

# Klasifikasi Penyakit Padi Menggunakan Convolutional Neural Network (CNN) Berbasis Citra Daun

Moh. Heri Susanto<sup>1</sup>, Irwan Budi Santoso<sup>2</sup>, Suhartono<sup>3</sup>, Ahmad Fahmi Karami<sup>4</sup>

<sup>1,2,3,4</sup> Program Studi Teknik Informatika, Fakultas Sains dan Teknologi, Universitas Islam Negeri Maulana Malik Ibrahim, Jalan Gajayana No. 50, Malang 65144, Indonesia

[Diserahkan: hh Bulan 20xx, Direvisi: 1 Januari 2025, Diterima: 1 Februari 2025]

Penulis Korespondensi: Moh. Heri Susanto (email: 200605110087@student.uin-malang.ac.id)

**INTISARI** — Penyakit padi berdampak signifikan terhadap produktifitas pertanian, sehingga model klasifikasi sangat penting untuk membedakan penyakit daun padi secara akurat. Berbagai model klasifikasi telah diusulkan untuk mengklasifikasikan penyakit padi berbasis citra daun, tetapi masih diperlukan peningkatan kinerja lebih lanjut. Dengan demikian, penelitian ini diusulkan menggunakan *Convolutional Neural Network* (CNN) untuk mengklasifikasikan penyakit padi berbasis citra daun. Dalam penelitian ini, menggunakan dataset citra daun yang memuat daun dalam kondisi *blight*, *blast*, *tungro*, dan *healthy*. Proses awal, data dilakukan tahap pemrosesan terlebih dahulu seperti *resize*, augmentasi, dan normalisasi. Setelah selesai dilakukan pemrosesan, dilakukan pembentukan arsitektur CNN yang bersifat kustom terdiri dari 4 *convolutional layer*, 4 *pooling layer*, dan 3 *fully connected layer*. Setiap *convolutional layer* menggunakan kernel berukuran 3x3, *stride* 1, dan aktivasi ReLU, sementara *pooling layer* menggunakan *max pooling* dengan kernel berukuran 3x3 dan *stride* 2. Ditambah penggunaan konfigurasi *batch size* 32 dan optimasi Adam, mampu didapatkan hasil pengujian terbaik pada percobaan dengan menggunakan *epoch* 100 dan *learning rate* 0,0002 yang menghasilkan akurasi *training* 0,9930, *loss* 0,0221, dan akurasi *testing* 0,9647. Hasil evaluasi model mampu mendapatkan keseimbangan antara presisi, *recall*, dan *f1-score* yaitu 0,9647 yang menunjukkan proses klasifikasi data berjalan dengan sangat baik, tanpa adanya bias terhadap kelas tertentu. Hasil penelitian ini menunjukkan bahwa model CNN yang disederhanakan dapat memberikan kinerja klasifikasi yang kompetitif tanpa memerlukan model CNN yang kompleks atau dengan teknik tambahan lainnya. Model CNN yang diusulkan mampu berkinerja lebih baik daripada model CNN yang ada seperti Inception-ResNet-V2, VGG-16, VGG-19, dan Xception.

**KATA KUNCI** — Penyakit Padi, Arsitektur CNN Khusus, Citra Daun, Optimasi Adam, Evaluasi Model

## I. PENDAHULUAN

Penyakit padi menimbulkan ancaman yang signifikan terhadap produktivitas pertanian, sehingga memerlukan metode klasifikasi yang akurat untuk membedakan berbagai jenis penyakit padi. Beberapa penyakit padi utama memengaruhi hasil panen, termasuk *blight*, *blast*, dan *tungro*. *Blight* adalah penyakit yang disebabkan oleh bakteri *Xanthomonas Oryzae Poryzae* (Xoo) dengan ciri-ciri lesi memanjang dengan warna lesi yang bervariasi mulai dari kuning hingga kecoklatan. Biasanya muncul pada daun yang baru berkembang dan seiring waktu lesi-lesi dapat menyatu hingga tampak seperti daun terbakar [1]. *Blast*, yang disebabkan oleh bakteri *Pyricularia Grisea*, menyebabkan malai bisa mengerut dan butiran padi bisa terisi setengah bahkan kosong. Dengan gejala daun berwarna abu-abu agak putih dan tepi pinggir berwarna coklat sedikit oranye, berbentuk belah ketupat dengan bagian tepi agak runcing [2]. *Tungro*, yang ditularkan oleh wereng hijau, yang menyebabkan menguning hingga oranye, pertumbuhan terhambat, dan berkurangnya produksi gabah [3]. Sebaliknya, tanaman padi sehat tidak menunjukkan tanda-tanda penyakit. Gambar 1 mengilustrasikan karakteristik visual jenis penyakit padi.

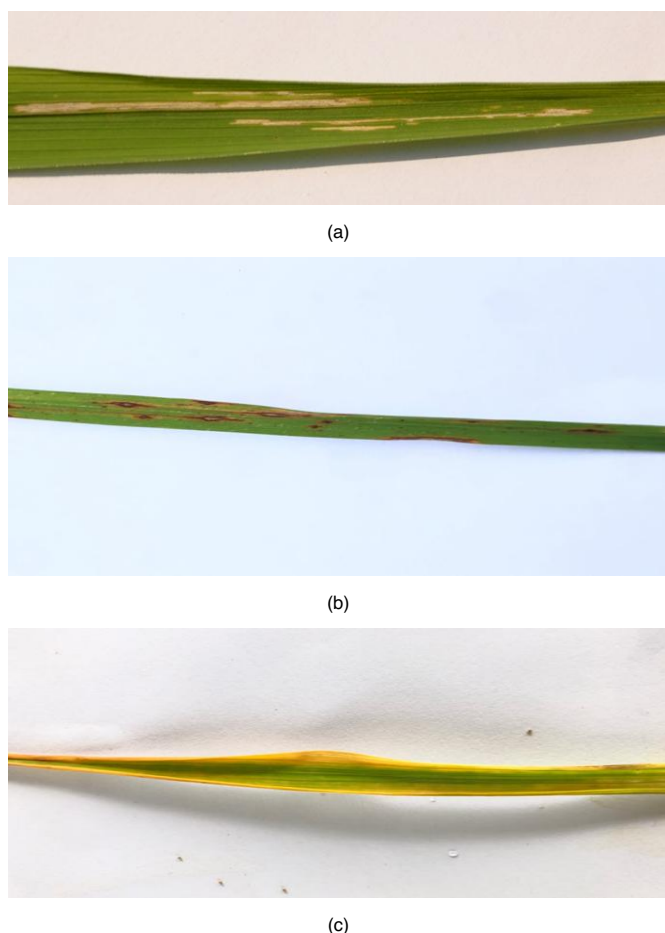
Beberapa penelitian sebelumnya telah dilakukan, dengan menggunakan metode klasifikasi seperti *Decision Tree*, *Artificial Neural Network*, *Convolutional Neural Network*, *Naïve Bayes*, *K-Nearest Neighbors*, dan *Support Vector Machine*. Meskipun masing-masing metode penelitian memberikan kontribusi yang signifikan, di dalamnya masih memiliki keterbatasan yang bisa disebabkan oleh banyak hal seperti data, keunggulan model atau arsitekturnya, dan lain-lain.

