Progresif: Jurnal Ilmiah Komputer
Jl. Ahmad Yani, K.M. 33,5 - Kampus STMIK Banjarbaru
Loktabat – Banjarbaru (Tlp. 0511 4782881), e-mail: puslit.stmikbjb@gmail.com
e-ISSN: 2685-0877
p-ISSN: 0216-3284

Implementasi *Transfer Learning Model InceptionV3* Untuk Deteksi Penyakit Daun Jagung Berbasis *Mobile*

DOI: http://dx.doi.org/10.35889/progresif.v22i3.3842

Dion Danianto[1*], Salamun Rohman Nudin[2]
Manajemen Informatika, Universitas Negeri Surabaya, Surabaya, Indonesia
*e-mail *Corresponding Author:* dion.22058@mhs.unesa.ac.id

Abstract

Maize represents a highly crucial agricultural staple within the Indonesian nation, where productivity is often affected by leaf diseases. Diseases like blight, common rust, and gray leaf spot prove hard to recognize by hand since the task demands time and risks personnel mistakes. Current research constructs a maize leaf disease classification model applying transfer learning based on InceptionV3 and evaluates the capabilities of three optimizing algorithms, specifically Adam, Stochastic Gradient Descent (SGD), and RMSProp. The dataset consists of 8,040 images collected from Kaggle and Mendeley, separated into four categories: blight, common rust, gray leaf spot, and healthy. Model training was executed using three dataset scenarios to assess the generalization ability of this suggested method. The empirical findings indicate that the Adam optimizer implemented on the merged dataset attained the highest effectiveness, reaching an exactness of 97.26%, as well as the highest precision, recall, and F1-score compared to SGD and RMSProp. The best-performing model was then converted to TensorFlow Lite and launched within a Flutter-driven Android software to help individuals with the initial spotting of maize leaf diseases.

Keywords: *Maize; Leaf Disease; Convolutional Neural Network (CNN); InceptionV3*

Abstrak

Jagung adalah salah satu komoditas pangan penting di Indonesia yang kerap mengalami kendala produksi akibat serangan penyakit pada daunnya. Penyakit seperti hawar, karat, dan bercak daun abu-abu sukar dikenali secara manual karena memerlukan waktu yang lama dan berisiko mengalami kesalahan. Studi ini mengembangkan model klasifikasi penyakit daun jagung menggunakan *transfer learning* berbasis InceptionV3 dengan membandingkan kinerja tiga algoritma optimasi, yaitu Adam, SGD, dan RMSProp. Dataset yang digunakan terdiri dari 8.040 citra dari Kaggle dan Mendeley yang terbagi ke dalam empat kelas, yaitu *blight*, *common rust*, *gray leaf spot*, dan *healthy*. Pelatihan dilakukan pada tiga skenario dataset untuk mengevaluasi kemampuan generalisasi model. Temuan memperlihatkan bahwa algoritma Adam pada data kombinasi memberikan performa tertinggi dengan akurasi 97,26% serta nilai *precision*, *recall*, dan F1-*score* tertinggi dibandingkan SGD dan RMSProp. Model terbaik dikonversi ke TensorFlow Lite dan diimplementasikan ke dalam aplikasi Android berbasis Flutter sebagai alat bantu mengenali gejala awal penyakit tanaman jagung.

Kata kunci: *Jagung; Penyakit Daun; CNN; InceptionV3*

1. Pendahuluan

Tanaman jagung merupakan salah satu tanaman sumber karbohidrat penting di dunia, menempati urutan ketiga setelah gandum dan padi, sedangkan di Indonesia berada pada urutan kedua setelah padi. Selain sebagai makanan pokok, jagung juga berperan strategis sebagai pakan ternak dan bahan baku industri, dengan lebih dari 55% alokasi untuk pakan ternak, 33% untuk konsumsi pangan, dan sisanya untuk industri, sehingga kebutuhannya terus meningkat [1]. Salah satu penyebab terbatasnya produksi jagung adalah gangguan organisme penyebab penyakit tanaman, yang berdampak pada turunnya hasil panen bahkan kegagalan panen [2].

Dengan demikian, gangguan produktivitas jagung akibat penyakit tidak hanya merugikan petani secara langsung, tetapi juga berdampak luas terhadap rantai penyediaan pangan dan pakan nasional.

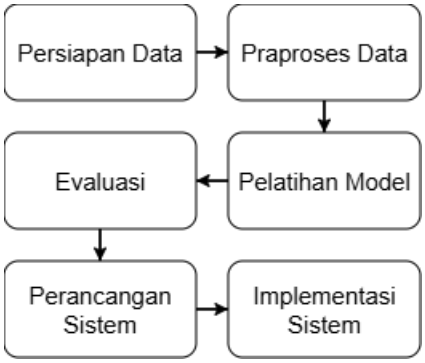
Gangguan pada vegetasi jagung umumnya hanya dikenali oleh petani yang rutin merawat tanaman, sehingga sulit dilacak dan ditangani jika tidak terdeteksi sejak awal [3]. Prosedur pengenalan penyakit saat ini dominan dilakukan melalui observasi lokasi secara manual, yang menghabiskan waktu lama, membutuhkan pakar berpengalaman, serta memiliki tingkat kekeliruan tinggi terutama pada area perkebunan luas dan pekerja yang kurang teredukasi. Akibatnya, pendekatan konvensional kurang praktis dan menuntut biaya tinggi sehingga sukar dijangkau petani mandiri, padahal deteksi dini sangat vital untuk mencegah kerusakan yang lebih parah dan penyebaran wabah ke area lebih luas [4][5]. Karena itu, diperlukan perangkat deteksi dini yang efisien dan tepat guna.

Berbagai studi telah dilakukan untuk menjawab permasalahan tersebut. Shandilya dkk. mengusulkan model hibrida CNN-ViT yang memadukan ekstraksi fitur lokal (CNN) dan informasi kontekstual global (Vision Transformer), mencapai akurasi 99,15% [6]. Aryanti dkk. menerapkan CNN untuk empat kondisi daun jagung menggunakan 4.188 citra, mencapai akurasi 95% dengan waktu inferensi 0,48 detik per gambar, meski diikuti kebutuhan komputasi yang lebih besar [7]. Untuk mengatasi hal ini, sejumlah penelitian mengusulkan *transfer learning* karena mampu meningkatkan akurasi sekaligus mempercepat pelatihan. Tirta dkk. menerapkan *transfer learning* DenseNet-201 dan meraih akurasi 96,65% dengan generalisasi baik pada empat kelas penyakit [8], sementara Sodikin dkk. menggunakan *transfer learning* MobileNetV2 pada 7.614 citra dengan rasio 80:20, memperoleh akurasi validasi 85% [9]. Herbert dkk. secara khusus menerapkan InceptionV3 untuk klasifikasi penyakit daun stroberi dan mencapai akurasi 99%, membuktikan arsitektur ini unggul untuk citra penyakit tanaman [10]. Sejalan dengan itu, penelitian mulai mengarah ke implementasi lapangan: Askale dkk. mengembangkan aplikasi *mobile* berbasis CNN untuk deteksi *real-time*, dengan VGG16 mencapai akurasi tertinggi 95% dan diimplementasikan ke Android [11], sedangkan Salsabilah dkk. mengembangkan sistem berbasis CNN dan *transfer learning* pada aplikasi GUI desktop dengan empat kelas yang sama seperti penelitian ini, mencapai akurasi 90,00%, namun masih terbatas pada perangkat desktop dan belum membahas optimasi untuk perangkat sumber daya terbatas seperti *smartphone* [12]. Berdasarkan tinjauan tersebut, mayoritas penelitian masih terbatas pada peningkatan akurasi model, sementara penelitian yang mengarah ke implementasi *mobile* belum mengevaluasi secara sistematis pengaruh algoritma optimasi terhadap performa klasifikasi maupun kesiapan model untuk perangkat terbatas. Celah inilah yang menjadi dasar penelitian ini.

