Jurnal Teknologi Terpadu

https://journal.nurulfikri.ac.id/index.php/jtt
ISSN : 2477-0043 ISSN ONLINE : 2460-7908

IDENTIFIKASI PENYAKIT DAUN PADA TANAMAN *SOLANACEAE* DAN *ROSACEAE* MENGGUNAKAN *DEEP LEARNING*

Allan Bil Faqih[1], Donny Avianto[2]

[1,2] Informatika, Universitas Teknologi Yogyakarta
Yogyakarta, D.I Yogyakarta, Indonesia 55164
allanbilfaqih214@gmail.com, donny@uty.ac.id

Abstract

With a projected global population of 9.7 billion by 2050, agriculture faces significant challenges in ensuring food security. One major obstacle is plant diseases that reduce crop yields by 40% per year. Previous research is often limited to disease detection in a single plant species, thus poorly reflecting multi-species needs in real agricultural practices. This research aims to develop and evaluate deep learning-based plant disease detection system using Convolutional Neural Networks (CNN) applied to two plant families, Solanaceae and Rosaceae. The dataset used was PlantVillage, containing 54,306 leaf images in JPEG format downloaded from GitHub, with data outside two families discarded during pre-processing. Three deep learning models were tested: transfer learning with InceptionV3 architecture and two custom CNNs (DFE and LCNN). LCNN model showed the best performance with training, validation, and testing accuracies of 99%, 99%, and 95%, respectively. In contrast, InceptionV3 achieved 96% training, 98% validation, and 92% testing accuracy, while DFE with 86% training, 94% validation, and 82% testing accuracy. Confusion matrix analysis showed difficulty distinguishing between healthy potatoes and potatoes with late blight, as well as cedar apple rust. These results highlights importance of developing specific model architectures rather than complex models for accurate multi-crop disease detection.

Keywords: *Computer Vision, Convolutional Neural Networks, Deep Learning, Plant Disease Detection, Transfer Learning*

Abstrak

Dengan proyeksi populasi global sebesar 9,7 miliar pada tahun 2050, pertanian menghadapi tantangan signifikan dalam memastikan ketahanan pangan. Salah satu kendala utama adalah penyakit tanaman yang menurunkan hasil panen hingga 40% per tahun. Penelitian sebelumnya sering terbatas pada deteksi penyakit pada satu spesies tanaman, sehingga kurang mencerminkan kebutuhan multi-spesies dalam praktik pertanian nyata. Penelitian ini bertujuan untuk mengembangkan dan mengevaluasi sistem deteksi penyakit tanaman berbasis *deep learning* menggunakan *Convolutional Neural Networks* (CNN) yang diterapkan pada dua famili tanaman, *Solanaceae* dan *Rosaceae*. *Dataset* yang digunakan adalah PlantVillage, berisi 54.306 citra daun dalam format JPEG yang diunduh dari GitHub, dengan data di luar dua famili tersebut dibuang selama pra-pemrosesan. Tiga model *deep learning* diuji: *transfer learning* dengan arsitektur *InceptionV3* dan dua CNN kustom (DFE dan LCNN). Model LCNN menunjukkan kinerja terbaik dengan akurasi pelatihan, validasi, dan pengujian masing-masing sebesar 99%, 99%, dan 95%. Sebaliknya, InceptionV3 mencapai akurasi pelatihan 96%, validasi 98%, dan pengujian 92%, sedangkan DFE dengan 86% pelatihan, 94% validasi, dan 82% akurasi pengujian. Analisis *confusion matrix* menunjukkan kesulitan membedakan antara kentang sehat dan kentang dengan *late blight*, serta *cedar apple rust*. Hasil ini menyoroti pentingnya pengembangan arsitektur model spesifik dibandingkan model yang kompleks untuk deteksi penyakit *multi-crop* secara akurat.

Kata kunci: *Computer Vision, Convolutional Neural Networks, Deep Learning, Deteksi Penyakit Tanaman, Transfer Learning*

1. PENDAHULUAN

Dengan populasi global yang diproyeksikan mencapai 9,7 miliar pada tahun 2050, sektor pertanian menghadapi tantangan signifikan dalam memenuhi permintaan produk pangan yang terus meningkat [1]. Di antara masalah utama yang dihadapi adalah penyakit tanaman, yang memiliki dampak besar pada produksi pertanian [2]. Penyakit tanaman dapat muncul dalam berbagai bentuk, menunjukkan gejala seperti layu, perubahan warna, pertumbuhan terhambat, dan kematian tanaman [3]. Kondisi ini menyebabkan kerugian tahunan sekitar 20-40% dari

Diterima 25 Oktober 2024, Direvisi 8 November 2024, Diterima untuk publikasi 4 Desember 2024

hasil pertanian, yang sangat merugikan terutama bagi petani kecil [4].

Secara tradisional, identifikasi penyakit tanaman dilakukan melalui observasi langsung oleh petani atau ahli agronomi. Namun, metode ini memakan waktu, membutuhkan pengetahuan khusus, dan sering kali tidak akurat jika dilakukan oleh mereka yang kurang berpengalaman. Luasnya lahan pertanian dan beragamnya jenis penyakit membuat deteksi dini menjadi semakin sulit [5]. Dengan memanfaatkan teknologi *Artificial Intelligence* (AI), khususnya *Convolutional Neural Networks* (CNN), terbuka peluang untuk mendeteksi penyakit tanaman secara lebih efisien dan efektif [6]. Penelitian sebelumnya menunjukkan bahwa CNN dapat mengidentifikasi jenis daun tanaman dengan akurasi tinggi [7], dan telah berhasil diimplementasikan dalam deteksi penyakit pada berbagai jenis tanaman [8], [9], [10], [11], [12], [13].

Tantangan utama dalam implementasi CNN terletak pada pengelolaan *dataset* yang besar dan kebutuhan komputasi yang tinggi selama proses pelatihan [14]. *Transfer Learning* muncul sebagai solusi yang menjanjikan, memberikan kemungkinan untuk menggunakan model yang telah dilatih untuk kasus yang lebih spesifik [15]. *InceptionV3*, sebuah model CNN yang populer, dipilih dalam penelitian ini karena kemampuannya yang telah terbukti dalam mempercepat pelatihan dengan tetap mempertahankan akurasi tinggi [16]. Penelitian sebelumnya juga menunjukkan bahwa *InceptionV3* dapat mempertahankan akurasi yang baik bahkan dengan *dataset* yang lebih kecil [17], dan efektivitasnya dalam klasifikasi penyakit tanaman [11].

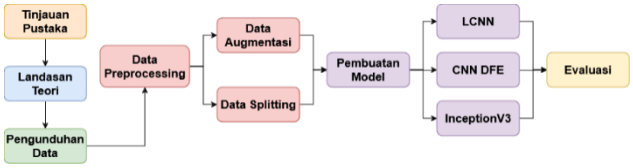
Penelitian ini menggunakan *dataset* PlantVillage, yang terdiri dari 54.306 citra daun yang memiliki format JPEG dari 14 spesies tanaman yang berbeda, termasuk 38 kategori daun yang sakit dan sehat [8]. Untuk penelitian ini, *dataset* tersebut dimodifikasi untuk fokus pada tanaman dari dua keluarga: *Solanaceae* (seperti paprika, tomat, dan kentang) dan *Rosaceae* (seperti apel, ceri, persik, dan stroberi).

Kontribusi utama penelitian ini adalah pengembangan dan mengevaluasi arsitektur *deep learning* untuk deteksi penyakit pada tanaman dari keluarga *Solanaceae* dan *Rosaceae*, dengan membandingkan dua pendekatan utama: *Transfer Learning* menggunakan *InceptionV3* dan CNN yang dirancang khusus. Berbeda dari penelitian sebelumnya yang umumnya terbatas pada deteksi penyakit untuk satu jenis tanaman saja [8], [9], [10], [11], [12], [13], melalui pendekatan multi-tanaman, penelitian ini tidak hanya menawarkan solusi praktis bagi petani yang umumnya menanam beragam jenis tanaman secara bersamaan, tetapi juga memberikan wawasan mendalam tentang keunggulan relatif dari setiap arsitektur melalui evaluasi dan perbandingan kedua pendekatan tersebut.

Hasil evaluasi diharapkan memberikan solusi yang lebih serbaguna untuk praktik pertanian dunia nyata, sekaligus mengatasi keterbatasan generalisasi dari penelitian-penelitian sebelumnya.

2. METODE PENELITIAN

Metode penelitian yang digunakan untuk mengembangkan sistem identifikasi penyakit tanaman menggunakan *deep learning* CNN akan dijelaskan dengan alur penelitian yang ditunjukkan pada Gambar 1 berikut.



