Jutisi: Jurnal Ilmiah Teknik Informatika dan Sistem Informasi
https://ojs.stmik-banjarbaru.ac.id/index.php/jutisi/index
Jl. Ahmad Yani, K.M. 33,5 - Kampus STMIK Banjarbaru
Loktabat – Banjarbaru (Tlp. 0511 4782881), e-mail: puslit.stmikbjb@gmail.com
e-ISSN: 2685-0893

Identifikasi Penyakit Daun Kentang Menggunakan *Transfer Learning* Model *EfficientNetB3* Berbasis *Mobile*

DOI: http://dx.doi.org/10.35889/jutisi.v15i4.3657

R.BG. Moch. Faishal Reza[1*], Salamun Rohman Nudin[2]
Manajemen Informatika, Universitas Negeri Surabaya, Surabaya, Indonesia
*e-mail *Corresponding Author*: rbgreza27@gmail.com

Abstract

Potato leaf disease is one of the main factors that can reduce crop quality and yield, thus requiring an accurate and efficient identification method. Manual identification is often inconsistent and time-consuming. This study aims to develop a deep learning-based model using the EfficientNetB3 architecture with a transfer learning approach to identify potato leaf diseases. The dataset was obtained from Kaggle, namely the Potato Disease Leaf Dataset (PLD), and processed through preprocessing stages including augmentation and normalization. The model was trained using the Adam optimizer to achieve optimal performance. The evaluation results indicate that the model achieved a test accuracy of 99.50%, with precision of 1.00, recall of 1.00, and F1-score of 1.00. The model was then deployed into a Flutter-based mobile application and demonstrated the ability to accurately identify real-world data.

Keywords: *Potato leaf disease identification; EfficientNetB3; Deep learning; Transfer Learning; TensorFlow Lite.*

Abstrak

Penyakit daun kentang merupakan salah satu faktor utama yang dapat menurunkan kualitas dan hasil panen sehingga diperlukan metode identifikasi yang akurat dan efisien. Identifikasi secara manual sering tidak konsisten dan membutuhkan waktu yang relatif lama. Penelitian ini bertujuan untuk mengembangkan model berbasis *deep learning* menggunakan arsitektur EfficientNetB3 dengan pendekatan *transfer learning* untuk mengidentifikasi penyakit daun kentang. Dataset yang digunakan berasal dari Kaggle, yaitu Potato Disease Leaf Dataset (PLD), yang diproses melalui tahap *preprocessing* berupa augmentasi dan normalisasi. Model dilatih menggunakan optimizer Adam guna memperoleh kinerja optimal. Hasil evaluasi menunjukkan bahwa model mencapai *test accuracy* sebesar 99,50% serta nilai *precision* sebesar 1,00, *recall* sebesar 1,00, dan *F1-score* sebesar 1,00. Model kemudian diimplementasikan ke dalam aplikasi berbasis *mobile* menggunakan Flutter serta mampu mengidentifikasi data dunia nyata secara akurat.

Kata kunci: *Identifikasi penyakit daun kentang; EfficientNetB3;* Deep learning*; Transfer Learning; TensorFlow Lite.*

1. Pendahuluan

Kentang (*Solanum tuberosum L.*) termasuk salah satu tanaman pangan penting di dunia yang berada pada posisi keempat setelah gandum, padi, dan jagung [1]. Di Indonesia, komoditas ini tidak hanya dimanfaatkan sebagai alternatif sumber makanan, tetapi juga memiliki peran ekonomi yang cukup besar dalam sektor pertanian [2]. Berdasarkan data dari Badan Pusat Statistik, produksi kentang di Indonesia terus menunjukkan tren peningkatan, bahkan pada tahun 2024 jumlahnya telah melampaui 12,7 juta ton per tahun [3]. Tingginya tingkat produksi serta kandungan nutrisi yang dimiliki menjadikan kentang sebagai komoditas yang strategis dalam mendukung ketahanan pangan, baik pada skala nasional maupun internasional [4]. Peningkatan produksi tersebut turut mendorong kebutuhan akan penerapan teknologi pertanian yang lebih maju guna menjaga kualitas hasil panen [5].

Produktivitas tanaman kentang sering mengalami penurunan akibat serangan penyakit pada daun, terutama hawar daun (*Late Blight*) yang disebabkan oleh *Phytophthora infestans* serta bercak kering (*Early Blight*) oleh *Alternaria* [6]. Penyakit ini dapat berkembang dengan cepat dan memberikan dampak signifikan terhadap hasil panen, bahkan berpotensi menyebabkan kegagalan panen apabila tidak ditangani secara tepat [7]. Metode identifikasi yang umum digunakan oleh petani masih bergantung pada pengamatan secara langsung secara manual, yang memiliki berbagai keterbatasan seperti tingkat subjektivitas yang tinggi, kebutuhan akan keahlian khusus seorang ahli patologi tanaman, serta kurang efisien jika diterapkan pada lahan pertanian yang luas [8]. Sehingga diperlukan suatu sistem deteksi penyakit yang lebih akurat dan praktis agar dapat digunakan secara efektif di lapangan atau dunia nyata [9].

Kemajuan teknologi kecerdasan buatan, terutama pada bidang *computer vision*, telah mendorong pemanfaatan metode *Deep Learning* yang memanfaatkan arsitektur *Convolutional Neural Network* (CNN) sebagai solusi untuk analisis citra. [10]. Pendekatan ini banyak digunakan dalam proses identifikasi citra, karena mampu menghasilkan kinerja yang tinggi [11]. Berbagai penelitian terdahulu menunjukkan bahwa arsitektur CNN seperti AlexNet, VGG, ResNet, dan DenseNet mampu memberikan tingkat akurasi yang baik dalam identifikasi penyakit tanaman [12], [13], [14]. Studi sebelumnya juga menyatakan bahwa arsitektur CNN modern mampu mengekstraksi fitur visual secara lebih optimal sehingga meningkatkan kemampuan identifikasi citra [15]. Firasari dan Cahyanti [16] menggunakan arsitektur InceptionResNetV2 untuk mengklasifikasikan penyakit daun kentang dan memperoleh akurasi sebesar 95,30%, tetapi masih ditemukan keterbatasan dalam membedakan kelas tertentu, terutama ketika gejala visual memiliki kemiripan. Alhammad dkk. [17] mengombinasikan *deep learning* dengan *Explainable AI* menggunakan Grad-CAM dan arsitektur VGG16 berbasis *transfer learning*, sehingga model tidak hanya menghasilkan akurasi tinggi, tetapi juga memberikan gambaran area citra yang menjadi dasar prediksi. Meslmani dkk. [18] mengembangkan model EfficientNetB3 untuk klasifikasi penyakit daun kentang pada tiga kategori, yaitu *Early Blight*, *Late Blight*, dan *Healthy*, dengan akurasi mencapai 92%. Selain itu, Saputra dkk. [19] membandingkan GoogleNet dan EfficientNetB3, yang menunjukkan bahwa EfficientNetB3 memiliki performa lebih baik dari sisi akurasi dan efisiensi komputasi. Meskipun begitu, sebagian besar studi yang telah dilakukan sebelumnya cenderung menekankan pada peningkatan ketepatan model tanpa menerapkannya dalam uji coba di lingkungan nyata atau lapangan. Kesenjangan yang masih ada adalah belum adanya pengembangan model identifikasi yang tidak hanya tepat, tetapi juga efisien, serta diimplementasikan secara langsung pada aplikasi mobile untuk kemudahan penggunaan di lapangan atau dunia nyata.

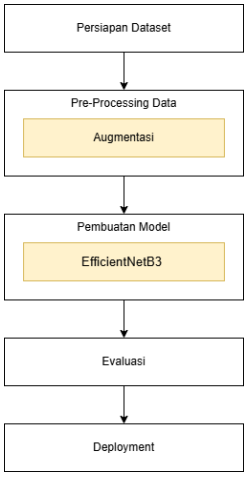
Berdasarkan gap tersebut, penelitian ini mengusulkan sistem identifikasi penyakit daun kentang menggunakan arsitektur EfficientNetB3 dengan pendekatan *transfer learning* yang diintegrasikan ke dalam aplikasi menggunakan Flutter berbasis *mobile*. EfficientNetB3 dipilih karena memiliki mekanisme *compound scaling* yang mampu menyelaraskan kedalaman jaringan, lebar jaringan, dan resolusi citra, sehingga dapat menghasilkan model yang efisien tanpa mengorbankan performa identifikasi [20]. Pendekatan *transfer learning* juga dinilai efektif karena dapat memanfaatkan bobot awal terhadap model yang telah dilatih pada dataset berskala besar, dengan begitu proses pelatihan menjadi efisien dan mampu menghasilkan ekstraksi fitur visual yang kuat meskipun dataset yang digunakan terbatas [21], [22]. Model yang dikembangkan kemudian dikonversi ke format TensorFlow Lite agar dapat dijalankan pada perangkat mobile dengan kapasitas komputasi terbatas [23], [24].Dengan demikian, kebaruan dalam penelitian ini terletak pada penggabungan EfficientNetB3 berbasis *transfer learning*, optimasi model menggunakan optimizer, konversi ke TensorFlow Lite, serta implementasi langsung ke dalam aplikasi menggunakan Flutter berbasis *mobile* yang dapat digunakan sebagai alat bantu deteksi awal oleh pengguna di lapangan atau dunia nyata.