Deep learning, sebagai cabang *machine learning* yang memanfaatkan lapisan pemrosesan nonlinier berjenjang untuk ekstraksi fitur, unggul dalam menangani data berskala besar dan berdimensi tinggi [13][14], khususnya *Convolutional Neural Network* yang meniru sistem visual manusia dan terbukti efektif mengenali fitur tekstur citra daun untuk klasifikasi penyakit tanaman berbasis Android [15]. Untuk mengatasi celah di atas, penelitian ini mengusulkan transfer learning dengan arsitektur InceptionV3, pengembangan dari GoogLeNet dan InceptionV2 dengan peningkatan efisiensi konvolusi [16], yang mereduksi kebutuhan data latih, waktu pelatihan, dan biaya komputasi [17], sekaligus menjawab kendala komputasi tinggi pada penelitian sebelumnya [7]. Kemampuan ekstraksi fitur multi-skala melalui *Inception Block* juga menjadikan arsitektur ini sesuai untuk klasifikasi citra berbasis tekstur dan warna seperti pada kasus ini [18][19]. Berdasarkan celah tersebut, penelitian ini bertujuan mengimplementasikan model deteksi penyakit daun jagung menggunakan *transfer learning* InceptionV3 ke dalam aplikasi Android berbasis Flutter, sekaligus membandingkan kinerja tiga algoritma optimasi (Adam, SGD, dan RMSProp) untuk mengevaluasi pengaruhnya terhadap akurasi klasifikasi. Perbandingan ketiga optimizer ini menjadi salah satu aspek kebaruan penelitian, karena memberikan analisis lebih komprehensif dalam menentukan metode optimasi paling efektif bagi performa InceptionV3 sekaligus implementasinya pada perangkat *mobile*.

2. Metodologi

Metode penelitian berfungsi sebagai landasan untuk menjalankan penelitian sehingga semua langkah bisa dijalankan dengan teratur serta terarah. Dalam penelitian ini, alur penelitian terdiri dari enam tahap utama, yaitu persiapan data, praproses data, pelatihan model, evaluasi, perancangan sistem, dan implementasi sistem yang ditunjukkan pada Gambar 1.



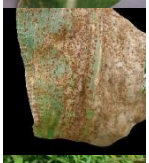
Gambar 1. Alur Penelitian

3.1 Persiapan Data

Dataset merupakan sekumpulan item data yang mencerminkan hasil dari aktivitas seperti pengukuran, pelaporan, pengumpulan, analisis, atau pengamatan. Dataset harus mencakup berbagai skenario, variasi, dan kompleksitas agar model yang dihasilkan mampu melakukan generalisasi yang baik [20]. Dalam penelitian ini dataset yang digunakan adalah data sekunder yang diperoleh dari Kaggle dan Mendeley yang terdiri dari 8.040 gambar. Dataset ini juga telah dianotasi untuk mengidentifikasi jenis penyakit pada daun jagung. Anotasi tersebut berfungsi sebagai acuan dalam proses klasifikasi penyakit jagung pada tahap pelatihan model.

Tabel 1 menunjukkan rincian dataset citra daun jagung yang digunakan dalam penelitian ini. Dataset terdiri atas empat kelas, yaitu *Blight*, *Common Rust*, *Gray Leaf Spot*, dan *Healthy*. Data diperoleh dari dua sumber, yaitu Kaggle sebanyak 4.188 citra dan Mendeley sebanyak 3.852 citra. Setelah digabungkan, total dataset yang digunakan berjumlah 8.040 citra, yang terdiri atas 2.131 citra *Blight*, 2.498 citra *Common Rust*, 1.087 citra *Gray Leaf Spot*, dan 2.324 citra *Healthy*.

Tabel 1. Label Dan Jumlah Dataset

Gambar	Kelas	Jumlah Data	
		Kaggle	Mendeley
	Blight	1146	985
	Common Rust	1306	1192
	Gray Leaf Spot	574	513
	Healthy	1162	1162
Jumlah Total		8040	

3.2 Praproses Data

Tahap praproses data dilakukan untuk mempersiapkan dataset sebelum digunakan dalam proses pelatihan model. Pada tahap ini, data diolah dan disesuaikan agar memiliki kualitas yang baik serta selaras dengan kebutuhan model klasifikasi yang dikembangkan. Pada proses

pertama dilakukan proses *resizing* data menjadi 299x299 piksel agar sesuai dengan ukuran input standar arsitektur InceptionV3 yang digunakan pada model *pre-trained* berbasis ImageNet [21].

Tabel 2. Parameter Augmentasi

Parameter	Nilai
Rescale	1./255
Horizontal_flip	True
Vertical_flip	True
Rotation_range	Hingga 40°
Zoom_range	0.2
Brightness_range	[0.8, 1.2]
Fill_Mode	Nearest

Kedua, dilakukan augmentasi data untuk meningkatkan variasi citra dan mendukung model mengenali objek pada berbagai kondisi. Metode augmentasi yang dipakai mencakup rotasi, flipping, zooming, serta pengaturan brightness. Tabel 2 menunjukkan rincian parameter augmentasi yang diterapkan pada tahap praproses data. Penerapan parameter augmentasi tersebut bertujuan untuk meningkatkan variasi data latih sehingga model mampu mengenali citra pada berbagai kondisi dan memperoleh kemampuan generalisasi yang lebih baik.

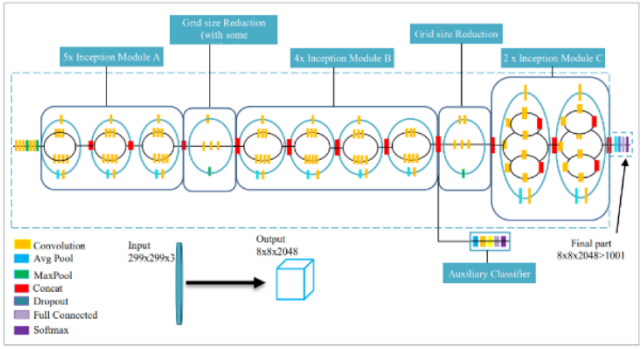
Tabel 3. Jumlah data setelah proses splitting

Dataset	Jumlah Data		
	Train	Validation	Testing
Kaggle	3350	419	419
Mendeley	3081	385	386
Kaggle dan Mendeley	6432	804	804

Pada tahap ketiga, dataset dipecah menjadi tiga porsi, yakni data latih, data validasi, serta data uji dengan rasio masing-masing sejumlah 80%, 10%, dan 10%. Pemisahan dataset tersebut ditujukan guna melatih model, memaksimalkan alur pembelajaran, serta mengevaluasi sejauh mana performa generalisasi model pada data yang tidak pernah dipakai saat tahapan pembelajaran.

