Implementasi Model MaxVit Untuk Deteksi Penyakit Daun Tanaman Bawang Merah Berbasis Mobile

Amanda Khoiroma'ul Soviyanti[1], Salamun Rohman Nudin[2]

[1,2]Manajemen Informatika, Fakultas Vokasi, Universitas Negeri Surabaya, Indonesia

E-mail: amandakhoiromaul.22063@mhs.unesa.ac.id*[1], salamunrohman@unesa.ac.id[2]

Abstrak. Bawang merah (*Allium cepa var. aggregatum*) merupakan komoditas hortikultura penting di Indonesia. Namun, produktivitasnya sering menurun akibat serangan penyakit daun yang sulit dikenali secara visual. Penelitian ini bertujuan mendeteksi penyakit daun bawang merah menggunakan model *Multi-Axis Vision Transformer* (MaxViT) dengan teknik *transfer learning* melalui klasifikasi citra daun. Dataset yang digunakan adalah Onion Dataset yang terdiri dari empat kelas, yaitu *Healthy, Purple Blotch, Leaf Blight,* dan *Iris Yellow Spot Virus*. Proses pelatihan model dilakukan menggunakan Python, TensorFlow, dan Google Colab dengan membandingkan *optimizer* Adam, AdamW, dan SGD. Evaluasi dilakukan menggunakan *confusion matrix* dengan metrik *accuracy, precision, recall, dan F1-score.* Hasil penelitian menunjukkan bahwa *optimizer* Adam menghasilkan performa terbaik dengan akurasi pengujian sebesar 98%. Model terbaik kemudian diimplementasikan ke dalam aplikasi *mobile* berbasis Flutter untuk mendukung deteksi penyakit daun bawang merah secara cepat dan mudah diakses.

Kata kunci: MaxViT; deteksi penyakit; bawang merah; transfer learning; Flutter

Abstract. *Shallots (Allium cepa var. aggregatum) are an important horticultural commodity in Indonesia. However, their productivity often declines due to leaf diseases that are difficult to identify visually. This study aims to detect shallot leaf diseases using the Multi-Axis Vision Transformer (MaxViT) model with a transfer learning approach through leaf image classification. The dataset used in this study was the Onion Dataset, which consists of four classes: Healthy, Purple Blotch, Leaf Blight, and Iris Yellow Spot Virus. The model training process was conducted using Python, TensorFlow, and Google Colab by comparing three optimizers: Adam, AdamW, and SGD. Model evaluation was performed using a confusion matrix with accuracy, precision, recall, and F1-score metrics. The results showed that the Adam optimizer achieved the best performance with a testing accuracy of 98%. The best-performing model was then implemented into a Flutter-based mobile application to support faster and more accessible early detection of shallot leaf diseases.*

Keywords: *MaxViT; disease detection; shallots; transfer learning; Flutter*

1. Pendahuluan

Bawang merah (*Allium cepa var. aggregatum*) merupakan salah satu komoditas *hortikultura* penting di Indonesia yang memiliki nilai ekonomi tinggi serta berperan dalam memenuhi kebutuhan pangan nasional. Berdasarkan data Kementerian Pertanian, produksi bawang merah nasional mencapai sekitar 1,98 juta ton pada tahun 2023 dengan nilai ekspor mencapai USD 11,67 juta [1]. Namun, produktivitas bawang merah masih sering mengalami penurunan akibat serangan berbagai penyakit daun yang menyebabkan kerusakan pada tanaman dan menurunkan hasil panen [2].

Penyakit *seperti Purple Blotch, Leaf Blight,* dan *Iris Yellow Spot Virus* pada daun bawang merah memiliki gejala visual yang mirip. Penyakit *Purple Blotch* biasanya menyebabkan munculnya lesi atau noda dengan warna ungu pada permukaan daun [3]. Selain itu, *Leaf Blight* ditandai dengan gejala klorosis dan nekrosis yang menyebar dari ujung daun [4], sedangkan *Iris Yellow Spot Virus* dapat menimbulkan pola mosaik pada daun [5]. Kemiripan gejala tersebut membuat proses identifikasi penyakit secara manual menjadi cukup sulit. Deteksi penyakit yang masih bergantung pada pengamatan visual memiliki batasan dari segi kecepatan dan akurasi diagnosis. Hal ini dapat mengakibatkan kesalahan dalam identifikasi dan keterlambatan dalam menangani penyakit pada tanaman [6]. Oleh karena itu, diperlukan metode berbasis teknologi yang dapat mendukung proses deteksi penyakit secara otomatis dan lebih akurat.

Perkembangan teknologi *computer vision* mempermudah dalam proses deteksi objek secara otomatis sehingga dapat mengurangi kesalahan pengamatan secara manual [7]. Teknologi ini memberikan kemampuan pada sistem untuk mengenali serta menganalisis informasi visual dari citra digital [8]. Proses kerja *computer vision* dirancang menyerupai sistem penglihatan manusia dalam melakukan identifikasi objek visual [9]. Dalam bidang pertanian, *artificial intelligence* dimanfaatkan untuk membantu mendeteksi gejala penyakit tanaman berdasarkan data yang diperoleh [10]. Pendekatan berbasis *deep learning* mampu melakukan ekstraksi fitur dan klasifikasi citra secara otomatis dengan tingkat akurasi yang tinggi [11]. Selain itu, teknik *transfer learning* banyak digunakan untuk mengatasi keterbatasan dataset dengan memanfaatkan model yang telah dilatih sebelumnya [12].

Beberapa penelitian sebelumnya telah menggunakan metode klasifikasi untuk penyakit daun bawang menggunakan pendekatan Convolutional Neural Network (CNN). Penggunaan kombinasi CNN dan SVM pada dataset lokal berhasil mencapai akurasi sebesar 75% [6]. Penelitian lain menggunakan arsitektur Xception memperoleh akurasi sebesar 97,37% pada klasifikasi penyakit daun bawang merah [13]. Selain itu, penelitian menggunakan Faster R-CNN dengan ResNet50 hanya memperoleh akurasi sebesar 65% [14].

Dengan pesatnya perkembangan teknologi *computer vision*, arsitektur berbasis *Vision Transformer* mulai menunjukkan hasil yang lebih baik dibandingkan CNN dalam tugas klasifikasi citra. Salah satu arsitektur terbaru adalah Multi Axis Vision Transformer (MaxViT) yang menggabungkan keunggulan CNN dengan mekanisme *self attention* untuk lebih efektif dalam menangkap fitur baik lokal maupun global [15]. Uji coba pada arsitektur ini menunjukkan bahwa MaxVit mencapai akurasi sebesar 95,75% dalam mengklasifikasikan penyakit pada daun bawang merah dengan menggunakan *Onion Dataset* [16]. Namun, penelitian tersebut masih berfokus pada penilaian model secara umum dan belum mengembangkan aplikasi khusus pada tanaman bawang merah berbasis perangkat mobile.

