Journal Software, Hardware and Information Technology
p-ISSN: 2808-3385 | e-ISSN: 2776-8961
Volume 6 Issue 2, June 2026

Evaluasi Efisiensi Model *Deep Learning* pada Deteksi Penyakit Cabai Dengan DenseNet201 dan Efficient-NetB0

Efficiency Evaluation of Deep Learning Models for Chili Disease Detection Using Dense-Net201 and EfficientNetB0

Santi Prayudani, Grace Putri Jelita Waruwu, Grace Mepa Stephoney Silaban
Jurusan Teknik Komputer dan Informatika, Politeknik Negeri Medan, Medan, Indonesia
santiprayudani@polmed.ac.id, graceputrijelita@students.polmed.ac.id, gracemepastephoney@students.polmed.ac.id

Notes

Submitted: 28-05-2026
Revised: 27-06-2026
Accepted: 28-06-2026
Published: 30-06-2026

How to cite: S. Prayudani, G. P. J. Waruwu, and G. M. S. Silaban, "Evaluasi Efisiensi Model Deep Learning pada Deteksi Penyakit Cabai Dengan DenseNet201 dan EfficientNetB0," *Journal Software, Hardware and Information Technology (SHIFT)*, vol. 6, no. 2, pp. 208–220, Jun. 2026, doi: 10.24252/shift.v6i2.299.

DOI: 10.24252/shift.v6i2.299

Copyright 2026© the Author(s)

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.

ABSTRACT

This study compares the performance of DenseNet201 and EfficientNetB0 architectures in detecting five types of conditions on chili plant leaves, namely healthy, leaf curl, leaf spot, whitefly, and yellowish. Manual disease identification by farmers has accuracy limitations and is time-consuming, hence a transfer learning-based deep learning approach is proposed as a solution. A secondary dataset of 500 digital images was divided into 80% training data, 10% validation data, and 10% testing data. The training process utilized the Adam optimization function, an initial learning rate of 0.001, and data augmentation techniques to prevent overfitting. Both models were evaluated based on standard classification metrics and memory size efficiency. The results showed that DenseNet201 achieved the highest accuracy of 92.00% with a storage model size of 141.25 MB and a training time of 5.4 minutes. On the other hand, EfficientNetB0 produced an accuracy of 88.00% with a model size 2.2 times smaller at 62.25 MB and a shorter training time of 4.3 minutes. There is a significant trade-off in both architectures; DenseNet201 is recommended for cloud computing systems that prioritize high accuracy (92.00%) with a precision of 92.32%, while EfficientNetB0 is more prospective for development on mobile devices because the number of parameters is much smaller (8,068,200 compared to 24,347,653).

Keywords: chili disease, CNN, deep learning, DenseNet201, EfficientNetB0.

ABSTRAK

Penelitian ini membandingkan kinerja arsitektur DenseNet201 dan EfficientNetB0 dalam mendeteksi lima jenis kondisi pada daun tanaman cabai, yaitu *healthy*, *leaf curl*, *leaf spot*, *whitefly*, dan *yellowish*. Identifikasi penyakit secara

manual oleh petani memiliki keterbatasan akurasi dan memakan waktu, sehingga pendekatan *deep learning* berbasis *transfer learning* diusulkan sebagai solusi. Dataset sekunder berjumlah 500 citra digital dibagi menjadi 80% data latih, 10% data validasi, dan 10% data uji. Proses pelatihan menggunakan fungsi optimasi Adam, *learning rate* awal 0,001, dan teknik *augmentasi data* untuk mencegah *overfitting*. Kedua model dievaluasi berdasarkan metrik klasifikasi standar dan efisiensi ukuran memori. Hasil penelitian menunjukkan bahwa DenseNet201 mencapai akurasi tertinggi sebesar 92,00% dengan ukuran penyimpanan model 141,25 MB dan waktu pelatihan 5,4 menit. Di sisi lain, EfficientNetB0 menghasilkan akurasi sebesar 88,00% dengan ukuran model yang 2,2 kali lebih kecil yakni 62,25 MB dan waktu pelatihan yang lebih singkat yaitu 4,3 menit. Terdapat *trade-off* yang signifikan pada kedua arsitektur; DenseNet201 direkomendasikan untuk sistem komputasi awan (*cloud*) yang mengutamakan akurasi tinggi (92,00%) dengan presisi 92,32%, sedangkan EfficientNetB0 lebih prospektif untuk dikembangkan pada perangkat *mobile* karena jumlah parameter yang jauh lebih sedikit (8.068.200 berbanding 24.347.653).

Kata kunci: CNN, deep learning, DenseNet201, EfficientNetB0, penyakit cabai.

1. PENDAHULUAN

Cabai (*Capsicum annuum L.*) merupakan salah satu komoditas hortikultura yang sangat penting dan strategis di Indonesia, serta nilai ekonominya yang cukup tinggi. Cabai juga memegang peranan besar, baik dalam konsumsi rumah tanga maupun industri pangan [1]. Namun, produksi cabai di tingkat nasional sering menghadapi masalah serius, terutama karena serangan berbagai penyakit daun. Contohnya, *leaf curl* (daun keriting), *leaf spot* (bercak daun), serangan *whitefly*, dan kondisi *yellowish* (menguning), yang semuanya sebenarnya bisa menurunkan kualitas dan hasil panen yang cukup signifikan [2][3]. Seringkali gejala-gejala penyakit ini terlambat dikenali, sehingga tindakan pengendalian yang seharusnya cepat dan tepat menjadi terhambat. Akibatnya, berdampak buruk pada pendapatan petani yang menggantungkan hidupnya pada tanaman ini [4]. Sehingga, diperlukan kebutuhan untuk memperbaiki sistem deteksi dan pengendalian agar kerugian yang dialami petani bisa diminimalkan.