3.3 Pelatihan Model

Arsitektur yang diaplikasikan pada studi ini adalah InceptionV3 mengguanakan metode *transfer learning*. InceptionV3 merupakan pengembangan lanjutan dari arsitektur GoogLeNet dan InceptionV2 yang dikembangkan oleh Christian Szegedy pada tahun 2015 [16]. Pendekatan *transfer learning* digunakan pada penelitian ini karena meningkatkan efisiensi model *deep learning* dengan mereduksi ukuran data latih, waktu pelatihan, serta biaya komputasi pada algoritma CNN [17].



Gambar 2. Arsitektur InceptionV3 [18]

Gambar 2 adalah arsitektur dari InceptionV3 yang digunakan pada penelitian ini. Arsitektur InceptionV3 merupakan pengembangan dari versi sebelumnya yang memiliki berbagai peningkatan, seperti penggunaan konvolusi 7×7, teknik label *smoothing*, serta penambahan *auxiliary classifier* untuk membantu proses distribusi label pada lapisan jaringan yang lebih dalam. Arsitektur ini memiliki modul utama yang disebut *Inception Block* yang berfungsi untuk melakukan ekstraksi fitur dari citra. Selain itu, model InceptionV3 menggunakan *Artificial Neural Network* (ANN) dengan lapisan *fully connected*, sedangkan fungsi *softmax* digunakan pada tahap klasifikasi akhir untuk menentukan kelas hasil prediksi [19].

Tabel 4. Parameter Arsitekur InceptionV3

Layer (Tipe)	Output Shape	Parameter
inceptionv3 (Conv2D)	(None, 149, 149, 32)	864
global_average_pooling2d_1 (GlobalAveragePooling2D)	(None, 2048)	0
batch_normalization_189 (BatchNormalization)	(None, 2048)	8192
fc1_256 (Dense)	(None, 256)	524544
dropout_1(Dropout)	(None, 256)	0
fc2_128 (Dense)	(None, 128)	32896
output (Dense)	(None, 4)	516
Total params		22,368,932
Trainable params		562,052
Non-trainable params		21,806,880

Berdasarkan ringkasan pada Tabel 4, model yang digunakan merupakan modifikasi dari arsitektur InceptionV3 untuk klasifikasi penyakit daun jagung dengan empat kelas. Lapisan awal berfungsi sebagai *feature extractor*, kemudian dilanjutkan dengan *Global Average Pooling 2D* untuk mereduksi dimensi tanpa menambah parameter. Selanjutnya, digunakan *Batch Normalization* untuk menstabilkan pelatihan, diikuti *Dense layer* 256 *neuron* dan *Dropout* untuk mengurangi *overfitting*. Proses pembelajaran dilanjutkan dengan *Dense layer* 128 *neuron*, sebelum akhirnya masuk ke lapisan output dengan 4 *neuron* beraktivasi *softmax* untuk menghasilkan output klasifikasi sesuai jumlah kelas. Secara keseluruhan, model memiliki sekitar 22 juta parameter, dengan sebagian besar merupakan parameter *non-trainable* dari *backbone* yang dibekukan dalam proses *transfer learning.*

Tabel 5. Konfigurasi pelatihan

Parameter	Pengaturan/Nilai
Input Size	299x299 pixel (RGB)
Batch Size	32
Optimizer	SGD, RMSProp, Adam
Learning Rate	Default Optimizer
Epochs	50
Patience (Early Stopping)	10 Epochs
ReduceLROnPlateau	True
ModelCheckpoint	True

Proses pelatihan model dilakukan selama 50 *epochs* dengan menggunakan *learning rate* bawaan dari masing-masing optimizer. Selain itu, diterapkan *callback EarlyStopping* untuk guna mengakhiri sesi pembelajaran dengan sendirinya apabila kinerja model berhenti meningkat, *ReduceLROnPlateau* digunakan untuk menyesuaikan *learning rate* secara adaptif guna meningkatkan stabilitas dan performa pelatihan model, serta *ModelCheckpoint* digunakan untuk menyimpan hasil terbaik model selama pelatihan. Setelah proses konfigurasi model selesai, tahap selanjutnya adalah melakukan kompilasi dan pelatihan model berdasarkan parameter yang telah ditentukan. Pada penelitian ini digunakan empat *optimizer*, yakni Adam, SGD, RMSProp, serta *baseline optimizer* sebagai pembanding performa model. Fungsi *loss* yang diaplikasikan

adalah *categorical crossentropy* karena tepat bagi proses kategorisasi multikelas pada citra penyakit daun jagung. Sementara itu, metrik *accuracy* digunakan untuk mengevaluasi kemampuan model dalam mengidentifikasi jenis penyakit secara tepat. Proses pelatihan dilakukan menggunakan tiga skenario dataset, yaitu dataset Kaggle, dataset Mendeley, serta dataset gabungan Kaggle dan Mendeley untuk membandingkan performa model pada setiap variasi data yang digunakan.

3.4 Evaluasi

Pada penelitian ini, evaluasi model dilakukan menggunakan *confusion matrix* dan *classification report*. *Confusion matrix* digunakan untuk menghitung nilai akurasi, presisi, *recall*, dan *f1-score* pada setiap metode pengujian. Metode ini merupakan alat evaluasi pada pembelajaran terawasi (*supervised learning*) yang dipakai untuk menilai kemampuan model dalam mengklasifikasikan data, termasuk pada kondisi kelas yang tidak seimbang [22][23]. Rumus perhitungan akurasi, presisi, *recall*, dan *f1-score* ditunjukkan pada Persamaan (1), (2), (3), dan (4).

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

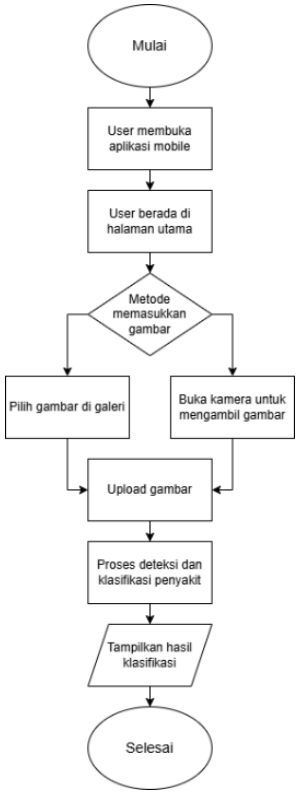
$$Precision = \frac{TP}{TP+FP} \quad (2)$$

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

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

3.5 Perancangan Sistem

Tahap perancangan sistem dilakukan dengan menyusun *flowchart* yang berfungsi untuk memvisualisasikan urutan proses dan interaksi yang terjadi dalam sistem secara sistematis.