2. Metodologi

2.1. Tahap Pengembangan Sistem

Pengembangan sistem dalam penelitian ini dijalankan melalui serangkaian tahapan yang tersusun secara sistematis. Proses diawali dengan pengumpulan dataset citra daun kentang yang kemudian dibagi menjadi data *training*, *validation*, dan *testing*. Selanjutnya, dilakukan tahap *pre-processing* yang mencakup normalisasi citra serta penyesuaian ukuran input menjadi 256×256 piksel. Tahap ini diikuti dengan penerapan *data augmentation* guna meningkatkan keberagaman data serta mengurangi risiko *overfitting*. Proses pemodelan dilakukan

menggunakan arsitektur EfficientNetB3 berbasis *transfer learning* dengan bobot *pre-trained ImageNet*, yang mencakup tahap *feature extraction* dan *fine-tuning* serta penambahan *fully connected layer*. Model dilatih menggunakan optimizer dengan penerapan mekanisme *callback* untuk mengoptimalkan proses pelatihan. Evaluasi model dilakukan menggunakan metrik *accuracy*, *precision*, *recall*, *F1-score*, serta *confusion matrix* pada data uji. Model dengan optimizer terbaik selanjutnya dikonversi ke dalam format TensorFlow Lite dengan menyertakan proses *preprocessing* sehingga dapat diimplementasikan pada aplikasi berbasis *mobile* dengan menggunakan Flutter sehingga mampu melakukan deteksi penyakit daun kentang secara *real-time*.



Gambar 1. Tahap Pengembangan Sistem

2.2. Persiapan Dataset

Dataset yang digunakan dalam penelitian ini berupa citra penyakit daun kentang yang diperoleh dari platform Kaggle, yaitu *Potato Disease Leaf Dataset* (PLD), yang terdiri atas tiga kategori, yaitu *Early Blight*, *Late Blight*, dan daun sehat (*healthy*). Jumlah keseluruhan data mencapai 4.072 citra yang dibagi ke dalam data *training* sebanyak 3.251 citra (80%), *validation* sebanyak 416 citra (10%), serta *testing* sebanyak 405 citra (10%). Seluruh citra kemudian disesuaikan ke ukuran resolusi 256×256 piksel agar sesuai dengan spesifikasi input model EfficientNetB3. Pembagian dataset ini bertujuan untuk memastikan bahwa proses pelatihan, validasi, dan pengujian model dapat berjalan dengan baik. Informasi detail terkait distribusi data pada masing-masing kelas disajikan pada Tabel 1.

Tabel 1. Label dan Jumlah Dataset

Kelas	Training	Validation	Testing
Early Blight	1303	163	162
Healthy	816	102	102
Late Blight	1132	151	141
Total	3251	416	405

Gambar 2. Potato Leaf Dataset

Gambar 2 menunjukkan contoh citra dataset penyakit daun kentang yang terbagi menjadi tiga kelas yaitu *Early Blight*, *Healthy*, dan *Late Blight*. Gambar pertama merupakan contoh daun kentang yang terinfeksi *Early Blight*, gambar kedua menunjukkan daun sehat (*Healthy*), dan gambar ketiga merupakan daun yang terinfeksi *Late Blight*. Perbedaan visual pada setiap kelas terlihat dari pola bercak, perubahan warna daun, serta tingkat kerusakan yang terjadi, yang menjadi dasar dalam proses identifikasi menggunakan model *Deep Learning*.

2.3. Pre-Procesing Data

Tahap *pre-processing* merupakan langkah awal yang dilakukan untuk menyiapkan citra sebelum digunakan dalam proses pelatihan model. Proses ini diawali dengan normalisasi citra menggunakan fungsi *preprocessing* dari EfficientNet, kemudian resolusi citra diubah menjadi 256×256 piksel untuk menyesuaikan dengan spesifikasi arsitektur model. Selain itu, diterapkan teknik *data augmentation* guna meningkatkan variasi data pelatihan sehingga model menjadi lebih tangguh terhadap berbagai kondisi citra [25]. Model yang dibangun digunakan untuk mengidentifikasikan penyakit daun kentang ke dalam tiga kategori, yaitu *Early Blight*, *Healthy*, dan *Late Blight*. Dataset yang digunakan telah dipisahkan sebelumnya menjadi data *training*, *validation*, dan *testing*, sehingga tidak diperlukan proses pemisahan tambahan.

Tabel 2. Parameter Pre-Processing dan Augmentasi Data

Parameter	Nilai
Ukuran Citra	256 x 256
Normalisasi	EfficentNet Preprocess
Rotasi	Hingga 25°
Width Shift	0.15
Height Shift	0.15
Shear	0.2
Zoom	0.3
Horizontal Flip	True
Brightness Range	[0.5, 1.5]
Fill Mode	Default

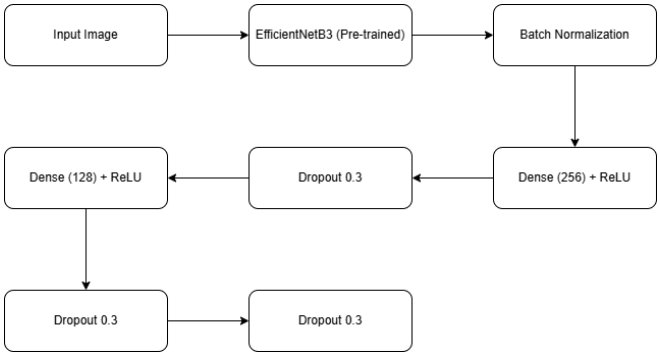
2.4. Pembuatan Model

Pada tahap ini dilakukan proses perancangan model dengan memanfaatkan arsitektur EfficientNetB3 yang telah menggunakan bobot *pre-trained* dari dataset ImageNet melalui pendekatan *transfer learning*. Dalam proses tersebut, model dasar digunakan sebagai *feature extractor*, di mana lapisan awal pada EfficientNetB3 dikunci (*freeze*) untuk mempertahankan fitur visual yang telah dipelajari sebelumnya. Pendekatan ini bertujuan untuk mempercepat proses pelatihan sekaligus meningkatkan kemampuan model dalam mengenali pola citra daun kentang.

Tahap *fine-tuning* diterapkan dengan melatih sebagian lapisan terakhir untuk menyesuaikan parameter terhadap dataset yang digunakan. Pada bagian akhir model ditambahkan *custom classification head* yang terdiri dari *Batch Normalization*, *Dense layer*, dan *Dropout* guna meningkatkan kemampuan generalisasi serta mengurangi risiko *overfitting*. Model kemudian dilatih menggunakan optimizer Adam dengan fungsi *loss categorical crossentropy* [26].

Selama proses pelatihan, digunakan beberapa mekanisme *callback* seperti *EarlyStopping*, *ReduceLROnPlateau*, dan *ModelCheckpoint*. *EarlyStopping* berfungsi untuk menghentikan pelatihan ketika tidak terjadi peningkatan performa, *ReduceLROnPlateau* digunakan untuk menyesuaikan *learning rate*, sedangkan *ModelCheckpoint* bertugas menyimpan model terbaik selama proses pelatihan berlangsung. Rancangan arsitektur EfficientNetB3 yang digunakan dalam penelitian ini ditampilkan pada Gambar 3.

Gambar 3 menunjukkan alur arsitektur model yang terdiri dari input citra yang diproses melalui model EfficientNetB3 sebagai *feature extractor*, dilanjutkan dengan lapisan *Batch Normalization*, *Dense layer*, dan *Dropout*, serta diakhiri dengan lapisan *Softmax* untuk menghasilkan identifikasi akhir. Arsitektur ini dirancang untuk menghasilkan performa yang optimal dengan tetap mempertahankan efisiensi komputasi sehingga sesuai untuk implementasi pada perangkat *mobile*.



Gambar 3. Rancangan Arsitektur EfficientNetB3

Tabel 3. Parameter Arsitektur Model EfficientnetB3

Layer (Tipe)	Output Shape	Parameter
input_layer_1 (InputLayer)	(None, 256, 256, 3)	0
efficientnetb3 (Functional)	(None, 1536)	10,783,535
batch_normalization (BatchNormalization)	(None, 1536)	6,144
dense (Dense)	(None, 256)	393,472
dropout (Dropout)	(None, 256)	0
dense_1 (Dense)	(None, 128)	32,896
dropout_1 (Dropout)	(None, 128)	0
dense_2 (Dense)	(None, 3)	387
Total		11,216,434

Model pada Tabel 3 memiliki total parameter sebesar 11.216.434 yang terdiri dari parameter terlatih dan tidak terlatih. Struktur model menunjukkan kombinasi EfficientNetB3 dengan beberapa lapisan identifikasi tambahan yang menghasilkan keseimbangan antara kemampuan dalam mengekstraksi fitur dan efisiensi komputasi. Dengan karakteristik tersebut, model dinilai sesuai untuk diterapkan pada perangkat yang memiliki keterbatasan kapasitas komputasi.