Gambar 1. Diagram Alur Penelitian

Alur ini menggambarkan langkah-langkah utama dalam pengembangan sistem identifikasi penyakit tanaman. Proses dimulai dengan tinjauan pustaka untuk memahami penelitian sebelumnya dan metode yang digunakan, serta untuk mencari celah penelitian. Selanjutnya, pada bagian Landasan Teori, akan dibahas metode atau konsep yang diterapkan dalam penelitian ini, termasuk CNN, *Transfer Learning*, dan model *InceptionV3*. Setelah itu, data berupa citra daun yang menunjukkan berbagai gejala penyakit diunduh dari PlantVillage di GitHub.

Kemudian, data tersebut menjalani *preprocessing*, termasuk membagi *dataset* menjadi data pelatihan dan validasi, serta melakukan *data augmentation* untuk memastikan keragaman *dataset*. Tiga model dibuat: satu model berdasarkan arsitektur *InceptfionV3* dan dua model CNN kustom untuk perbandingan. Setelah pelatihan, model-model ini mengidentifikasikan penyakit tanaman dari citra daun.

Model-model tersebut dievaluasi menggunakan metrik kinerja seperti *confusion matrix* dan *classification report*, dan model akan disimpan untuk digunakan di luar lingkungan pelatihan.

2.1 Tinjauan Pustaka

Penelitian mengenai penggunaan CNN untuk klasifikasi berbagai objek telah banyak dilakukan oleh para peneliti. Ramadhani et al. melakukan penelitian tentang klasifikasi jenis tumbuhan berdasarkan citra daun menggunakan arsitektur CNN VGG-16. Penelitian tersebut menggunakan *dataset* sebanyak 2300 citra daun yang terbagi menjadi 23 kelas. Hasil penelitian menunjukkan tingkat akurasi mencapai 92,6% pada proses *training* dan 92% pada proses klasifikasi dengan pengujian 50 citra. Meskipun demikian, peneliti menyimpulkan bahwa efektivitas model dalam mengklasifikasi jenis tumbuhan masih kurang baik [7].

Implementasi CNN untuk klasifikasi penyakit pada daun berbagai tanaman telah dibuktikan efektivitasnya oleh beberapa penelitian. Henry et al. mengembangkan model CNN yang menjadi *state of the art* untuk klasifikasi

penyakit pada daun tomat dengan *dataset* sebanyak 18.162 citra dan mencapai akurasi *training* 94,06% dengan *loss function* 7,8% [8]. Sulistiyana dan Anardani memberikan kontribusi penting dengan membuktikan superioritas CNN (98%) dibanding SVM (87%) dalam deteksi penyakit jagung, menunjukkan bahwa CNN lebih efektif dalam mengekstrak fitur kompleks dari citra daun [10]. Rijal et al. juga menunjukkan efektivitas CNN dalam mengklasifikasikan penyakit pada Padi dengan tingkat akurasi validasi mencapai 80% [13].

Penggunaan *transfer learning* dalam implementasi CNN juga menunjukkan hasil yang menjanjikan. Setiawan et al. menggunakan arsitektur *ResNet152V2* untuk klasifikasi penyakit pada daun tomat dan berhasil mencapai akurasi 97% [11]. Hasil ini menunjukkan bahwa penggunaan *transfer learning* dapat meningkatkan efektivitas klasifikasi. Hal ini diperkuat oleh penelitian Vicky et al. yang menggunakan arsitektur *Inception* untuk mengidentifikasi penyakit pada daun alpukat, dimana penggunaan *transfer learning* membantu dalam proses identifikasi penyakit melalui citra digital dengan tingkat akurasi sebesar 80% [9]. Sementara itu, Pratama et al. menggunakan model *transfer learning* CNN-nya menghadirkan inovasi dengan mengatasi masalah *overfitting* menggunakan *dropout* pada klasifikasi penyakit daun pisang, mencapai akurasi 92% yang stabil bahkan dengan *dataset* terbatas [12].

Meskipun penelitian-penelitian tersebut menunjukkan hasil yang menjanjikan, terdapat beberapa keterbatasan signifikan. Pendekatan *single-crop* yang digunakan membatasi aplikasi praktis di lapangan. Model-model ini memerlukan pelatihan ulang untuk setiap jenis tanaman baru, sehingga kurang efisien untuk implementasi skala besar. Selain itu, kemampuan generalisasi model terhadap berbagai jenis tanaman belum teruji secara menyeluruh.

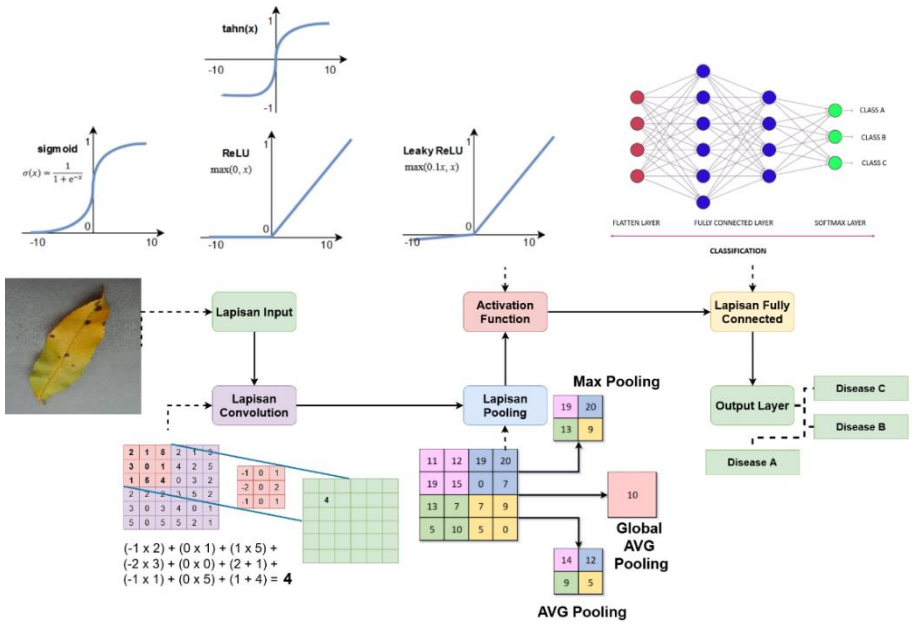
Berdasarkan analisis *state of the art* dan keterbatasan tersebut, penelitian ini mengusulkan penggunaan model yang dapat menangani *multiple crops* sekaligus. Pendekatan ini diharapkan dapat mempertahankan akurasi tinggi yang telah dicapai penelitian sebelumnya, sambil mengatasi keterbatasan dalam hal generalisasi dan implementasi praktis. Selain itu, akan dilakukan perbandingan beberapa arsitektur *deep learning* untuk mengidentifikasi model yang paling optimal dalam menangani kasus *multiple crops*. Dengan fokus pada tanaman dari keluarga *Solanaceae* dan *Rosaceae*, penelitian ini bertujuan mengembangkan solusi yang lebih komprehensif dan mudah diakses bagi petani dalam mendeteksi penyakit tanaman secara cepat dan akurat.

2.2 Landasan Teori

2.2.1 *Convolutional Neural Networks*

Convolutional Neural Network (CNN) merupakan jenis arsitektur jaringan syaraf yang dirancang khusus untuk mengolah data dalam bentuk matriks 2 dimensi (2D), seperti citra. CNN merupakan evolusi dari *Multilayer Perceptron* (MLP) dan termasuk ke dalam kategori *deep learning*, dengan kemampuan untuk memproses kumpulan data yang besar dengan berbagai parameter. Dengan memanfaatkan lapisan konvolusi, CNN mampu menangkap fitur-fitur dalam data secara efektif, menjadikannya alat yang tepat untuk tugas-tugas seperti pengenalan pola pada citra [18].

CNN pada umumnya terdiri dari lima komponen utama yang memungkinkan pemrosesan citra secara efisien, yaitu lapisan *input*, lapisan konvolusi, lapisan *pooling*, fungsi aktivasi, dan lapisan *fully-connected* [18]. Seperti yang terlihat pada Gambar 2 di bawah.

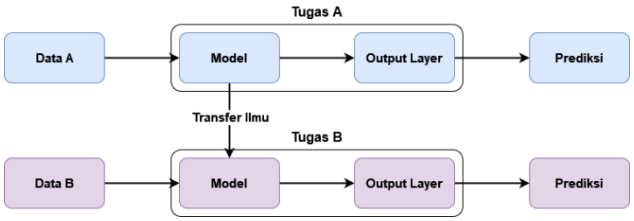


Gambar 2. Arsitektur Komponen CNN pada umumnya

CNN dikenal mampu memberikan performa yang sangat baik dalam tugas terkait identifikasi dan klasifikasi citra. Dalam konteks identifikasi penyakit tanaman, CNN dapat mengenali gejala penyakit dengan menganalisis perubahan warna, bentuk, dan tekstur pada daun tanaman, meskipun penelitian tersebut hanya berfokus pada tanaman tomat [11]. Penelitian sebelumnya juga telah menunjukkan bahwa CNN dapat mencapai akurasi yang tinggi dalam klasifikasi daun tanaman [7].