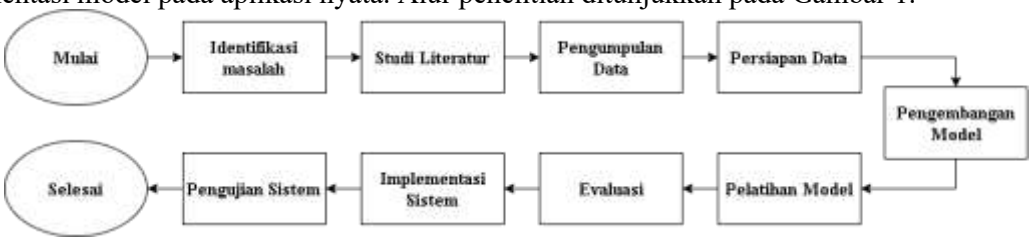
Performa klasifikasi model dipengaruhi oleh algoritma optimasi yang digunakan selama proses pelatihan. *optimizer* Adam dikenal mampu menyesuaikan *learning rate* secara adaptif sehingga proses pelatihan dapat berlangsung lebih stabil [17]. Selain itu, *optimizer* AdamW menggunakan mekanisme *weight decay* untuk mendukung regulasi model, sedangkan *optimizer* SGD memperbarui parameter model secara bertahap berdasarkan data yang digunakan [18]. Cara kerja yang berbeda dari setiap *optimizer* dapat berdampak pada hasil pelatihan dan performa klasifikasi yang diperoleh. Oleh karena itu, analisis perbandingan diperlukan untuk menentukan *optimizer* yang paling tepat untuk model MaxViT dalam pengklasifikasian penyakit pada daun bawang merah.

Penelitian ini menerapkan arsitektur MaxViT dengan teknik *transfer learning* untuk deteksi penyakit daun bawang merah menggunakan *Onion Dataset* yang terdiri dari empat kelas, yaitu *Healthy, Purple Blotch, Leaf Blight,* dan *Iris Yellow Spot Virus* [19]. Evaluasi model dilakukan menggunakan *confusion matrix* dengan metrik *accuracy, precision, recall, dan F1-score* [20]. Model terbaik kemudian diimplementasikan ke dalam aplikasi *mobile* berbasis Flutter bernama *ShallotEye*. Penggunaan

perangkat Android dipilih karena mampu memberikan akses informasi yang cepat dan praktis di lapangan [21], sedangkan Flutter digunakan karena dapat mendukung pengembangan aplikasi lintas platform dengan performa yang responsif melalui satu basis kode [22].

2. Metode

Penelitian ini menggunakan pendekatan eksperimen dengan mengadopsi tahapan dari IBM Data Science Methodology yang meliputi identifikasi masalah, pengumpulan data, persiapan data, pengembangan model, evaluasi model, dan implementasi sistem [10]. Metode ini dipilih karena sesuai untuk pengembangan sistem berbasis kecerdasan buatan yang melibatkan proses pengolahan data hingga implementasi model pada aplikasi nyata. Alur penelitian ditunjukkan pada Gambar 1.



Gambar 1. Alur Metode Penelitian

2.1. Identifikasi Masalah

Pada tahap ini dilakukan pemilihan topik penelitian serta penentuan rumusan masalah dan tujuan yang akan dicapai. Langkah ini bertujuan agar penelitian memiliki arah yang jelas dan fokus terhadap permasalahan yang dikaji. Proses identifikasi diawali dengan melakukan observasi mengenai potensi penerapan *deep learning*, khususnya model MaxViT dengan teknik *transfer learning*, dalam mengklasifikasi jenis penyakit pada daun tanaman bawang merah secara akurat.

2.2. Studi Literatur

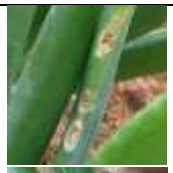
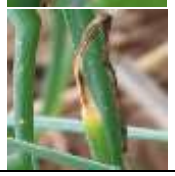
Tahap Studi literatur yaitu melakukan penelusuran berbagai referensi dan literatur yang relevan dengan topik penelitian, baik berupa buku, jurnal ilmiah, artikel, maupun sumber daring lainnya. Tujuan dari tahap ini adalah untuk memperoleh pemahaman yang lebih mendalam mengenai konsep, teori, serta metode yang mendukung penelitian. Literatur yang dikaji mencakup penelitian terkait penerapan teknik *transfer learning,* penggunaan arsitektur MaxViT dalam pengolahan citra, serta pemanfaatan *deep learning* dalam klasifikasi penyakit tanaman, khususnya pada daun bawang merah.

2.3. Pengumpulan Data

Dataset yang digunakan dalam penelitian ini merupakan dataset sekunder berupa Onion Dataset yang dikembangkan oleh Aishwarya dan Reddy [19]. Dataset pada Tabel 1 berisi citra daun bawang dengan empat kategori kelas penyakit dengan total dataset yang digunakan sebanyak 4.493 citra. Dataset ini dipilih karena memiliki variasi visual gejala penyakit daun yang relevan dengan kondisi bawang merah.

Tabel 1. Onion Dataset

Kategori	Jumlah data	Contoh gambar
Healthy	1.278	
Iris Yellow Spot Virus (IYSV)	1.272	

Purple blotch	735	
Leaf blight	1.217	

2.4. Persiapan Data

Pada tahap ini, data dipersiapkan agar memiliki kualitas yang baik sebelum digunakan dalam proses pelatihan model MaxViT. Dataset dibagi menjadi tiga bagian, yaitu *training set, validation set, dan testing set* dengan rasio 80:10:10. Pembagian ini didasarkan pada pertimbangan bahwa model memerlukan cakupan data yang luas untuk mengenali keragaman fitur visual, sehingga rasio terbesar dialokasikan pada data latih. Sementara data validasi dan data uji masing-masing mendapat rasio yang setara agar evaluasi performa dapat dilakukan secara objektif dan tidak memihak.

Ketidakseimbangan distribusi sampel pada Onion Dataset, dimana kelas *Purple Blotch* hanya memuat 735 citra, jauh dibawah kelas *Healthy* yang mencapai 1.278 citra menjadi pertimbangan utama dalam menentukan metode pembagian dataset. Oleh sebab itu, pembagian dilakukan melalui pendekatan *stratified sampling* untuk menjaga agar proporsi setiap kelas tetap terwakili secara seimbang di masing-masing subset, sehingga model tidak cenderung berpihak pada kelas dengan jumlah sampel yang lebih dominan

Seluruh citra diproses melalui tahap *resizing* dan normalisasi. Ukuran citra diseragamkan menjadi 224×224 piksel agar sesuai dengan kebutuhan input model MaxViT. Nilai piksel dinormalisasi ke dalam rentang antara 0 hingga 1 melalui pembagian 255,0. Proses ini membantu model dalam mempelajari pola data secara lebih stabil serta menyesuaikan format input dengan bobot *pre-trained* pada proses *transfer learning*. Tahap *preprocessing* ini menghasilkan data yang lebih konsisten dan siap digunakan pada proses pelatihan model.