Sehingga, penelitian tentang klasifikasi penyakit padi masih memiliki potensi untuk dikembangkan. Oleh karena itu, penyempurnaan lebih lanjut diperlukan untuk meningkatkan akurasi dan performa metode klasifikasi, guna memastikan kategorisasi penyakit yang lebih andal.

Seperti penelitian sebelumnya tentang klasifikasi penyakit padi yang hanya berfokus pada data citra daun sehat dan bercak coklat. Metode yang diusulkan ialah *Gray-level Co-occurrence Matrix* (GLCM) dan *Intensity-Level Based Multi-Fractal Dimension* (ILMFD), diikuti oleh klasifikasi dengan teknik *Support Vector Machine* (SVM), *Artificial Neural Network* (ANN), dan *Neuro-Genetic Algorithm* (Neuro-GA) untuk dijadikan perbandingan. Hasil menunjukkan Neuro-GA lebih baik daripada model ANN, tetapi SVM terbukti lebih unggul dari Neuro-GA, mencapai akurasi 96,68% [4].

Penelitian lain, menggunakan variasi dataset yang telah tersedia pada *UCI Machine Learning Repository* terdiri dari *bacterial leaf blight*, *brown spots*, dan *left smut*. Metode *Artificial Neural Network* (ANN) digunakan dalam proses klasifikasi dengan 100 *epoch* mencapai akurasi 83%, presisi 84%, *recall* 84%, dan *F1-score* 83% [5]. Penelitian ini masih memiliki keterbatasan yaitu jumlah dataset hanya 120 yang tergolong kecil dan tidak ada proses augmentasi data.

Dengan dataset yang sama, peneliti lain juga melaporkan bahwa metode yang diusulkan yaitu *transfer learning* dengan *pretrained Resnet101* diperoleh akurasi validasi 100% dengan nilai *loss* 5,61% [6]. Performa validasi tersebut menunjukkan bahwa model mampu mempelajari pola pada dataset validasi dengan sangat efektif. Akan tetapi karena penelitian ini hanya memproses hingga tahap validasi, sementara tahap



**Gambar 1.** Gambar jenis penyakit daun padi, (a) *blight*, (b) *blast*, (c) *tungro*.

pengujiannya tidak dilakukan, belum bisa dipastikan model dapat melakukan generalisasi dengan baik pada data yang tidak terlihat. Tahap pengujian dalam penelitian sangat penting dilakukan agar kemampuan model dapat dievaluasi.

Studi lain juga mengklasifikasikan penyakit padi dengan jenis penyakit *leaf blight*, *brown spots*, dan *leaf smut*. *Gray-Level Co-occurrence Matrix* (GLCM) digunakan sebagai ekstraksi fitur dan *K-Nearest Neighbor* (KNN) sebagai model klasifikasi, diperoleh akurasi uji 65,83% [7]. Ini membuktikan bahwa metode KNN juga masih belum mampu mendapatkan performa yang baik.

Selain itu juga, para ahli yang lain melakukan penelitian dengan menggunakan *Decision Tree*. Dalam penelitian ini, fitur-fitur diekstraksi menggunakan segmentasi dan morfologi. Didapatkan akurasi pengujian 90% [8]. Sementara data yang digunakan hanya berjumlah 120 dan metode *Decision Tree* tidak efektif untuk menangani data yang kompleks.

Penggunaan model *Naive Bayes* yang digunakan untuk mengklasifikasikan penyakit padi *blight*, *blast*, dan *tungro*. Hasil akurasi mencapai 76% dan akurasi keyakinan 100% pada kelas *blight* [9]. Hasil yang didapatkan menunjukkan bahwa kinerja model kurang optimal dan teknik augmentasi yang tidak dilakukan menjadikan keterbatasan dalam menghadapi keragaman data.

Selanjutnya, penelitian lain juga menggunakan variasi dataset yang terdiri dari *tungro*, *blast*, *leaf blight*, dan *brownspot* dengan metode *Convolutional Neural Network* (CNN). Dengan jumlah keseluruhan data sebanyak 320, didapatkan hasil akurasi sebesar 91,4% [10]. Dataset yang relatif kecil dan tidak ada penanganan terhadap *overfitting*

seperti augmentasi/*cross validation* menjadi keterbatasan dalam penelitian ini. Selain itu juga, untuk CNN sendiri masih memiliki banyak potensi pengembangan arsitektur mulai dari jumlah *hidden layer*, *epoch*, *learning rate*, *batch size*, ukuran kernel, metode penghitungan *loss*, metode optimasi, dan penambahan dataset.

Riset yang relevan juga menunjukkan semakin bervariasi dan jumlah data yang besar menunjukkan adanya performa model yang meningkat. Dengan data yang digunakan pada data latih berjumlah 4000 dan data uji berjumlah 300, didapatkan akurasi uji 95,67%. Penelitian ini menggunakan arsitektur *InceptionResNetV2* dan metode *transfer learning* [11]. Meskipun akurasi yang didapatkan cukup tinggi, data yang digunakan memiliki karakteristik *background* yang tidak seimbang. Pada kelas *blast* dan *blight* memiliki latar belakang bertumpuk, sementara pada kelas *brownspot* dan *healthy* hanya berupa satu lembar daun padi dengan latar belakang putih polos. Sehingga memungkinkan model mengalami *overfitting* terhadap data. Dalam hal ini, model tidak fokus terhadap fitur dari daun, melainkan hanya fokus pada latar belakang.

Dengan demikian penelitian ini, mengusulkan penggunaan metode *Convolutional Neural Network* (CNN) untuk mengklasifikasikan penyakit padi berbasis citra daun. Dataset citra daun dipilih karena pada umumnya gejala penyakit padi muncul pertama kali pada daun dalam bentuk perubahan warna dan terdapat bercak sebelum menyebar ke bagian lain seperti batang atau malai. Sementara, CNN dipilih karena mampu secara efektif dalam mempelajari dan mengekstraksi fitur-fitur penting secara manual. Dalam artian, CNN tidak membutuhkan proses ekstraksi fitur tambahan. karena di dalamnya sudah mengandung proses ekstraksi fitur yaitu konvolusi. Dan telah terbukti, dalam proses pengklasifikasian objek dalam gambar, CNN mengungguli metode *machine learning* lainnya seperti SVM [12].

