model terhadap data latih, di mana garis biru adalah *Training Loss* dan garis oranye adalah *Validation*

Loss. Mirip seperti grafik akurasi, nilai *loss* mengalami penurunan drastis secara mulus pada awal pelatihan fase pertama. Penurunan yang drastis ini menunjukkan efektivitas optimizer Adam dengan learning rate 0,001 dalam meminimalkan kesalahan klasifikasi secara cepat. Sebaliknya, pada tahap *fine-tuning*, grafik validasi mulai menunjukkan ketidakstabilan (fluktuasi) dibandingkan grafik training. Hal ini menandakan bahwa model mulai mengalami kesulitan untuk memperbaiki bobot tanpa merusak fitur yang sudah terbentuk, yang menegaskan bahwa model dari fase pertama adalah model yang paling *robust*.

3. Analisis *Confusion Matrix*: Analisis komparatif pada *confusion matrix* dilakukan untuk menelusuri sumber kesalahan prediksi secara spesifik antar kelas. Gambar 4.3 dan Gambar 4.4 menampilkan *confusion matrix* dari pelatihan fase pertama dan kedua.



GAMBAR 8
(*Confusion Matrix* Model Fase 1)



GAMBAR 9
(*Confusion Matrix* Model Fase 2)

Pada fase *base model* training, kesalahan klasifikasi paling signifikan dalam penelitian ini terjadi antara *Gray Leaf Spot* dan *Blight*. Kesalahan ini mengindikasikan bahwa kedua penyakit tersebut kemiripan pola, keduanya memiliki pola lesi berwarna coklat pada daun, lesi pada *Gray Leaf Spot* cenderung berbentuk persegi panjang, sedangkan lesi pada *Blight* berbentuk elips memanjnag, ini menyebabkan model kesulitan untuk membedakan antara *Gray Leaf Spot* dan *Blight.* Setelah dilakukan *fine-tuning*, jumlah kesalahan ini menurun untuk prediksi *Blight* dengan aktual *Gray Leaf Spot.*

4. Analisis *Evaluation Metrics*: Evaluasi kinerja model dilakukan menggunakan metrik *Accuracy*, *Precision*, *Recall*, dan *F1-Score* untuk setiap kelas. Analisis ini bertujuan untuk melihat seberapa baik model menangani ketimpangan data dan membedakan fitur penyakit yang memiliki kemiripan pola. Tabel 4.2 dan Tabel 4.3 Menampilkan tabel *evaluation metrics* untuk setiap kelas.

TABEL 4
(BASE MODEL TRAINING EVALUATION METRICS)

Metrik Evaluasi	*Blight*	*Common_Rust*	*Gray Leaf Spot*	*Healthy*
Precision	0,8992	0,9772	0,8723	1,00
Recall	0,9554	0,9922	0,7455	0,9917
F1-Score	0,9264	0,9846	0,8039	0,9958
Accuracy	0,9495			

TABEL 5
(BASE MODEL TRAINING EVALUATION METRICS)

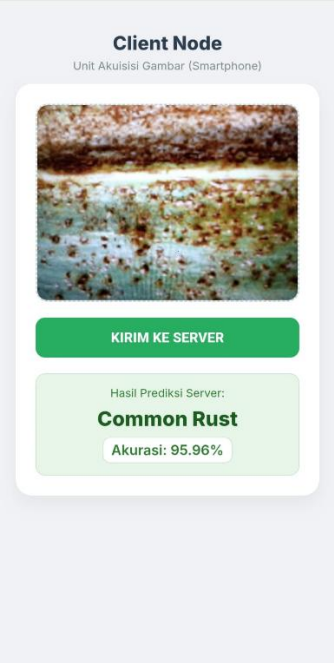
Metrik Evaluasi	*Blight*	*Common_Rust*	*Gray Leaf Spot*	*Healthy*
Precision	0,9052	0,9771	0,8400	1,00
Recall	0,9375	0,9922	0,7636	0,9917
F1-Score	0,9211	0,9846	0,8000	0,9958
Accuracy	0,9471			