Identifikasi penyakit dengan pendekatan tradisional yang hanya mengandalkan pengamatan visual saja oleh petani memiliki keterbatasan yang cukup signifikan. Cara ini subjektif dan kurang maksimal, terutama karena gejala penyakit yang satu dan lainnya kadang terlihat mirip [5]. Oleh karena itu, diperlukan kebutuhan akan sistem deteksi dini yang bukan hanya cepat tapi juga akurat dan objektif, serta menjadi semakin penting untuk menunjang praktik pertanian presisi (*precision agriculture*) [6]. Kemajuan dalam bidang *Deep Learning,* khususnya melalui dalam pemanfaatan *Convolutional Neural Network* (CNN), muncul sebagai solusi yang menjanjikan. Pendekatan ini, berdasarkan analisis citra daun digital, yang sudah terbukti cukup efektif dan berpotensi jadi alat bantu yang sangat berguna bagi para petani [7].

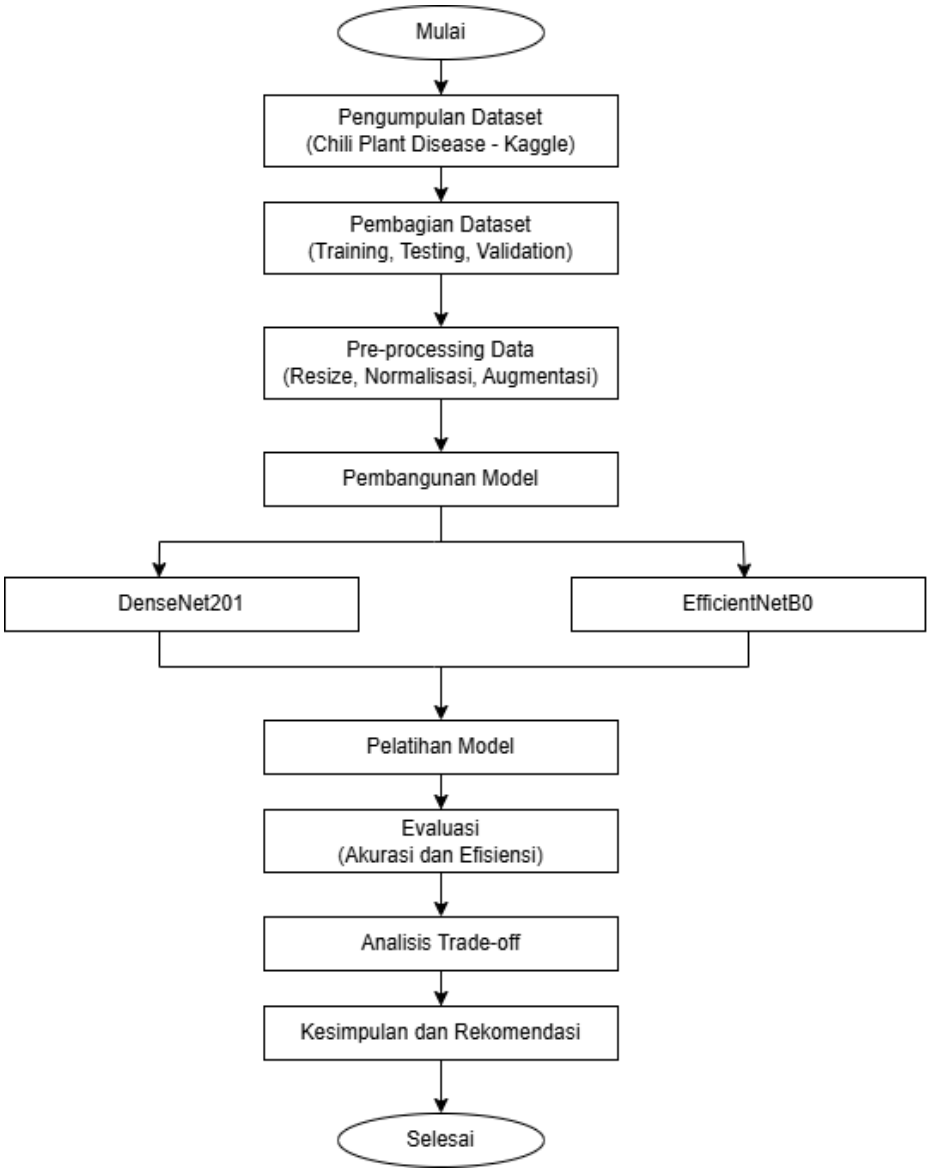
Dalam beberapa tahun terakhir, pendekatan *Transfer Learning* (TL) yang diaplikasikan pada arsitektur CNN menarik perhatian banyak peneliti. Hal ini terutama disebabkan oleh kemampuannya untuk memanfaatkan bobot model yang sudah dilatih sebelumnya (*pre-trained*) yang secara signifikan bisa meningkatkan akurasi klasifikasi, terutama pada kondisi dataset yang terbatas [8]. Banyak penelitian yang menunjukkan bahwa arsitektur seperti ResNet101 mampu mencapai akurasi validasi hingga 98,12% dalam mendeteksi penyakit pada tanaman cabai. Selain itu, model dengan koneksi padat, semisal DenseNet201, juga kerap diulik karena kemampuannya dalam menangkap kompleksitas fitur visual yang ada pada penyakit cabai [9][10]. Meski begitu, ada masalah yang cukup serius, yaitu efisiensi model saat diterapkan secara langsung di lapangan. DenseNet201 memang akurat, tapi ukuran modelnya cukup besar, dan ini jelas membebani sumber daya komputasi yang digunakan [11]. Jadi, walaupun menjanjikan, aspek efisiensi tidak bisa diabaikan begitu saja.

Sebagai pilihan lain, arsitektur EfficientNetB0 menggunakan pendekatan yang disebut *compound scaling*, yang pada dasarnya mengatur tiga aspek model sekaligus kedalaman, lebar, dan resolusi dengan koefisien gabungan untuk mengoptimalkan efisiensi parameter [12]. Namun, sampai sekarang, riset yang secara khusus menilai apakah model yang relatif ringan ini benar-benar bisa mempertahankan tingkat akurasi yang sebanding saat langsung dibandingkan dengan DenseNet201 pada dataset penyakit cabai yang tergolong kecil masih sedikit. Aspek efisiensi komputasi juga kurang diperhatikan secara menyeluruh dalam konteks ini jadi, ini cukup menarik untuk ditelusuri lebih lanjut [13][14].