Keberhasilan CNN dalam mengklasifikasikan objek dalam gambar, serta keunggulannya dibandingkan metode pembelajaran mesin lainnya, menjadikannya pilihan ideal untuk mendapatkan performa yang tangguh dan andal dalam mendiagnosis penyakit padi. Penting untuk diklarifikasi bahwa studi ini berfokus pada klasifikasi dalam bidang *Intelligent Systems*, tanpa membahas pada implementasi waktu nyata atau otomatisasi dalam lingkungan pertanian. Penelitian ini berkontribusi pada aspek mendasar tersebut, yang nantinya dapat dimanfaatkan dalam aplikasi pertanian cerdas yang lebih luas. Penelitian ini bertujuan untuk membuat arsitektur CNN yang dirancang khusus untuk klasifikasi penyakit padi, memastikan bahwa model tersebut mencapai kinerja yang kompetitif dibandingkan dengan model klasifikasi yang ada.

Meskipun penelitian sebelumnya mungkin telah mencapai akurasi yang baik, akan selalu ada ruang untuk perbaikan. Kemampuan ini menjadikan CNN sebagai alat yang ampuh untuk mengklasifikasikan penyakit daun padi berdasarkan karakteristik visualnya.

Berdasarkan pembahasan di atas, penelitian ini memberikan beberapa kontribusi utama. Pertama, mengoptimalkan arsitektur CNN sederhana untuk klasifikasi penyakit padi, yang bertujuan meningkatkan akurasi dan kinerja tanpa meningkatkan kompleksitas model. Kedua, melakukan evaluasi mendetail terkait akurasi, presisi, *recall*, dan *f1-score* sebagai alat ukur kinerja model dalam membedakan berbagai penyakit padi. Ketiga, melakukan analisis perbandingan kinerja antara model klasifikasi CNN yang diusulkan dan model CNN lainnya, yang dapat

memberikan pemahaman tentang efektivitas pendekatan yang diusulkan.

## II. METODOLOGI

### A. Pengumpulan Data

Pada penelitian ini, dataset penyakit padi berdasarkan citra daun dikumpulkan melalui Kaggle. Pemilihan dataset ini didasarkan karena data telah diverifikasi dan divalidasi oleh Kaggle sehingga dapat digunakan untuk analisis lebih lanjut seperti klasifikasi. Dataset yang digunakan terdiri dari jenis daun penyakit padi *blight* sebanyak 220 data, *blast* 200 data, *tungro* 80 data, dan penambahan data *healthy* 210 data, dengan keseluruhan data sebanyak 710 data. Tabel I merupakan contoh sampel yang digunakan dalam penelitian ini.

### B. Skema Data Pelatihan dan Data Pengujian

Pada penelitian ini, dataset penyakit padi dilakukan proses pemisahan data menjadi dua kelompok utama yaitu pelatihan dan pengujian. Data latih diambil sebanyak 80% dari data untuk digunakan dalam pelatihan model. Ini merupakan proses model melakukan pembelajaran terhadap data untuk mengenali pola atau fitur yang relevan. Proses pelatihan yang dilakukan meliputi *resize*, *augmentation*, dan *normalization*. Setelah itu, dilakukan pembentukan model CNN untuk mendapatkan bobot dan bias yang baru. Kemudian, bobot dan bias yang baru ini akan disimpan untuk digunakan pada saat pengujian.

Setelah model dilatih, 20% kumpulan data yang tersisa digunakan untuk data uji. Proses pengujian juga dilakukan dengan cara yang sama, yaitu *resize* dan *normalization*. Data yang telah diproses ini kemudian dimasukkan ke dalam CNN, dengan menggunakan bobot dan bias yang dipelajari sebelumnya untuk mengklasifikasikan data uji, menghasilkan hasil klasifikasi akhir berupa jenis kondisi padi.

### C. Pra-proses





Praprosesan citra merupakan langkah penting dalam menyiapkan data untuk model CNN, yang melibatkan teknik seperti *resizing* dan *augmentasi*. Karena fitur warna sangat penting untuk klasifikasi, maka tidak ada konversi warna yang diterapkan.

Pertama, dilakukan proses *resize* citra untuk memastikan keseluruhan data agar seragam. Hal ini karena CNN memerlukan input citra dengan ukuran keseluruhan data yang konsisten. Jika ukuran citra bervariasi, model CNN tidak dapat memprosesnya. Dalam penelitian ini, citra diubah ukurannya menjadi 64 x 64 piksel. Nilai piksel ini dipilih karena relatif kecil dan tidak terlalu membebankan proses komputasi.

Kemudian *augmentasi* digunakan untuk meningkatkan keragaman data pelatihan dan meningkatkan generalisasi model [13] dengan menerapkan rotasi yang memungkinkan model belajar objek dari posisi yang berbeda. Dan juga diterapkan teknik *augmentasi* lain seperti *flip*, *shift*, *shear*, dan *zoom*, yang memungkinkan model mengenali objek dalam orientasi yang berbeda, memperkaya representasi data dan mengurangi *overfitting*. Teknik-teknik ini akan membantu kemampuan model pada saat proses pengujian dan meningkatkan akurasi dengan adanya orientasi dan representasi data yang melimpah.

Terakhir, diterapkan normalisasi pada data pelatihan dan data pengujian. Dengan cara menskalakan nilai piksel antara 0 dan 1 untuk memastikan konsistensi dan meningkatkan efisiensi pembelajaran model. Hal ini membantu untuk

TABEL I  
CONTOH SAMPEL GAMBAR DARI DATASET YANG DIGUNAKAN

No.	Kondisi Daun Padi	Gambar
1.	<i>Blast</i>	
2.	<i>Blight</i>	
3.	<i>Tungro</i>	
4.	<i>Healthy</i>	

mencegah gradien terlalu besar yang dapat memperlambat proses pelatihan.