2.5. Pengembangan Model

Tahap pengembangan model dilakukan menggunakan perangkat keras laptop ASUS VivoBook X415KA dengan spesifikasi Intel Celeron N4500 berkecepatan 1.10 GHz dengan 2 core dan 2 thread, RAM 8 GB, serta SSD 256 GB. Karena spesifikasi tersebut tergolong kategori *entry level*, proses pelatihan dialihkan secara daring memanfaatkan layanan *cloud* Google Colab Pro dengan spesifikasi perangkat keras berupa GPU NVIDIA Tesla T4 berkapasitas memori 15 GB. Seluruh proses dilakukan dengan menggunakan bahasa pemrograman *Python* dan pustaka *TensorFlow/Keras* karena efisiensi dan fleksibilitasnya dalam membangun arsitektur jaringan saraf tiruan [23]. Pada tahap ini dilakukan perancangan alur pemrosesan data, pemilihan arsitektur dasar, dan penyesuaian lapisan klasifikasi untuk mendeteksi penyakit daun bawang merah. Sebelum citra diproses oleh model, dilakukan normalisasi data dengan mengubah nilai piksel dari rentang 0–255 menjadi 0–1 melalui pembagian dengan 255.0 untuk memastikan skala input konsisten agar proses pelatihan model dapat berjalan dengan lebih stabil dan efisien.

Arsitektur dasar yang digunakan dalam penelitian ini adalah MaxViT Tiny dengan teknik *transfer learning* menggunakan *pre-trained weights* dari *ImageNet*. Seluruh lapisan pada model dasar diatur dalam kondisi *trainable* agar model dapat mempelajari karakteristik visual penyakit daun bawang merah secara menyeluruh. Karena MaxViT bawaan dirancang untuk klasifikasi umum, bagian *output* asli diganti dengan *custom classifier head*. Fitur hasil ekstraksi diproses menggunakan lapisan *GlobalAveragePooling2D* untuk mereduksi dimensi spasial tanpa menghilangkan informasi penting.

Lapisan *batch normalization* ditambahkan untuk membantu menstabilkan proses pembelajaran. Setelah itu, digunakan *dense layer* dengan 256 neuron dan fungsi aktivasi ReLU yang diikuti *dropout* 0,5 untuk mengurangi risiko *overfitting*, serta *dense layer* kedua berisi 128 neuron dengan aktivasi ReLU

dan *dropout* 0,3 sebagai regulasi tambahan. Pada tahap akhir, lapisan *dense* keluaran dengan 4 neuron dan fungsi aktivasi *softmax* digunakan untuk menghasilkan probabilitas klasifikasi pada empat kelas, yaitu *Healthy, Purple Blotch, Leaf Blight*, dan *Iris Yellow Spot Virus*.

2.6. Pelatihan Model

Tahap pelatihan model bertujuan untuk memperoleh konfigurasi terbaik melalui perbandingan tiga algoritma *optimizer* yang berbeda. Untuk menjaga konsistensi eksperimen, seluruh skenario menggunakan parameter yang sama, yaitu 50 *epoch, batch size* 16, dan *learning rate* 1e-4. Pelatihan dilakukan dengan memperbarui seluruh bobot model agar dapat mempelajari karakteristik penyakit pada daun bawang merah secara langsung. Selain itu, diterapkan mekanisme *callback* berupa *Early Stopping* untuk mencegah *overfitting* serta *ReduceLROnPlateau* untuk menyesuaikan *learning rate* secara dinamis guna mencapai konvergensi optimal. Detail skenario eksperimen disajikan pada Tabel 2.

Tabel 2. Skenario Pelatihan Model

Skenario	*Epoch*	Batch Size	Learning Rate	*Optimizer*
1	50	16	1e-4	No *Optimizer*
2	50	16	1e-4	Adam
3	50	16	1e-4	AdamW
4	50	16	1e-4	SGD

2.7. Evaluasi Model

Evaluasi dilakukan menggunakan metode *confusion matrix* berukuran 4×4 untuk menganalisis hasil klasifikasi pada empat kelas penyakit yaitu *Healthy*, *Purple Blotch*, *Leaf Blight*, dan *Iris Yellow Spot Virus*. Berdasarkan hasil *confusion matrix*, performa model diukur menggunakan 4 metrik evaluasi, yaitu *accuracy, precision, recall, dan F1-score* seperti berikut:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (1)$$

$$Precision = \frac{TP}{TP + FP} \quad (2)$$

$$Recall = \frac{TP}{TP + FN} \quad (3)$$

$$F1\ Score = 2 \times \frac{Recall\ \times precision}{Recall + precision} \quad (4)$$

Model terbaik dipilih berdasarkan nilai akurasi tertinggi, nilai *loss* terendah, serta keseimbangan nilai *precision*, *recall*, dan *F1-score*. Penggunaan *multi-metrik* ini sangat krusial mengingat adanya ketidakseimbangan jumlah sampel antarkelas pada dataset yang digunakan [24], sehingga performa model dapat dievaluasi secara menyeluruh dan terhindar dari bias. Kombinasi metrik ini memastikan model yang terpilih tidak hanya memiliki akurasi tinggi, tetapi juga *recall* dan *precision* yang seimbang dalam meminimalkan kesalahan prediksi positif palsu maupun negatif palsu di setiap kelas penyakit [25].

2.8. Implementasi Aplikasi Mobile

Model terbaik hasil evaluasi dikonversi dari format *.h5* menjadi *.tflite* untuk memperkecil ukuran file dan meningkatkan kecepatan inferensi pada perangkat dengan sumber daya terbatas [26]. Model tersebut kemudian diintegrasikan ke dalam aplikasi *mobile* berbasis Flutter yang diberi nama *ShallotEye*. Aplikasi ini memungkinkan pengguna mendeteksi penyakit daun bawang merah melalui unggahan file dari galeri maupun pengambilan foto secara langsung menggunakan kamera *smartphone*. *ShallotEye* dirancang untuk menampilkan hasil deteksi beserta *confidence score* secara instan, sehingga memudahkan petani atau pengguna dalam mengidentifikasi kondisi tanaman di lapangan.

2.9. Pengujian Sistem

Pengujian sistem dilakukan untuk mengevaluasi kinerja aplikasi *ShallotEye* secara *real-time* di lapangan. Skenario pengujian meliputi beberapa parameter, seperti variasi jarak pengambilan gambar, kondisi pencahayaan, sudut pengambilan citra, serta kompleksitas latar belakang. Variasi jarak pengambilan gambar dilakukan pada rentang 10-20 cm, sedangkan kondisi pencahayaan meliputi kondisi terang, redup, dan cahaya buatan. Selain itu, pengujian juga dilakukan dengan beberapa sudut kemiringan kamera saat mengarahkan citra ke daun bawang merah. Pada setiap skenario, sistem dievaluasi berdasarkan kemampuan dalam mengenali objek daun, keberhasilan proses deteksi, serta nilai kepercayaan yang dihasilkan untuk menganalisis stabilitas aplikasi secara objektif.

3. Hasil dan Pembahasan

Bagian ini menyajikan hasil penelitian, mulai dari tahap persiapan data hingga implementasi sistem pada aplikasi *mobile ShallotEye*.

3.1 Hasil Persiapan Data

Tahap *preprocessing* menghasilkan dataset yang siap digunakan untuk pelatihan model MaxViT. *Onion Dataset* terdiri dari 4.493 citra daun bawang merah yang terbagi ke dalam empat kelas, yaitu *Healthy*, *Purple Blotch*, *Leaf Blight*, dan *Iris Yellow Spot Virus*. Setelah melalui proses validasi data, dataset dibagi menjadi data latih, data validasi, dan data uji dengan rasio 80:10:10. Seluruh citra diseragamkan menjadi ukuran 224 × 224 piksel dan dinormalisasi agar sesuai dengan kebutuhan input model MaxViT. Distribusi dataset hasil *preprocessing* ditunjukkan pada Tabel 3.