2.2.2 *Transfer Learning*

Transfer Learning (TL) adalah teknik *machine learning* yang memungkinkan pengetahuan yang diperoleh dari hasil pemecahan suatu masalah untuk diterapkan pada masalah lain namun terkait, misalnya yang masih berkaitan dengan citra. Hal ini sangat berguna ketika data yang tersedia untuk mencapai target tugas itu terbatas, sehingga memungkinkan model untuk memanfaatkan pengetahuan dari domain atau tugas lain untuk meningkatkan kinerja [14], seperti yang digambarkan pada Gambar 3 di bawah.



Gambar 3. Cara Kerja *Transfer Learning*

TL umumnya dibagi menjadi tiga pendekatan utama berdasarkan hubungan antara domain sumber dan target yaitu, *Transfer Learning* Induktif, Transduktif, dan Tanpa Pengawasan [14]. Dalam penelitian ini, TL Induktif digunakan, karena tugas identifikasi penyakit tanaman akan mendapat manfaat dari penggunaan model yang sudah terlatih seperti *InceptionV3*. Dalam TL Induktif, tugas sumber dan target itu berbeda tetapi memiliki domain yang sama. Misalnya, dengan memanfaatkan model yang sudah dilatih sebelumnya pada *dataset* yang luas seperti *ImageNet* (yang terdiri dari 20.000 kategori objek), untuk menyempurnakannya tugas-tugas tertentu, seperti mengidentifikasi penyakit tanaman dari citra daun.

Kemampuan *Transfer Learning* untuk menggunakan kembali pengetahuan yang sudah ada untuk tugas-tugas baru, tidak hanya mengurangi kebutuhan akan kumpulan data berlabel besar, tetapi juga secara signifikan mempercepat proses pelatihan, sehingga menjadikannya teknik yang sangat penting dalam sistem AI modern [14].

2.2.3 *InceptionV3*

InceptionV3 adalah model *pre-trained* yang tersedia di *TensorFlow*, sebagai kelanjutan dari versi sebelumnya seperti *InceptionV1* dan *InceptionV2*. Dirilis pada tahun 2015, model ini dirancang untuk meningkatkan performa dalam berbagai tugas *computer vision* [19]. Gambar 4 di bawah memperlihatkan bahwa model ini memiliki banyak layer.

Gambar 4. Arsitektur *InceptionV3*

Dengan struktur yang mendalam dan komprehensif ini, *InceptionV3* mampu mencapai akurasi tinggi dalam berbagai tugas pengenalan citra, menjadikannya salah satu model yang populer dalam bidang *computer vision* [15].

2.3 Pengunduhan Data

Data yang diterapkan dalam penelitian ini berasal dari *dataset* PlantVillage yang tersedia di Github. *Dataset* tersebut berisi total 54,306 sampel citra berformat JPEG dengan 38 kelas penyakit maupun sehat yang berasal dari 14 jenis tanaman yaitu Apel, Bluberi, Ceri, Jagung, Anggur, Jeruk, Persik, Lada, Kentang, Rasberi, Kedelai, Labu, Stroberi, dan Tomat. *Dataset* ini sudah disusun secara rapi, dengan setiap penyakit tanaman ditandai dengan cermat [8].

Dalam penelitian ini, *dataset* dimodifikasi sebagaimana ditunjukkan pada Tabel 1 dengan menghilangkan data tanaman di luar spesies *Solanaceae* dan *Rosaceae* untuk memudahkan proses penelitian. Modifikasi ini menghasilkan total 25 kelas. Tanaman yang dihilangkan dari *dataset* meliputi Bluberi, Rasberi, Kedelai, Labu, Jagung, Anggur, dan Jeruk karena tidak termasuk dalam kedua spesies yang menjadi fokus penelitian. Selain itu, setiap kelas diberi kode untuk meningkatkan keterbacaan.

Tabel 1. Kelas yang ada pada *Dataset* PlantVillage

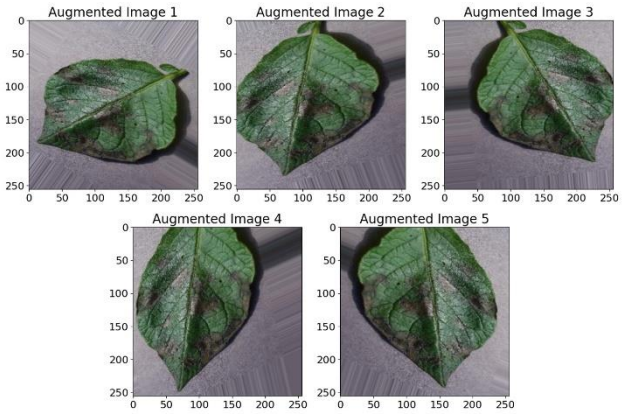
***Dataset* PlantVillage**	
Apple Apple scab (A-SCAB)	*Strawberry healthy (S-HLTH)*
Apple Black rot (A-ROT)	*Strawberry Leaf scorch (S-LSCR)*
Apple Cedar apple rust (A-CRUST)	*Tomato Bacterial spot (T-BSPT)*
Apple healthy (A-HLTH)	*Tomato Early blight (T-EBLT)*
Cherry healthy (C-HLTH)	*Tomato healthy (T-HLTH)*

Dataset PlantVillage	
Cherry Powdery mildew (C-PMIL)	*Tomato Late blight (T-LBLT)*
Peach Bacterial spot (PE-BSPT)	*Tomato Leaf Mold (T-LM)*
Peach healthy (PE-HLTH)	*Tomato Septoria leaf spot (T-SSP)*
Pepper bell Bacterial spot (BP-BSPT)	*Tomato Spider mites Two-spotted spider mite (T-SM)*
Pepper bell healthy (BP-HLTH)	*Tomato Target Spot (T-TS)*
Potato Early blight (P-EBLT)	*Tomato Tomato mosaic virus (T-MV)*
Potato healthy (P-HLTH)	*Tomato Tomato Yellow Leaf Curl Virus (T-YCV)*
Potato Late blight (P-LBLT)	

2.4 *Data Preprocessing*

Dataset PlantVillage yang telah diperoleh akan melalui tahap *preprocessing* agar data citra dapat diolah oleh model dengan tingkat akurasi yang optimal.

Tahap pertama adalah membuat label dari nama folder dan memasukkannya ke dalam *dataframe* yang dibagi menjadi *train*, dan *validation* (*val*) dengan rasio 80:20. Selanjutnya, *train set* diacak (*randomized*) untuk memastikan model dapat belajar dari data secara efisien. Setelah itu, dilakukan *augmentasi* pada data *train* dan *validation* dengan menerapkan teknik seperti rotasi, *horizontal flip*, pergeseran lebar dan tinggi (*width and height shift*), *shear*, serta *zoom*. Gambar 5 berikut ini adalah contoh citra yang sudah di-*augmentasi*.



Gambar 5. Contoh Hasil *Data Augmentation* pada *Dataset*

Kelima citra di atas merupakan citra yang telah di-*augmentasi* dengan teknik yang telah disebut sebelumnya. Perbedaan pada setiap citra tersebut adalah nilai *augmentasi* semua teknik yang bersifat acak, dengan rentang yang sudah disesuaikan, yang dapat dilihat pada Tabel 6 Parameter Dasar bagian *Data Augmentation*. Berikut adalah penjelasan perbedaan pada masing-masing citra: Terlihat pada *Augmented Image 1*, gambar telah dilakukan rotasi ke kanan dengan rentang 32 derajat, pengecilan sekitar 0,8, serta terlihat adanya kemiringan pada perspektif sehingga telah dilakukan *shearing*. Pada *Augmented Image 2*, gambar telah digeser ke kiri dan atas, rotasi ke kiri, dan *shearing*. Lalu pada *Augmented Image 3*, dilakukan *horizontal flip* serta pergeseran ke kanan dan atas. Pada *Augmented Image 4*, gambar diperbesar, digeser ke kiri dan atas, serta terdapat rotasi ke kiri. Terakhir pada *Augmented Image 5*, gambar dilakukan *horizontal flip* lalu dirotasi ke kanan dan diperbesar.

2.5 Pembuatan Model

Pada penelitian ini, pembuatan model diawali dengan pembuatan *callback*, agar memudahkan proses perawatan pada saat pelatihan. *Callback* kustom ini memantau dan menyesuaikan *learning rate* selama pelatihan berdasarkan kinerja model. Parameter seperti *patience*, *stop_patience*, *threshold*, dan *factor* menentukan kapan *learning rate* harus diturunkan jika akurasi atau *loss* tidak membaik. *Callback* ini juga memonitor *epoch*, akurasi, dan *loss* terbaik, menyimpan *weights* optimal untuk mencegah *overfitting*, serta dapat menghentikan pelatihan secara otomatis atau manual jika diperlukan. Selain itu, metrik pelatihan seperti akurasi dan *loss* dipantau dan ditampilkan secara *real-time*.

Pada model ini, arsitektur *InceptionV3* digunakan hingga lapisan terakhir sebelum *fully connected layer* untuk menghasilkan vektor fitur, seperti yang terlihat pada Tabel 2 di bawah. Lapisan-lapisan tambahan kemudian dibangun di atas vektor fitur ini untuk menyesuaikan model dengan *dataset* penyakit daun.