### D. Arsitektur CNN Usulan

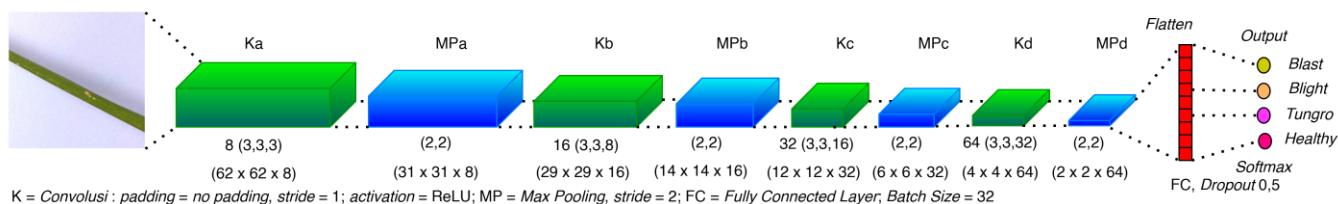
Gambar 2 merupakan arsitektur CNN yang diusulkan untuk mengklasifikasikan penyakit padi. Dirancang untuk mengoptimalkan deteksi fitur pada citra berukuran 64x64 piksel yang memiliki 4 *hidden layer*. *Hidden layer* pertama, memiliki *convolutional layer* (Ka) dengan penerapan filter 3x3 sebanyak 8 dan dengan data input 3 (RGB). 62x62x8 adalah output dari *convolutional layer* (Ka), proses mendapatkan dimensi output tersebut didapatkan dari perhitungan konvolusi 'no padding'. Setelah melalui proses konvolusi, selanjutnya akan diproses oleh ReLU (*Rectified Linear Unit*) untuk menangani masalah nonlinearitas. Selanjutnya, lapisan *max pooling* (Mpa) memproses dengan menggunakan filter 2x2 dan 31x31x8 adalah output dari lapisan *max pooling* (Mpa).

*Hidden layer* kedua, memiliki *convolutional layer* (Kb) yang diterapkan 16 filter berukuran 3x3, dengan 8 adalah dimensi inputnya yang didapatkan dari output pada proses *hidden layer* pertama. 29x29x8 adalah output dari *convolutional layer* (Kb) dengan cara menerapkan perhitungan konvolusi 'no padding'. ReLU diterapkan untuk menangani data yang nonlinearitas. Selanjutnya *max pooling layer* memproses dengan menggunakan filter berukuran 2x2 sehingga didapatkan output berukuran 14x14x16.

*Hidden layer* ketiga, *convolutional layer* (Kc) diterapkan 32 filter berukuran 3x3, dengan data input yang diperoleh dari output *hidden layer* kedua yaitu 16. Lapisan ini juga menerapkan konvolusi 'no padding'. Sehingga berdasarkan aturan tersebut didapatkan output konvolusi (Kc) 12x12x32. Selanjutnya, untuk fungsi aktivasinya juga diterapkan ReLU. Kemudian, diproses oleh lapisan *max pooling* yang memiliki filter berukuran 2x2 dan output *max pooling*-nya adalah 6x6x32.

*Hidden layer* terakhir, memiliki *convolutional layer* (Kd) dengan filter berukuran 3x3 yang berjumlah 64 filter. Data inputnya diperoleh dari proses *hidden layer* ketiga yaitu 32.





Gambar 2. Arsitektur Usulan CNN

Dengan aturan konvolusi ‘no padding’ didapatkan keluaran  $4 \times 4 \times 64$ . Fungsi aktivasi juga sama yaitu menggunakan ReLU. Kemudian, lapisan *max pooling* memproses menggunakan filter  $2 \times 2$ , sehingga mendapatkan output  $2 \times 2 \times 64$ .

Setelah proses ekstraksi fitur selesai, hasilnya diratakan yang sebelumnya bersifat multidimensi menjadi satu dimensi oleh *flatten layer*. Sehingga dapat diproses oleh lapisan *fully connected*. Ouputnya berupa data mentah atau logit. Kemudian teknik *dropout* digunakan untuk mengatasi *overfitting* yang bernilai 0,5. Dan akhirnya, pada lapisan *softmax* diproses untuk mengonversi yang sebelumnya bersifat data mentah menjadi probabilitas klasifikasi akhir. Yang memungkinkan model untuk menentukan jenis penyakit padi dengan memilih akurasi tertinggi berdasarkan input citra dengan resolusi kecil. Detail setiap langkah akan dijelaskan lebih lanjut dalam bagian berikut.

### 1) Convolutional Layer

Dalam arsitektur yang diusulkan, *Convolutional layer* bertugas mengekstraksi fitur dari data input [14] yang berupa citra daun padi. Bekerja dengan cara menerapkan serangkaian filter ke citra input untuk dilakukan proses ekstraksi fitur. Fitur-fitur ini meliputi tekstur, warna atau bentuk corak yang muncul pada citra daun padi. Persamaan (1) merupakan proses perhitungan output dalam *convolutional layer* yang dinyatakan secara matematis [15].

$$Y(p, q) = \sum_i \sum_j X(p, q) * F(p - i, q - j) \quad (1)$$

Dengan  $Y(p, q)$  adalah output dari operasi konvolusi pada koordinat  $(p, q)$ ,  $X(p, q)$  adalah input gambar pada koordinat  $(p, q)$ ,  $F(i, j)$  adalah filter atau kernel konvolusi pada posisi  $(i, j)$ ,  $p$  dan  $q$  menunjukkan koordinat output, dan  $i$  dan  $j$  menunjukkan indeks spasial pada filter. Masing-masing filter akan menghasilkan satu fitur map setelah melakukan konvolusi dengan input. Jadi, jika ada 8 filter dalam satu *convolutional layer*, maka ada 8 fitur map yang dihasilkan.

Pada saat *convolutional layer* bekerja, *stride* dan *padding* berperan penting dalam menentukan proses ekstraksi fitur dari citra daun padi. Pada penelitian ini, nilai *stride* yang digunakan adalah 1. Ini memastikan bahwa filter konvolusi bergerak 1 piksel untuk setiap kali perpindahan ke posisi berikutnya. Sementara *padding* tidak digunakan karena untuk mengurangi dimensi tinggi dan lebar input. Dengan tidak menggunakan *padding*, model akan mengurangi beban komputasi sambil tetap mempertahankan fitur yang relevan dengan penyakit padi.

Pada persamaan (2) dan (3), merupakan rumus untuk menentukan dimensi output konvolusi menggunakan aturan konvolusi tanpa *padding* [16] yang digunakan dalam penelitian ini.

$$W_2 = \frac{W_1 - F_1}{S} + 1 \quad (2)$$

$$H_2 = \frac{H_1 - F_2}{S} + 1 \quad (3)$$

Dalam hal ini,  $W_1$  dan  $H_1$  adalah lebar dan tinggi dari input matriks.  $F_1$  dan  $F_2$  ialah lebar dan tinggi filter/kernel. Dan  $S$  merupakan *stride* atau langkah kernel yang diterapkan pada data input.