Tabel 4. Konfigurasi Pelatihan

Parameter	Nilai/Pengaturan
Arsitektur Input	256 x 256 pixel (RGB)
Batch Size	16
Optimizer	Adam
Learning Rate	Adaptive (ReduceLROnPlateau)
Epochs	50
Early Stopping	5 patience

Konfigurasi pelatihan pada Tabel 4 menunjukkan bahwa model dilatih menggunakan ukuran input citra sebesar 256 × 256 piksel dengan batch size 16. Proses optimasi dilakukan dengan memanfaatkan algoritma *Adam*, sedangkan penyesuaian *learning rate* dilakukan secara adaptif menggunakan mekanisme *ReduceLROnPlateau* untuk menjaga stabilitas pelatihan. Jumlah pelatihan ditetapkan sebanyak 50 *epochs*, dengan penerapan *Early Stopping* selama 5 *epochs* untuk menghentikan proses pelatihan jika tidak menunjukkan peningkatan performa pada data validasi.

2.5. Evaluasi

C*onfusion matrix* dan *classification report* digunakan sebagai acuan utama dalam evaluasi model. *Confusion matrix* merupakan metode analisis berbentuk tabel yang menampilkan empat kemungkinan hasil identifikasi, yaitu *True Positive* (TP), *True Negative* (TN), *False Positive* (FP), dan *False Negative* (FN). Pendekatan ini digunakan untuk menilai kinerja model dengan cara membandingkan label sebenarnya dengan hasil prediksi yang dihasilkan.

Hasil evaluasi selanjutnya digunakan untuk menghitung beberapa metrik kinerja, antara lain *accuracy*, *precision*, *recall*, dan *F1-score*. Nilai-nilai tersebut mencerminkan performa

menyeluruh terhadap kemampuan model saat melakukan penyakit daun kentang. Perhitungan metrik evaluasi dirangkum dalam bentuk *classification report* untuk memudahkan analisis performa model. Rumus yang digunakan untuk menentukan nilai *accuracy*, *precision*, *recall*, dan *F1-score* ditampilkan pada persamaan (1), (2), (3), dan (4).

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

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

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

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

2.6. Deployment

Integrasi model ke dalam sistem aplikasi merupakan tahap penting dalam siklus pengembangan *deep learning* agar hasil identifikasi dapat dimanfaatkan secara langsung oleh pengguna. Pada tahap ini, model TensorFlow yang telah melalui proses pelatihan sebelumnya diubah ke dalam format TensorFlow Lite (*.tflite*) untuk meningkatkan efisiensi proses inferensi, terutama pada perangkat dengan keterbatasan kapasitas komputasi seperti perangkat *mobile*. Proses konversi dilakukan dengan menggabungkan model dengan tahapan *preprocessing* sehingga model dapat menerima input citra secara langsung tanpa memerlukan proses tambahan di sisi aplikasi. Model yang berhasil dikonversi kemudian diintegrasikan ke dalam aplikasi berbasis *mobile* dengan menggunakan Flutter dengan memanfaatkan *library* TensorFlow Lite, sehingga aplikasi mampu melakukan identifikasi citra daun kentang secara *real-time*. Tahapan ini menghasilkan sebuah aplikasi yang dapat digunakan secara praktis dan efisien untuk mendeteksi penyakit daun kentang.

4. Hasil dan Pembahasan

3.1 Hasil Pelatihan dan Perbandingan Optimizer

Dalam tahap ini dilakukan pengujian kinerja model dalam mengidentifikasikan citra menggunakan beberapa variasi optimizer, yaitu tanpa optimizer tambahan (*baseline*), Adam, SGD, dan RMSProp. Pengujian dilakukan dengan parameter pelatihan yang seragam guna memastikan perbandingan yang objektif, meliputi ukuran batch, jumlah *epoch*, serta konfigurasi *early stopping*. Hasil dari pengujian tersebut disajikan pada Tabel 5.

Tabel 5. Hasil Perbandingan Optimizer

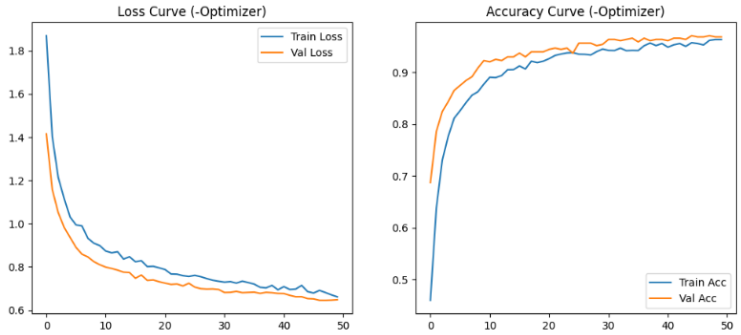
Optimizer	Batch Size	Epoch	Early Stopping	Train Accuracy	Val Accuracy	Test Accuracy
Non Optimizer	16	50	50 Epoch	96,37%	96,88%	98,27%
Adam	16	50	50 Epoch	**99,88%**	**99,76%**	**99,50%**
SGD	16	50	50 Epoch	90,56%	95,19%	93,58%
RMSProp	16	50	40 Epoch	99,78%	99,76%	99,25%

Berdasarkan hasil pada Tabel 5, model dengan optimizer Adam menunjukkan performa terbaik dibandingkan metode lainnya. Hal ini dapat dilihat dari nilai akurasi yang diperoleh, yaitu akurasi mencapai 99,88%, akurasi validasi 99,76%, dan akurasi pengujian mencapai 99,50%. Proses pelatihan menggunakan Adam berlangsung hingga epoch ke-50, yang mengindikasikan bahwa model mampu mencapai konvergensi dengan performa yang tinggi dan stabil. Optimizer RMSProp juga memperlihatkan bahwa performa yang sangat baik dengan akurasi pelatihan mencapai 99,78%, akurasi validasi 99,76%, dan akurasi pengujian mencapai 99,25%. Meskipun nilainya sedikit di bawah Adam, RMSProp menunjukkan kestabilan performa yang hampir setara, dengan proses pelatihan berhenti pada epoch ke-40 yang mengindikasikan efisiensi pelatihan yang lebih baik.

Optimizer SGD menghasilkan performa yang lebih rendah dibandingkan kedua optimizer tersebut, dengan akurasi pelatihan mencapai 90,56%, akurasi validasi 95,19%, dan akurasi pengujian mencapai 93,58%. Proses pelatihan menggunakan SGD berlangsung hingga epoch maksimum yaitu 50 epoch, yang mengindikasikan bahwa model belum mencapai konvergensi optimal. Sementara itu, skenario non optimizer (*baseline*) menghasilkan akurasi pelatihan mencapai 96,37%, akurasi validasi 96,88%, dan akurasi pengujian mencapai 98,27%. Temuan ini mengindikasikan bahwa pemilihan optimizer memberikan pengaruh yang signifikan terhadap performa model, sehingga Adam dipilih sebagai konfigurasi terbaik dalam penelitian ini.

3.2 Analisis Kinerja Non Optimizer

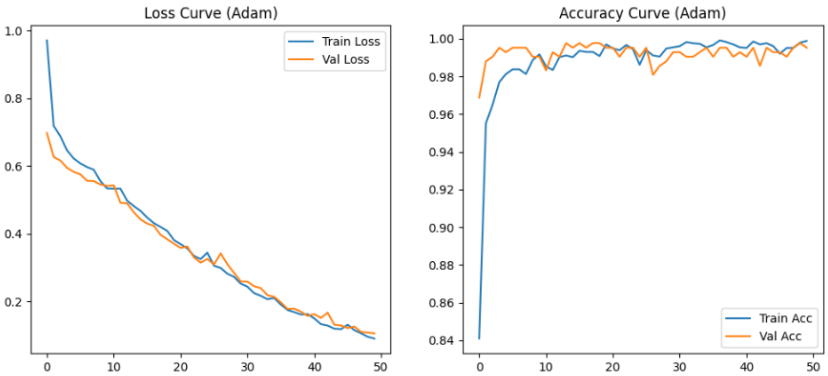
Grafik tanpa penggunaan optimizer tambahan (*non-optimizer*) terlihat bahwa proses pelatihan model mengalami peningkatan performa secara bertahap namun tidak secepat penggunaan optimizer tertentu. Pada fase awal (epoch 0–10), nilai *accuracy* meningkat cukup signifikan dari sekitar 45% hingga mendekati 85%–90%, sementara nilai *loss* mengalami penurunan tajam pada data pelatihan dan validasi dengan selisih yang relatif kecil. Memasuki epoch 10–30, peningkatan *accuracy* masih terjadi namun mulai melambat dan cenderung stabil di kisaran 93%–95%, sedangkan nilai *loss* menurun secara bertahap hingga berada di sekitar 0,7. Pada fase akhir pelatihan (epoch 30–50), kurva *training accuracy* dan *validation accuracy* terlihat semakin mendekati dan stabil, namun masih terdapat sedikit perbedaan di antara keduanya. Sementara itu, kurva *training loss* dan *validation loss* terus menurun secara perlahan dengan pola yang cukup serupa.