Tabel 3. Hasil Pemisahan Dataset

Jenis Data	Persentase	Jumlah Data
Train set	80%	3.594
Validation set	10%	450
Test set	10%	449
Total	100%	4.493

3.2 Hasil Pengembangan Model

Pengembangan model kustom berbasis arsitektur MaxViT-Tiny berhasil diimplementasikan dan diverifikasi melalui ringkasan arsitektur model. Hasil kompilasi menunjukkan bahwa model memiliki total 838 layer terintegrasi secara terstruktur. Ringkasan arsitektur model yang meliputi nama layer, tipe operasional, fungsi aktivasi, dan jumlah parameter disajikan pada Tabel 4.

Tabel 4. Arsitektur Model

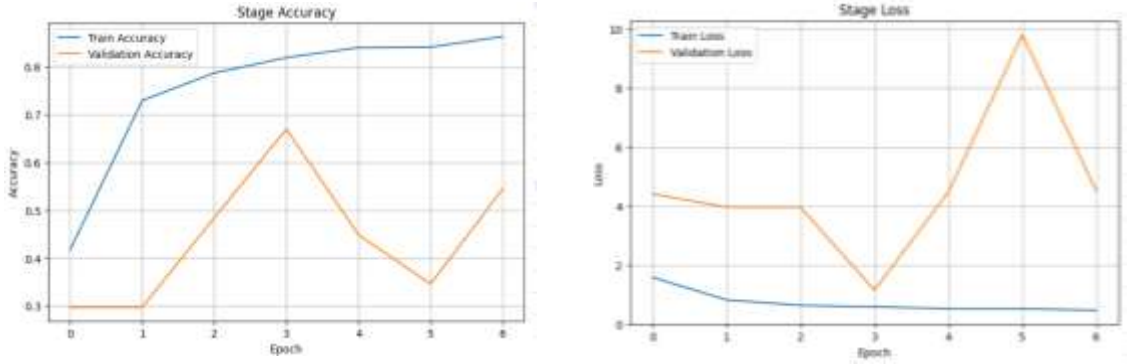
Layer	Tipe	Parameter
MaxViT-Tiny (backbone)	MaxViT	30,187,464
GlobalAveragePooling2D	GlobalAveragePooling2D	-
BatchNormalization	BatchNormalization	256
Dense(256, relu)	Dense	4,160
Dropout(0.5)	Dropout	-
Dense(128, relu)	Dense	16,640
Dropout(0.3)	Dropout	-
Dense(4, softmax) - Output	Dense	16,448
Total Layer		838
Total Parameter		30,354,252
Parameter Trainable		30,305,612
Parameter Non-Trainable		48,640

Berdasarkan hasil kompilasi, model memiliki total 30.354.252 parameter, dengan 30.187.464 parameter berasal dari *backbone* MaxViT-Tiny dan sisanya dari lapisan kustom *classifier head*. Dari total parameter tersebut, sebanyak 30.305.612 termasuk trainable parameters yang diperbarui selama

proses pelatihan, sedangkan 48.640 parameter lainnya termasuk *non trainable parameters* yang berasal dari lapisan *batch normalization*. Hasil ini menunjukkan bahwa arsitektur model berhasil dibangun dengan baik dan siap digunakan pada tahap pelatihan serta pengujian.

3.3 Hasil Pelatihan Model Skenario 1 Tanpa Optimizer

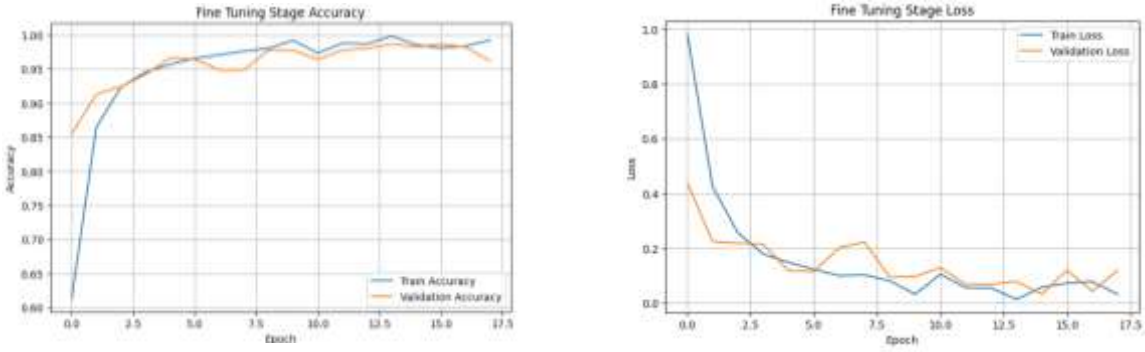
Pada grafik Gambar 2, proses pelatihan model tanpa penggunaan *optimizer* menunjukkan performa yang kurang stabil. Pada *epoch* awal 0-3, akurasi model sempat meningkat hingga sekitar 67%. Namun, pada *epoch* 4–6 akurasi validasi mengalami penurunan yang cukup drastis hingga mencapai sekitar 35%. Selain itu, nilai *validation loss* juga meningkat tajam hingga menyentuh angka 10 pada *epoch* ke-5. Kondisi ini menunjukkan bahwa model belum mampu melakukan generalisasi dengan baik dan cenderung mengalami *overfitting*. Oleh karena itu, proses pelatihan dihentikan pada *epoch* ke-6 oleh *callback Early Stopping* karena tidak terdapat peningkatan performa yang stabil.



Gambar 2. Pelatihan Model Skenario 1

3.4 Hasil Pelatihan Model Skenario 2 Menggunakan Optimizer Adam

Penggunaan *optimizer* Adam pada grafik Gambar 3 menunjukkan performa pelatihan yang lebih stabil dibandingkan skenario sebelumnya. Pada *epoch* 0-5, akurasi model meningkat cukup cepat dari sekitar 61% hingga mencapai lebih dari 95%. Selanjutnya pada *epoch* 6–17, performa model mulai stabil dengan akurasi tertinggi mendekati 99%. Nilai *training los*s dan *validation loss* juga terus mengalami penurunan secara konsisten hingga berada di bawah 0,1. Berbeda dengan skenario 1, pada skenario ini tidak terlihat fluktuasi yang ekstrem, sehingga dapat dikatakan bahwa *optimizer* Adam mampu meningkatkan stabilitas proses pelatihan model dengan baik.