### 2) Rectified Linear Unit (ReLU)

Dalam penelitian ini, fungsi aktivasi ReLU digunakan untuk mengatasi masalah nonlinearitas data penyakit padi berbasis citra daun secara efektif. Persamaan (4) adalah rumus ReLU yang digunakan dengan cara mengubah nilai negatif menjadi 0 dan mempertahankan nilai positif [17].

$$\hat{y}_a(p, q) = \begin{cases} 0 & Y(p, q) < 0 \\ Y(p, q) & Y(p, q) \geq 0 \end{cases} \quad (4)$$

$Y(p, q)$  adalah output dari *convolutional layer* dan  $\hat{y}_a(p, q)$  adalah output dari lapisan ReLU. Dengan menghilangkan nilai negatif, ReLU memastikan bahwa model hanya mempertahankan pola fitur penyakit yang relevan pada daun padi. Dengan hanya mempertahankan aktivasi yang positif, fungsi ReLU meningkatkan kemampuan jaringan untuk fokus pada fitur yang membedakan antara daun sehat dan daun yang terkena penyakit tertentu.

### 3) Pooling Layer

Setelah data diproses oleh aktivasi ReLU, selanjutnya data akan diproses oleh *pooling layer* untuk mengurangi dimensi spasial peta fitur yang dapat mengurangi beban komputasi dan mengurangi kompleksitas parameter dalam jaringan [18]. Sesuai dengan Gambar 2 jenis *pooling layer* yang digunakan yaitu *max pooling* dengan cara mengambil nilai maksimum dari setiap peta fitur. Ini memastikan fitur yang paling penting dan dominan pada daun yang terindikasi penyakit padi dipertahankan sambil menghilangkan informasi yang kurang signifikan. Dan diterapkan *stride* 2, yang berarti filter bergerak dua piksel sekaligus, mengurangi ukuran peta fitur. Persamaan (5) dan (6) adalah rumus untuk menentukan dimensi outputnya [19].

$$W_2 = \frac{W_1 - F_1}{S} \quad (5)$$

$$H_2 = \frac{H_1 - F_2}{S} \quad (6)$$

### 4) Flatten Layer

Setelah proses *pooling layer* pada *hidden layer* terakhir, data akan diproses oleh *flatten layer* untuk mengubah data yang sebelumnya berbentuk multidimensi menjadi satu dimensi. Ini merupakan langkah penting, agar data yang berisi fitur yang telah diekstraksi dapat diproses lebih lanjut yaitu pada *fully connected layer*.

### 5) Fully Connected Layer

*Fully Connected Layer* adalah bagian terpenting dari arsitektur model pada penelitian ini. Pada tahap ini, model mulai mengidentifikasi pola penyakit tertentu, berdasarkan fitur yang telah diekstraksi, yang kemudian diolah dan digabungkan menjadi output mentah (logit). Setiap setelah lapisan *fully connected*, diterapkan ReLU dan *Dropout* 50% (0.5). *Dropout* berfungsi untuk mengatasi *overfitting* [6] dengan cara menonaktifkan 50% neuronnya secara acak selama pelatihan. Hal ini mencegah ketergantungan model terhadap neuron tertentu dan meningkatkan kemampuannya dalam menggeneralisasi ke citra daun padi yang baru terlihat, sehingga meningkatkan akurasi model.

#### 6) Softmax dan Negative Log-Likelihood (NLL) Loss

Pada tahap ini, *softmax* berfungsi mengubah data mentah (logit) dari *fully connected layer* menjadi distribusi probabilitas. Dengan memastikan nilai tidak negatif dan jumlah keseluruhan probabilitas ialah satu. Persamaan (7) ialah rumus untuk menentukan keluaran *softmax* [20].

$$S(\widetilde{y}_{ak}) = \frac{e^{\widetilde{y}_{ak}}}{\sum_{i=1}^K e^{\widetilde{y}_{ak}}} \quad (7)$$

Dalam hal ini,  $S$  adalah output *softmax* di kelas ke- $k$ ,  $\widetilde{y}_{ak}$  adalah nilai output dari *fully connected layer*,  $e$  adalah bilangan euler sekitar 2,718, dan  $K$  adalah total kelas.

Setelah mendapatkan distribusi probabilitas, dilakukan perhitungan kerugian menggunakan *Negative Log Likelihood (NLL) Loss*. *NLL Loss* merupakan perhitungan kerugian yang mirip dengan *cross entropy loss*. Perbedaannya, jika *cross entropy loss* dilakukan perhitungan secara berpisah dengan *softmax*. Namun, jika *NLL loss* sudah mengandung *softmax* di dalamnya. *NLL* memberikan penalti terhadap prediksi yang salah [20] dilakukan dengan cara (8).

$$E = -\hat{y} \log \widetilde{y}_{ak} \quad (8)$$

Dalam hal ini,  $\hat{y}$  adalah label sebenarnya (bernilai 1 untuk benar dan 0 jika salah) dan  $\widetilde{y}_{ak}$  adalah probabilitas prediksi model untuk kelas ke- $k$ . Disini, keluaran *NLL* dapat dikatakan prediksi probabilitasnya lebih baik apabila nilai keluarannya lebih rendah. Keluaran *NLL* yang lebih rendah menunjukkan model berhasil mengidentifikasi pola penyakit padi pada citra daun dengan lebih baik.

#### 7) Backpropagation dan Perhitungan Gradien

Setelah mendapatkan nilai *Loss*, nilai tersebut akan digunakan pada proses *backpropagation*. Dengan tujuan untuk mendapatkan bobot dan bias yang baru. Pada penelitian ini menggunakan optimasi Adam.

Optimasi Adam ialah salah satu metode yang digunakan untuk memperbaiki bobot dan bias [21]. Ini merupakan algoritma pengoptimalan laju pembelajaran adaptif yang dirancang khusus untuk melatih *deep neural network* [22]. Dan juga merupakan gabungan dari dua optimasi yaitu RMSprop dan Momentum. Pada persamaan (9), (10), (11), dan (12) adalah rumus yang digunakan untuk memperbaiki bobot [23].

$$m_w = \beta_1 * m_{w-1} + (1 - \beta_1) * \left(\frac{\partial E}{\partial w}\right) \quad (9)$$

$$\frac{\partial E}{\partial w} = \delta \cdot x \quad (10)$$

$$v_w = \beta_2 * v_{w-1} + (1 - \beta_2) * \left(\frac{\partial E}{\partial w}\right)^2 \quad (11)$$

$$w = w - \left(\frac{\alpha * m_w}{\sqrt{(v_w + \epsilon)}}\right) \quad (12)$$

Sementara untuk mendapatkan bias yang baru tertera pada (13), (14), (15), dan (16).