Gambar 4. Grafik Loss dan Accuracy Non Optimizer

3.3 Analisis Kinerja Optimizer Adam

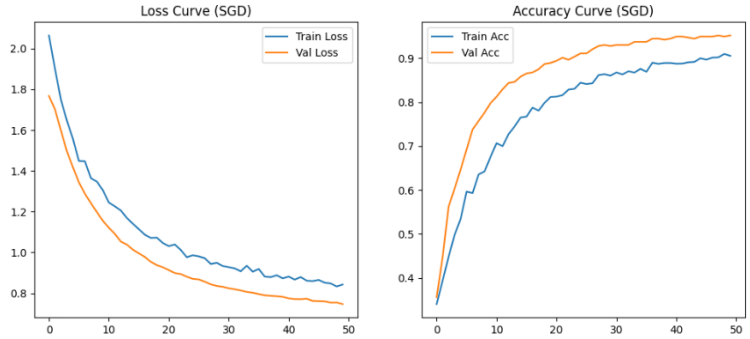
Grafik dengan optimizer Adam terlihat bahwa proses pelatihan model mengalami peningkatan performa secara bertahap sepanjang epoch, dimana pada fase awal (epoch 0–10) nilai *accuracy* meningkat cukup cepat dari sekitar 85% hingga mendekati 97%, sementara nilai *loss* mengalami penurunan yang signifikan pada data pelatihan dan validasi dengan selisih yang relatif kecil. Memasuki epoch 10–30, peningkatan *accuracy* mulai melambat dan cenderung stabil di kisaran 98% hingga 99%, sedangkan nilai *loss* terus menurun hingga berada di bawah 0,2. Pada fase akhir pelatihan (epoch 30–50), kurva *training accuracy* dan *validation accuracy* tampak stabil dan saling berdekatan, begitu pula dengan *training loss* dan *validation loss* yang tidak menunjukkan perbedaan yang signifikan.



Gambar 5. Grafik Loss dan Accuracy Optimizer Adam

3.4 Analisis Kinerja Optimizer SGD

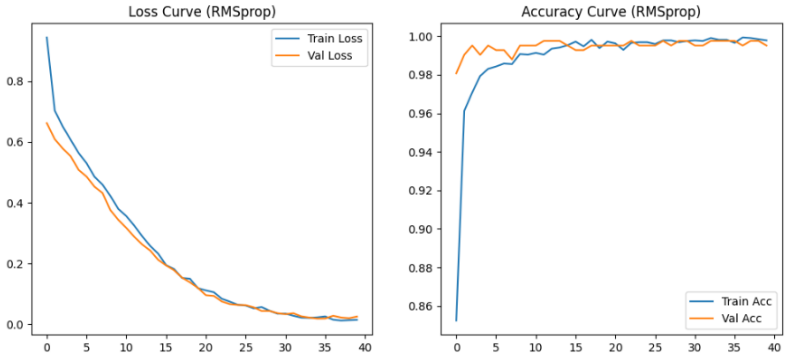
Grafik dengan optimizer SGD terlihat bahwa proses pelatihan model mengalami peningkatan performa secara bertahap, namun cenderung lebih lambat dibandingkan optimizer lainnya. Pada fase awal (epoch 0–10), nilai *accuracy* meningkat dari sekitar 35% hingga mendekati 75%, sementara nilai *loss* mengalami penurunan yang cukup signifikan meskipun masih berada pada nilai yang relatif tinggi. Memasuki epoch 10–30, peningkatan *accuracy* terus berlangsung secara perlahan hingga mencapai kisaran 85%–90%, sedangkan nilai *loss* menurun secara bertahap namun tidak secepat fase awal. Pada fase akhir pelatihan (epoch 30–50), kurva *training accuracy* dan *validation accuracy* mulai mendekati kondisi stabil, namun masih terdapat jarak di antara keduanya. Selain itu, pada kurva *loss*, nilai *training loss* dan *validation loss* terus menurun tetapi dengan selisih yang masih terlihat, menunjukkan proses konvergensi yang lebih lambat selama pelatihan.



Gambar 6. Grafik Loss dan Accuracy Optimizer SGD

3.5 Analisis Kinerja Optimizer RMSProp

Grafik dengan optimizer RMSProp terlihat bahwa proses pelatihan model berlangsung dengan cukup cepat dan stabil. Pada fase awal (epoch 0–5), nilai *accuracy* meningkat secara signifikan dari sekitar 85% hingga mendekati 98%, sementara nilai *loss* mengalami penurunan tajam baik pada data pelatihan maupun validasi dengan selisih yang relatif kecil. Memasuki epoch 5–20, peningkatan *accuracy* masih terjadi namun mulai melambat dan mendekati nilai maksimum di kisaran 99%, sedangkan nilai *loss* terus menurun hingga berada di bawah 0,1. Pada fase akhir pelatihan (epoch 20–40), kurva *training accuracy* dan *validation accuracy* terlihat sangat stabil dan saling berdekatan, begitu pula dengan *training loss* dan *validation loss* yang hampir berhimpit.



Gambar 7. Grafik Loss dan Accuracy Optimizer RMSProp

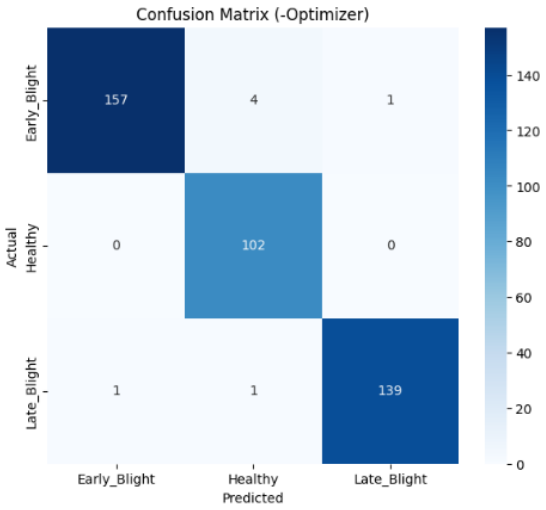
3.6 Hasil Evaluasi Pengujian Semua Optimizer

Model yang dirancang menghasilkan evaluasi berupa *confusion matrix* yang mencakup metrik *accuracy*, *precision*, *recall*, dan *F1-score*. Skenario non optimizer menghasilkan *accuracy* sebesar 98% dengan nilai metrik yang relatif tinggi pada setiap kelas. Pada penggunaan optimizer Adam, model menunjukkan performa terbaik dengan *accuracy* mencapai 100% serta nilai *precision*, *recall*, dan *F1-score* yang mendekati sempurna di seluruh kelas. Sementara itu, penggunaan optimizer SGD menghasilkan penurunan performa dengan *accuracy* sebesar 94%, di mana nilai *recall* dan *F1-score* pada beberapa kelas belum sebaik optimizer lainnya. Optimizer RMSProp juga memberikan hasil yang baik dengan *accuracy* mencapai 99% serta nilai metrik

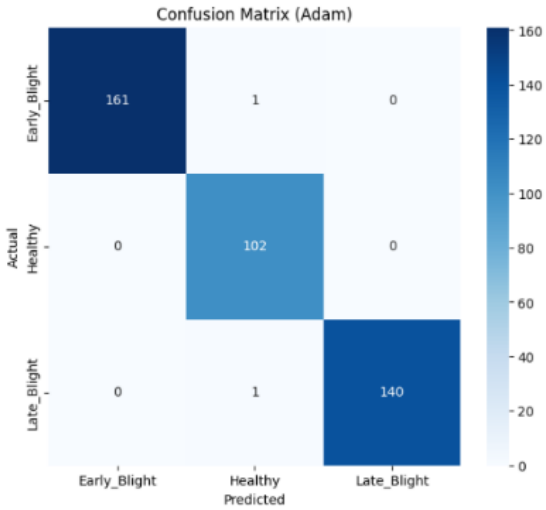
yang stabil pada setiap kelas. Hasil evaluasi ini mengindikasikan bahwa pemilihan optimizer berpengaruh signifikan terhadap performa model, dengan optimizer Adam memberikan hasil terbaik dibandingkan metode lainnya sebagaimana ditunjukkan pada Tabel 6.