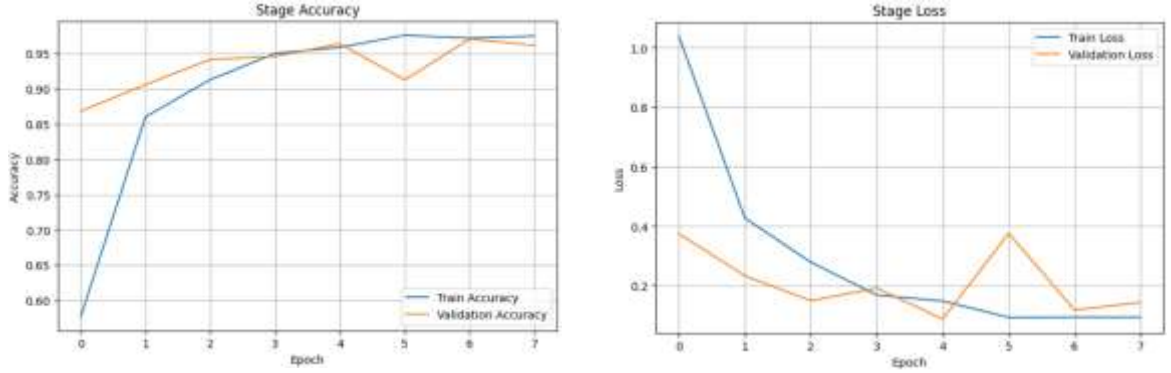


Gambar 3. Pelatihan Model Skenario 2

3.5 Hasil Pelatihan Skenario 3 Menggunakan Optimasi AdamW

Skenario 3 dengan *optimizer* AdamW pada Gambar 4 menunjukkan proses pelatihan yang cukup baik sejak *epoch* awal. Pada *epoch* 0–4, akurasi validasi telah mencapai lebih dari 85% dan terus meningkat hingga sekitar 96%. Meskipun sempat terjadi sedikit penurunan pada akurasi validasi di *epoch* ke-5, performa model kembali stabil pada *epoch* berikutnya dengan akurasi akhir mencapai sekitar 97%. Selain itu, nilai *training loss* juga terus menurun secara konsisten hingga berada di bawah 0,2. Hasil

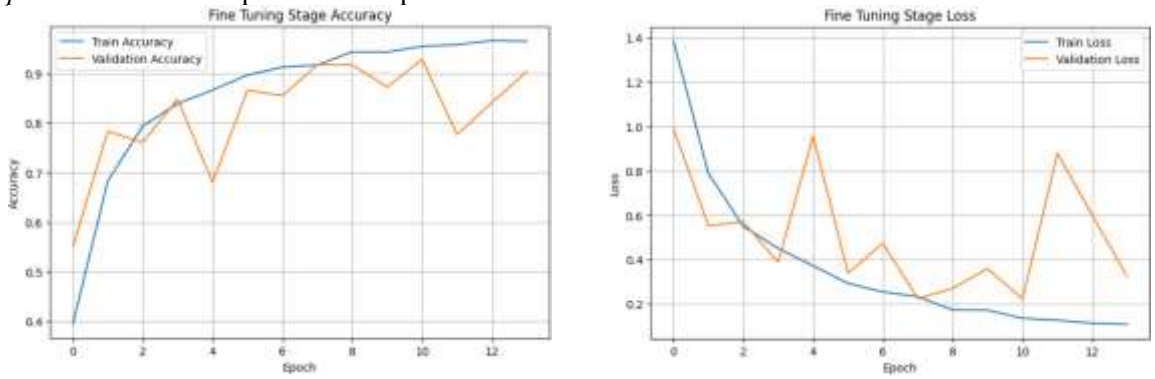
tersebut menunjukkan bahwa *optimizer* AdamW mampu memberikan performa yang cukup stabil meskipun masih terdapat sedikit fluktuasi pada data validasi selama proses pelatihan.



Gambar 4. Pelatihan Model Skenario 3

3.6 Hasil Pelatihan Model Skenario 4 Menggunakan optimizer SGD

Penggunaan *optimizer* SGD pada skenario 4 menunjukkan peningkatan akurasi data latih yang cukup stabil hingga mencapai lebih dari 95%. Namun, performa pada data validasi masih terlihat kurang stabil selama proses pelatihan. Dalam grafik Gambar 5 terlihat beberapa penurunan akurasi validasi pada *epoch* tertentu, seperti pada *epoch* ke-4 dan *epoch* ke-11, di mana akurasi turun hingga sekitar 68% dan 78%. Kondisi serupa juga terlihat pada grafik *validation loss* yang beberapa kali mengalami peningkatan meskipun *training loss* terus menurun secara konsisten hingga berada di bawah 0,2. Hal ini menunjukkan bahwa *optimizer* SGD masih belum mampu menghasilkan performa yang lebih stabil dari *optimizer* Adam maupun AdamW pada data validasi.



Gambar 5. Pelatihan Model Skenario 4

3.7 Perbandingan Hasil Pelatihan Model

Perbandingan performa setiap skenario model dilakukan berdasarkan nilai akurasi pada data *training, validation, dan testing* yang disajikan pada Tabel 5. Hasil perbandingan menunjukkan *optimizer* Adam menghasilkan performa terbaik dengan akurasi pengujian sebesar 98%, diikuti oleh AdamW 95%, SGD 91%, dan tanpa *optimizer* 65%. Keunggulan *optimizer* Adam didukung oleh kemampuannya memperbarui laju pembelajaran secara adaptif dengan memanfaatkan estimasi momen pertama (m_t) dan momen kedua (v_t) pada setiap lapisan jaringan. Perpaduan antara prinsip momentum dan RMSProp membuat Adam sangat responsif dalam mengenali variasi fitur spasial rapat dan lesi kecil pada daun bawang merah [17]. Sebaliknya, AdamW terbatas dalam mengeksplorasi ruang hipotesis karena pemisahan mekanisme *weight decay* mempersempit fleksibilitas pembaruan bobot model. Adapun SGD rentan terjebak pada minimum lokal karena tidak adanya dukungan memori momentum yang kuat.

Faktor lain yang berkontribusi pada tingginya akurasi adalah volume dataset. Dengan total 4.493 citra berhasil memenuhi kebutuhan data arsitektur MaxViT berbasis *transformer* yang tidak memiliki *inductive bias* bawaan seperti CNN. Besarnya jumlah data latih memungkinkan mekanisme *self-*

attention bekerja optimal dalam menangkap relasi konteks global antarpiksel sekaligus menekan risiko *underfitting*.

Dibandingkan dengan penelitian sebelumnya dengan akurasi 95,75% [16], hasil ini menunjukkan peningkatan performa hingga 98%. Signifikansi peningkatan sebesar 2,25% ini terlihat nyata pada pemangkasan tingkat *error rate* dari 4,25% menjadi hanya 2%. Penurunan *error rate* ini sangat krusial saat model diimplementasikan ke aplikasi mobile *ShallotEye* di lapangan guna meminimalkan risiko salah diagnosis penyakit tanaman oleh petani secara fatal

Tabel 5. Perbandingan Hasil Skenario

Skenario	Model	Hasil		
		Train	Val	Test
1	No *Optimizer*	0.6696	0.8194	0.65
2	**Adam**	**0.9856**	**0.9821**	**0.98**
3	AdamW	0.9592	0.9643	0.95
4	SGD	0.9547	0.9286	0.91