Gambar 3. Alur Kerja Sistem

Flowchart pada Gambar 3 menggambarkan tahapan yang dilalui pengguna saat menggunakan aplikasi hingga memperoleh hasil deteksi penyakit pada daun jagung. Alur kerja ini dirancang agar sederhana, efisien, dan mudah digunakan oleh pengguna.

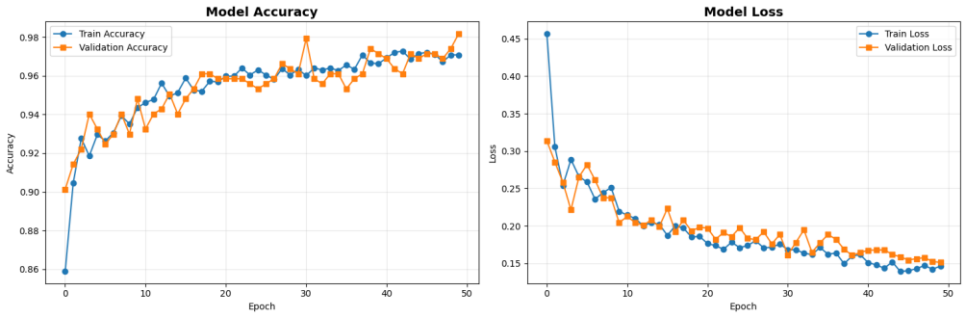
3.6 Implementasi Sistem

Setelah proses evaluasi model selesai, model terbaik diimplementasikan ke sistem aplikasi guna dimanfaatkan secara langsung oleh pengguna. Sistem dibangun menggunakan *framework Flutter*. Flutter adalah *framework cross-platform* buatan Google yang memfasilitasi pengembangan aplikasi *mobile* berkinerja tinggi untuk platform Android dan iOS dalam satu basis kode yang sama [24]. Pada tahap ini, pustaka *TensorFlow* digunakan untuk mengkonversi model yang dihasilkan dalam format .h5 menjadi format .tflite milik *TensorFlow* [25]. Setelah proses konversi selesai, integrasi model dan pengembangan antarmuka dilakukan dengan menggunakan Android Studio dengan *framework* Flutter hingga terbentuk aplikasi yang dapat mendeteksi penyakit daun jagung.

4. Hasil dan Pembahasan

4.1 Hasil Pelatihan Model

Model dilatih menggunakan tiga skenario dataset, yaitu Kaggle, Mendeley, serta gabungan Kaggle dan Mendeley dengan empat variasi optimizer, yaitu Baseline, SGD, RMSProp, dan Adam. Seluruh proses pelatihan dilakukan menggunakan data latih sebesar 80%, sedangkan evaluasi selama proses pembelajaran menggunakan data validasi sebesar 10%.



Gambar 4. Grafik Akurasi dan Loss dari Pelatihan Model

Grafik pada Gambar 4 merupakan salah satu grafik pelatihan dengan menggunakan Dataset Mendeley *optimizer* Adam, grafik menunjukkkan peningkatan performa model dalam pelatihan. Grafik akurasi memberikan hasil yang terbaik dengan validasi akurasi sebesar 98.18% pada *epoch* ke-50 dan cenderung stabil hingga akhir pelatihan. Hal ini didukung oleh grafik *loss* yang menunjukkan penurunan secara konsisten, menunjukkan bahwa model dapat belajar dengan optimal dan memiliki tingkat kesalahan yang sangat kecil. Selain itu, tidak ditemukan perbedaan yang signifikan antara kurva akurasi pelatihan dan validasi, serta kurva *loss* pelatihan dan validasi yang menunjukkan tren penurunan yang serupa. Keadaan tersebut menandakan bahwa model menunjukkan kemampuan generalisasi yang mumpuni dan tidak memunculkan gejala *overfitting* yang mencolok selama proses pelatihan.

Tabel 6. Hasil Akurasi Train dan Validation Model

Dataset	Optimizer	Train		Validation	
		Accuracy	Loss	Accuracy	Loss
Kaggle	Baseline Optimizer	94.69%	22.17%	92.60%	26.39%
	SGD	93.79%	21.27%	92.60%	23.40%
	RMSProp	95.25%	20.07%	92.60%	24.77%
	Adam	94.72%	22.62%	93.32%	25.13%
Mendeley	Baseline Optimizer	97.18%	13.60%	96.36%	15.85%
	SGD	94.94%	18.99%	94.29%	19.74%
	RMSProp	96.69%	15.30%	96.62%	20.39%

Dataset	Optimizer	Train		Validation	
		Accuracy	Loss	Accuracy	Loss
	Adam	97.08%	14.62%	98.18%	15.13%
Kaggle + Mendeley	Baseline Optimizer	94.99%	20.72%	95.02%	20.44%
	SGD	94.93%	18.62%	95.02%	18.17%
	RMSProp	96.69%	13.44%	96.62%	13.57%
	Adam	96.89%	16.00%	96.52%	15.88%

Tabel 6 menunjukkan hasil akurasi pelatihan dan validasi model pada tiga variasi dataset dengan empat *optimizer*. Pada dataset Kaggle, Adam menghasilkan *validation accuracy* tertinggi sebesar 93,32%, sedangkan SGD memperoleh *validation loss* terendah sebesar 23,40%. Pada dataset Mendeley, Adam memberikan performa terbaik dengan *validation accuracy* 98,18% dan *validation loss* 15,13%. Sementara itu, pada dataset gabungan Kaggle dan Mendeley, Adam kembali menghasilkan *validation accuracy* tertinggi sebesar 96,52% dengan *validation loss* 15,88%, menunjukkan performa validasi yang lebih baik dibandingkan *optimizer* lainnya.

4.2 Hasil Pengujian dan Evaluasi Model

Tahap pengujian model dilakukan setelah tahap pelatihan model selesai untuk mengukur performa dari model yang telah dilatih menggunakan data yang belum pernah dipelajari pada data training. Pada tahap ini dilakukan pengujian performa model menggunakan data uji (t*est*) sebesar 10%.