$$m_b = \beta_1 * m_{b-1} + (1 - \beta_1) * \left(\frac{\partial E}{\partial b}\right) \quad (13)$$

$$\frac{\partial E}{\partial b} = \delta \quad (14)$$

$$v_b = \beta_2 * v_{b-1} + (1 - \beta_2) * \left(\frac{\partial E}{\partial b}\right)^2 \quad (15)$$

$$b = b - \left(\frac{\alpha * m_b}{\sqrt{(v_b + \epsilon)}}\right) \quad (16)$$

Pada persamaan ini,  $m_w$  dan  $m_b$  adalah momentum *term* pada bobot dan bias;  $v_w$  dan  $v_b$  adalah *velocity term* pada bobot dan bias;  $\beta_1$  adalah parameter untuk momen pertama (biasanya 0,9);  $\beta_2$  adalah parameter untuk momen kedua (biasanya 0,999);  $\frac{\partial E}{\partial w}$  dan  $\frac{\partial E}{\partial b}$  adalah gradien *loss function* terhadap bobot dan bias;  $m_{w-1}$  dan  $m_{b-1}$  adalah momen bobot dan bias pada waktu sebelumnya;  $v_{w-1}$  dan  $v_{b-1}$  adalah *velocity* bobot dan bias pada waktu sebelumnya;  $w$  dan  $b$  ialah parameter bobot dan bias yang diperbarui;  $\alpha$  ialah *learning rate*;  $\epsilon$  ialah konstanta kecil untuk mencegah pembagian dengan nol (biasanya  $10^{-7}$ );  $\delta$  ialah *loss* untuk neuron keluaran; dan  $x$  ialah input yang terkait dengan bobot tersebut dalam jaringan *neural*.

Proses *backpropagation* akan terus berjalan hingga mencapai *epoch* yang ditentukan. Setelah itu, bobot dan bias yang baru disimpan dan digunakan untuk proses *testing*.

#### 8) Evaluasi

Untuk mengukur kinerja model terhadap data pengujian dapat dilakukan dengan cara evaluasi model. Metode yang digunakan meliputi akurasi dilakukan menggunakan (17), presisi menggunakan (18), *recall* menggunakan (19), dan *F1-Score* menggunakan (20). Akurasi adalah teknik untuk mengukur seberapa sering model memprediksi data uji secara keseluruhan dengan benar. Semakin nilai mendekati 1 maka dapat dikatakan model memiliki performa yang bagus. Dan berlaku sebaliknya, semakin nilai mendekati 0 maka dapat dikatakan model belum mencapai performa terbaik. Presisi adalah suatu teknik yang mengukur prediksi positif yang benar dari keseluruhan kelas positif. *Recall* adalah suatu teknik yang mengukur seberapa banyak data positif yang berhasil diprediksi dari keseluruhan data yang sebenarnya positif. Dan *F1-Score* adalah suatu teknik yang mengukur rata-rata dari *recall* dan *precision*.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (17)$$

$$Precision = \frac{TP}{TP+FP} \quad (18)$$

$$Recall = \frac{TP}{TP+FN} \quad (19)$$

$$F1 - Score = 2 \frac{Recall \times Precision}{Recall + Precision} \quad (20)$$

Selain itu, untuk mengetahui gambaran secara detail dapat diketahui dengan cara menerapkan *confusion matrix*. Seperti pada Tabel II, yang berisi rangkuman terkait kinerja model klasifikasi seperti *True Positive* (TP), *False Positive* (FP), *True Negative* (TN), dan *False Negative* (FN).

Jika pada kasus prediksi *blight*, akan bernilai TP apabila *blight* benar diprediksi sebagai *blight*. Akan bernilai FN apabila semua elemen di baris *blight* yang bukan TP. Akan bernilai FP apabila semua elemen di kolom *blight* bukan TP. Dan akan

TABEL II  
CONFUSION MATRIX

Aktual	Kelas Prediksi			
	<i>Blight</i> (A)	<i>Blast</i> (B)	<i>Tungro</i> (C)	<i>Healthy</i> (D)
<i>Blight</i> (A)	AA	AB	AC	AD
<i>Blast</i> (B)	BA	BB	BC	BD
<i>Tungro</i> (C)	CA	CB	CC	CD
<i>Healthy</i> (D)	DA	DB	DC	DD

bernilai TN, apabila semua elemen yang bukan di baris dan di kolom *blight*. Untuk di prediksi kelas yang lain penentuan TP, TN, FP, dan FN dilakukan dengan cara yang sama.

### III. HASIL DAN PEMBAHASAN

#### A. Hasil Pengujian

Tahap awal dalam penelitian ini, menggunakan beberapa skema uji coba untuk mendapatkan hasil yang terbaik. Skema uji coba ini dilakukan dengan melibatkan variasi *epoch* dan *learning rate*. Variasi *epoch* dilakukan untuk merujuk pada jumlah iterasi terhadap keseluruhan data, sedangkan variasi *learning rate* dilakukan untuk menentukan kecepatan pembaruan bobot pada saat proses *training*. Tujuan dari variasi ini adalah untuk menemukan kombinasi parameter yang paling efektif dalam meningkatkan performa yang model.

Uji coba dilakukan secara bertahap, mulai A1 hingga B4. Model A1 hingga A4 digunakan sebagai uji coba dengan parameter dengan *epoch* 30 dan variasi *learning rate* mulai dari 0,001, 0,0001, 0,0002 hingga 0,0003. *Learning rate* 0,001 dipilih karena nilai tersebut relatif rendah dan *learning rate* 0,0002 hingga 0,0003 dipilih untuk dijadikan sebagai pembanding. Sementara untuk model B1 hingga B4 digunakan sebagai uji coba dengan menggunakan *epoch* 100 dan variasi *learning rate* 0,001, 0,0001, 0,0002, dan 0,0003. Disini *epoch* ditingkatkan bertujuan untuk mengetahui pengaruh *epoch* yang diperbesar terhadap performa model.

Setelah berhasil menyelesaikan uji coba berdasarkan skema yang sudah dibuat, diperoleh hasil uji coba A1 yaitu dengan *epoch* 30 dan *learning rate* 0,001 model menunjukkan akurasi *training* 0,9018. Didapatkan nilai *loss* yang cukup rendah dikarenakan nilainya yang mendekati 0 yaitu 0,2258. Durasi *training* pada uji coba ini sekitar 112 menit 0,8 detik. Akurasi *testing* mencapai 0,9295, hal ini menunjukkan bahwa akurasi *testing* lebih tinggi daripada akurasi *training* atau yang disebut *underfitting*. Ada kemungkinan model tidak mampu mempelajari fitur kompleks yang ada di data pelatihan, tetapi dapat melakukan generalisasi yang cukup baik di data pengujian.