Tabel 6. Hasil Evaluasi Pengujian Semua Optimizer

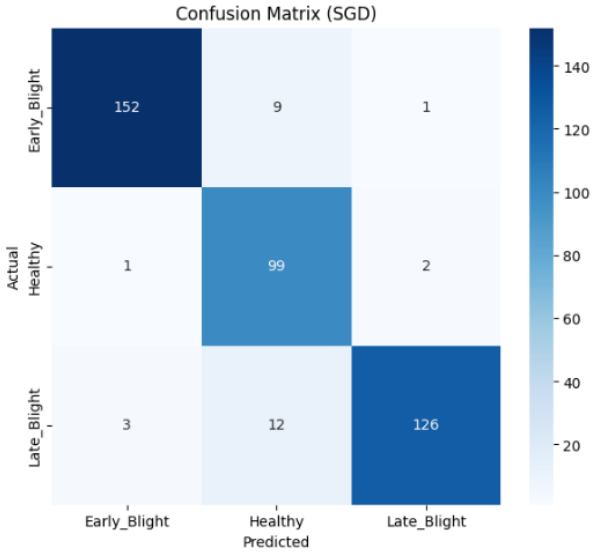
Skenario	Kategori	Presicion	Recall	F1-Score	Accuracy
Non Optimizer	Daun Sehat	95%	100%	98%	98%
	Daun Early Blight	99%	97%	98%	
	Daub Late Blight	99%	99%	99%	
Adam	Daun Sehat	98%	100%	99%	100%
	Daun Early Blight	100%	99%	100%	
	Daub Late Blight	100%	99%	100%	
SGD	Daun Sehat	83%	97%	90%	94%
	Daun Early Blight	98%	94%	96%	
	Daub Late Blight	98%	91%	94%	
RMSprop	Daun Sehat	98%	99%	99%	99%
	Daun Early Blight	99%	99%	99%	
	Daub Late Blight	100%	100%	100%	



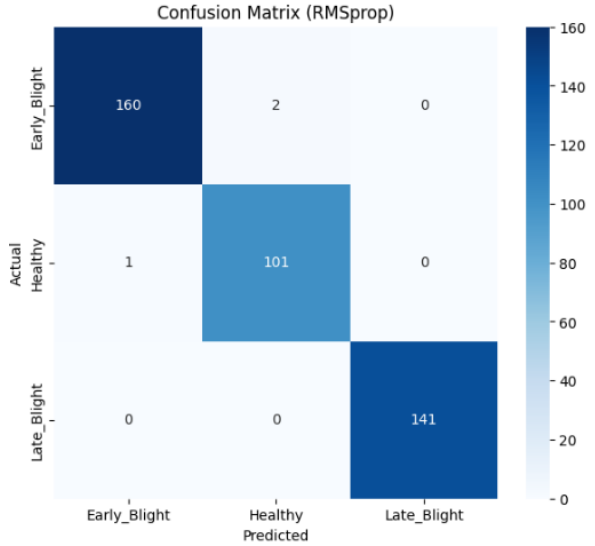
Gambar 8. Confusion matrix hasil pengujian model Menggunakan Non Optimizer



Gambar 9. Confusion matrix hasil pengujian model Menggunakan Optimizer Adam



Gambar 10. Confusion matrix hasil pengujian model Menggunakan Optimizer SGD



Gambar 11. Confusion matrix hasil pengujian model Menggunakan Optimizer RMSPProp

Berdasarkan *confusion matrix* pada Gambar 8-11, dapat disimpulkan bahwa penggunaan optimizer Adam dan RMSProp mampu mengidentifikasikan setiap kelas dengan baik serta menghasilkan tingkat kesalahan yang relatif rendah dibandingkan dengan skenario tanpa optimizer maupun penggunaan SGD. Pada optimizer Adam, hampir seluruh data berhasil diidentifikasikan dengan benar, dengan hanya sedikit kesalahan, seperti pada kelas *Early Blight* yang diprediksi sebagai *Healthy* sebanyak satu data serta pada kelas *Late Blight* yang juga memiliki satu kesalahan prediksi. RMSProp menunjukkan performa yang sebanding dengan tingkat kesalahan yang sangat kecil, seperti dua data *Early Blight* yang diprediksi sebagai *Healthy* dan satu data *Healthy* yang diprediksi sebagai *Early Blight*. Berbeda dengan kedua metode tersebut, skenario tanpa optimizer masih menghasilkan beberapa kesalahan, terutama pada kelas *Early Blight* dan *Late Blight*, sedangkan penggunaan SGD menunjukkan jumlah kesalahan yang lebih tinggi, khususnya pada kelas *Late Blight* yang cukup sering diidentifikasi sebagai *Healthy*. Temuan ini mengindikasikan bahwa pemilihan *hyperparameter*, terutama jenis optimizer, memberikan pengaruh signifikan terhadap performa model dibandingkan konfigurasi lainnya.

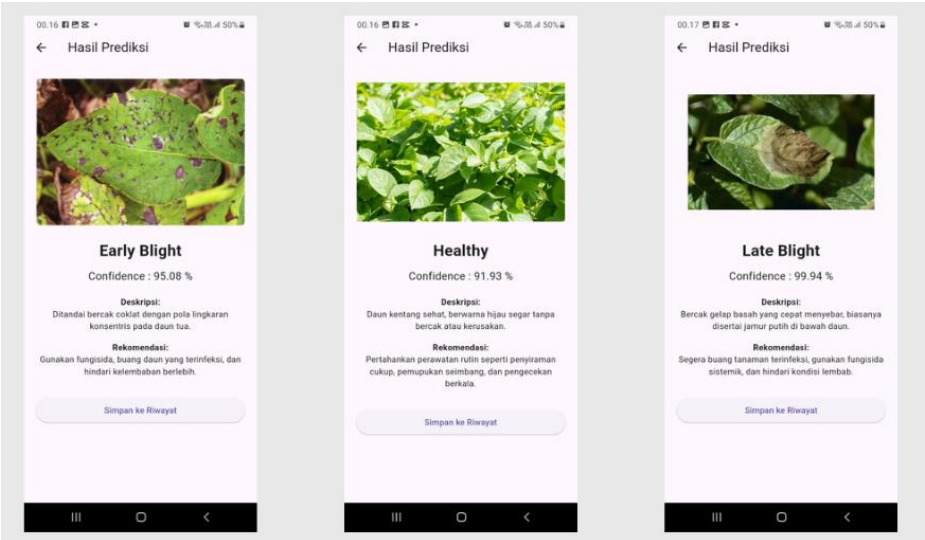
3.7 Evaluasi Model pada Data Uji

Pada hasil data uji ditampilkan beberapa contoh citra daun yang telah diidentifikasikan oleh model, di mana tulisan berwarna hijau menunjukkan bahwa hasil prediksi sesuai dengan label asli. Berdasarkan sampel yang ditampilkan, model mampu mengidentifikasikan citra dengan baik pada tiga kategori, yaitu *Early Blight*, *Healthy*, dan *Late Blight*. Model berhasil mengenali karakteristik visual masing-masing kelas, seperti bercak pada *Early Blight*, kondisi daun tanpa gejala pada *Healthy*, serta kerusakan yang lebih parah pada *Late Blight*. Kesesuaian antara label asli dan hasil prediksi pada seluruh sampel mengindikasikan bahwa model memiliki kinerja yang baik pada data uji serta mampu melakukan generalisasi secara efektif terhadap data yang belum pernah dilatih, sebagaimana ditunjukkan pada Gambar 12.

Gambar 12. Prediksi Dataset Uji

3.8 Implementasi dan Hasil Prediksi pada Aplikasi

Pengujian dilakukan dengan menggunakan citra daun yang diperoleh dari dunia nyata untuk mengevaluasi kemampuan model dalam mengidentifikasikan data baru di luar dataset pelatihan maupun pengujian sebelumnya. Pada tampilan aplikasi, sistem dapat menampilkan hasil prediksi dalam bentuk kategori penyakit, yaitu *Early Blight*, *Healthy*, dan *Late Blight*, yang dilengkapi dengan nilai *confidence* sebagai indikator tingkat keyakinan model terhadap hasil prediksi yang dihasilkan.



Gambar 13. Hasil Test Pada Aplikasi

Berdasarkan hasil yang ditunjukkan pada Gambar 13, setiap prediksi dilengkapi dengan informasi tambahan berupa deskripsi kondisi daun serta rekomendasi penanganan yang sesuai. Model mampu mengidentifikasikan citra daun dengan baik, ditunjukkan oleh tingkat keyakinan sebesar 95,08% pada kelas *Early Blight*, 91,93% pada kelas *Healthy*, dan 99,94% pada kelas *Late Blight*. Hasil tersebut mengindikasikan bahwa model memiliki kemampuan yang cukup baik saat mengenali pola visual dari masing-masing penyakit meskipun data berasal dari sumber eksternal. Integrasi model ke dalam aplikasi *mobile* juga menunjukkan bahwa sistem dapat

digunakan secara praktis sebagai alat bantu identifikasi penyakit daun kentang secara yang mudah digunakan oleh pengguna.