Tabel 7. Hasil Akurasi dan Loss pada Data Uji

Dataset	Optimizer	Test	
		Accuracy	Loss
Kaggle	Baseline Optimizer	93.56%	27.87%
	SGD	92.84%	23.52%
	RMSProp	92.84%	27.90%
	Adam	92.84%	26.90%
Mendeley	Baseline Optimizer	96.37%	16.53%
	SGD	94.56%	19.67%
	RMSProp	95.60%	17.53%
	Adam	96.37%	15.88%
Kaggle + Mendeley	Baseline Optimizer	95.40%	18.97%
	SGD	95.40%	16.60%
	RMSProp	95.60%	14.44%
	Adam	97.26%	15.85%

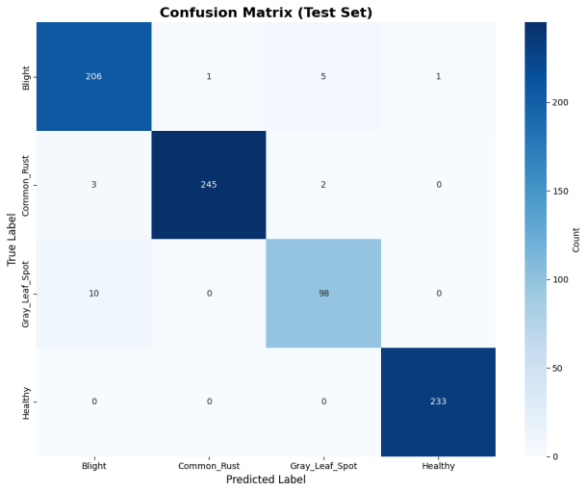
Tabel 7 menunjukkan hasil pengujian model menggunakan data uji sebesar 10% pada tiga variasi dataset dengan empat optimizer. Pada dataset Kaggle, Baseline Optimizer memperoleh akurasi tertinggi sebesar 93,56%, sedangkan SGD menghasilkan loss terendah sebesar 23,52%. Pada dataset Mendeley, Baseline Optimizer dan Adam sama-sama mencapai akurasi 96,37%, namun Adam memberikan loss yang lebih rendah, yaitu 15,88%. Sementara itu, pada dataset gabungan Kaggle dan Mendeley, Adam menghasilkan performa terbaik dengan akurasi 97,26% dan loss 15,85%. Hasil tersebut menunjukkan bahwa penggunaan dataset yang lebih beragam, dikombinasikan dengan optimizer Adam, mampu meningkatkan kemampuan model dalam menggeneralisasi data yang belum pernah digunakan selama proses pelatihan.

Tabel 8. Perbandingan Hasil Klasifikasi dengan Dataset Gabungan Kaggle dan Mendeley

Optimizer	Kelas	Precision	Recall	F1-Score
Baseline Optimizer	Blight	0.88	0.96	0.92
	Common_Rust	1.00	0.98	0.99
	Gray_Leaf_Spot	0.91	0.79	0.85

Optimizer	Kelas	Precision	Recall	F1-Score
	Healthy	1.00	1.00	1.00
SGD	Blight	0.90	0.94	0.92
	Common_Rust	0.98	0.99	0.98
	Gray_Leaf_Spot	0.91	0.80	0.85
	Healthy	1.00	1.00	1.00
RMSProp	Blight	0.92	0.96	0.94
	Common_Rust	0.99	0.98	0.98
	Gray_Leaf_Spot	0.95	0.86	0.90
	Healthy	0.99	1.00	1.00
Adam	Blight	0.94	0.97	0.95
	Common_Rust	1.00	0.98	0.99
	Gray_Leaf_Spot	0.93	0.91	0.92
	Healthy	1.00	1.00	1.00

Pengujian menggunakan dataset gabungan Kaggle dan Mendeley dengan *optimizer* Adam dipilih sebagai hasil terbaik karena menghasilkan akurasi tertinggi sebesar 97,26%. *Classification report* pada Tabel 8 menunjukkan bahwa model memiliki performa tinggi pada kelas *Healthy* dan *Common Rust*, dengan nilai *precision*, *recall*, dan *F1-score* yang mendekati 1,00. Kelas *Gray Leaf Spot* menunjukkan performa terendah dibandingkan kelas lainnya, meskipun hasil klasifikasinya tetap berada pada kategori baik. Penggunaan *optimizer* Adam meningkatkan performa klasifikasi pada kelas *Gray Leaf Spot* dibandingkan *baseline optimizer* maupun SGD, serta memberikan hasil yang paling stabil pada seluruh kelas. Hasil pengujian pada dataset gabungan juga menunjukkan performa yang lebih baik dibandingkan penggunaan dataset Kaggle maupun Mendeley secara terpisah.

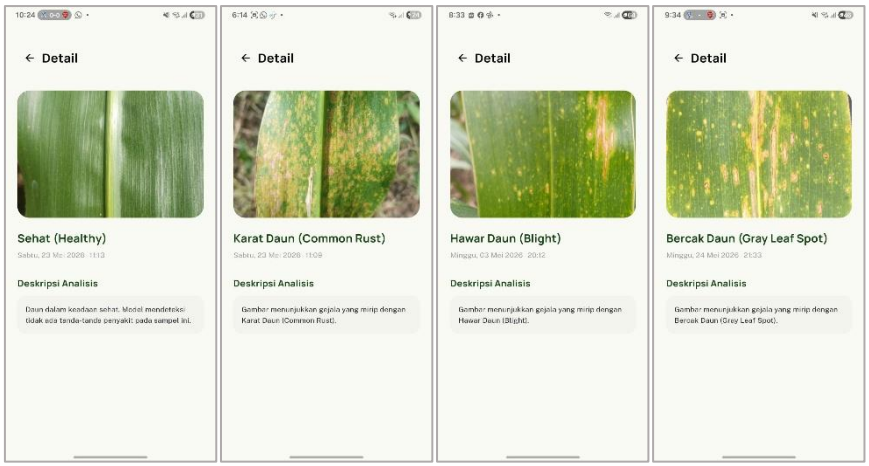


Gambar 5. Visualisasi Confusion Matrix

Visualisasi confusion matrix pada Gambar 5 menunjukkan performa klasifikasi terbaik yang diperoleh menggunakan data uji dengan dataset gabungan dengan *optimizer* Adam. Model berhasil mengklasifikasikan kelas *Healthy* secara sempurna dengan 233 citra terklasifikasi benar tanpa kesalahan, diikuti kelas *Common Rust* sebanyak 245 citra benar dari 250 citra, dan kelas *Blight* sebanyak 206 citra benar dari 213 citra. Kekeliruan pengkategorian tertinggi muncul pada kelas *Gray Leaf Spot* yakni 10 citra salah diprediksi sebagai *Blight*, ini kemungkinan disebabkan oleh kemiripan karakteristik visual antara kedua kelas tersebut. Hasil ini membuktikan bahwa model mempunyai kemampuan klasifikasi secara konsisten serta bisa mengurangi kesalahan prediksi pada keseluruhan, meskipun peluang peningkatan akurasi masih terbuka terutama pada kelas-kelas dengan kemiripan visual yang tinggi.

4.3 Hasil Implementasi Model

Berdasarkan hasil pengujian pada data uji, model InceptionV3 dengan skenario dataset gabungan Kaggle dan Mendeley serta *optimizer* Adam dipilih sebagai model terbaik. Model tersebut kemudian dikonversi ke dalam format *TensorFlow Lite* dan diintegrasikan ke dalam aplikasi Android yang dikembangkan menggunakan *Flutter* untuk memungkinkan proses klasifikasi penyakit daun jagung dilakukan secara langsung melalui perangkat *mobile*.