Menanggapi permasalahan yang ada, penelitian ini menerapkan metode eksperimental komparatif dengan konfigurasi dan parameter yang sama pada kedua arsitektur menggunakan dataset "*Chili Plant Disease"* yang bersumber dari *Kaggle* terdiri atas 500 citra dalam 5 kelas penyakit. Penelitian ini memberikan kontribusi baru dengan membandingkan DenseNet201 dan EfficientNetB0. Sampai sekarang, belum ada penelitian yang membandingkan kedua arsitektur ini secara langsung pada dataset penyakit cabai dengan parameter pelatihan yang sama. Yang tidak hanya berfokus pada akurasi klasifikasi, melainkan secara spesifik mengukur efisiensi komputasi dan ukuran penyimpanan model. Penelitian ini bertujuan untuk: membangun dan melatih model klasifikasi penyakit cabai menggunakan DenseNet201 dan EfficientNetB0; mengevaluasi kinerja kedua model dari segi akurasi maupun efisiensi penyimpanan; serta menganalisis *trade-off* antara kinerja dan efisiensi sebagai dasar rekomendasi arsitektur yang paling layak (*feasible*) untuk implementasi pada perangkat *mobile* atau IoT di sektor pertanian [15].

2. METODE PENELITIAN

Penelitian ini menggunakan metode eksperimental untuk membandingkan kinerja dua arsitektur yaitu DenseNet201 dan EfficientNetB0, dalam mengklasifikasikan penyakit pada daun cabai. Kedua arsitektur ini akan diuji dengan parameter yang sama untuk memastikan hasil perbandingan yang adil dan objektif. Proses penelitian secara keseluruhan dapat dilihat pada Gambar 1.



Gambar 1. Alur Penelitian

Gambar 1 menunjukkan alur penelitian yang dilakukan secara sistematis. Penelitian diawali dari proses pengumpulan data, kemudian dilanjutkan dengan pembagian dataset dan tahap *pre-processing.* Selanjutnya dilakukan pembangunan model arsitektur secara paralel, diikuti proses pelatihan model. Setelah itu, dilakukan evaluasi akurasi dan efisiensi penyimpanan model, kemudian dilanjutkan dengan analisis *trade-off* antara performa dan ukuran model. Hingga tahap akhir penelitian adalah penarikan kesimpulan dan rekomendasi arsitektur yang paling layak untuk digunakan pada sub bab 2.4 tabel 2.

2.1. Dataset

Dataset yang digunakan dalam penelitian ini adalah dataset sekunder "*Chili Plant Disease*" yang diperoleh dari *Kaggle* [16]. Dataset ini terdiri dari 500 citra digital daun cabai yang dibagi menjadi 5 kelas penyakit, yaitu: *healthy, leaf curl, leaf spot, whitefly,* dan *yellowish.*

Dataset ini terbagi menjadi tiga direktori utama sehingga memudahkan evaluasi yang objektif. Pembagian data dilakukan dengan perbandingan 80:10:10, di mana rasio ini dipilih karena merupakan pembagian yang umum diterapkan pada dataset berskala kecil hingga menengah. Tujuannya adalah untuk memastikan model memiliki data latih yang cukup untk mempelajari pola fitur. Selain itu, pembagian ini juga menyisakan data validasi dan data uji yang representatif, sehinggal evaluasi dapat dilakukan secara objektif. Artinya, 400 citra (80%) dari total data digunakan sebagai data latih untuk membentuk bobot model. Sementara itu, 50 citra (10%) dari total data digunakan sebagai data validasi untuk memantau kinerja model. Kemudian, 50 citra (10%) dari total data digunakan sebagai data uji untuk mengevaluasi kinerja model secara keseluruhan. Detail distribusi dataset dapat dilihat pada Tabel 1.

Tabel 1. Distribusi Dataset

Kelas	Data Latih	Data Uji	Data Validasi	Total
Healthy	80	10	10	100
Leaf Curl	80	10	10	100
Leaf Spot	80	10	10	100
Whitefly	80	10	10	100
Yellowish	80	10	10	100
Total	400	50	50	500

2.2. Pre-processing Data

Tahap *pre-processing* dilakukan untuk memastikan konsistensi dan kesiapan data masukan bagi kedua arsitektur model melalui tiga langkah sistematis berikut [17].

Pengubahan Ukuran Citra (Resizing)

Seluruh citra diubah ukurannya menjadi dimensi standar 224×224 piksel agar sesuai dengan ukuran masukan (*input layer*) kedua arsitektur model.

Normalisasi

Normalisasi diterapkan dengan cara yang berbeda-beda tergantung pada spesifikasi teknis masing-masing model. Pada DenseNet201, teknik *rescaling* standar digunakan dengan membagi nilai piksel dengan 255. Hal ini membuat nilai berada pada rentang 0 hingga 1, seperti yang terlihat pada Persamaan (1):

$$x' = \frac{x}{255} \tag{1}$$

di mana x adalah nilai piksel asli dan x′ adalah nilai piksel setelah normalisasi.

Sedangkan pada EfficientNetB0, fungsi *preprocess_input* bawaan digunakan. Fungsi ini melakukan standarisasi sesuai dengan distribusi bobot *pre-trained ImageNet* menggunakan normalisasi *mean-std*, seperti yang terlihat pada Persamaan (2):

$$x' = \frac{x-\mu}{\sigma} \tag{2}$$

di mana μ adalah nilai rata-rata dan σ adalah standar deviasi dari distribusi dataset *ImageNet.*

Augmentasi Data

Teknik augmentasi data diterapkan secara langsung pada data pelatihan, untuk membuat model lebih tangguh dan mengatasi masalah kurangnya data. Beberapa transformasi yang

digunakan antara lain: *rotation range 20°, horizontal flip, vertical flip, zoom range 0,2 dan shear range 0,2.*

2.3. Arsitektur Model

Penelitian ini membandingkan dua jenis arsitektur *Convolutional Neural Network* atau CNN dengan menggunakan arsitektur *DenseNet201* dan *EfficientNetB0*. Eksperimen dirancang menggunakan skenario pelatihan dan konfigurasi parameter yang identik (*controlled environment*) pada kedua model guna menjamin validitas hasil perbandingan.