Tabel 2. Arsitektur Model *InceptionV3*

Layer (type)	*Output Shape*	*Param #*
keras_layer (KerasLayer)	*(None,* 2048*)*	21802784
batch_normalization	*(None,* 2048*)*	8192
dense (Dense)	*(None,* 256*)*	524544
dropout (Dropout)	*(None,* 256*)*	0
dense_1 (Dense)	*(None,* 25*)*	6425
Total params	:	22,342,973
Trainable params	:	536,093
Non-trainable params	:	21,806,880

2.6 Evaluasi

2.6.1 Metode Evaluasi

Evaluasi model dalam penelitian ini mengimplementasikan tiga metode analisis: *training curve*, *Classification Report*, dan *Confusion Matrix*. Ketiga metode ini dipilih untuk memberikan evaluasi komprehensif terhadap performa model yang dikembangkan.

Pada *Classification Report* disajikan evaluasi performa model menggunakan empat metrik fundamental. *Accuracy* mengukur rasio prediksi benar terhadap total prediksi. *Precision* mengevaluasi ketepatan prediksi positif model. *Recall* mengukur kemampuan model dalam mendeteksi kasus positif yang seharusnya terklasifikasi. *F1-Score* memberikan nilai tengah antara *precision* dan *recall*. Metrik-metrik ini dihitung menggunakan formula:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$$

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

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

$$F1\text{-}Score = 2\ x \frac{(Presisi\ x\ Recall)}{(Presisi+Recall)}$$

Dimana *true positive* (TP) mengacu pada jumlah yang terdeteksi positif secara benar, *true negative* (TN) adalah jumlah yang terdeteksi negatif secara benar, *false positive* (FP) merujuk pada jumlah yang terdeteksi positif secara salah, dan *false negative* (FN) mengacu pada jumlah yang terdeteksi negatif secara salah.

2.6.2 Rencana Eksperimen

Rencana eksperimen dirancang untuk mengevaluasi performa model melalui berbagai parameter yang relevan. Dalam eksperimen ini, model diuji menggunakan *dataset* yang telah diolah sebelumnya. Rancangan eksperimen ini juga mempertimbangkan variasi kondisi pelatihan dan pengujian untuk mendapatkan hasil yang komprehensif. Rencana Eksperimen dapat dilihat pada Tabel 3 berikut.

Tabel 3. Rencana Eksperimen

Eksperimen	Keterangan
Model Arsitektur	*InceptionV3*, CNN DFE, LCNN
Optimizer	*Adam, Adamax*
Pembagian Data	80% *Train*, 20% *Validation*; 70% Train, 30% *Validation*

Pada model CNN *Deep Feature Extractor* (DFE), arsitektur lapisan *Keras* dari *InceptionV3* disederhanakan dengan mengganti lapisan-lapisan tersebut dengan serangkaian lapisan konvolusi dan *pooling*. Jumlah filter konvolusi (32 dan 64) dipilih berdasarkan hasil penelitian sebelumnya yang menunjukkan bahwa nilai ini optimal untuk mendeteksi fitur pada citra tanaman. Ukuran *dense layer* sebesar 256 juga ditentukan dengan merujuk pada penelitian serupa, yang menyatakan bahwa nilai ini memberikan keseimbangan antara kapasitas model dan efisiensi komputasi. Setiap lapisan konvolusi diikuti oleh aktivasi *ReLU*, yang berfungsi untuk memperkenalkan *non-linearitas*, sebelum diakhiri dengan lapisan *Flatten* untuk mengubah data tiga dimensi menjadi satu dimensi, sehingga memudahkan proses klasifikasi. Arsitektur DFE dirancang untuk menangkap fitur-fitur penting dari citra *input* dengan efisien, sebagaimana terlihat pada Tabel 4 berikut.

Tabel 4. Arsitektur Model CNN DFE

Layer (type)	*Output Shape*	*Param #*
conv2d (Conv2D)	(*None*, 222, 222, 32)	896
max_pooling2d (MaxPooling2D)	(*None*, 111, 111, 32)	0
conv2d_1 (Conv2D)	(*None*, 109, 109, 64)	18,496
max_pooling2d_1 (MaxPooling2D)	(*None*, 54, 54, 64)	0
conv2d_2 (Conv2D)	(*None*, 52, 52, 64)	36,928
flatten (Flatten)	(*None*, 173056)	0
dense (Dense)	(*None*, 256)	44,302,592
dropout (Dropout)	(*None*, 256)	0
Total params	:	44,365,337
Trainable params	:	44,365,337
Non-trainable params	:	0

Sementara itu, pada model *Layered* CNN (LCNN), penambahan *batch normalization* di antara setiap lapisan konvolusi dan *pooling* bertujuan untuk meningkatkan stabilitas dan kecepatan pelatihan dengan menormalkan distribusi aktivasi. Hal ini memungkinkan model untuk beradaptasi dengan perubahan distribusi data selama proses pelatihan. Di samping itu, arsitektur LCNN juga dimodifikasi dengan penggunaan lapisan *Dense* yang lebih beragam dan penerapan *dropout* yang lebih tinggi untuk mencegah *overfitting*. Penggunaan 128 *filter* pada *conv2d_2* merupakan peningkatan dari *layer* sebelumnya (64) untuk memungkinkan deteksi fitur yang lebih kompleks pada level yang lebih dalam. Regularisasi L2 diterapkan pada lapisan *Dense* untuk memberikan kontrol tambahan terhadap kompleksitas model dan menjaga generalisasi. Pendekatan ini memastikan bahwa LCNN memiliki kapasitas yang baik untuk belajar dari data tanpa kehilangan kemampuan generalisasi, serta meningkatkan akurasi pada pengujian model. Arsitektur Model LCNN dapat dilihat pada Tabel 5 berikut.

Tabel 5. Arsitektur Model LCNN

Layer (type)	*Output Shape*	*Param #*
conv2d (Conv2D)	(*None*, 222, 222, 32)	896
batch_normalization (BatchNormalization)	(*None*, 222, 222, 32)	128
max_pooling2d (MaxPooling2D)	(*None*, 111, 111, 32)	0
conv2d_1 (Conv2D)	(*None*, 109, 109, 64)	18,496
batch_normalization_1 (BatchNormalization)	(*None*, 109, 109, 64)	256
max_pooling2d_1 (MaxPooling2D)	(*None*, 54, 54, 64)	0
conv2d_2 (Conv2D)	(*None*, 52, 52, 128)	73,856
batch_normalization_2 (BatchNormalization)	(*None*, 52, 52, 128)	512
Total params	:	19,402,969
Trainable params	:	19,402,009
Non-trainable params	:	960

Pemilihan *Adam* dan *Adamax* didasarkan pada rekam jejak keduanya yang sangat baik dalam tugas-tugas *deep learning*, terutama dalam klasifikasi citra. *Adam* dikenal karena kemampuannya yang unggul dalam mengombinasikan kecepatan konvergensi dengan stabilitas, sedangkan *Adamax*, sebagai variasi dari *Adam*, sering kali menunjukkan performa yang lebih baik pada data dengan nilai gradien yang lebih tinggi [20].

Skema 80% pelatihan dan 20% validasi dipilih karena umum digunakan dan memberikan data cukup untuk

melatih model serta mencegah *overfitting*. menguji generalisasi model dengan evaluasi yang lebih menyeluruh melalui jumlah data validasi yang lebih besar [21].

2.6.3 Penyimpanan Model

Setelah proses pelatihan dan evaluasi selesai, model disimpan dalam format *SavedModel*, yang merupakan format standar untuk menyimpan model *TensorFlow* beserta *metadata* terkait. Setelah itu, file model diarsipkan untuk memudahkan penyimpanan dan distribusi. Model yang telah disimpan ini dapat diimplementasikan di luar lingkungan IDE untuk melakukan proses identifikasi penyakit pada daun, baik di perangkat lokal maupun di server, sesuai kebutuhan aplikasi.

3. HASIL DAN PEMBAHASAN

3.1 Hasil Pelatihan Model

Setelah model dibuat, langkah selanjutnya adalah melatih model tersebut. Pelatihan akan dilakukan sesuai dengan rencana eksperimen yang telah dibuat sebelumnya. Namun, sebelum proses pelatihan dimulai, perlu dibahas terlebih dahulu parameter dasar yang akan digunakan secara konsisten di semua model, seperti yang tertera pada Tabel 6 berikut.