Gambar 6. Hasil Deteksi pada Data Lapangan

Gambar 6 menunjukkan hasil implementasi model klasifikasi penyakit daun jagung pada aplikasi Android menggunakan citra dari lingkungan nyata (*real-world environment*). Citra yang digunakan merupakan data baru yang tidak pernah dilibatkan pada proses pelatihan maupun pengujian model sebelumnya, sehingga digunakan untuk mengevaluasi kemampuan model dalam melakukan prediksi pada kondisi nyata. Aplikasi berhasil mengidentifikasi keempat kelas penyakit, yaitu *Healthy*, *Common Rust*, *Blight*, dan *Gray Leaf Spot*, serta menampilkan hasil klasifikasi dan informasi pendukung mengenai kondisi daun. Hasil pengujian menunjukkan bahwa model mampu memberikan prediksi yang sesuai pada seluruh sampel yang diuji, meskipun citra berasal dari lingkungan yang berbeda dengan data pelatihan. Temuan ini menunjukkan bahwa model memiliki kemampuan generalisasi yang baik dan berpotensi diterapkan sebagai alat bantu identifikasi penyakit daun jagung secara cepat melalui perangkat *mobile*.

4.4 Pembahasan

Penelitian ini berhasil mengembangkan dan mengimplementasikan model klasifikasi penyakit daun jagung berbasis *transfer learning* menggunakan arsitektur InceptionV3 ke dalam aplikasi *mobile*. Berdasarkan hasil pengujian pada data uji, model yang menggunakan dataset gabungan Kaggle dan Mendeley dengan *optimizer* Adam menghasilkan performa terbaik dengan akurasi sebesar 97,26% dan *loss* sebesar 15,85%. Hasil tersebut menunjukkan bahwa kombinasi arsitektur InceptionV3 dengan konfigurasi *optimizer* yang sesuai mampu menghasilkan performa klasifikasi yang tinggi dan dapat diimplementasikan pada perangkat *mobile*. Dengan demikian, tujuan penelitian untuk membandingkan performa beberapa *optimizer* pada model InceptionV3 serta mengimplementasikan model terbaik ke dalam aplikasi *mobile* telah berhasil dicapai.

Keunggulan *optimizer* Adam dibandingkan *Stochastic Gradient Descent* (SGD) dan RMSProp dapat dijelaskan melalui mekanisme kerjanya yang mengombinasikan *momentum* dan *adaptive learning rate*, sehingga proses pembaruan bobot berlangsung secara adaptif dan stabil selama pelatihan [26]. Karakteristik tersebut memungkinkan model mencapai proses konvergensi yang lebih baik pada proses *fine-tuning* InceptionV3 dibandingkan *optimizer* lainnya. Temuan lain adalah Adam tidak selalu memberikan performa terbaik pada setiap skenario dataset. Pada dataset Kaggle, *Baseline Optimizer* menghasilkan akurasi yang sedikit lebih tinggi dibandingkan Adam, sedangkan pada dataset gabungan Adam menunjukkan peningkatan performa yang paling konsisten. Temuan ini menunjukkan bahwa *optimizer* adaptif seperti Adam cenderung memberikan keuntungan yang lebih besar ketika model dilatih menggunakan dataset dengan variasi karakteristik citra yang lebih beragam karena pembaruan bobot dilakukan secara adaptif terhadap setiap parameter.

Penggunaan dataset gabungan Kaggle dan Mendeley juga berkontribusi terhadap peningkatan performa model. Variasi data yang lebih beragam memungkinkan model mempelajari representasi fitur penyakit daun jagung secara lebih komprehensif, sehingga kemampuan generalisasi terhadap data yang belum pernah dilihat sebelumnya meningkat. Hal ini ditunjukkan dari peningkatan akurasi pengujian dibandingkan penggunaan dataset tunggal. Dengan bertambahnya variasi kondisi pencahayaan, orientasi daun, serta karakteristik visual penyakit, model mampu membangun representasi fitur yang lebih baik sehingga menghasilkan prediksi yang lebih konsisten pada data uji.

Di sisi lain, hasil *confusion matrix* menunjukkan bahwa kesalahan klasifikasi paling banyak terjadi pada kelas *Gray Leaf Spot*, di mana sebanyak 10 dari 108 citra diprediksi sebagai *Blight*. Kondisi tersebut dapat dijelaskan oleh tingginya kemiripan karakteristik visual antara kedua penyakit, terutama pada fase awal infeksi yang sama-sama menampilkan lesi memanjang berwarna kecokelatan dengan pola penyebaran yang serupa. Kemiripan tersebut menyebabkan fitur yang diekstraksi oleh jaringan konvolusi memiliki tingkat tumpang tindih yang tinggi sehingga meningkatkan kemungkinan terjadinya misklasifikasi [6].

Secara konseptual, tingginya nilai *precision* dan *recall* pada hampir seluruh kelas menunjukkan bahwa kemampuan ekstraksi fitur multi-skala pada InceptionV3 mendukung performa klasifikasi yang diperoleh. Arsitektur *Inception Block* memungkinkan model memproses fitur menggunakan beberapa ukuran filter secara bersamaan, sehingga mampu menangkap karakteristik penyakit baik dalam bentuk tekstur halus maupun pola bercak yang lebih luas. Selain itu, penggunaan *Global Average Pooling* membantu mengurangi jumlah parameter tanpa menghilangkan informasi spasial yang penting, sehingga model tetap memiliki kompleksitas komputasi yang relatif ringan untuk dikonversi ke dalam format *TensorFlow Lite* dan diimplementasikan pada perangkat *mobile*.

Tabel 9. Perbandingan Hasil Penelitian

Penelitian	Metode	Implementasi	Hasil
Shandilya dkk. [6]	Hybrid CNN–Vision Transformer (CNN-ViT)	Model klasifikasi	Akurasi 99,15%
Aryanti dkk. [7]	CNN	Model klasifikasi	Akurasi 95%
Tirta dkk. [8]	Transfer Learning DenseNet-201	Model klasifikasi	Akurasi 96,65%
Sodikin dkk. [9]	Transfer Learning MobileNetV2	Model klasifikasi	Akurasi validasi 85%
Askale dkk. [11]	CNN (VGG16)	Aplikasi Android	Akurasi 95%
Salsabilah dkk. [12]	CNN + Transfer Learning	GUI Desktop	Akurasi 90,00%
Penelitian ini	**Transfer Learning InceptionV3 + Perbandingan Adam, SGD, RMSProp**	**Aplikasi Android (Flutter)**	**Akurasi 97,26% (Adam)**

Jika dibandingkan dengan penelitian terdahulu, hasil penelitian ini menunjukkan peningkatan performa sekaligus memberikan perspektif baru terhadap proses optimasi model. Salsabilah dkk. [12] melaporkan akurasi sebesar 90,00% menggunakan pendekatan *transfer learning* pada empat kelas penyakit daun jagung, sedangkan penelitian ini memperoleh akurasi 97,26% melalui penggunaan dataset gabungan dan evaluasi beberapa *optimizer*. Perbedaan tersebut diduga dipengaruhi oleh meningkatnya keragaman data pelatihan serta pemilihan *optimizer* yang lebih sesuai untuk proses *fine-tuning*. Di sisi lain, penelitian Shandilya dkk. [6] melaporkan akurasi yang lebih tinggi menggunakan arsitektur hibrida CNN-*Vision Transformer* (CNN-ViT). Namun, pendekatan tersebut memiliki kompleksitas komputasi yang lebih tinggi sehingga kurang sesuai untuk implementasi pada perangkat bergerak dengan sumber daya terbatas. Temuan penelitian ini menunjukkan bahwa arsitektur CNN seperti InceptionV3 masih mampu menghasilkan performa klasifikasi yang mampu menghasilkan performa klasifikasi yang tinggi apabila dipadukan dengan strategi optimasi yang tepat.