DenseNet201

DenseNet201 adalah arsitektur *Convolutional Neural Network* (CNN) dengan 201 lapisan yang dibuat oleh Gao Huang. Arsitektur ini dirancang untuk mengatasi permasalahan *vanishing gradient* yang sering terjadi pada jaringan saraf tiruan dengan lapisan yang sangat dalam. DenseNet201 menerapkan konsep *dense connectivity*, yaitu setiap lapisan menerima masukan dari seluruh lapisan sebelumnya dan mengirimkan keluarannya ke seluruh lapisan berikutnya. Hal ini memungkinkan aliran gradien berjalan dengan lebih baik dan proses pembelajaran pada lapisan awal tetap berjalan [11]. Konsep ini dapat dirumuskan dalam Persamaan (3):

$$x_l = H_l([x_0, x_1, \dots, x_{l-1}]) \tag{3}$$

di mana x_l adalah output lapisan ke- l, H_l adalah fungsi transformasi non-linear, dan $[x_0, x_1, \dots, x_{l-1}]$ adalah gabungan peta fitur dari semua lapisan sebelumnya.

Pada penelitian ini, DenseNet201 digunakan sebagai model dasar untuk ekstraksi fitur dengan bobot *pre-trained* dari *ImageNet* yang dibekukan (*frozen*). Di atas model dasar ini, ditambahkan *classifier head* kustom dengan urutan lapisan sebagai berikut:

a. *Dropout* dengan tingkat 0,2: menonaktifkan 20% neuron secara acak untuk mencegah *overfitting.*
b. *Flatten*: mengubah output model dasar menjadi vektor satu dimensi.
c. Dense Layer(x2): dua lapisan *fully connected* dengan masing-masing 64 neuron dan fungsi aktivasi ReLU;
d. *Ouput Layer*: dengan 5 neuron dan fungsi aktivasi Softmax untuk menghasilkan distribusi probabilitas pada 5 kelas penyakit.

EfficientNetB0

EfficientNetB0 adalah salah satu jenis *Convolutional Neural Network* (CNN) yang menggunakan metode *compound scaling.* Metode ini memungkinkan penambahan skala pada tiga aspek jaringan sekaligus, yaitu kedalaman (*depth*), lebar (*width*), dan resolusi (*resolution*), menggunakan koefisien komposit ϕ [18]. Persamaan *compound scaling* ditunjukkan pada Persamaan (4), (5), dan (6):

$$d = \alpha^{\phi} \tag{4}$$

$$w = \beta^{\phi} \tag{5}$$

$$r = \gamma^{\phi} \tag{6}$$

dengan batasan $\alpha \cdot \beta^2 \cdot \gamma^2 \approx 2$, $\alpha \geq 1$, $\beta \geq 1$, $\gamma \geq 1$, di mana d, w, r masing-masing adalah faktor skala kedalaman, lebar, dan resolusi; α, β, γ adalah konstanta yang ditentukan melalui pencarian arsitektur (*neural architecture search*); dan ϕ adalah koefisien komposit yang mengendalikan sumber daya komputasi yang tersedia [18].

Classifier head yang ditambahkan pada model dasar EfficientNetB0 dirancang serupa dengan DenseNet201 untuk memastikan perbandingan yang konsisten [19]. Konfigurasi *classifier head* tersebut terdiri dari *Dropout* dengan tingkat 0,2 untuk mencegah *overfitting*, dilanjutkan dengan *Flatten* untuk mengubah *output* menjadi vektor satu dimensi. Setelah itu, diterapkan dua *Dense Layer* berturut-turut yang masing-masing memiliki 64 neuron dengan fungsi aktivasi ReLU, dan diakhiri oleh *Output Layer* dengan 5 neuron serta fungsi aktivasi *Softmax* untuk menghasilkan distribusi probabilitas pada 5 kelas penyakit.

2.4. Konfigurasi Pelatihan

Eksperimen pada penelitian ini dilakukan menggunakan spesifikasi perangkat keras dan perangkat lunak sebagaimana tercantum pada Tabel 2.

Tabel 2. Spesifikasi Perangkat Keras dan Lunak Eksperimen

Komponen	Spesifikasi
Platform	Google Colaboratory
GPU	NVIDIA Tesla T4
VRAM	16 GB
RAM	12,7 GB
Bahasa Pemrograman	Python 3.10
Framework Deep Learning	TensorFlow 2.12.0
API Model	Keras 2.12.0
Komputasi Numerik	NumPy 1.23.5
Evaluasi Model	Scikit-learn 1.2.2
Visualisasi	Matplotlib 3.7.1, Seaborn 0.12.2

Kedua model dilatih menggunakan konfigurasi hiperparameter yang identik sebagaimana disajikan pada Tabel 3.

Tabel 3. Konfigurasi Hiperparameter Pelatihan

Parameter	Nilai
Optimizer	Adam
Learning Rate	0.001
Loss Function	Sparse Categorical Crossentropy
Batch Size	32
Epoch	150
Dropout Rate	0.2

Pelatihan model juga mengimplementasikan teknik *callbacks* yang terdiri dari *Early Stopping* dengan batas toleransi (*patience*) 150 epoch untuk menghentikan pelatihan jika *val_loss* tidak membaik, *ModelCheckpoint* untuk menyimpan model dengan akurasi validasi terbaik, serta *ReduceLROnPlateau* yang akan menurunkan *learning rate* sebesar 50% (faktor 0.5) apabila performa model mulai stagnan selama 7 epoch. Hal ini bertujuan untuk mempercepat konvergensi dan menemukan parameter lokal minimum yang optimal.

2.5. Evaluasi Model

Evaluasi dilakukan secara menyeluruh untuk memastikan model bekerja dengan baik. Evaluasi ini menggunakan beberapa metrik standar untuk klasifikasi multi-kelas dan membandingkan efisiensi penyimpanan model [20].