Berdasarkan perbandingan tabel di atas, nilai *recall* dari kelas *Gray Leaf Spot* pada fase *base model training* tercatat sebesar 0,74. Nilai ini relatif lebih rendah dibandingkan kelas lainnya, yang selaras dengan temuan pada confusion matrix di mana model cenderung salah mengenali *Gray Leaf Spot* sebagai *Blight.* Setelah dilakukan *fine-tuning*, nilai *recall* dari *Gray Leaf Spot* hanya mengalami peningkatan menjadi 0,76. Namun, peningkatan kecil ini diikuti dengan penurunan nilai *precision* yang cukup signifikan dari 0,87 menjadi 0,84. Hal ini menyebabkan nilai *F1-Score* untuk kelas tersebut stagnan pada angka 0,80. Penurunan ini menunjukkan bahwa proses *unfreezing* pada 50 lapisan terakhir tidak memberikan perbaikan yang berarti; model justru menjadi kurang presisi

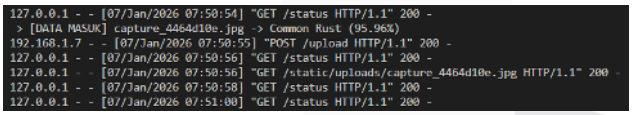
dan lebih banyak menghasilkan prediksi *false positives*. Kondisi ini menegaskan bahwa *Base Model* memiliki kinerja yang lebih *robust* dan stabil dibandingkan model hasil *fine-tuning*.

C. Pengujian Fungsionalitas *Client-Server*

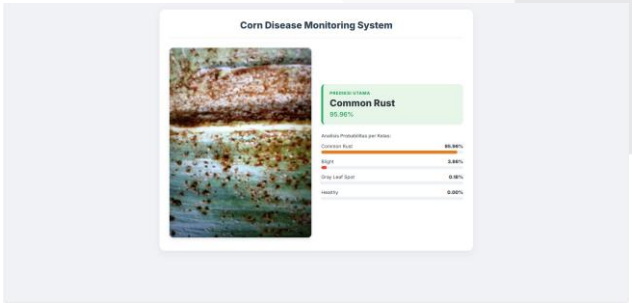
Pengujian ini bertujuan untuk membuktikan bahwa komunikasi antara sisi *Client* dan sisi *Server* berjalan lancar. Gambar 10 menunjukkan tampilan dari perangkat akuisisi, dam Gambar 11 menunjukkan bahwa *server* berhasil menerima gambar yang dikirim. Kemudian *browser* akan mengekstrak file JSON dari *server* untuk ditampilkan di *dashboard monitoring* pada Gambar 12



GAMBAR 10
(Pengiriman Gambar dari Perangkat Akuisisi)



GAMBAR 11
(*Log Server* Menerima Gambar)



GAMBAR 12
(Tampilan *Dashboard*)

V. KESIMPULAN

Penelitian ini menyimpulkan bahwa penerapan arsitektur Deep Learning MobileNetV3Large dengan metode *Transfer Learning* sangat efektif untuk mendeteksi penyakit daun jagung, di mana model fase pelatihan awal (*Base Model*) terbukti lebih unggul dibandingkan model hasil *fine-tuning*. Berdasarkan hasil uji komparasi, *Base Model* mampu mencapai akurasi validasi tertinggi sebesar 94,95% dengan stabilitas grafik yang lebih baik, sedangkan upaya *fine-tuning* justru memicu saturasi konvergensi dan penurunan presisi pada kelas minoritas. Oleh karena itu, bobot dari fase *transfer learning* ditetapkan sebagai model *final* karena kemampuannya yang optimal dalam mengekstraksi fitur visual penyakit tanpa mengalami gejala *overfitting* maupun degradasi performa yang muncul pada skenario pelatihan lanjutan.