Tabel 6. Parameter Dasar

Kategori	Parameter	Nilai
Input & Preprocessing	*Image Size*	299 x 299 *pixels* (*InceptionV3*), 224 x 224 *pixels* (CNN kustom)
	Batch Size	32 (*generator*), 64 (*extractor*), 40 (*training*)
Data Augmentation	*Rescale*	1/255
	Rotation Range	40°
	Width/Height Shift	0.2
	Shear/Zoom Range	0.2
Training	*Base Learning Rate*	0.001
	Loss Function	*categorical*_crossentropy
	Epochs	20
Callbacks	*Patience*	1
	Stop Patience	3
	LR Reduction Factor	0.5
	Accuracy Threshold	0.9
Batch Normalization	*Momentum*	0.99
	Epsilon	0.001

Parameter dasar dalam penelitian ini dipilih berdasarkan praktik terbaik dan hasil penelitian sebelumnya untuk memastikan efisiensi dan performa model. Ukuran *input* 299x299 piksel digunakan untuk *InceptionV3* sesuai spesifikasinya [22], sementara ukuran 224x224 untuk CNN kustom dipilih karena efisien dalam menangkap fitur dan komputasi [23]. *Batch size* bervariasi (32, 64, 40) untuk mengoptimalkan stabilitas pelatihan dan pemanfaatan memori. Data *augmentasi* mencakup *rescaling* (1/255) untuk normalisasi, rotasi 40°, pergeseran posisi (0.2), serta *shear* dan *zoom* (0.2) untuk meningkatkan tingkat *robustness* pada model terhadap variasi data.

Base learning rate 0.001 dan fungsi *loss categorical_crossentropy* dipilih untuk stabilitas pelatihan pada masalah klasifikasi multikelas, dengan *epochs* maksimum 20 dan *early stopping* (*patience* 3) untuk mencegah *overfitting*. *Callback* seperti reduksi *learning rate* (faktor 0.5) dan momentum 0.99 digunakan untuk mempercepat konvergensi. Epsilon 0.001 pada *batch normalization* menjaga stabilitas numerik selama pelatihan. Parameter ini dipilih untuk memberikan keseimbangan optimal antara kapasitas model dan efisiensi komputasi.

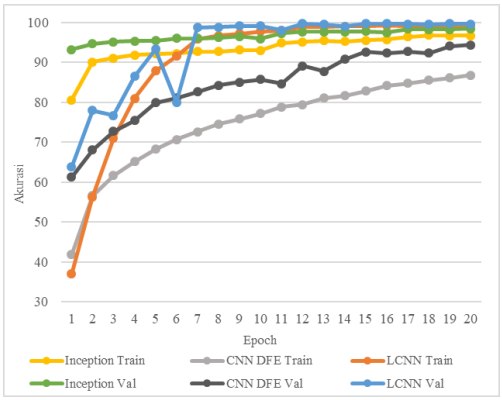
Setelah parameter dasar ditetapkan, eksperimen dilaksanakan. Proses pelatihan dilakukan menggunakan *Kaggle IDE* karena memiliki kecepatan tinggi, biaya yang efektif, dan kemudahan penggunaan. Hasil eksperimen ditunjukkan pada Tabel 7, dengan durasi pelatihan sekitar 1 jam untuk setiap model berbasis *InceptionV3* dan sekitar 30 menit untuk setiap model CNN kustom.

Tabel 7. Hasil Eksperimen

Eksperimen				Akurasi	*Loss*	
No	**Model**	***Optimizer***	**Rasio**	***F1-Score***	***Train***	***Val***
1	*InceptionV3*	*Adamax*	80:20	98%	0.43	0.4
2	*InceptionV3*	*Adamax*	70:30	98%	0.47	0.4
3	*InceptionV3*	*Adam*	80:20	97%	0.5	0.4
4	*InceptionV3*	*Adam*	70:30	98%	0.44	0.4
5	CNN DFE	*Adamax*	80:20	94%	1.12	0.9
6	CNN DFE	*Adamax*	70:30	93%	1.19	1
7	CNN DFE	*Adam*	80:20	83%	2.45	2.2
8	CNN DFE	*Adam*	70:30	80%	2.28	2
9	LCNN	*Adamax*	80:20	99%	0.19	0.2
10	LCNN	*Adamax*	70:30	98%	0.2	0.2
11	LCNN	*Adam*	80:20	89%	1.73	1.8
12	LCNN	*Adam*	70:30	93%	1.11	0.8

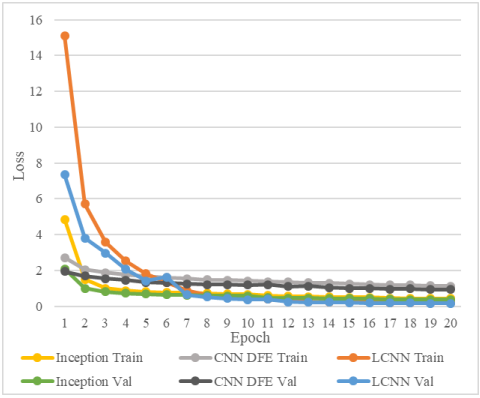
Hasil eksperimen diperoleh dari pengujian berbagai konfigurasi model sesuai pada tabel 6 pada *Kaggle IDE*. Dari tabel hasil, terlihat bahwa model InceptionV3 dan LCNN menunjukkan performa paling konsisten dengan akurasi tinggi (98-99%) dan *loss* rendah (0.2-0.4), sementara model CNN DFE memiliki performa sedikit lebih rendah dengan akurasi 80-94%.

Maka dari itu dari antara 12 eksperimen yang dilakukan, 3 eksperimen yang memiliki akurasi tertinggi di setiap jenis model diambil, yaitu eksperimen yang mewakili 3 model CNN yang digunakan: Model *InceptionV3*, CNN DFE, dan LCNNs. Eksperimen yang terpilih adalah eksperimen di-*highlight* pada tabel di atas yaitu 1, 5, dan 9. Untuk menggambarkan lebih lanjut kinerja mereka, kurva akurasi untuk hasil ini ditampilkan pada Gambar 6 berikut.



Gambar 6. Akurasi *Train* dan *Val InceptionV3* vs CNN DFE vs LCNN

Metrik *Loss* untuk ketiga model ditampilkan pada Gambar 7 di bawah ini, dimana pada pelatihan, nilai *Loss* masing-masing model adalah 0.43 untuk *InceptionV3*, 1.12 untuk CNN DFE, dan 0.19 untuk LCNN. Sementara itu, untuk validasi, nilai *Loss* bernilai sebesar 0.4 untuk *InceptionV3*, 0.9 untuk CNN DFE, dan 0.2 untuk LCNN. Hasil ini menunjukkan bahwa ketiga model bekerja dengan baik selama pelatihan, dengan LCNN menunjukkan kinerja terbaik. Nilai *Loss* yang lebih rendah mencerminkan kemampuan yang lebih baik dalam memprediksi data pelatihan, dengan LCNN unggul dalam hal ini diikuti oleh *InceptionV3* dan CNN DFE.



Gambar 7. *Loss Train* dan *Val InceptionV3* vs CNN DFE vs LCNN

Di antara 12 eksperimen yang dilakukan, hasil performa menunjukkan variasi yang signifikan antar model dan parameter yang digunakan. Analisis detail untuk semua model adalah sebagai berikut.

Model LCNN menunjukkan performa terbaik dengan akurasi *training* mencapai 99% (eksperimen 9) menggunakan *optimizer Adamax* dan rasio 80:20. Model ini juga mencapai nilai *loss* terendah yaitu 0.19 untuk *training* dan 0.2 untuk validasi. Hal ini menunjukkan bahwa arsitektur berlapis pada LCNN sangat efektif dalam mengekstrak fitur-fitur penting dari citra. Konsistensi antara nilai *loss training* dan *validasi* yang hampir sama (0.19 dan 0.2) juga mengindikasikan bahwa model ini memiliki generalisasi yang baik.

Model *InceptionV3* menunjukkan performa yang stabil di semua eksperimen dengan akurasi *training* berkisar 97-98%. Khususnya pada eksperimen 1, model mencapai akurasi 98% dengan *loss training* 0.43 dan *loss validasi* 0.4. Perbedaan yang kecil antara *loss training* dan *validasi* menunjukkan model tidak mengalami *overfitting* yang signifikan. Performa model relatif konsisten baik menggunakan *optimizer Adam* maupun *Adamax*, yang menunjukkan *robustness* dari arsitektur *InceptionV3*.

Model CNN DFE menunjukkan variasi performa yang lebih besar. Dengan *optimizer Adamax* (eksperimen 5 dan 6), model mencapai akurasi training 93-94% dengan *loss* yang lebih tinggi dibanding model lainnya (1.12-1.19). Namun performa menurun signifikan dengan *optimizer Adam*, dimana akurasi turun hingga 80-83% dengan *loss* yang meningkat hingga 2.28-2.45. Hal ini mengindikasikan bahwa CNN DFE lebih sensitif terhadap pemilihan *optimizer*.