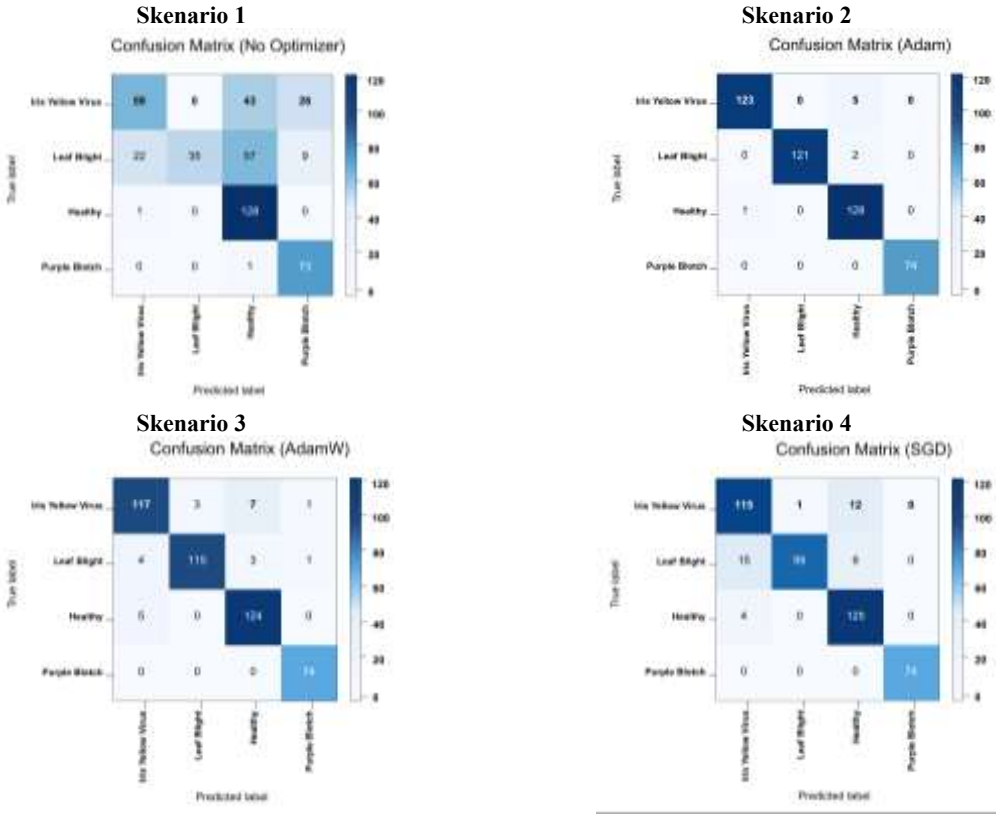
3.8 Hasil Evaluasi Pengembangan Model

Pengujian dilakukan menggunakan *test set* untuk melihat performa model dalam mengklasifikasikan penyakit secara nyata. Hasil evaluasi per kategori untuk setiap skenario disajikan secara lengkap pada Tabel 6. Tabel tersebut menunjukkan nilai metrik evaluasi seperti *precision, recall, dan F1-score* pada tiap skenario pengujian. Berdasarkan hasil tersebut, dapat dianalisis skenario yang memberikan performa klasifikasi paling optimal dan konsisten pada setiap kelas penyakit.

Tabel 6. Hasil Evaluasi Model Semua Skenario

Skenario	Kategori	Precision	Recall	F1-score	Accuracy
1	Iris Yellow Virus	72%	46%	56%	65%
	Leaf blight	100%	28%	44%	
	Healthy	56%	99%	72%	
	Purple blotch	68%	99%	80%	
2	Iris Yellow Virus	99%	96%	98%	98%
	Leaf blight	100%	98%	99%	
	Healthy	95%	99%	97%	
	Purple blotch	100%	100%	100%	
3	Iris Yellow Virus	93%	91%	92%	95%
	Leaf blight	97%	93%	95%	
	Healthy	93%	96%	94%	
	Purple blotch	97%	100%	99%	
4	Iris Yellow Virus	86%	90%	88%	91%
	Leaf blight	99%	80%	89%	
	Healthy	86%	97%	91%	
	Purple blotch	100%	100%	100%	

Untuk mengevaluasi performa klasifikasi model MaxViT yang telah dilatih, juga dilakukan pengujian menggunakan *confusion matrix*. Pengujian ini bertujuan untuk mengetahui tingkat akurasi model dalam memprediksi setiap kelas penyakit pada data uji. Hasil pengujian model dengan *optimizer* Adam disajikan pada Gambar 6.

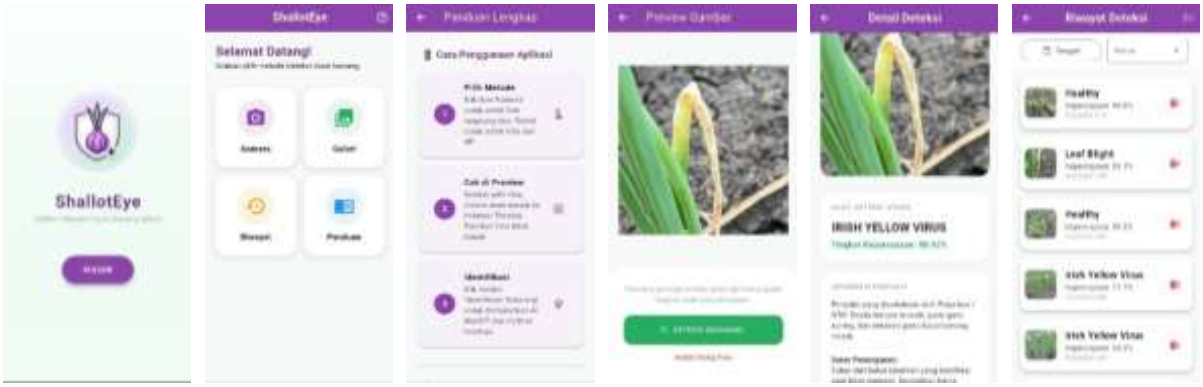


Gambar 6. Confusion Matrix Semua Skenario

Skenario 2 dengan *optimizer* Adam menunjukkan hasil klasifikasi terbaik dengan tingkat kesalahan yang relatif kecil dibandingkan skenario lainnya. Model mampu membedakan setiap kelas dengan cukup jelas. Skenario 3 dan 4 menunjukkan performa yang cukup konsisten, meskipun masih terdapat beberapa kesalahan, terutama berupa *false negative* pada kelas *Leaf Blight*. Skenario 1 menunjukkan penurunan performa yang signifikan, dengan kesalahan klasifikasi yang lebih tinggi pada kelas *Iris Yellow Spot Virus* dan *Leaf Blight*. Hal ini menunjukkan bahwa penggunaan *optimizer* berperan penting dalam membantu model mencapai konvergensi yang lebih baik serta meningkatkan akurasi secara keseluruhan.

3.9 Implementasi Aplikasi Mobile

Implementasi sistem diwujudkan dalam aplikasi *mobile* bernama *ShallotEye* untuk mendeteksi penyakit daun bawang merah secara praktis. Aplikasi ini dikembangkan dengan antarmuka sederhana yang mudah digunakan oleh pengguna. Tampilan antarmuka aplikasi ditunjukkan pada Gambar 7.



(a) (b) (c) (d) (e) (f)

Gambar 7. Tampilan Aplikasi *ShallotEye*: (a) Halaman pembuka, (b) Halaman utama, (c) Halaman panduan, (d) Halaman Preview, (e) Halaman detail deteksi, (f) Halaman riwayat

Aplikasi *ShallotEye* terdiri atas halaman pembuka, halaman utama, panduan, preview, hasil deteksi, dan riwayat. Halaman utama menyediakan fitur kamera dan galeri untuk memasukkan citra daun bawang merah yang akan dianalisis. Halaman panduan disediakan untuk membantu pengguna memahami alur penggunaan aplikasi dan teknik pengambilan gambar yang tepat.