Secara keseluruhan, penelitian ini berhasil mengintegrasikan model cerdas tersebut ke dalam sistem deteksi berbasis arsitektur *Local Client-Server* yang solutif untuk kondisi perkebunan. Penggunaan kamera *smartphone* sebagai perangkat akuisisi gambar memberikan fleksibilitas tinggi dan kemudahan operasional bagi pengguna di lapangan. Melalui komunikasi nirkabel pada jaringan lokal, gambar daun jagung dapat dikirimkan langsung dari *smartphone* ke *server* lokal untuk diproses tanpa ketergantungan pada koneksi internet publik. Sinergi antara keandalan model dalam mendeteksi penyakit dan kepraktisan penggunaan perangkat seluler ini membuktikan bahwa sistem yang dibangun telah memenuhi tujuan penelitian, yaitu menyediakan alat bantu diagnosis yang akurat, dan mudah diakses.

REFERENSI

[1] A. S. Dewi, D. H. Setiawan, and R. Novitaningrum, “Potensi dan Pengembangan Jagung Hibrida dI Indonesia,” Nov. 2022, doi: https://doi.org/10.47701/sintech.v3i1.2518.

[2] C. Bi, S. Xu, N. Hu, S. Zhang, Z. Zhu, and H. Yu, “Identification Method of Corn Leaf Disease Based on Improved Mobilenetv3 Model,” Agronomy, vol. 13, no. 2, Feb. 2023, doi: 10.3390/agronomy13020300.

[3] L. Li, S. Zhang, and B. Wang, “Plant Disease Detection and Classification by Deep Learning - A Review,” Mar. 2021, Institute of Electrical and Electronics Engineers Inc. doi: 10.1109/ACCESS.2021.3069646.

[4] E. Aditya and A. Kusuma Wardhana, “Exploration of Machine Learning Algorithms and Class Imbalance Handling with Deep Feature Extraction Using ResNet50 for Plant Disease Detection,” Oct. 2025. doi: https://doi.org/10.30871/jaic.v9i5.10338.

[5] J. Zhang, X. Yang, X. Fu, B. Wang, and H. Li, “LDL-MobileNetV3S: an enhanced lightweight MobileNetV3-small model for potato leaf disease diagnosis through multi-module fusion,” Front Plant Sci, vol. 16, Oct. 2025, doi: 10.3389/fpls.2025.1656731.

[6] A. Gao, A. J. Geng, Y. P. Song, L. L. Ren, Y. Zhang, and X. Han, “Detection of maize leaf diseases using improved MobileNet V3-small,” International Journal of Agricultural and Biological Engineering, vol. 16, no. 3, pp. 225–232, May 2023, doi: 10.25165/j.ijabe.20231603.7799.

[7] W. Maximilliano and N. Rachmat, “Comparative Analysis of MobileNetV3-Large and Small for Corn Leaf

Disease Classification," Brilliance: Research of Artificial Intelligence, vol. 5, no. 1, pp. 325–332, Jul. 2025, doi: 10.47709/brilliance.v5i1.6259.

[8] S. Choudhary, N. Dhote, and M. L. Kolhe, "Detection of Plant Leaf Diseases Using Internet of Things and Machine Learning Based Drone System," in Advances in Transdisciplinary Engineering, IOS Press BV, Jul. 2024, pp. 703–712. doi: 10.3233/ATDE240516.

[9] E. Ozbilge, M. K. Ulukok, O. Toygar, and E. Ozbilge, "Tomato Disease Recognition Using a Compact Convolutional Neural Network," IEEE Access, vol. 10, pp. 77213–77224, Jul. 2022, doi: 10.1109/ACCESS.2022.3192428.

[10] V. Naresh, G. Yogeswararao, B. Nageswararao Naik, R. Malmathanraj, and P. Palanisamy, "Empirical Analysis of Squeeze and Excitation-Based Densely Connected CNN for Chili Leaf Disease Identification," IEEE Transactions on Artificial Intelligence, vol. 5, no. 4, pp. 1681–1692, Apr. 2024, doi: 10.1109/TAI.2024.3364126.