Ditinjau dari kontribusi ilmiahnya, penelitian ini menganalisis dampak penggunaan optimizer Adam, SGD, dan RMSProp terhadap kinerja arsitektur InceptionV3 dalam

mengklasifikasikan penyakit daun jagung. Sejumlah penelitian sebelumnya, seperti Aryanti dkk. [7], Tirta dkk. [8], Sodikin dkk. [9], dan Shandilya dkk. [6], mengembangkan solusi klasifikasi dengan memanfaatkan pendekatan deep learning yang berbeda, yaitu CNN, DenseNet-201, MobileNetV2, serta kombinasi CNN dan Vision Transformer. Sementara itu, penelitian ini menggunakan InceptionV3 sebagai arsitektur utama dan memusatkan kajian pada evaluasi pengaruh berbagai algoritma optimasi terhadap kinerja model. Pendekatan tersebut memperluas sudut pandang penelitian klasifikasi penyakit daun dengan menunjukkan bahwa pencapaian performa model tidak hanya ditentukan oleh arsitektur jaringan yang digunakan, tetapi juga oleh strategi optimasi yang diterapkan selama proses pelatihan.

Penelitian ini masih menyimpan beberapa batasan di luar hasil evaluasi yang positif ini. Model hanya dikembangkan untuk mengidentifikasi tiga jenis penyakit daun jagung dan satu kelas sehat, serta seluruh data pelatihan berasal dari dataset publik sehingga variasi kondisi lapangan nyata masih terbatas. Selain itu, model masih mengalami kesalahan klasifikasi pada kelas yang memiliki karakteristik visual yang sangat mirip, khususnya *Gray Leaf Spot* dan *Blight*. Oleh karena itu, penelitian selanjutnya dapat mengembangkan metode segmentasi citra atau *object detection*, seperti YOLO, untuk memisahkan area daun dari latar belakang sehingga proses ekstraksi fitur menjadi lebih optimal. Pengembangan dataset yang lebih beragam dengan variasi intensitas pencahayaan, kondisi lingkungan, serta penambahan jenis penyakit juga diharapkan mampu meningkatkan kemampuan generalisasi model pada implementasi di lingkungan nyata.

5. Simpulan

Studi ini berhasil mengembangkan sistem deteksi penyakit daun jagung menggunakan metode *transfer learning* berbasis InceptionV3 yang mampu mengklasifikasikan empat kategori, yakni *blight*, *common rust*, *gray leaf spot*, dan *healthy*. Proses pengolahan data mencakup penggabungan dataset Kaggle serta Mendeley, augmentasi data, normalisasi, serta pemisahan data menjadi data pelatihan, validasi, dan pengujian. Model kemudian dievaluasi menggunakan tiga *optimizer*, yaitu SGD, RMSProp, dan Adam. Hasil pengujian menunjukkan bahwa *optimizer* Adam menghasilkan performa tertinggi dengan akurasi sebesar 97,26%, lebih tinggi dibandingkan RMSProp dan SGD. Evaluasi menggunakan *confusion matrix* dan *classification report* memperlihatkan bahwa model memiliki kemampuan klasifikasi yang baik pada seluruh kelas penyakit. Model yang telah dikembangkan selanjutnya diimplementasikan ke dalam aplikasi Android menggunakan Flutter dan TensorFlow Lite, sehingga mampu melakukan deteksi penyakit daun jagung secara langsung melalui kamera maupun galeri dengan hasil yang cepat dan praktis bagi pengguna. Penelitian selanjutnya dapat dilakukan dengan menambah variasi dataset melalui penambahan jenis penyakit daun jagung yang lebih beragam serta membandingkan atau menerapkan arsitektur *deep learning* lain yang berpotensi memberikan performa yang lebih optimal baik dari segi akurasi maupun efisiensi sistem.

Daftar Referensi

[1] V. Ardilla, J. Supriyanti, S. Diana Putri, R. Fevria, and M. Larashinda, “Inventarisasi Penyakit-Penyakit Pada Tanaman Jagung (Zea Mays L.) Di Lahan Percobaan Dinas Pertanian Kabupaten Sijunjung ”, *snagro*, vol. 2, pp. 37–44, Agustus 2025.

[2] U. Khaira, I. Weni, and W. Wilia, “Rancang Bangun Aplikasi Deteksi Penyakit Tanaman Jagung Melalui Citra Daun Berbasis Android Menggunakan Algoritma Convolutional Neural Network”, *Publikasi Elektronik Pengembangan Aplikasi Digital Untuk Negeri*, vol. 5, no. 1, pp. 1–11, Apr. 2024, doi: 10.23960/pepadun.v5i1.210.

[3] Q. N. Azizah, “Klasifikasi Penyakit Daun Jagung Menggunakan Metode Convolutional Neural Network AlexNet”, *sudo J. Tek. Inform.*, vol. 2, no. 1, pp. 28–33, Feb. 2023, doi: 10.56211/sudo.v2i1.227.

[4] M. Yusuf, Khoirunnisa, D. Kurniawan, dan T. Agustin, “Klasifikasi penyakit tanaman jagung dengan kecerdasan buatan berbasis CNN,” *Semin. Nas. AMIKOM Surakarta*, vol. 2, pp. 355–368, 2024, doi: 0.47065/jieee.v5i1.2701.

[5] D. P. Dharmawan, E. Y. Puspaningrum, dan A. L. Nurlaili, “Klasifikasi Bunga Herbal Menggunakan Model Inceptionv3 Dan Vision Transformer”, *JATI (Jurnal Mahasiswa Teknik Informatika)*, vol. 9, no. 3, pp. 5474–5482, 2025, doi: 10.36040/jati.v9i3.14308.