Gambar yang dipilih atau diambil ditampilkan pada halaman *preview* untuk memastikan kesesuaiannya sebelum proses deteksi dilakukan. Hasil deteksi menampilkan jenis penyakit, tingkat kepercayaan prediksi, serta informasi singkat mengenai penyakit dan saran penanganannya. Aplikasi juga dilengkapi fitur riwayat untuk menyimpan hasil deteksi yang dapat difilter maupun dihapus sesuai kebutuhan. Secara keseluruhan, aplikasi yang dikembangkan berhasil mengintegrasikan model deteksi ke dalam sistem mobile dan dapat digunakan untuk membantu identifikasi penyakit daun bawang merah secara langsung di lapangan.

3.10 Skenario Pengujian Deteksi Sistem

Pengujian lapangan dilakukan untuk mengevaluasi performa aplikasi dalam mendeteksi penyakit daun bawang merah pada berbagai kondisi lingkungan. Berdasarkan hasil pengujian jarak pada Tabel 7, sistem memperoleh akurasi terbaik pada jarak 10 cm sebesar 100% dan sedikit menurun menjadi 99,98% pada jarak 15 cm. Akurasi kemudian turun menjadi 97,47% pada jarak 20 cm, yang menunjukkan bahwa peningkatan jarak memengaruhi kemampuan model dalam menangkap detail gejala penyakit pada daun.

Tabel 7. Skenario Pengujian Jarak

Skenario Pengujian	Kondisi Uji	Hasil Terdeteksi	Akurasi (%)	Gambar
Jarak	10 cm	Ya	100	
	15 cm	Ya	99,98	

	20 cm	Ya	97,47	

Pada pengujian pencahayaan Tabel 8, kondisi cahaya terang memberikan hasil terbaik dengan akurasi sebesar 99,72%. Sementara itu, akurasi menurun pada kondisi cahaya redup menjadi 79,01% dan kembali menurun pada cahaya buatan sebesar 68,42%. Hasil ini menunjukkan bahwa intensitas cahaya memengaruhi kualitas fitur visual yang diterima model.

Tabel 8. Skenario Pengujian Pencahayaan

Skenario Pengujian	Kondisi Uji	Hasil Terdeteksi	Akurasi (%)	Gambar
Pencahayaan	Cahaya terang alami (*Outdoor)*	Ya	99,72	
	Cahaya redup	Ya	79,01	
	Cahaya Buatan	Ya	68,42	

Pengujian skenario sudut pengambilan gambar pada Tabel 9 menunjukkan bahwa sudut 0° menghasilkan akurasi tertinggi sebesar 99,93%, sedangkan sudut 45° mengalami penurunan menjadi 90,04%. Pada sudut 90°, akurasi kembali meningkat hingga 99,11%, sehingga model masih mampu mengenali objek dengan baik meskipun terdapat perubahan orientasi gambar.

Tabel 9. Skenario Pengujian Sudut Kamera

Skenario Pengujian	Kondisi Uji	Hasil Terdeteksi	Akurasi (%)	Gambar
Sudut Kamera	0° (tegak lurus)	Ya	99,93%	

	45° (miring)	Ya	90,04	
	90° (samping)	Ya	99,11	

Pada pengujian latar belakang pada Tabel 10, sistem memperoleh akurasi hampir sempurna sebesar 99,99% pada latar polos. Akurasi menurun menjadi 81,78% pada latar alami area pertanian dan sistem tidak berhasil mendeteksi objek pada latar yang sangat kompleks. Hal ini menunjukkan bahwa keberadaan objek lain di sekitar daun dapat memengaruhi proses ekstraksi fitur dan hasil deteksi model.

Tabel 10. Skenario Pengujian Latar Belakang

Skenario Pengujian	Kondisi Uji	Hasil Terdeteksi	Akurasi (%)	Gambar
Latar Belakang	Polos	Ya	99,99	
	Alami	Ya	81,78	
	Kompleks	Tidak	-	

4. Kesimpulan

Penelitian ini berhasil mengimplementasikan model *Multi-Axis Vision Transformer* (MaxViT) dengan teknik *transfer learning* untuk mendeteksi penyakit daun bawang merah melalui aplikasi *mobile ShallotEye*. Berdasarkan hasil pengujian, *optimizer* Adam memberikan performa terbaik dibandingkan *optimizer* lainnya dengan akurasi pengujian sebesar 98%, diikuti AdamW sebesar 95%, SGD sebesar 91%, dan model tanpa *optimizer* sebesar 65%. Capaian akurasi tersebut terbukti berhasil meningkatkan dan mengungguli performa akurasi model MaxViT pada penelitian terdahulu. Evaluasi menggunakan *confusion matrix* menunjukkan bahwa model mampu mengklasifikasikan empat kelas citra, yaitu *Healthy, Purple Blotch, Leaf Blight, dan Iris Yellow Spot Virus*, dengan tingkat kesalahan yang rendah.

Pengujian sistem di lapangan juga menunjukkan performa yang optimal pada jarak 10 cm, kondisi pencahayaan terang, sudut pengambilan 0°, dan latar belakang polos dengan tingkat akurasi yang tinggi. Namun, performa sistem mengalami penurunan pada kondisi cahaya buatan, sudut pengambilan miring, dan latar belakang yang kompleks akibat adanya gangguan visual di sekitar objek daun.

Pengembangan penelitian selanjutnya dapat difokuskan pada penambahan variasi dataset agar kemampuan generalisasi model terhadap kondisi lingkungan nyata menjadi lebih baik. Integrasi teknologi *Internet of Things* (IoT) juga dapat diterapkan untuk mendukung proses pemantauan kesehatan tanaman bawang merah secara otomatis dan berkelanjutan di area pertanian.

5. Ucapan Terima Kasih

Penulis mengucapkan terimakasih kepada Bapak Taufiq Dwi Purnama selaku pemilik lahan budidaya bawang merah di Desa Rejoso, Kabupaten Nganjuk, Provinsi Jawa Timur yang telah memberikan izin tempat penelitian serta membantu proses validasi lapangan dalam pengujian aplikasi. Penulis juga menyampaikan terimakasih kepada semua pihak yang telah memberikan bantuan, bimbingan dan dukungan sehingga penelitian ini dapat diselesaikan dengan baik.

Referensi

[1] Sekretariat Jenderal Kementerian Pertanian, "Analisis Kinerja Perdagangan Komoditas Bawang Merah 2024," Jakarta, 2024. [Daring]. Tersedia pada: https://satudata.pertanian.go.id/details/publikasi/626

[2] F. A. Pratama, B. Irawan, dan A. Premana, "IDENTIFIKASI PENYAKIT TANAMAN BAWANG MERAH DENGAN METODE K- MEANS PADA PLATFORM ANDROID," *JUTECH J. Educ. Technol.*, vol. 6, hal. 265–279, 2025, doi: 10.31932/jutech.v6i1.5186.

[3] H. Hersanti, S. M. L. H. Aldriana, Y. Maharani, dan S. Hartati, "Keefektivan Campuran Kitosan Nano dan Silika Nano dalam Menekan Alternaria porri dan Penyakit Bercak Ungu pada Tanaman Bawang Merah," *Agrikultura*, vol. 36, no. 1, hal. 51–61, 2025, doi: 10.24198/agrikultura.v36i1.59886.