3.9 Pembahasan

Hasil penelitian mengindikasikan bahwa model yang dirancang dengan arsitektur EfficientNetB3 serta optimizer Adam mampu memberikan kinerja yang optimal dalam mengidentifikasikan penyakit daun kentang. Hal ini tercermin dari nilai *accuracy* yang tinggi pada data pelatihan, validasi, dan pengujian, di mana *test accuracy* mencapai 99,50% serta didukung oleh nilai *precision* sebesar 1,00, *recall* sebesar 1,00, dan *F1-score* sebesar 1,00. Setiap kelas, yaitu *Early Blight*, *Healthy*, dan *Late Blight*, juga menunjukkan hasil evaluasi yang konsisten. Secara keseluruhan, model mampu menghasilkan prediksi yang akurat serta memiliki keseimbangan yang baik dalam mengenali seluruh kategori. Hasil tersebut konsisten dengan penelitian yang telah dilaporkan oleh Meslmani dkk. [18] yang menyatakan bahwa arsitektur EfficientNetB3 dengan optimizer Adam memberikan performa yang tinggi dalam identifikasi penyakit daun kentang.

Tingginya performa model dapat dijelaskan melalui karakteristik arsitektur EfficientNetB3 yang memiliki kemampuan ekstraksi fitur visual secara efisien. Salah satu keunggulan EfficientNet adalah penerapan metode *compound scaling*, yaitu pendekatan yang meningkatkan dimensi model melalui penyesuaian kedalaman jaringan, lebar jaringan, serta resolusi citra secara seimbang [27]. Keseimbangan tersebut membuat model mampu mengenali ciri visual penting pada citra daun, seperti pola bercak, perubahan warna, tekstur permukaan daun, dan tingkat kerusakan jaringan daun. Pada kasus penyakit daun kentang, perbedaan antara daun sehat, *Early Blight*, dan *Late Blight* sering kali terletak pada detail visual yang halus. Oleh sebab itu, kemampuan model dalam mengekstraksi fitur visual menjadi faktor penting yang menentukan keberhasilan identifikasi. Hasil penelitian ini memperkuat konsep bahwa arsitektur CNN modern mampu mengenali pola citra secara otomatis melalui proses pembelajaran berlapis, mulai dari fitur sederhana seperti tepi dan warna hingga fitur kompleks seperti pola penyakit pada daun.

Selain faktor arsitektur, penggunaan *transfer learning* juga berperan dalam meningkatkan performa model. Pada penelitian ini, EfficientNetB3 menggunakan parameter awal yang berasal dari model pralatih (pre-trained model) menggunakan dataset ImageNet., kemudian disesuaikan kembali melalui proses pelatihan pada dataset daun kentang. Strategi ini membuat model tidak memulai proses pembelajaran dari nol, melainkan memanfaatkan pengetahuan visual awal yang telah terbentuk sebelumnya. Pendekatan seperti ini efektif digunakan ketika jumlah dataset spesifik relatif terbatas, karena model tetap mampu memperoleh representasi fitur yang kuat dengan proses pelatihan yang lebih efisien [21], [27]. Penelitian ini menggunkan *transfer learning* membantu model mengenali pola penyakit daun kentang secara lebih cepat dan stabil, terutama karena dataset yang digunakan hanya terdiri atas tiga kelas utama. Dengan demikian, hasil akurasi tinggi yang diperoleh tidak hanya disebabkan oleh jumlah epoch pelatihan, tetapi juga oleh pemanfaatan pengetahuan visual awal dari model pra-latih.

Perbandingan optimizer pada tabel 5 juga terbukti memberikan pengaruh penting terhadap performa model. Berdasarkan hasil perbandingan, optimizer Adam menghasilkan *test accuracy* tertinggi sebesar 99,50%, disusul RMSProp sebesar 99,25%, skenario tanpa optimizer tambahan sebesar 98,27%, dan SGD sebesar 93,58%. Kinerja Adam yang lebih baik dapat dijelaskan karena Adam menggunakan estimasi adaptif terhadap momen pertama dan kedua dari gradien, sehingga proses pembaruan bobot menjadi lebih stabil dan efisien [28]. Pada data citra, terutama citra daun dengan variasi warna, tekstur, dan pola bercak, stabilitas pembaruan bobot sangat penting agar model tidak mudah terjebak pada proses konvergensi yang lambat. Sebaliknya, SGD cenderung membutuhkan pengaturan *learning rate* yang lebih sensitif dan proses pelatihan yang lebih panjang. Hal ini menjelaskan mengapa model dengan SGD menunjukkan akurasi lebih rendah serta belum mencapai konvergensi optimal hingga akhir pelatihan.

Berdasarkan grafik *training* dan *validation* pada Gambar 4-7 setiap skenario juga menunjukkan bahwa model dengan optimizer Adam memiliki pola pembelajaran yang lebih stabil. Kurva *accuracy* meningkat secara cepat pada awal epoch dan kemudian cenderung stabil pada epoch berikutnya, sedangkan nilai *loss* mengalami penurunan bertahap tanpa perbedaan mencolok antara data pelatihan dan validasi. Kondisi ini mengindikasikan bahwa model tidak mengalami *overfitting* secara berarti. Penerapan *data augmentation*, *dropout*, *batch normalization*, *early stopping*, dan *ReduceLROnPlateau* turut berkontribusi dalam menjaga

kemampuan generalisasi model. Data *augmentation* dapat memperluas variasi citra pelatihan melalui transformasi seperti rotasi, pergeseran, zoom, dan perubahan kecerahan, sehingga model menjadi lebih tangguh terhadap variasi citra [29]. Sementara itu, *dropout* dan *batch normalization* membantu mengurangi ketergantungan model pada pola tertentu yang terlalu spesifik terhadap data pelatihan. Kombinasi teknik tersebut menjadikan model lebih mampu mengenali terhadap citra baru yang tidak termasuk dalam data pelatihan sebelumnya.

Meskipun hasil model sangat tinggi, performa identifikasi belum sepenuhnya mencapai 100% pada seluruh pengujian. Berdasarkan confusion matrix, masih ditemukan beberapa kesalahan identifikasi, misalnya citra *Early Blight* yang diidentifikasi sebagai *Healthy*, serta citra *Late Blight* yang mengalami kesalahan identifikasi. Kesalahan ini dapat terjadi karena beberapa faktor. Pertama, gejala awal penyakit pada daun kentang dapat memiliki kemiripan visual dengan kondisi daun sehat, terutama apabila bercak penyakit masih kecil atau belum menyebar luas. Kedua, terdapat kemiripan warna dan tekstur antara *Early Blight* dan *Late Blight*, khususnya pada citra dengan pencahayaan kurang merata. Ketiga, variasi latar belakang, sudut pengambilan gambar, kualitas kamera, dan tingkat fokus citra dapat memengaruhi kemampuan model dalam mengekstraksi fitur penyakit. Dalam kajian identifikasi penyakit tanaman berbasis citra, variasi kondisi lapangan seperti pencahayaan, latar belakang, dan kualitas citra memang sering menjadi faktor yang menyebabkan penurunan performa model ketika diuji pada data dunia nyata [30]. Oleh karena itu, kesalahan prediksi yang masih muncul tidak menunjukkan kegagalan model, tetapi menunjukkan adanya batas generalisasi yang perlu diperkuat melalui dataset yang lebih beragam.

Apabila dibandingkan dengan penelitian sebelumnya, hasil yang diperoleh pada penelitian ini mendukung sekaligus memperluas temuan sebelumnya. Penelitian Meslmani dkk. [18] menunjukkan bahwa EfficientNetB3 mampu mencapai akurasi 92% dalam identifikasi penyakit daun kentang. Hasil penelitian ini menunjukkan akurasi yang lebih tinggi, yaitu 99,50%, sehingga memperkuat bukti bahwa EfficientNetB3 efektif digunakan untuk klasifikasi penyakit daun kentang. Penelitian Firasari dan Cahyanti [16] menggunakan InceptionResNetV2 dan memperoleh akurasi sebesar 95,30%, tetapi masih mengalami kendala dalam membedakan kelas tertentu. Penelitian Alhammad dkk. [17] memperoleh akurasi tinggi melalui VGG16 dan pendekatan *Explainable AI*, tetapi model yang digunakan relatif kompleks dan belum sepenuhnya diarahkan pada implementasi mobile. Dengan demikian, penelitian ini memberikan perluasan dari penelitian sebelumnya karena tidak hanya mengejar akurasi model, tetapi juga mengintegrasikan model ke dalam aplikasi mobile berbasis Flutter melalui TensorFlow Lite.