[11] C. Janiesch, P. Zschech, and K. Heinrich, "Machine learning and deep learning," 2021, doi: 10.1007/s12525-021-00475-2/Published.

[12] M. M. Taye, "Theoretical Understanding of Convolutional Neural Network: Concepts, Architectures, Applications, Future Directions," Mar. 01, 2023, MDPI. doi: 10.3390/computation11030052.

[13] S. Qian, Y. Hu, and Ning. Chenran, MobileNetV3 for Image Classification. Institute of Electrical and Electronics Engineers, Inc., 2021. doi: 10.1109/ICBAIE52039.2021.9389905.

[14] A. Hosna, E. Merry, J. Gyalmo, Z. Alom, Z. Aung, and M. A. Azim, "Transfer learning: a friendly introduction," J Big Data, vol. 9, no. 1, Dec. 2022, doi: 10.1186/s40537-022-00652-w.

[15] M. Sholihin, Moh. R. Zamroni, L. Anifah, M. F. M. Fudzee, and M. N. Ismail, "Fine-Tuned Transfer Learning with InceptionV3 for Automated Detection of Grapevine Leaf Diseases," Jurnal Teknik Informatika (Jutif), vol. 6, no. 5, pp. 3683–3696, Oct. 2025, doi: 10.52436/1.jutif.2025.6.5.4717.

[16] H. Ghazouani, W. Barhoumi, E. Chakroun, and A. Chehri, "Dealing with Unbalanced Data in Leaf Disease Detection: A Comparative Study of Hierarchical Classification, Clustering-based Undersampling and Reweighting-based Approaches," in Procedia Computer Science, Elsevier B.V., 2023, pp. 4891–4900. doi: 10.1016/j.procs.2023.10.489.

[17] M. Hassanin, S. Anwar, I. Radwan, F. S. Khan, and A. Mian, "Visual Attention Methods in Deep Learning: An In-Depth Survey," May 2024, doi: https://doi.org/10.1016/j.inffus.2024.102417.

[18] H. Kopetz and W. Steiner, Real-time systems: Design principles for distributed embedded applications. Springer, 2022. doi: 10.1007/978-3-031-11992-7.

[19] M. N. O. Sadiku and C. M. Akujuobi, Fundamentals of Computer Networks. Springer Science+Business Media, 2022. doi: 10.1007/978-3-031-09417-0.

[20] J. Wendroth and B. Jaeger, "A Brief Overview on HTTP," Nov. 2022, doi: 10.2313/NET-2022-11-1_11.

[21] S. Di Meglio, L. Libero, L. Starace, and S. Di Martino, "Starting a New REST API project? A Performance Benchmark of Frameworks and Execution Environments," 2023. [Online]. Available: https://luistar.github.io/

[22] Z. Zhang and T. Lv, "JSVE: JSON Schema Variant Extraction Based on Clustering and Formal Concept Analysis," IEEE Access, vol. 13, pp. 145517–145527, 2025, doi: 10.1109/ACCESS.2025.3599650.

[23] J. Erbani, P. É. Portier, E. Egyed-Zsigmond, and D. Nurbakova, "Confusion Matrices: A Unified Theory," IEEE Access, 2024, doi: 10.1109/ACCESS.2024.3507199.

[24] Ž. Vujović, "Classification Model Evaluation Metrics," International Journal of Advanced Computer Science and Applications, vol. 12, no. 6, pp. 599–606, 2021, doi: 10.14569/IJACSA.2021.0120670.

[25] S. Ghose, "Corn or Maize Leaf Disease Dataset," Kaggle. Accessed: Jan. 07, 2026. [Online]. Available: https://www.kaggle.com/datasets/smaranjitghose/corn-or-maize-leaf-disease-dataset