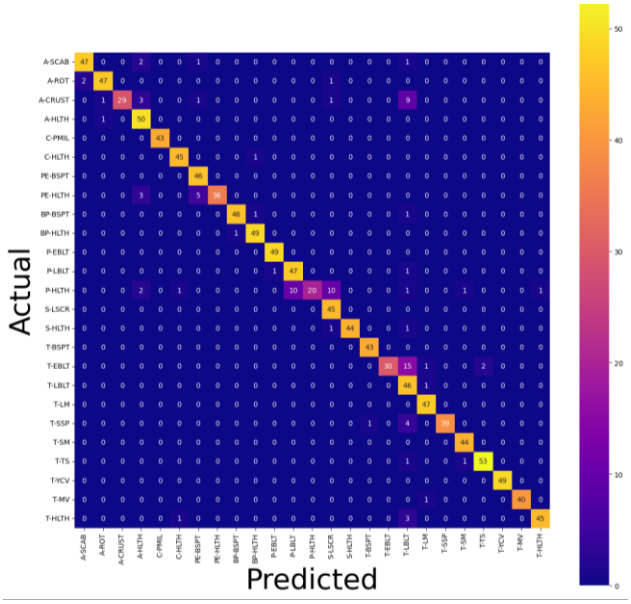
Dari segi stabilitas, LCNN dan *InceptionV3* menunjukkan performa yang lebih konsisten dibanding CNN DFE. Hal ini terlihat dari gap antara nilai *loss training* dan validasi yang lebih kecil, serta konsistensi akurasi di berbagai konfigurasi parameter. LCNN unggul dalam mencapai akurasi tertinggi dan loss terendah, sementara *InceptionV3* menunjukkan stabilitas yang baik dengan performa yang konsisten di semua eksperimen.

Setelah hasil dari ketiga model diperoleh, pengujian lebih lanjut terhadap model perlu dilakukan untuk mengevaluasi performa dalam skenario dunia nyata, memastikan keandalan dan keakuratannya sebelum penerapan.

3.2 Pengujian Model

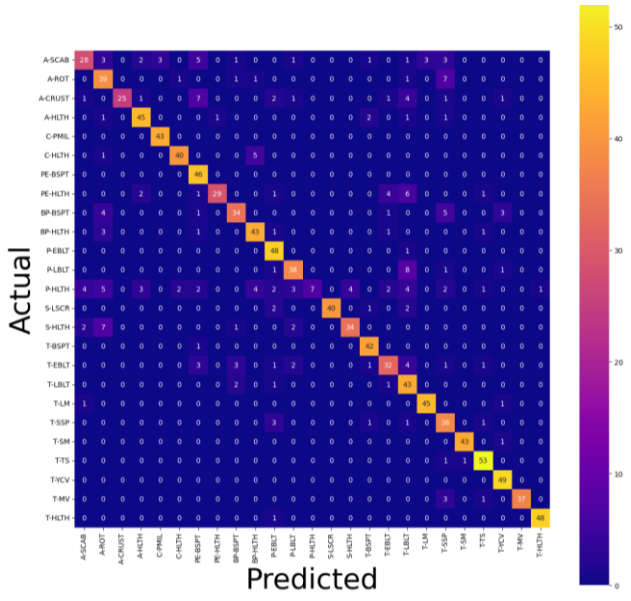
Pengujian berfungsi untuk menentukan keefektifan model dalam mengidentifikasi objek (dalam penelitian ini, daun) dan untuk memastikan bahwa model yang telah dilatih tidak mengalami *overfitting*. Oleh karena itu, proses pengujian ini sangat penting. Pengujian dilakukan dengan menggunakan model yang telah disimpan sebelumnya, yang kemudian dijalankan dalam lingkungan yang berbeda, dalam hal ini, *VS Code*. Model yang digunakan adalah ketiga *best* model yaitu model nomor 1 (*InceptionV3*), nomor 5 (CNN DFE) dan nomor 9 (LCNN). Model ini akan diuji dengan Data uji yang terdiri dari 40 hingga 50 sampel untuk setiap kelas penyakit (atau kategori sehat). Seperti yang telah dijelaskan sebelumnya, tingkat keberhasilan pengujian model ini dievaluasi dengan menggunakan metode seperti *classification report* dan *confusion matrix*.

Hasil dari *confusion matrix* untuk model *InceptionV3* dapat dilihat pada Gambar 8. Dalam confusion matrix berikut, setiap kelas penyakit direpresentasikan dengan singkatan yang sesuai dengan daftar kelas pada Tabel 1, seperti *'A-SCAB'* untuk *Apple Scab*.



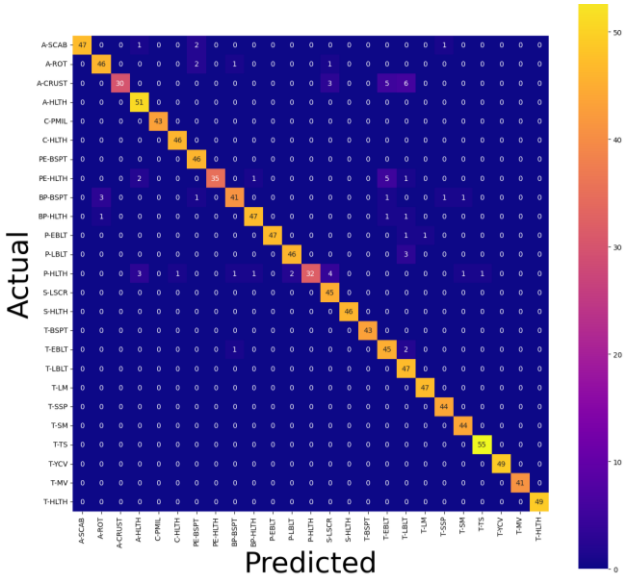
Gambar 8. *Confusion Matrix InceptionV3*

Dapat diamati bahwa sejumlah besar daun diprediksi dengan benar, yang mengindikasikan keefektifan model secara keseluruhan. Namun, ada beberapa prediksi yang salah juga. Khususnya, di daerah tengah *confusion matrix* tersebut terdapat kesalahan yang melibatkan daun kentang sehat yang salah diklasifikasikan sebagai memiliki penyakit daun kentang *late blight* dan daun stroberi *leaf scorch*, yang menghasilkan total 20 positif palsu. Lalu, pada gambar 9 di bawah, disajikan *confusion matrix* model CNN DFE untuk dianalisis mengenai kinerja kesulitan prediksi pada model.



Gambar 9. *Confusion Matrix* CNN DFE

Berdasarkan hasil yang ditunjukkan di atas, model CNN DFE ini memiliki kesulitan dalam mendeteksi daun *Potato Healthy* dengan nilai *True Positive* hanya sebanyak 7. Ini menunjukkan bahwa model kurang mampu membedakan daun *potato healthy* dari yang terinfeksi, mengindikasikan perlunya penyesuaian model. Selanjutnya, analisis *confusion matrix* model LCNN pada Gambar 10 menyajikan wawasan lebih lanjut mengenai kinerja dan area-area di mana model mengalami kesulitan.



Gambar 10. *Confusion Matrix* LCNN

Hasil dari ketiga model (*InceptionV3*, CNN DFE, dan LCNN) menunjukkan perbedaan kinerja yang signifikan. *InceptionV3* dan LCNN menunjukkan hasil yang sangat mirip dan cenderung lebih baik, dengan beberapa kesalahan spesifik seperti kesalahan klasifikasi *cedar apple rust* pada LCNN. Sementara itu, CNN DFE menunjukkan lebih banyak kesulitan, terutama dalam mendeteksi daun *potato healthy*. Untuk memberikan gambaran yang lebih komprehensif tentang kinerja ketiga model dalam mengklasifikasikan berbagai jenis penyakit tanaman, tingkat akurasi dan *F1-score* untuk masing-masing model dapat dilihat pada *classification report* yang disajikan dalam Tabel 8, Tabel 9, dan Tabel 10 berikut ini.

Tabel 8. Nilai Data *Test* pada Model *InceptionV3*

Class	*Precision*	*Recall*	*F1-Score*	*Support*
A-SCAB	0.96	0.92	0.94	51
A-ROT	0.96	0.94	0.95	50
A-CRUST	1.00	0.66	0.79	44
A-HLTH	0.83	0.98	0.90	51
C-PMIL	1.00	1.00	1.00	43
C-HLTH	0.96	0.98	0.97	46
PE-BSPT	0.87	1.00	0.93	46
PE-HLTH	1.00	0.82	0.90	44
BP-BSPT	0.98	0.96	0.97	48
BP-HLTH	0.96	0.98	0.97	50
P-EBLT	0.98	1.00	0.99	49
P-LBLT	0.82	0.96	0.89	49
P-HLTH	1.00	0.43	0.61	46
S-LSCR	0.78	1.00	0.87	45
S-HLTH	1.00	0.96	0.98	46
T-BSPT	0.98	1.00	0.99	43
T-EBLT	1.00	0.62	0.77	48
T-LBLT	0.55	0.98	0.71	47
T-LM	0.94	1.00	0.97	47
T-SSP	1.00	0.89	0.94	44
T-SM	0.96	1.00	0.98	44
T-TS	0.96	0.96	0.96	55
T-YCV	1.00	1.00	1.00	49