[4] A. Asrul, "Virulensi beberapa isolat Pantoea ananatis penyebab penyakit hawar daun bakteri (bacterial leaf blight) pada varietas bawang merah," *Agromix*, vol. 11, no. 2, hal. 136–150, 2020, doi: 10.35891/agx.v11i2.1946.

[5] I. A. R. M. Siregar dan S. H. Hidayat, "DAS-ELISA and RT-PCR Methods for Detection of Potyviruses Infecting Shallot," *J. Fitopatol. Indones.*, vol. 20, no. 5, hal. 255–262, 2025, doi: 10.14692/jfi.20.5.255-262.

[6] A. Zalvadila, Purnawansyah, L. Syafie, dan H. Darwis, "Klasifikasi Penyakit Tanaman Bawang Merah Menggunakan Metode SVM dan CNN," *J. Inform. J. Pengemb. IT*, vol. 8, no. 3, hal. 255–260, 2023, doi: 10.30591/jpit.v8i3.5341.

[7] H. Susilawati dan G. A. Nugroho, "Deteksi Baut Kereta Api Menggunakan Artificial Intelligence," *KONSTELASI Konvergensi Teknol. dan Sist. Inf.*, vol. 5, no. 1, hal. 3–8, 2025, doi: 10.24002/konstelasi.v5i1.11379.

[8] A. Sopian, D. Setiadi, A. Suryatno, dan R. Agustino, "Computer Vision : Deteksi Masker Wajah Prediksi Usia Jenis Kelamin dengan Teknik Deep Learning Menggunakan Algoritma Convolutional Neural Network (CNN)," vol. 10, no. 2, hal. 720–733, 2024, doi: 10.37012/jtik.v10i2.2395.

[9] R. A. Kumalasanti, C. Rooyen, dan M. Christy, "Deteksi Ekspresi Wajah Menggunakan Visual Geometry Group - 16 Layer Convolutional Neural Network," *KONSTELASI Konvergensi Teknol. dan Sist. Inf.*, vol. 101, no. 101, hal. 128–135, 2026, doi: 10.24002/konstelasi.v101i101.13450.

[10] Y. Xu *et al.*, "Artificial intelligence: A powerful paradigm for scientific research," *Innovation*, vol. 2, no. 4, 2021, doi: 10.1016/j.xinn.2021.100179.

[11] I. D. Mienye dan T. G. Swart, "A Comprehensive Review of Deep Learning : Architectures , Recent Advances , and Applications," *Information*, 2024, doi: 10.3390/info15120755.

[12] X. Chen *et al.*, "Transfer learning for deep neural network-based partial differential equations solving," *Adv. Aerodyn.* , vol. 3, no. 1, 2021, doi: 10.1186/s42774-021-00094-7.

[13] D. P. Pamungkas dan M. F. Amrulloh, "Analisis Hasil Klasifikasi Penyakit Daun Bawang Merah Menggunakan Cnn Arsitektur Exception," *JIPI (Jurnal Ilm. Penelit. dan Pembelajaran Inform.*, vol. 10, no. 1, hal. 359–366, 2025, doi: 10.29100/jipi.v10i1.5875.

[14] M. I. Kresnawan, D. P. Pamungkas, dan U. Mahdiyah, "Identifikasi Penyakit Tanaman Bawang Merah Menggunakan Faster R-CNN dan Arsitektur ResNet50," *Pros. SEMNAS INOTEK (Seminar Nas. Inov. Teknol.*, vol. 8, hal. 319–326, 2024, doi: 10.29407/inotek.v8i1.4947.
[15] Z. Tu *et al.*, "MaxViT: Multi-axis Vision Transformer," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2022, vol. 13684 LNCS, hal. 459–479. doi: 10.1007/978-3-031-20053-3_27.
[16] H. Rahman, H. Imran, A. Hossain, M. Siddiqul, dan A. Sakib, "Explainable vision transformers for real-time chili and onion leaf disease.pdf," hal. 1823–1833, 2025, doi: 10.30574/ijsra.2025.15.1.1163.
[17] M. Reyad, A. M. Sarhan, dan M. Arafa, "A modified Adam algorithm for deep neural network optimization," *Neural Comput. Appl.*, vol. 35, no. 23, hal. 17095–17112, 2023, doi: 10.1007/s00521-023-08568-z.
[18] N. S. Mahajaya, P. Desiana, W. Ayu, dan R. R. Huizen, "Pengaruh Optimizer Adam , AdamW , SGD , dan LAMB terhadap Model Vision Transformer pada Klasifikasi," *Pros. Semin. Has. Penelit. Inform. dan Komput. 2024*, vol. 1, no. 2, hal. 818–823, 2024, [Daring]. Tersedia pada: https://spinter.stikom-bali.ac.id/index.php/spinter/article/view/222
[19] M. P. Aishwarya dan A. P. Reddy, "Dataset of chilli and onion plant leaf images for classification and detection," *Data Br.*, vol. 54, hal. 110524, 2024, doi: 10.1016/j.dib.2024.110524.
[20] S. Sathyanarayanan dan B. R. Tantri, "Confusion Matrix-Based Performance Evaluation Metrics," *africanjournalofbiomedicalresearch*, vol. 27, no. 4, hal. 4023–4031, 2024, doi: 10.53555/AJBR.v27i4S.4345.
[21] H. I. Permana, N. A. Prasetyo, dan I. Susanto, "Perancangan Aplikasi Android ' Satria ' Pendukung Smart City di Kabupaten Banyumas dengan Metode Sekuensial Linear," *KONSTELASI Konvergensi Teknol. dan Sist. Inf.*, vol. 2, no. 1, hal. 1–12, 2022, doi: 10.24002/konstelasi.v2i1.5596.
[22] N. K. Gupta, "Impact of Flutter Technology in Software Development," *Int. J. Res. Publ. Rev.*, vol. 5, no. 7, hal. 3749–3752, 2024, doi: 10.55248/gengpi.5.0724.1928.
[23] M. Ramchandani, H. Khandare, P. Singh, dan P. Rajak, "Survey : Tensorflow in Machine Learning," *J. Phys. Conf. Ser.*, vol. 2273, no. 1, hal. 1–11, 2022, doi: 10.1088/1742-6596/2273/1/012008.
[24] T. Saito dan M. Rehmsmeier, "The Precision-Recall Plot Is More Informative than the ROC Plot When Evaluating Binary Classifiers on Imbalanced Datasets," *PLoS One*, hal. 1–21, 2015, doi: 10.1371/journal.pone.0118432.
[25] S. Bej, N. Davtyan, M. Wolfien, dan M. Nassar, "LoRAS : an oversampling approach for imbalanced datasets," *Mach. Learn.*, vol. 110, no. 2, hal. 279–301, 2021, doi: 10.1007/s10994-020-05913-4.
[26] F. Prananta, S. Riyadi, dan K. Khotimah, "DETEKSI MULTI PENYAKIT PADI BERBASIS CNN MOBILENETV2 PADA," in *Prosiding Seminar Nasional CIASTECH*, 2025, vol. 8, no. 1, hal. 428–437. doi: 10.31328/ciastech.v8i1.7501.