Accuracy

Mengukur seberapa banyak prediksi yang benar dari total data uji. Rumusnya dapat dilihat pada Persamaan (7).

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (7)$$

Precision

Mengukur seberapa akurat prediksi positif yang dihasilkan model. Rumusnya dapat dilihat pada Persamaan (8).

$$Precision = \frac{TP}{TP+FP} \quad (8)$$

Recall

Mengukur kemampuan model untuk medetekti semua sampel positif yang sebeneranya. Rumusnya dapat dilihat pada Persamaan (9).

$$Recall = \frac{TP}{TP+FN} \quad (9)$$

F-1 Score

Merupakan rata-rata harmonik dari *Precision* dan *Recall* yang memberikan keseimbangan penilaian antara keduanya. Rumusnya dapat dilihat pada Persamaan (10).

$$F1 - Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (10)$$

di mana TP(*True Positive*) adalah jumlah prediksi positif yang benar, TN(*True Negative*) adalah jumlah prediksi negatif yang benar, FP(*False Positive*) adalah jumlah prediksi positif yang salah, dan FN(*False Negative*) adalah jumlah prediksi negatif yang salah.

Confusion Matrix

Visualisasi *Confusion Matrix* digunakan untuk menganalisis secara detail bagaimana kesalahan klasifikasi terjadi pada tiap kelas penyakit cabai. Tujuannya adalah untuk mengetahui pasangan kelas mana yang paling sering tertukar oleh model [21].

Efisiensi Penyimpanan Model

Untuk efisiensi penyimpanan, membandingkan ukuran file model dalam satuan MB setelah proses pelatihan selesai. Metrik ini menunjukkan berapa banyak memori yang dibutuhkan jika model diterapkan pada perangkat mobile.

3. HASIL PENELITIAN DAN PEMBAHASAN

Pada bagian ini dijelaskan hasil pengujian dari kedua arsitektur model berdasarkan parameter hiperparameter yang telah ditetapkan sebelumnya. Analisis berfokus pada kinerja metrik klasifikasi matriks kebingungan (*confusion matrix*), serta evaluasi komparatif mengenai efisiensi ukuran model.

3.1. Kinerja Model Klasifikasi

Pengujian dilakukan menggunakan testing data sebanyak 50 citra (10% dari total 500 citra dataset). Evaluasi kinerja dari arsitektur DenseNet201 dan EfficientNetB0 diukur menggunakan metrik *accuracy*, *precision*, *recall*, dan *f1-score*. Hasil pengujian secara komprehensif disajikan pada Tabel 4.

Tabel 4. Perbandingan Hasil Evaluasi Model

Arsitektur Model	*Accuracy (%)*	*Precision (%)*	*Recall (%)*	*F1-Score (%)*
DenseNet201	92,00	92,32	92,00	91,98
EfficientNetB0	88,00	89,17	88,00	87,88

Merujuk pada Tabel 4, DenseNet201 menunjukkan keunggulan yang konsisten pada semua metrik klasifikasi dengan tingkat akurasi mencapai 92,00%. Konektivitas antar lapisan yang padat (*dense connectivity*) pada arsitektur ini terbukti efektif dalam mempertahankan informasi gradien selama proses ekstraksi fitur citra daun cabai, sehingga menghasilkan pengklasifikasian yang lebih akurat.

3.2. Analisis Matriks Kebingungan (Confusion Matrix)

Evaluasi lanjutan dilakukan dengan menganalisis hasil prediksi pada masing-masing kelas kondisi daun (*healthy*, *leaf curl*, *leaf spot*, *whitefly*, dan *yellowish*):

a. DenseNet201: Model ini mencatatkan tingkat pengenalan sempurna (*Recall* 100%) pada kelas *whitefly* dan *yellowish*. Kesalahan identifikasi terendah berada pada kelas *healthy* (Recall 80%), di mana model terkadang keliru membedakan tekstur daun sehat dengan gejala awal penyakit. Kondisi ini dapat terjadi karena pada tahap awal infeksi, perubahan visual pada daun sehat sangat halus dan belum membentuk pola yang konsisten, sehingga menyulitkan model dalam memisahkan kedua kelas tersebut.

b. EfficientNetB0: Meski secara keseluruhan memiliki akurasi 88%, model ini sangat andal dalam mengenali kelas *leaf spot* dan *whitefly* dengan *Recall* masing-masing 100%. Tantangan utama bagi EfficientNetB0 terletak pada pengenalan kelas *leaf curl* yang hanya mencapai akurasi *Recall* sebesar 70%, menunjukkan adanya kesulitan dalam membedakan kerutan daun akibat hama secara presisi. Hal ini kemungkinan disebabkan oleh kemiripan visual antara gejala leaf curl dengan kondisi daun yang menggulung secara alami akibat faktor lingkungan, sehingga fitur yang diekstrak model belum cukup diskriminatif untuk kedua kondisi tersebut.

3.3. Analisis Efisiensi Penyimpanan dan Trade-off

Evaluasi efisiensi sangat krusial untuk menentukan kelayakan penerapan model pada perangkat *mobile* atau *Internet of Things* (IoT) yang umumnya memiliki keterbatasan sumber daya [22]. Perbandingan parameter dan ukuran penyimpanan model ditunjukkan pada Tabel 5.

Tabel 5. Perbandingan Efisiensi Model

Arsitektur Model	DenseNet201	EfficientNetB0
Total Parameter	24.347.653	8.068.200
Ukuran Model (MB)	141,25	62,25
Waktu Training	5,4 menit	4,3 menit

Tabel 5 memaparkan bahwa untuk mencapai akurasi 92%, DenseNet201 membutuhkan jumlah parameter yang sangat besar, mencapai lebih dari 24 juta, yang berimbas pada ukuran akhir model sebesar 141,25 MB. Sebaliknya, arsitektur EfficientNetB0 menunjukkan efisiensi yang sangat baik. Melalui metode *compound scaling*, model ini hanya membutuhkan sekitar 8 juta parameter, menghasilkan ukuran file yang jauh lebih ringan yakni 62,25 MB.