Class	*Precision*	*Recall*	*F1-Score*	*Support*
T-MV	1.00	0.98	0.99	41
T-HLTH	0.98	0.92	0.95	49
Accuracy			**0.92**	**1175**
Macro Avg	0.94	0.92	0.92	1175
Weighted Avg	0.94	0.92	0.92	1175

Tabel 9. Nilai Data *Test* pada Model CNN DFE

Class	*Precision*	*Recall*	*F1-Score*	*Support*
A-SCAB	0.78	0.55	0.64	51
A-ROT	0.62	0.78	0.69	50
A-CRUST	1.00	0.57	0.72	44
A-HLTH	0.85	0.88	0.87	51
C-PMIL	0.93	1.00	0.97	43
C-HLTH	0.93	0.87	0.90	46
PE-BSPT	0.69	1.00	0.81	46
PE-HLTH	0.97	0.66	0.78	44
BP-BSPT	0.81	0.71	0.76	48
BP-HLTH	0.81	0.86	0.83	50
P-EBLT	0.76	0.98	0.86	49
P-LBLT	0.81	0.78	0.79	49
P-HLTH	1.00	0.15	0.26	46
S-LSCR	1.00	0.89	0.94	45
S-HLTH	0.89	0.74	0.81	46
T-BSPT	0.88	0.98	0.92	43
T-EBLT	0.76	0.67	0.71	48
T-LBLT	0.57	0.91	0.70	47
T-LM	0.94	0.96	0.95	47
T-SSP	0.60	0.86	0.71	44
T-SM	0.98	0.98	0.98	44
T-TS	0.90	0.96	0.93	55
T-YCV	0.88	1.00	0.93	49
T-MV	1.00	0.90	0.95	41
T-HLTH	0.98	0.98	0.98	49
Accuracy			**0.82**	**1175**
Macro Avg	0.85	0.82	0.82	1175
Weighted Avg	0.85	0.82	0.82	1175

Tabel 10. Nilai Data *Test* pada Model LCNN

Class	*Precision*	*Recall*	*F1-Score*	*Support*
A-SCAB	1.00	0.92	0.96	51
A-ROT	0.92	0.92	0.92	50
A-CRUST	1.00	0.68	0.81	44
A-HLTH	0.89	1.00	0.94	51
C-PMIL	1.00	1.00	1.00	43
C-HLTH	0.98	1.00	0.99	46
PE-BSPT	0.90	1.00	0.95	46
PE-HLTH	1.00	0.80	0.89	44
BP-BSPT	0.93	0.85	0.89	48
BP-HLTH	0.96	0.94	0.95	50
P-EBLT	1.00	0.96	0.98	49
P-LBLT	0.96	0.94	0.95	49
P-HLTH	1.00	0.70	0.82	46
S-LSCR	0.85	1.00	0.92	45
S-HLTH	1.00	1.00	1.00	46
T-BSPT	1.00	1.00	1.00	43
T-EBLT	0.79	0.94	0.86	48
T-LBLT	0.77	1.00	0.87	47
T-LM	0.98	1.00	0.99	47
T-SSP	0.96	1.00	0.98	44
T-SM	0.96	1.00	0.98	44
T-TS	0.98	1.00	0.99	55
T-YCV	1.00	1.00	1.00	49
T-MV	1.00	1.00	1.00	41
T-HLTH	1.00	1.00	1.00	49
Accuracy			**0.95**	**1175**
Macro Avg	0.95	0.95	0.95	1175
Weighted Avg	0.95	0.95	0.95	1175

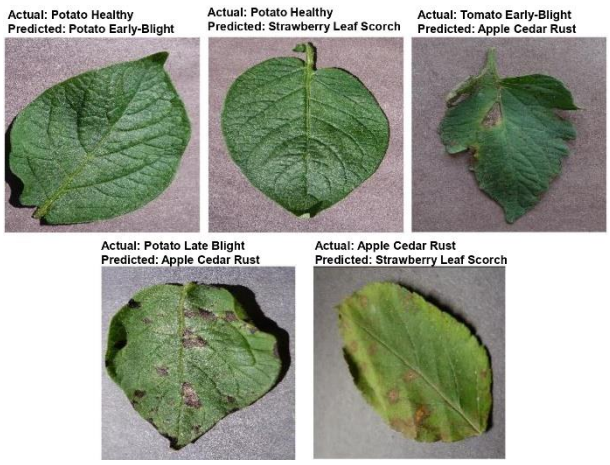
Berdasarkan hasil pengujian pada ketiga model, dapat dilihat bahwa kinerja mereka bervariasi. *InceptionV3* mencapai akurasi 92%, CNN DFE mencapai akurasi 82%, sedangkan LCNN menunjukkan kinerja terbaik dengan akurasi 95%.

Model LCNN menunjukkan performa yang paling unggul dengan nilai rata-rata *F1-score*, *precision*, dan *recall* yang tertinggi (0.95), diikuti oleh *InceptionV3* (0.92), dan CNN DFE (0.82). Ini menunjukkan bahwa LCNN memiliki kemampuan terbaik dalam mengklasifikasikan berbagai kondisi daun dengan tingkat keseimbangan yang baik antara *precision* dan *recall*, tetapi masih mengalami kesulitan pada kelas *Potato Healthy* (*recall* 0.70). *InceptionV3* juga menunjukkan kinerja yang sangat baik, dengan performa yang konsisten di berbagai kelas. Namun, model ini mengalami sedikit kesulitan dalam beberapa kelas seperti *Cedar apple rust* (*recall* 0.66) dan *Potato Healthy* (*recall* 0.43). CNN DFE, meskipun memiliki akurasi terendah di antara ketiga model, masih menunjukkan kinerja yang cukup baik untuk beberapa kelas. Namun, model ini mengalami kesulitan signifikan dalam mengklasifikasikan beberapa kelas, terutama *Potato Healthy* (*recall* hanya 0.15) dan *Apple Scab* (*recall* 0.55).

Secara keseluruhan, LCNN menunjukkan kinerja terbaik dalam tugas klasifikasi ini, diikuti oleh *InceptionV3*, sementara CNN DFE memerlukan perbaikan lebih lanjut untuk meningkatkan akurasinya, terutama dalam kelas-kelas yang kinerjanya rendah.

3.3 Keterbatasan

Meskipun ketiga model menunjukkan kinerja yang menjanjikan dengan akurasi *InceptionV3* mencapai 0.92, LCNN 0.95, dan CNN DFE 0.82, analisis lebih lanjut mengungkapkan beberapa keterbatasan yang perlu diperhatikan. Salah satu keterbatasan utama adalah terjadinya hasil positif palsu, seperti yang ditunjukkan pada Gambar 11. Keadaan ini menunjukkan bahwa model tersebut masih menghadapi tantangan dalam membedakan antara beberapa penyakit tanaman dan kondisi daun yang sehat dengan akurasi yang tinggi.



Gambar 11. Sampel Data *False Positive*

Citra 3, 4, dan 5 kemungkinan menghasilkan *false positive* karena kemiripan gejala antar penyakit yang diidentifikasi, serta kesamaan bentuk dan warna daun dari jenis-jenis tanaman tersebut. Sebagai contoh, pola kerusakan daun pada penyakit *apple cedar rust* yang memiliki presisi 1.00 tetapi *recall* hanya 0.66-0.68 pada model InceptionV3 dan LCNN, dan *tomato early blight* dengan presisi 1.00 tetapi recall 0.62-0.67 pada model *InceptionV3* dan CNN DFE cenderung serupa, sehingga sulit dibedakan oleh model.

Sementara itu, penyebab kesalahan pada citra 1 dan 2, di mana daun kentang sehat teridentifikasi sebagai penyakit *potato early blight* (yang memiliki *recall* 0.98-1.00 dan presisi 0.87-0.90 pada ketiga model) atau *strawberry leaf scorch* (dengan *recall* 0.89-1.00 dan presisi 0.78-0.85 pada ketiga model), tampak kurang jelas. Data ini menunjukkan bahwa model cenderung *over-confident* dalam mengidentifikasi kedua penyakit tersebut (*recall* tinggi), yang bisa menjelaskan kenapa daun sehat bisa salah teridentifikasi sebagai penyakit ini.

4. KESIMPULAN

Penelitian ini berhasil mengembangkan sistem deteksi penyakit tanaman yang efektif dan adaptif untuk mengidentifikasi berbagai penyakit pada dua keluarga tanaman berbeda. Model CNN kustom (LCNN) yang dikembangkan menunjukkan kinerja unggul dengan tingkat akurasi pelatihan dan validasi sebesar 99%, serta pengujian sebesar 95%. Sementara itu, *InceptionV3* mencapai akurasi pelatihan 96%, validasi 98%, dan pengujian 92%, sedangkan CNN DFE memiliki akurasi pelatihan 86%, validasi 94%, dan pengujian 82%. Temuan ini membuktikan bahwa arsitektur yang lebih sederhana namun dirancang spesifik dapat lebih efektif daripada model yang lebih kompleks untuk kasus deteksi penyakit tanaman.