Lalu dilanjutkan uji coba pada A2, yaitu dengan menggunakan *epoch* 30 dan *learning rate* 0,0001. Pada saat uji coba, diperoleh untuk akurasi *training* 0,8305, dan nilai *loss* 0,3880 dengan durasi 128 menit 0,8 detik. Sementara pada data uji, didapatkan akurasi 0,8802. Berdasarkan hasil tersebut, menurunkan nilai *learning rate* akan menyebabkan durasi yang lebih lama. Ini berarti pembelajaran dengan langkah kecil akan menyebabkan model untuk mencapai titik yang optimal membutuhkan lebih banyak waktu dibandingkan dengan menerapkan langkah yang lebih besar. Terlihat jelas pada akurasi *training*, *loss*, dan akurasi *testing* yang tidak lebih optimal dibandingkan uji coba A1 sebelumnya.

Selanjutnya uji coba A3 dengan menggunakan *epoch* 30 dan *learning rate* 0,0002. Hasil uji coba diperoleh untuk

akurasi *training* meningkat menjadi 0,9234, *loss* menurun menjadi 0,1921, durasi 112 menit 6,4 detik, dan akurasi *testing* meningkat menjadi 0,9436. Berdasarkan hasil tersebut, *learning rate* yang sedikit ditingkatkan, model dapat memperoleh performa yang lebih bagus. Ini menunjukkan model membutuhkan *learning rate* yang lebih besar dari uji coba A2 dan lebih kecil dari uji coba A1 untuk mencapai konvergensi yang lebih cepat dan efektif tanpa menyebabkan peningkatan *loss*. Namun demikian, *underfitting* masih terjadi dalam uji coba ini.

Dan untuk uji coba A4 dengan menggunakan *epoch* 30 dan *learning rate* 0,0003 model menunjukkan penurunan kinerja secara keseluruhan. Hasil akurasi *training* mencapai 0,9179, *loss* 0,2069, durasi *training* 112 menit 3,7 detik, dan akurasi *testing* 0,9084. Meskipun akurasi *training* relatif tinggi, akurasi *testing* menunjukkan penurunan dibandingkan uji coba A1 dan A3, yang menunjukkan adanya *overfitting*. Sementara selama pelatihan, model bekerja sangat baik, yang ditandai dengan adanya perbandingan antara akurasi *training* dan akurasi *testing* yang tidak terlalu jauh.

Berikutnya uji coba B1 dengan menggunakan *epoch* 100 dan *learning rate* 0,001. Akurasi *training* mencapai 0,9850 dengan *loss* 0,0393 yang relatif lebih rendah daripada uji coba sebelumnya. Durasi *training* meningkat menjadi 364 menit 8,7 detik. Hal ini dikarenakan jumlah *epoch* yang ditingkatkan menyebabkan model membutuhkan lebih banyak waktu dalam melakukan pembelajaran. Akurasi *testing* diperoleh 0,9154 menunjukkan adanya *overfitting* ketika model terlalu banyak menghafal pada data pelatihan. Uji coba B1 memberikan wawasan dengan meningkatkan jumlah *epoch* yang memberikan durasi waktu pembelajaran lebih lama, tidak secara otomatis meningkatkan kinerja model.

Pada uji coba B2 menggunakan *epoch* 100 dan *learning rate* 0,0001. Dengan menurunkan nilai *learning rate*, model menunjukkan penurunan nilai akurasi *training* yang hanya mencapai 0,9421 dan nilai *loss* yang meningkat menjadi 0,1366. Durasi *training* pada uji coba ini sekitar 364 menit 8,7 detik. Sementara untuk nilai akurasi *testing* meningkat mencapai 0,9225. Hal ini, model masih menunjukkan adanya *overfitting* meskipun nilainya tidak terlalu jauh. Meskipun demikian, model telah mampu menggeneralisasi lebih baik pada data pengujian dibandingkan dengan uji coba B1.

Uji coba B3 menggunakan *epoch* 100 dan *learning rate* 0,0002. Akurasi *training* yang diperoleh mencapai 0,9930, ini merupakan akurasi tertinggi dibandingkan uji coba sebelumnya. Nilai *loss* menunjukkan penurunan dan sangat mendekati nilai 0 dengan mencapai 0,0221. Akurasi *testing* juga meningkat signifikan menjadi 0,9647, yang menunjukkan kemampuan model melakukan generalisasi dengan sangat baik. Durasi *training* tercatat 363 menit 10,8 detik yang lebih singkat daripada uji coba B1 dan B2.

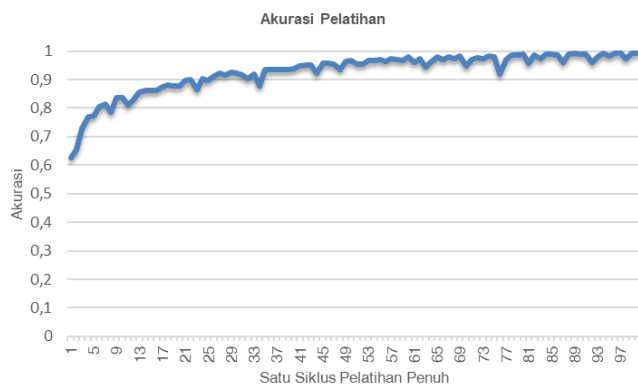
Terakhir uji coba B4 menggunakan *epoch* 100 dan *learning rate* 0,0003. Akurasi *training* yang diperoleh adalah 0,9910, dengan nilai *loss* sebesar 0,0246, dan durasi yang didapatkan sekitar 359 menit 23,2 detik. Sedangkan akurasi *testing* diperoleh 0,9225 yang menunjukkan adanya *overfitting*. Kombinasi parameter ini dinilai kurang, yang menyebabkan model gagal mencapai konvergensi dan kehilangan kemampuan untuk belajar dari data pelatihan.

Tabel III merupakan tabel hasil uji coba keseluruhan untuk mempermudah dilakukan analisis. Terlihat Tabel III, hasil menunjukkan bahwa model yang dilatih dengan 100 *epoch* cenderung menghasilkan akurasi dan *loss* yang lebih baik

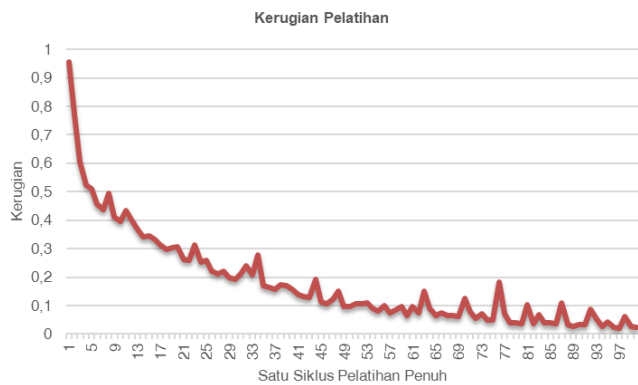


TABEL III  
HASIL UJI COBA KESELURUHAN

Uji Coba	Akurasi Training	Loss	Durasi Training	Akurasi Testing
A1	0,9018	0,2258	112 min 0,8 s	0,9295
A2	0,8305	0,3880	128 min 7,8 s	0,8802
A3	0,9234	0,1921	112 min 6,4 s	0,9436
A4	0,9179	0,2069	112 min 3,7 s	0,9084
B1	0,9850	0,0393	377 min 24,3 s	0,9154
B2	0,9421	0,1366	364 min 8,7 s	0,9225
B3	0,9930	0,0221	363 min 10,8 s	0,9647
B4	0,9910	0,0246	359 min 23,2 s	0,9225



(a)



(b)

Gambar 3. Grafik yang didapatkan pada performa terbaik yaitu uji coba B3, (a) akurasi pelatihan, (b) kerugian pelatihan.

dibandingkan model yang dilatih dengan 30 *epoch*. Yang dimungkinkan oleh semakin banyak *epoch*, model memiliki waktu yang lebih lama untuk mempelajari data sehingga mampu mendapatkan *loss* yang lebih rendah.