[6] G. Shandilya, S. Gupta, H. G. Mohamed, S. Bharany, A. U. Rehman, dan S. Hussen, “Enhanced Maize Leaf Disease Detection and Classification Using an Integrated CNN-ViT

Model," *Food Science & Nutrition*, vol. 13, no. 7, e70513, 2025, doi: 10.1002/fsn3.70513.
[7] A. Aryanti, D. Apriliani, W. Z. Putri, D. Ramadhani, T. Malinda, and D. Andreansyah, "Identifikasi CNN dalam Deteksi Penyakit Daun Jagung Berbasis Pengenalan Gambar,", *MIND (Multimedia Artificial Intelligent Networking Database. Journal MIND Journal,* vol. 10, no. 2, pp. 195–205, 2025, doi: 10.26760/mindjournal.v10i2.195-205.
[8] M. S. Tirta, R. Kurniawan, dan A. Zulius, "Model Transfer Learning Dalam Klasifikasi Penyakit Daun Jagung Menggunakan Arsitektur Densenet-201," Journal of Informatics, Electrical and Electronics Engineering, vol. 4, no. 3, pp. 91–99, 2025, doi: 10.47065/jieee.v4i3.2347.
[9] S. Sodikin, T. Khotimah, dan A. Jazuli, "Penerapan Transfer Learning Menggunakan Mobile NetV2 untuk Klasifikasi Penyakit Daun Jagung Berbasis Citra," JEMSI, vol. 6, no. 6, pp. 4276–4282, 2025, doi: 10.38035/jemsi.v6i6..
[10] A. Herbert, P. Sitohang, T. I. Hermanto, dan C. D. Lestari, "Tumbuhan Stroberi Menggunakan Metode Convolutional Neural Network Arsitektu," *JITET (Jurnal Inform. dan Tek. Elektro Ter.*, vol. 12, no. 3, pp. 4146–4154, 2024. doi: http://dx.doi.org/10.23960/jitet.v12i3$1.5274
[11] G. T. Askale, A. B. Yibel, B. M. Taye, dan G. D. Wubneh, "Mobile based deep CNN model for maize leaf disease detection and classification," *Plant Methods*, vol. 21, no. (1)72, 2025, doi: 10.1186/s13007-025-01386-5.
[12] Salsabilah, N., Riadi, A., dan Chamid, A., "CNN-Based Automatic Detection of Corn Leaf Diseases Using Desktop GUI Application," Progresif: Jurnal Ilmiah Komputer, vol. 22, no. 1, pp. 228–238, 2026, doi: 10.35889/progresif.v22i1.3501.
[13] M. H. Diponegoro, S. S. Kusumawardani, dan I. Hidayah, "Tinjauan Pustaka Sistematis: Implementasi Metode Deep Learning pada Prediksi Kinerja Murid," *J. Nas. Tek. Elektro dan Teknol. Inf.*, vol. 10, no. 2, pp. 131–138, 2021, doi: 10.22146/jnteti.v10i2.1417.
[14] S. Sharma dan P. Chaudhary, "Machine learning and deep learning," dalam Quantum Computing and Artificial Intelligence: Training Machine and Deep Learning Algorithms on Quantum Computers, P. Raj, A. Kumar, A. K. Dubey, S. Bhatia, dan O. M. S., Eds. Berlin, Boston: De Gruyter, 2023, pp. 71–84, doi: 10.1515/9783110791402-004..
[15] R. W. Hasibuan *dkk.*, "Implementasi Convolutional Neural Network Dalam Mendeteksi Tingkat Kematangan Buah Kakao", *JATI (Jurnal Mahasiswa Teknik Informatika)*, vol. 9, no. 4, pp. 6479–6487, 2025, doi: 10.36040/jati.v9i4.14116.
[16] M. Heenaye-Mamode, M. Khan, P. Reesaul, et al., "Detection of Alzheimer's disease using pre-trained deep learning models through transfer learning: A review," Artif. Intell. Rev., vol. 57, Art. no. 275, 2024, doi: 10.1007/s10462-024-10914-z.
[17] T. S. Winanto, C. Rozikin, and A. Jamaludin, "Analisa Performa Arsitektur Transfer Learning Untuk Mengindentifikasi Penyakit Daun Pada Tanaman Pangan", *JAIC*, vol. 7, no. 1, pp. 74–87, Jul. 2023, doi: 10.30871/jaic.v7i1.5991.
[18] O. Iparraguirre-Villanueva, V. Guevara-Ponce, O. R. Paredes, F. Sierra-Liñan, J. Zapata-Paulini, dan M. Cabanillas-Carbonell, "Convolutional Neural Networks with Transfer Learning for Pneumonia Detection," International Journal of Advanced Computer Science and Applications (IJACSA), vol. 13, no. 9, pp. 544–551, 2022, doi: 10.14569/IJACSA.2022.0130963.
[19] V. Wulandari, W. J. Sari, Z. H. Al-sawaff, dan S. Manickam, "Comparative Analysis of Weather Image Classification Using CNN Algorithm with InceptionV3 , DenseNet169 and NASNetMobile Architecture Models," *PREDATECS*, vol. 2, no. 2, pp. 81-92, Jan. 2025, doi: 10.57152/predatecs.v2i2.1608.
[20] G. Hermawan, "Memahami Peran Dataset dalam Penelitian Kecerdasan Buatan: Kualitas, Aksesibilitas, dan Tantangan," October 2024, doi: 10.13140/RG.2.2.34468.49288.
[21] J. Cao, M. Yan, Y. Jia, X. Tian, and Z. Zhang, "Application of a modified Inception-v3 model in the dynasty-based classification of ancient murals," EURASIP J. Adv. Signal Process., vol. 2021, no. 49, 2021, doi: 10.1186/s13634-021-00740-8.
[22] E. Helmud, E. Helmud, F. Fitriyani, and P. Romadiana, "Classification Comparison Performance of Supervised Machine Learning Random Forest and Decision Tree Algorithms Using Confusion Matrix", *SISFOKOM*, vol. 13, no. 1, pp. 92–97, Feb. 2024, doi: 10.32736/sisfokom.v13i1.1985.
[23] A. Rizky, W. Widhiarso, and A. Rahman, "Klasifikasi kelayakan ban sepeda motor menggunakan metode Convolutional Neural Network," Progresif: Jurnal Ilmu Komputer, vol.

21, no. 2, pp. 623–633, 2025, doi: 10.35889/progresif.v21i2.2917.
[24] C. Kartiko, A. C. Wardhana, dan D. P. Rakhmadani, “Pengembangan Mobile Learning Management System Dengan User Centered Design (UCD) Menggunakan Flutter Framework,”, *mib*, vol. 6, no. 2, pp. 960–968, Apr. 2022, doi: 10.30865/mib.v6i2.3524.
[25] I. Susilawati, dan M. Muhammad, “Model CNN Berbasis Efficientnet-Lite Untuk Deteksi Retinopati Diabetik Menggunakan Tensorflow Lite,”, *JATI (Jurnal Mahasiswa Teknik Informatika),* vol. 9, no. 5, pp. 8754–8761, 2025, doi: 10.36040/jati.v9i5.15192.
[26] A. Mohammad, S. Surono, dan A. Thobirin, “Performance Evaluation Of Transfer Learning Models Based On Optimization In Agricultural Pest Classification”, *jitk*, vol. 11, no. 4, pp. 1401–1410, Mei 2026, doi: 10.33480/jitk.v11i4.7800.