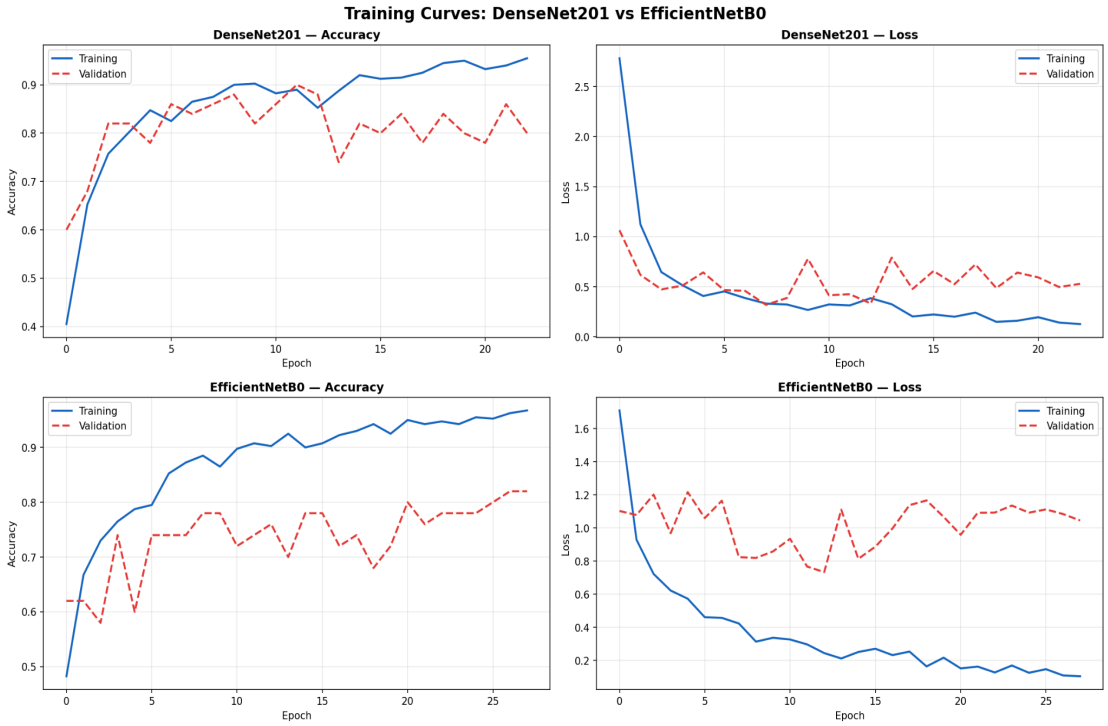
Hasil ini menegaskan adanya *trade-off* antara performa dan efisiensi memori. Apabila sistem deteksi penyakit cabai dirancang untuk beroperasi secara *offline* di perangkat *smartphone*

petani, EfficientNetB0 merupakan pilihan yang jauh lebih ideal karena berukuran 2,2 kali lebih kecil dan memiliki waktu pelatihan yang lebih singkat (4,3 menit). Namun, jika aplikasi berbasis *cloud* dengan kapasitas penyimpanan besar, DenseNet201 dapat digunakan untuk memaksimalkan presisi deteksi penyakit.

Guna memperkuat validitas hasil penelitian ini, dilakukan perbandingan dengan beberapa penelitian terdahulu yang relevan. Setiarini et al. [13] menggunakan DenseNet169 untuk identifikasi penyakit daun cabai dan memperoleh akurasi 94,44%, lebih tinggi dari DenseNet201 dalam penelitian ini (92,00%). Perbedaan ini kemungkinan disebabkan oleh perbedaan dataset dan konfigurasi fine-tuning yang digunakan. Sementara itu, Perdani et al. [18] menggunakan EfficientNet untuk klasifikasi glaukoma dan memperoleh performa yang baik dengan ukuran model yang kompak, konsisten dengan temuan penelitian ini bahwa EfficientNetB0 unggul dalam efisiensi. Khasanah dan Fachrie [11] menggunakan DenseNet20 untuk klasifikasi penyakit cabai dan menunjukkan bahwa arsitektur DenseNet secara umum efektif untuk domain ini. Secara keseluruhan, hasil penelitian ini konsisten dengan tren literatur yang menunjukkan adanya trade-off antara kompleksitas model dan efisiensi komputasi [10][14].

3.4. Analisis Grafik Pelatihan (Training Curves)

Untuk mengevaluasi stabilitas dan konvergensitas kedua arsitektur selama proses pelatihan, dilakukan analisis terhadap grafik *loss* dan *accuracy* yang terbentuk selama 150 *epoch*. Grafik perbandingan performa proses pelatihan dan validasi untuk kedua model disajikan pada Gambar 2.



Gambar 2. Grafik Perbandingan Kinerja Akurasi dan Loss Selama Proses Pelatihan

Berdasarkan Gambar 2, kurva pelatihan untuk arsitektur DenseNet201 menunjukkan penurunan nilai *loss* yang sangat stabil dan konvergen secara cepat mulai dari *epoch* ke-20. Hal ini menandakan bahwa konektivitas padat (*dense blocks*) pada DenseNet201 berhasil mempermudah aliran gradien informasi di setiap lapisan, sehingga meminimalkan risiko kehilangan

informasi tekstur daun cabai selama iterasi berlangsung. Kurva akurasi validasi pada DenseNet201 juga berimpit secara konsisten dengan kurva akurasi pelatihan, yang membuktikan bahwa model terhindar dari kendala *overfitting*.

Di sisi lain, kurva pelatihan pada model EfficientNetB0 menunjukkan fluktuasi minor pada awal *epoch* sebelum akhirnya mencapai kondisi konvergen yang stabil pada *epoch* ke-45. Meskipun nilai akhir akurasinya berada 4% di bawah DenseNet201, kestabilan kurva *loss* validasi dari EfficientNetB0 tetap terjaga dengan baik tanpa adanya lonjakan (*spike*) yang ekstrem. Keberhasilan pengendalian stabilitas pada kedua arsitektur model ini didukung secara signifikan oleh penerapan teknik *data augmentation* pada generator data serta mekanisme *callbacks Early Stopping* yang secara otomatis menghentikan pelatihan saat model telah mencapai titik optimal, sehingga mencegah pemborosan daya komputasi.