Hasil penelitian ini juga sejalan dengan temuan Tadele dkk. [31], yang menunjukkan bahwa *deep learning* dapat digunakan secara efektif untuk deteksi penyakit tanaman berbasis citra. Namun, penelitian Mulugeta dkk. juga menegaskan bahwa performa tinggi pada dataset terkontrol belum tentu selalu sama ketika model diterapkan pada kondisi lapangan yang lebih kompleks. Hal tersebut relevan dengan penelitian ini, karena meskipun model telah diuji menggunakan citra dunia nyata pada aplikasi mobile, jumlah data lapangan yang digunakan masih terbatas. Oleh karena itu, hasil implementasi mobile dalam penelitian ini dapat dipandang sebagai tahap awal menuju sistem identifikasi lapangan, tetapi masih memerlukan pengujian lebih luas pada berbagai kondisi lingkungan, varietas tanaman, jenis kamera, dan tingkat keparahan penyakit.

Implementasi model ke dalam aplikasi mobile menunjukkan bahwa hasil penelitian tidak hanya memiliki nilai komputasional, tetapi juga memiliki nilai praktis. Model yang berhasil dikonversi ke format TensorFlow Lite dapat digunakaan pada perangkat mobile sehingga proses identifikasi tidak hanya bergantung pada komputer dengan spesifikasi tinggi. TensorFlow Lite dikembangkan agar model deep learning dapat dijalankan secara efisien pada perangkat mobile maupun perangkat edge yang memiliki sumber daya komputasi terbatas [32]. Dalam penelitian ini, aplikasi mampu menampilkan hasil prediksi beserta nilai *confidence*, seperti 95,08% pada kelas *Early Blight*, 91,93% pada kelas *Healthy*, dan 99,94% pada kelas *Late Blight*. Nilai *confidence* tersebut memberikan informasi tambahan kepada pengguna mengenai tingkat keyakinan model terhadap hasil prediksi. Hal ini penting karena dalam penggunaan lapangan, sistem identifikasi penyakit tanaman tidak hanya perlu memberikan label penyakit, tetapi juga perlu memberikan indikasi tingkat kepercayaan agar pengguna dapat mempertimbangkan tindakan lanjutan secara lebih bijak.

Kontribusi penelitian ini terletak pada pengembangan sistem identifikasi penyakit daun kentang yang menggabungkan aspek akurasi, efisiensi model, dan kemudahan implementasi.

Dari sisi pengembangan ilmu pengetahuan, penelitian ini memperkuat bukti bahwa arsitektur EfficientNetB3 berbasis *transfer learning* dapat digunakan secara efektif untuk klasifikasi citra penyakit tanaman. Dari sisi metodologis, penelitian ini menunjukkan bahwa pemilihan optimizer berpengaruh terhadap performa model, dengan Adam memberikan hasil paling stabil dibandingkan SGD dan RMSProp. Dari sisi praktis, penelitian ini menghasilkan aplikasi mobile yang dapat membantu proses identifikasi penyakit daun kentang secara lebih cepat dan mudah. Dengan demikian, penelitian ini memberikan kontribusi pada bidang *computer vision* pertanian, khususnya dalam pengembangan sistem deteksi penyakit tanaman yang lebih aplikatif dan mendekati kebutuhan pengguna lapangan.

Penelitian ini masih memiliki beberapa keterbatasan. Pertama, dataset yang digunakan hanya mencakup tiga kelas, yaitu *Early Blight*, *Healthy*, dan *Late Blight*, sehingga sistem belum mampu mengidentifikasi jenis penyakit daun kentang lain atau gangguan tanaman yang memiliki gejala serupa. Kedua, dataset utama masih berasal dari sumber publik, sehingga variasi citra belum sepenuhnya merepresentasikan kondisi lapangan yang kompleks. Ketiga, pengujian aplikasi pada data dunia nyata masih terbatas pada beberapa sampel citra, sehingga belum dapat menggambarkan performa sistem secara menyeluruh pada berbagai kondisi pencahayaan, jenis kamera, sudut pengambilan gambar, dan latar belakang. Keempat, penelitian ini belum mengukur secara rinci aspek efisiensi aplikasi seperti ukuran model, waktu inferensi, penggunaan memori, konsumsi daya, dan respons aplikasi pada berbagai spesifikasi perangkat mobile. Padahal, aspek tersebut penting untuk menilai kelayakan sistem pada perangkat pengguna dengan kemampuan komputasi yang berbeda-beda.

Berdasarkan keterbatasan tersebut, penelitian selanjutnya dapat diarahkan pada beberapa pengembangan. Pertama, dataset perlu diperluas dengan menambahkan citra dari kondisi lapangan secara langsung, termasuk variasi pencahayaan, latar belakang alami, usia daun, tingkat keparahan penyakit, dan varietas kentang yang berbeda. Kedua, jumlah kelas penyakit dapat ditambah agar sistem mampu mengenali penyakit atau gangguan tanaman lain yang relevan dalam budidaya kentang. Ketiga, model dapat dikembangkan dengan pendekatan optimasi lanjutan seperti *quantization*, *pruning*, atau penggunaan arsitektur yang lebih ringan agar performa mobile menjadi lebih efisien. Keempat, penelitian berikutnya dapat menambahkan metode *Explainable AI* seperti Grad-CAM untuk menampilkan bagian daun yang menjadi dasar keputusan model [33]. Dengan adanya visualisasi tersebut, pengguna dapat lebih memahami alasan model memberikan prediksi tertentu, sehingga kepercayaan terhadap sistem dapat meningkat. Kelima, pengujian lapangan perlu dilakukan dengan melibatkan petani atau penyuluh pertanian agar sistem dapat dinilai tidak hanya dari sisi akurasi teknis, tetapi juga dari sisi kemudahan penggunaan, kebermanfaatan, dan kesiapan implementasi pada praktik budidaya kentang.

5. Simpulan

Model yang dikembangkan dengan arsitektur EffcientNetB3 serta optimizer Adam memberikan kinerja paling optimal dalam identifikasi penyakit daun kentang dibandingkan metode lainnya. Model mampu mencapai *test accuracy* sebesar 99,50%, serta didukung oleh nilai *precision* sebesar 1,00, *recall* sebesar 1,00, dan *F1-score* sebesar 1,00 yang menunjukkan hasil identifikasi yang konsisten pada seluruh kelas. Implementasi model ke dalam aplikasi *mobile* mengindikasikan bahwa sistem dapat dimanfaatkan secara praktis sebagai alat bantu identifikasi penyakit daun kentang yang akurat dan mudah digunakan oleh petani di lapangan atau dunia nyata. Penelitian selanjutnya disarankan untuk menggunakan dataset dengan jumlah kelas yang lebih banyak agar model mampu mengidentifikasi variasi penyakit daun yang lebih beragam, sehingga meningkatkan kemampuan generalisasi dan cakupan identifikasi sistem.

Daftar Referensi

[1] N. Shabrina *et al.*, “A Novel Dataset of Potato Leaf Disease in Uncontrolled Environment,” *Data Brief*, vol. 52, p. 109955, Dec. 2023, doi: 10.1016/j.dib.2023.109955.

[2] K. Prasetyo and R. Mahenra, “Analisis Kinerja Convolutional Neural Networks Baseline untuk Identifikasi Jenis Jenis Penyakit Kentang: Performance Analysis of Baseline Convolutional Neural Networks for Identifying Potato Disease Types,” *MALCOM: Indonesian Journal of Machine Learning and Computer Science*, vol. 5, no. 2, pp. 609–615, Mar. 2025, doi: 10.57152/malcom.v5i2.1722.

[3] N. R. Vicky Yanto, “Implementasi CBAM pada Arsitektur ResNet50 dalam Klasifikasi Penyakit Daun Tanaman Kentang,” *Jutisi: Jurnal Ilmiah Teknik Informatika dan Sistem Informasi*, vol. 15, no. 1, pp. 39–49, Feb. 2026, doi: http://dx.doi.org/10.35889/jutisi.v15i1.3411.

[4] S. A. Salihu *et al.*, “Detection and Classification of Potato Leaves Diseases Using Convolutional Neural Network and Adam Optimizer,” in *Procedia Computer Science*, Elsevier B.V., 2025, pp. 2–17. doi: 10.1016/j.procs.2025.04.159.

[5] S. Singla and R. Gupta, “Enhanced Image Classification Performance on a Potato-Tomato Dataset using EfficientNetB3,” in *2024 3rd International Conference for Advancement in Technology (ICONAT)*, 2024, pp. 1–6. doi: 10.1109/ICONAT61936.2024.10775274.

[6] R. A. Sholihati, I. A. Sulistijono, A. Risnumawan, and E. Kusumawati, “Potato Leaf Disease Classification Using Deep Learning Approach,” in *IES 2020 - International Electronics Symposium: The Role of Autonomous and Intelligent Systems for Human Life and Comfort*, Institute of Electrical and Electronics Engineers Inc., Sep. 2020, pp. 392–397. doi: 10.1109/IES50839.2020.9231784.

[7] G. Sangar and V. Rajasekar, “Optimized classification of potato leaf disease using EfficientNet-LITE and KE-SVM in diverse environments,” *Front. Plant Sci.*, vol. 16, p. 1499909, May 2025, doi: 10.3389/fpls.2025.1499909.