Analisis lebih lanjut mengungkapkan bahwa LCNN memiliki kemampuan generalisasi yang lebih baik dan efektivitas dalam menangani kasus-kasus sulit, seperti membedakan daun sehat dan yang terinfeksi. Keseimbangan yang baik antara *precision* dan *recall* di berbagai kelas penyakit juga menunjukkan konsistensi kinerja LCNN. Temuan ini menegaskan pentingnya merancang arsitektur model yang sesuai dengan karakteristik spesifik tugas. Meskipun demikian, beberapa keterbatasan teridentifikasi melalui *confusion matrix*, terutama dalam membedakan antara daun kentang yang sehat dengan yang terinfeksi *late blight*, serta kesalahan dalam mengidentifikasi *cedar apple rust* pada daun apel. Keterbatasan ini terlihat pada ketiga model yang diuji, menunjukkan adanya tantangan intrinsik dalam klasifikasi penyakit tanaman tertentu.

Untuk pengembangan ke depan, model ini direncanakan akan diimplementasikan ke dalam aplikasi *mobile* dengan arsitektur *client-server*. Salah satu pengembangan yang dapat dipertimbangkan adalah memperluas cakupan *dataset* dengan menambahkan lebih banyak jenis tanaman. Pengujian lapangan yang lebih ekstensif dan pengembangan sistem peringatan dini yang terintegrasi dengan rekomendasi penanganan seperti penggunaan *IoT* juga disarankan untuk meningkatkan nilai praktis dari sistem ini. Langkah-langkah ini diharapkan dapat memperluas aplikasi sistem deteksi penyakit tanaman dan meningkatkan efektivitasnya dalam kondisi nyata di lapangan.

DAFTAR PUSTAKA

[1] A. Dias Rizkia Artha Putri, L. Sukma Winata, A. Tanggono, and F. Disa Durry, "Mitigasi Krisis Pangan Global Warming: SDGs Pencegahan Malnutrisi (Literature Review)," *Prosiding Seminar Nasional Kusuma*, vol. 2, pp. 179–187, Oct. 2024.

[2] D. Sofia Laeliyah, N. Noreen Noor, A. Sabillah, U. Kamal, and M. Adymas Hikal Fikri, "Kebijakan Hukum Pengelolaan Food Loss And Waste Melalui USDA (United States Departement Of Agriculture And Public Domain Policy)," *Kultura: Jurnal Ilmu Hukum, Sosial, Dan Humaniora*, vol. 2, no. 6, pp. 25–41, May 2024.

[3] A. H. Afzal *et al.*, "Mechanisms of Disease Resistance in Plants For Sustainable Agriculture," *Current Research in Agriculture and Farming*, vol. 4, pp. 1–13, Oct. 2023, doi: 10.18782/2582-7146.207.

[4] M. Junaid and A. Gokce, "Global Agricultural Losses And Their Causes," *Bulletin of Biological and Allied Sciences Research*, vol. 2024, no. 1, p. 66, Feb. 2024, doi: 10.54112/bbasr.v2024i1.66.

[5] M. Türkoğlu and D. Hanbay, "Plant disease and pest detection using deep learning-based features," *Turkish Journal of Electrical Engineering & Computer Sciences*, vol. 27, no. 3, pp. 1636–1651, May 2019, doi: 10.3906/elk-1809-181.

[6] S. P. Mohanty, D. P. Hughes, and M. Salathé, "Using Deep Learning for Image-Based Plant Disease Detection," *Front Plant Sci*, vol. 7, p. 1419, Sep. 2016, doi: 10.3389/fpls.2016.01419.

[7] I. Rizki Ramadhani, A. Nilogiri, and A. Qurrota, "Klasifikasi Jenis Tumbuhan Berdasarkan Citra Daun Menggunakan Metode Convolutional Neural Network," *Jurnal Smart Teknologi*, vol. 3, no. 3, pp. 249–260, Mar. 2022.

[8] G. Henry, A. Panjaitan, and F. Simatupang, "Pemodelan Klasifikasi Penyakit Daun Tanaman Tomat dengan Convolutional Neural Network Algorithm," *KLIK: Kajian Ilmiah Informatika dan Komputer*, vol. 4, no. 5, Apr. 2024, doi: 10.30865/klik.v4i5.1646.

[9] J. Vicky, F. Ayu, and B. Julianto, "Implementasi Pendeteksi Penyakit pada Daun Alpukat

Menggunakan Metode CNN," *Seminar Nasional Teknologi & Sains*, vol. 2, no. 1, pp. 155–162, 2023.

[10] F. Sulistiyana and S. Anardani, "Aplikasi Deteksi Penyakit Tanaman Jagung Dengan Convolutional Neural Network dan Support Vector Machine," *Seminar Nasional Teknologi Informasi dan Komunikasi*, vol. 6, no. 1, pp. 423–432, 2023.

[11] D. Setiawan, T. Wira, and E. Suryawijaya, "Algoritma RESNET152V2 Dalam Melakukan Klasifikasi Penyakit Pada Daun Tanaman Tomat," *Journal of Computer Science and Technology (JCS-TECH)*, vol. 3, no. 2, pp. 37–42, Nov. 2023.

[12] M. D. Pratama, R. Gustriansyah, and E. Purnamasari, "Klasifikasi Penyakit Daun Pisang menggunakan Convolutional Neural Network (CNN)," *Jurnal Teknologi Terpadu*, vol. 10, no. 1, pp. 1–6, Jul. 2024, doi: 10.54914/jtt.v10i1.1167.

[13] M. Rijal, A. M. Yani, and A. Rahman, "Deteksi Citra Daun untuk Klasifikasi Penyakit Padi menggunakan Pendekatan Deep Learning dengan Model CNN," *Jurnal Teknologi Terpadu*, vol. 10, no. 1, pp. 56–62, Jul. 2024, doi: 10.54914/jtt.v10i1.1224.

[14] A. Hosna, E. Merry, J. Gyalmo, Z. Alom, Z. Aung, and M. A. Azim, "Transfer learning: a friendly introduction," *J Big Data*, vol. 9, no. 1, p. 102, Oct. 2022, doi: 10.1186/s40537-022-00652-w.

[15] Xiaoling Xia, Cui Xu, and Bing Nan, "Inception-v3 for flower classification," in *2017 2nd International Conference on Image, Vision and Computing (ICIVC)*, IEEE, Jun. 2017, pp. 783–787. doi: 10.1109/ICIVC.2017.7984661.

[16] K. Zairan Maulana and A. Susanto, "Implementasi Arsitektur CNN Inception V3 untuk Identifikasi Spesies Burung Endemik di Indonesia," *Jurnal Pustaka Robot Sister (Jurnal Pusat Akses Kajian Robotika, Sistem Tertanam, dan Sistem Terdistribusi)*, vol. 2, no. 1, pp. 22–27, Jan. 2024, doi: 10.55382/jurnalpustakarobotsister.v2i1.775.

[17] S. Kornblith, J. Shlens, and Q. V. Le, "Do Better ImageNet Models Transfer Better?," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, Jun. 2019, pp. 2656–2666. doi: 10.1109/CVPR.2019.00277.

[18] M. M. Taye, "Theoretical Understanding of Convolutional Neural Network: Concepts, Architectures, Applications, Future Directions," *Computation*, vol. 11, no. 3, p. 52, Mar. 2023, doi: 10.3390/computation11030052.

[19] M. Mujahid, F. Rustam, R. Álvarez, J. Luis Vidal Mazón, I. de la T. Díez, and I. Ashraf, "Pneumonia Classification from X-ray Images with Inception-V3 and Convolutional Neural Network," *Diagnostics*, vol. 12, no. 5, p. 1280, May 2022, doi: 10.3390/diagnostics12051280.

[20] K. R. Prilianti, T. H. P. Brotosudarmo, S. Anam, and A. Suryanto, "Performance comparison of the convolutional neural network optimizer for photosynthetic pigments prediction on plant digital image," in *AIP Conference Proceedings*, American Institute of Physics Inc., Mar. 2019, p. 020020. doi: 10.1063/1.5094284.

[21] I. O. Muraina, "Ideal Dataset Splitting Ratios In Machine Learning Algorithms: General Concerns For Data Scientists And Data Analysts," in *7th International Mardin Artuklu Scientific Researches Conference*, Mardin: Mardin Artuklu Kongresi, Feb. 2022, pp. 496–504.

[22] S. Ramaneswaran, K. Srinivasan, P. M. D. R. Vincent, and C.-Y. Chang, "Hybrid Inception v3 XGBoost Model for Acute Lymphoblastic Leukemia Classification," *Computational and Mathematical Methods in Medicine*, vol. 2021, pp. 1–10, Jul. 2021, doi: 10.1155/2021/2577375.

[23] M. L. Richter, W. Byttner, U. Krumnack, A. Wiedenroth, L. Schallner, and J. Shenk, "(Input) Size Matters for CNN Classifiers," 2021, pp. 133–144. doi: 10.1007/978-3-030-86340-1_11.