Ditinjau dari aspek efisiensi komputasi, hasil penelitian ini juga konsisten dengan temuan Hanh et al. [19] yang menunjukkan bahwa model berbasis EfficientNet mampu mencapai performa kompetitif dengan waktu pelatihan lebih singkat dibandingkan arsitektur yang lebih dalam. Begitu pula Guan et al. [22] yang menegaskan bahwa model ringan berbasis *deep learning* dapat mempertahankan akurasi yang memadai dengan sumber daya komputasi minimal. Selisih performa sebesar 4,00% antara DenseNet201 dan EfficientNetB0 dalam penelitian ini dapat dijelaskan dari perbedaan mendasar arsitekturnya. DenseNet201 menerapkan *dense connectivity* yang memungkinkan *reuse* fitur dari seluruh lapisan sebelumnya, sehingga lebih unggul dalam mengekstrak fitur diskriminatif pada dataset kecil [9]. Sebaliknya, EfficientNetB0 yang mengutamakan efisiensi melalui *compound scaling* belum sepenuhnya optimal pada dataset berukuran 500 citra, meskipun kestabilan pelatihannya tetap terjaga dengan baik.

4. KESIMPULAN

Berdasarkan eksperimen dan evaluasi yang telah dilakukan, dapat disimpulkan bahwa penerapan *Transfer Learning* berbasis *Deep Learning* mampu mengenali penyakit tanaman cabai secara otomatis. Arsitektur DenseNet201 memberikan performa akurasi tertinggi sebesar 92,00%, lebih unggul dibandingkan EfficientNetB0 yang mencapai 88,00%. Akan tetapi, dari perspektif efisiensi komputasi, EfficientNetB0 jauh lebih unggul dengan ukuran akhir model 62,25 MB (2,2 kali lebih ringan dibandingkan DenseNet201 yang berukuran 141,25 MB) serta waktu pelatihan yang lebih singkat. Analisis *trade-off* menunjukkan bahwa pemilihan arsitektur sangat bergantung pada *platform* target. Model DenseNet201 direkomendasikan untuk implementasi sistem berbasis server (*cloud*) yang mengutamakan keakuratan tingkat tinggi. Sementara itu, EfficientNetB0 sangat direkomendasikan untuk diterapkan pada perangkat *mobile* pintar (aplikasi Android/iOS) bagi petani, mengingat performanya yang tetap solid meski beroperasi pada perangkat dengan memori dan daya baterai terbatas.

PERNYATAAN PENGGUNAAN ARTIFICIAL INTELLIGENCE

Penulis memanfaatkan Gemini untuk mendukung penyusunan draf awal, perbaikan tata bahasa, dan penataan alur penulisan artikel. Hasil bantuan AI tidak digunakan secara langsung tanpa pemeriksaan, melainkan ditinjau dan disesuaikan berdasarkan konteks serta temuan penelitian. Penulis bertanggung jawab atas keseluruhan isi naskah.

KONTRIBUSI PENULIS

Santi Prayudani: Konsep, Metode, Eksperimen, Artikel, Supervisi, Review, Persetujuan AKhir Naskah.

Grace Putri Jelita Waruwu: Eksperimen, Penulisan Artikel, Supervisi, Review.

Grace Mepa Stephoney Silaban: Konsep, Metode, Eksperimen, Penulisan Artikel, Review.

PENDANAAN

Penelitian ini tidak menerima hibah khusus dari lembaga pendanaan mana pun, baik sektor publik, komersial, maupun nirlaba.

KONFLIK KEPENTINGAN

Penulis menyatakan tidak terdapat konflik kepentingan terkait publikasi artikel ini.

DAFTAR PUSTAKA

[1] L. Y. Rizki, N. Silvana, T. N. Huda, K. J. Utami, and H. Saksono, "Transformasi Bisnis Komoditas Cabai di NTB: Analisis Disparitas Pertumbuhan Cabai Merah dan Cabai Rawit2013–2022," *ALETHEIA: Jurnal Sosial & Humaniora, Inovasi, Ekonomi, dan Edukasi*, vol. 1, no. 2, pp. 105–115, Dec. 2024, doi: https://orcid.org/0009-0006-7501-4716.

[2] K. L. Waly, H. R. Amanupunnyo, and G. N. C. Tuhumury, "Kajian Keberadaan Jenis-Jenis Penyakit-penyakit Tanaman Penting pada Cabai (Capsicum sp.)," *Jurnal Pertanian Kepulauan (JPK)*, vol. 9, no. 1, pp. 42–49, Mar. 2025, doi: 10.30598/jpk.2025.9.1.42.

[3] C. A. Reyvalda, O. S. Bachri, and N. A. Ramdhan, "Klasifikasi Penyakit Daun Tanaman Cabai Menggunakan Convolutional Neural Network (CNN)," *RIGGS: Journal of Artificial Intelligence and Digital Business*, vol. 4, no. 4, pp. 10814–10820, Jan. 2026, doi: https://doi.org/10.31004/riggs.v4i4.5328.

[4] M. Anwar, Y. Kristian, and E. Setyati, "Classification Of Chili Plant Diseases Equipped With Leaf And Fruit Image Segmentation Using Yolo V7," *Journal of Information Technology and Computer Science (INTECOMS)*, vol. 6, no. 1, pp. 540–548, Jun. 2023, doi: 10.31539/intecoms.v6i1.6071.

[5] P. A. Azli, M. Minarni, A. Syahrani, G. Y. Swara, and A. Anisya, "Segmentasi Citra Daun Tomat untuk Klasifikasi Penyakit Tanaman Menggunakan Support Vector Machine (SVM)," *Jurnal Informatika: Jurnal Pengembangan IT*, vol. 10, no. 4, pp. 997–1008, Sep. 2025, doi: 10.30591/jpit.v10i4.9404.