[8] V. J. A. Davincylin and D. Hermanto, “Klasifikasi Tingkat Kematangan Buah Kelapa Sawit Menggunakan EfficientNet-B7,” *Jutisi : Jurnal Ilmiah Teknik Informatika dan Sistem Informasi*, vol. 14, no. 3, p. 2133, Dec. 2025, doi: 10.35889/jutisi.v14i3.3399.

[9] M. Arafat, S. Khatun, M. Halder, M. N. Sheikh, M. N. Adnan, and S. Kabir, “An Efficient Convolution Neural Network-based Novel Framework for Potato Leaf Diseases Classification and Identification,” vol. 52, pp. 2411–2428, Jul. 2025.

[10] T.-Y. Lee, I.-A. Lin, J.-Y. Yu, J. Yang, and Y.-C. Chang, “High Efficiency Disease Detection for Potato Leaf with Convolutional Neural Network,” *SN Comput. Sci.*, vol. 2, p. 297, Jul. 2021, doi: 10.1007/s42979-021-00691-9.

[11] M. I. Rasyid and L. M. Wisudawati, “Klasifikasi Hama Ulat Pada Citra Daun Sawi Berbasis Convolutional Neural Network Dengan Model Xception,” *Jutisi: Jurnal Ilmiah Teknik Informatika dan Sistem Informasi*, vol. 13, pp. 870–880, Aug. 2024, Accessed: Apr. 20, 2026. [Online]. Available: https://ojs.stmik-banjarbaru.ac.id/index.php/jutisi/article/view/3399

[12] A. Raza, A. H. Pitafi, M. K. Shaikh, and K. Ahmed, “RETRACTED: Optimizing potato leaf disease recognition: Insights DENSE-NET-121 and Gaussian elimination filter fusion,” *Heliyon*, vol. 11, no. 3, p. e42318, 2025, doi: https://doi.org/10.1016/j.heliyon.2025.e42318.

[13] Muhammad Mahtab, Zainab Sadiq, Muhammad Raoof, and Sohail Masood Bhatti, “Enhancing Heart Disease Detection in Echocardiogram Images Using Optimized EfficientNetB3 Architecture,” *Journal of Computing & Biomedical Informatics*, vol. 7, no. 02, pp. 10–15, Sep. 2024, [Online]. Available: https://jcbi.org/index.php/Main/article/view/568

[14] J. Sinamenye, A. Chatterjee, and R. Shrestha, “Potato plant disease detection: leveraging hybrid deep learning models,” *BMC Plant Biol.*, vol. 25, no. 1, p. 647, May 2025, doi: 10.1186/s12870-025-06679-4.

[15] J. Akther, A. A. Nayan, and M. Harun-Or-roshid, “Potato Leaves Blight Disease Recognition and Categorization Using Deep Learning,” *Engineering Journal*, vol. 27, no. 9, pp. 27–38, Sep. 2023, doi: 10.4186/ej.2023.27.9.27.

[16] E. Firasari and F. L. D. Cahyanti, “Classification Of Potato Leaf Diseases Using Convolutional Neural Network,” *Jurnal Techno Nusa Mandiri*, vol. 20, no. 2, pp. 89–94, Oct. 2023, doi: 10.33480/techno.v20i2.4655.

[17] S. Alhammad, D. Khafaga, W. Elhady, F. Samy, and K. Hosny, “Deep learning and explainable AI for classification of potato leaf diseases,” *Front. Artif. Intell.*, vol. 7, p. 1449329, Feb. 2025, doi: 10.3389/frai.2024.1449329.

[18] H. Meslmani, E. Q. Ali, and A. M. Hussein, “Optimized Deep Learning Framework for Potato Leaf Disease Classification Using EfficientNetB3,” *Information Sciences with Applications*, vol. 6, pp. 50–59, Jun. 2025, doi: 10.61356/j.iswa.2025.6600.

[19] A. D. Saputra, D. Hindarto, B. Rahman, and H. Santoso, “Comparison of Accuracy in Detecting Tomato Leaf Disease with GoogleNet VS EfficientNetB3,” *SinkrOn*, vol. 8, no. 2, pp. 647–656, Apr. 2023, doi: 10.33395/sinkron.v8i2.12218.

[20] A. Yasid, R. T. Wahyuningrum, A. T. Ni’Mah, and I. H. Ayani, “Rice Leaf Diseases Classification using Deep Learning Based on EfficientNetB3 Architecture with Transfer Learning,” in *2023 International Conference on Technology, Engineering, and Computing Applications (ICTECA)*, Semarang: IEEE, Dec. 2023, pp. 1–6. doi: 10.1109/ICTECA60133.2023.10490596.

[21] T. Ozcan and E. Polat, “BorB: A Novel Image Segmentation Technique for Improving Plant Disease Classification With Deep Learning Models,” *IEEE Access*, vol. 13, pp. 71822–71839, 2025, doi: 10.1109/ACCESS.2025.3563160.

[22] C. Y. Chang and C. C. Lai, “Potato Leaf Disease Detection Based on a Lightweight Deep Learning Model,” *Mach. Learn. Knowl. Extr.*, vol. 6, no. 4, pp. 2321–2335, Dec. 2024, doi: 10.3390/make6040114.

[23] U. S. D. B. G. G. D. J. Barman, “Comparative Assessment of Deep Learning to Detect the Leaf Diseases of Potato based on Data Augmentation,” Shillong, Meghalaya, India: IEEE, 2020, pp. 5175–5183. Accessed: Oct. 15, 2025. [Online]. Available: https://doi.org/10.1109/ComPE49325.2020.9200067

[24] H. Ghosh, I. S. Rahat, K. Shaik, S. Khasim, and M. Yesubabu, “Potato Leaf Disease Recognition and Prediction using Convolutional Neural Networks,” *EAI Endorsed Transactions on Scalable Information Systems*, vol. 10, no. 6, pp. 1–8, Sep. 2023, doi: 10.4108/eetsis.3937.

[25] M. Rivaldo and D. Udjulawa, “Performance Comparison of EfficientNetB0 in Potato Leaf Disease Classification with Adam and SGD,” *Brilliance: Research of Artificial Intelligence*, vol. 5, no. 2, pp. 1224–1231, Dec. 2025, doi: 10.47709/brilliance.v5i2.7482.

[26] R. Mahum *et al.*, “A novel framework for potato leaf disease detection using an efficient deep learning model,” *Human and Ecological Risk Assessment*, vol. 29, no. 2, pp. 303–326, 2023, doi: 10.1080/10807039.2022.2064814.

[27] P. Johri *et al.*, “Advanced deep transfer learning techniques for efficient detection of cotton plant diseases,” *Front. Plant Sci.*, vol. 15, p. 1441117, Dec. 2024, doi: 10.3389/fpls.2024.1441117.

[28] H. Pebrian and E. Hartati, “Potato Leaf Disease Classification Using MobileNetV3 Architecture With Adam and Stochastic Gradient Descent Optimizers,” *Green Intelligent Systems and Applications*, vol. 6, no. 1, pp. 81–95, Mar. 2026, doi: 10.53623/gisa.v6i1.1063.

[29] E. Hassan, M. Y. Shams, N. A. Hikal, and S. Elmougy, “The effect of choosing optimizer algorithms to improve computer vision tasks: a comparative study,” *Multimed. Tools Appl.*, vol. 82, no. 11, pp. 16591–16633, May 2023, doi: 10.1007/s11042-022-13820-0.

[30] T. Astuti, A. Umar, R. Wahyudi, and Z. Rifai, “Enhancing Potato Leaf Disease Detection: Implementation of Convolutional Vision Transformers with Synthetic Datasets from Stable Diffusion,” *JOIV : International Journal on Informatics Visualization*, vol. 8, no. 4, pp. 2054–2065, Dec. 2024, doi: 10.62527/joiv.8.4.2167.

[31] A. Tadele, W. Jifara, E. Bogale, T. Desiyo, and A. Mokonnen, “Early Detection and Classification of Potato Leaf Disease Using Convolutional Neural Networks,” *Applied Computational Intelligence and Soft Computing*, vol. 2025, pp. 1–16, Nov. 2025, doi: 10.1155/acis/7614841.

[32] J. Rashid, I. Khan, G. Ali, S. H. Almotiri, M. A. Alghamdi, and K. Masood, “Multi-level deep learning model for potato leaf disease recognition,” *Electronics (Switzerland)*, vol. 10, no. 17, p. 2064, Sep. 2021, doi: 10.3390/electronics10172064.

[33] S. U. Khan, A. Alsuhaibani, A. Alabduljabbar, F. Almarshad, Y. N. Altherwy, and T. Akram, “A review on automated plant disease detection: motivation, limitations, challenges, and recent advancements for future research,” May 01, 2025, *Springer International Publishing*. doi: 10.1007/s44443-025-00040-3.