[6] D. Rofianto, R. Maulini, D. Sahlinal, D. Meilantika, and T. Pujiana, "Deteksi Dini Penyakit Tanaman Jagung Berbasis Transfer Learning dengan Arsitektur DenseNet121," *Jurnal Sistem Komputer dan Informatika (JSON)*, vol. 7, no. 2, pp. 559–567, Dec. 2025, doi: 10.30865/json.v7i2.9094.

[7] B. Bhagawanta and C. B. Santoso, "Deteksi Penyakit Daun Kapas Dengan Deep Learning Berbasis Convolutional Neural Network (CNN)," *Idealis: Indonesia Journal Information System*, vol. 8, no. 2, pp. 220–227, Jul. 2025, doi: 10.36080/idealis.v8i2.3517.

[8] A. J. Rozaqi, A. Sunyoto, and M. R. Arief, "Implementasi Transfer Learning pada Algoritma Convolutional Neural Network Untuk Identifikasi Penyakit Daun Kentang," *Procedia of Engineering and Life Science*, vol. 1, no. 1, Mar. 2021, doi: https://doi.org/10.21070/pels.v1i1.820.

[9] R. A. Setyadi, S. Rahman, D. Manurung, M. Hasanah, and A. Indrawati, "Implementasi Transfer Learning Untuk Klasifikasi Penyakit Pada Daun Cabai Menggunakan CNN," *Jurnal Teknologi Informasi*, vol. 5, no. 2, p. 304, Aug. 2024, doi: 10.46576/djtechno.

[10] M. Bakr, S. Abdel-Gaber, M. Nasr, and M. Hazman, "DenseNet Based Model for Plant Diseases Diagnosis," *European Journal of Electrical Engineering and Computer Science*, vol. 6, no. 5, pp. 1–9, Sep. 2022, doi: http://dx.doi.org/10.24018/ejece.2022.6.5.458.

[11] N. U. Khasanah and M. Fachrie, "Klasifikasi Jenis Penyakit Tanaman Cabai Menggunakan Arsitektur DenseNet20," *SINTECH (Science and Information Technology) Journal*, vol. 7, no. 3, pp. 148–158, Dec. 2024, doi: https://doi.org/10.31598/sintechjournal.v7i3.1708.

[12] R. Apriliansyah, A. Handayanto, and N. D. Saputro, "Penerapan Arsitektur EfficientNetB0 pada Model Convolutional Neural Network untuk Deteksi Dini Mata Katarak," *Journal of Computer Engineering, System and Science*, vol. 10, no. 2, pp. 435–446, Jul. 2025, doi: https://doi.org/10.24114/cess.v10i2.66913.

[13] P. R. Setiarini, B. C. Oktariadi, and A. C. Siregar, "Identifikasi Penyakit Daun Cabai Menggunakan Arsitektur DenseNet169," *Jurnal CoSciTech (Computer Science and Information Technology)*, vol. 6, no. 3, pp. 430–437, Dec. 2025, doi: 10.37859/coscitech.v6i3.10631.

[14] N. Ullah *et al.*, "An Effective Approach For Plant Leaf Diseases Classification Based On A Novel DeepPlantNet Deep Learning Model," *Front. Plant Sci.*, vol. 14, pp. 01–16, Oct. 2023, doi: 10.3389/fpls.2023.1212747.

[15] S. Pancono, N. Indroasyoko, and A. Irfan Setiawan, "Pemantauan dan Deteksi Penyakit Daun Tomat Berbasis IoT dan CNN dengan Aplikasi Android," *Indonesian Journal of Computer Science*, vol. 13, no. 3, pp. 4692–4790, Jun. 2024, doi: 10.33022/ijcs.v13i3.4083.

[16] D. D. Prakoso, "Chili Plant Disease," *Kaggle*, 2021, [Online]. Available: https://www.kaggle.com/datasets/dhenyd/chili-plant-disease

[17] R. M. Aritonang, M. Y. Florestiyanto, and B. Yuwono, "Human Skin Disease Detection using Convolutional Neural Network Method with Hyperparameter Tuning to Determine the Best Parameter Combination," *Jurnal Informatika dan Teknologi Informasi*, vol. 20, no. 2, pp. 215–225, Jun. 2023, doi: 10.31515/telematika.v20i2.9161.

[18] W. R. Perdani, R. Magdalena, and N. K. Caecar Pratiwi, "Deep Learning untuk Klasifikasi Glaukoma dengan menggunakan Arsitektur EfficientNet," *ELKOMIKA: Jurnal Teknik Energi Elektrik, Teknik Telekomunikasi, & Teknik Elektronika*, vol. 10, no. 2, pp. 322–333, Apr. 2022, doi: 10.26760/elkomika.v10i2.322.

[19] B. T. Hanh, H. Van Manh, and N. V. Nguyen, "Enhancing The Performance Of Transferred Efficientnet Models In Leaf Image-Based Plant Disease Classification," *Journal of Plant Diseases and Protection*, vol. 129, no. 3, pp. 623–634, Jun. 2022, doi: 10.1007/s41348-022-00601-y.

[20] R. Faturrahman, Y. S. Hariyani, and S. Hadiyoso, "Klasifikasi Jajanan Tradisional Indonesia berbasis Deep Learning dan Metode Transfer Learning," *ELKOMIKA: Jurnal Teknik Energi Elektrik, Teknik Telekomunikasi, & Teknik Elektronika*, vol. 11, no. 4, pp. 945–957, Oct. 2023, doi: 10.26760/elkomika.v11i4.945.

[21] P. Novantara, Risteruw Leonardo Firmansyah, and Marrilyn Arismawati, "Deteksi Hama Penyakit Daun Padi Dengan Menggunakan Teknik Optimasi Deep Learning Convolutional Neural Network," *Bit-Tech (Binary Digital - Technology)*, vol. 7, no. 3, pp. 975–983, Apr. 2025, doi: 10.32877/bt.v7i3.2284.

[22] H. Guan, C. Fu, G. Zhang, K. Li, P. Wang, and Z. Zhu, "A Lightweight Model For Efficient Identification Of Plant Diseases And Pests Based On Deep Learning," *Front. Plant Sci.*, vol. 14, pp. 01–13, Jul. 2023, doi: 10.3389/fpls.2023.1227011.