Pada uji coba A1 hingga A4 dan B1 hingga B4, model mampu mencapai akurasi tertinggi dengan menggunakan *learning rate* 0,0002. Ini menunjukkan *learning rate* yang lebih kecil mampu membantu model belajar lebih stabil dan mendapatkan *loss* yang lebih rendah. Walaupun pada *learning rate* yang lebih kecil dari 0,0002 yaitu 0,0001 belum mampu menghasilkan performa yang lebih baik, ini mungkin disebabkan oleh *learning rate* yang terlalu kecil sehingga

terlalu lambat dan menghambat model dalam melakukan pembelajaran.

Jadi performa terbaik diperoleh pada uji coba B3 dengan menggunakan *epoch* 100 dan *learning rate* 0,0002 yang menghasilkan akurasi *training* 0,9930, *loss* 0,0221, dan akurasi *testing* 0,9647. Variasi *epoch* dan *learning rate* secara signifikan memengaruhi kinerja model, yang menunjukkan pentingnya penyetelan parameter dalam mengoptimalkan model CNN untuk klasifikasi penyakit padi.

Gambar 3(a) merupakan grafik akurasi pada saat *training* di Uji Coba B3. Terlihat, pada permulaan akurasi pelatihan menunjukkan adanya kenaikan dengan cepat yang berarti model mampu memahami pola data pelatihan dengan sangat baik. Namun, secara bersamaan model mengalami ketidakstabilan dan terjadi secara berulang. Ketidakstabilan ini, menunjukkan bahwa model mengalami kesulitan pembelajaran dalam melakukan penyesuaian terhadap data yang kompleks. Tampaknya ini dikarenakan adanya fitur yang baru terlihat oleh model. Meskipun demikian, di akhir *epoch* model berhasil mendapatkan nilai akurasi tertinggi.

Dan Gambar 3(b) adalah grafik *loss* saat melakukan pelatihan di Uji Coba B3. Pada tahap awal nilai *loss* sangat mendekati 1, ini dikarenakan pada penelitian ini bobot diinisialisasi secara acak sehingga model belum menyesuaikan bobotnya dengan baik terhadap data. Namun setelah beberapa *epoch* terlihat adanya penurunan hingga mendekati 0 yang menunjukkan bahwa model telah mampu menyesuaikan bobotnya terhadap data pelatihan. Ini berarti kesalahan prediksi telah diminimalkan.

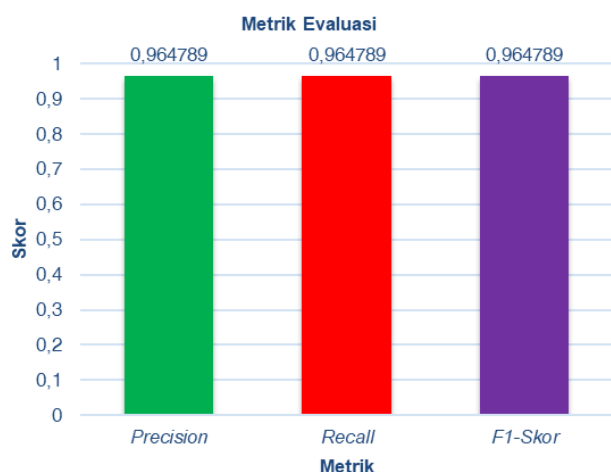
## B. Pembahasan

Berdasarkan performa terbaik (B3), selain akurasi yang digunakan untuk mengukur kinerja model atau *evaluation matrices*, juga terdapat presisi, *recall*, dan *f1-score*. Hasil *evaluation matrices* ditampilkan pada Gambar 4.

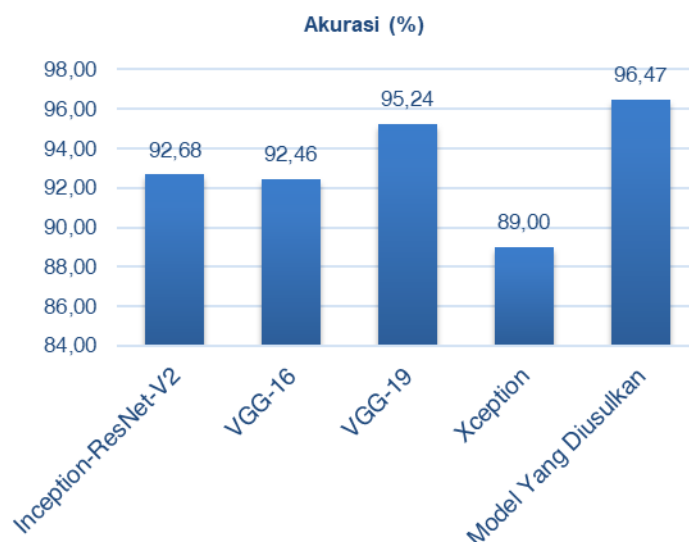
Dalam kasus penelitian ini, yaitu Klasifikasi Penyakit Padi, *recall* menjadi matriks yang lebih penting dibandingkan dengan *precision* dan *f1-score*. Dikarenakan, *recall* digunakan untuk mengukur seberapa banyak data positif yang berhasil diprediksi dari keseluruhan data yang sebenarnya positif. Hal ini artinya *recall* berfungsi untuk memastikan bahwa model dapat memprediksi semua daun yang terkena penyakit padi.

Dalam konteks pertanian, kegagalan dalam melakukan prediksi daun padi yang berpenyakit akan menyebabkan ancaman yang serius dibandingkan kesalahan dalam melakukan prediksi pada daun padi yang sehat. Oleh karena itu, *recall* menjadikan prioritas utama yang harus diperhatikan untuk meminimalkan resiko daun yang terindikasi penyakit tidak terdeteksi.

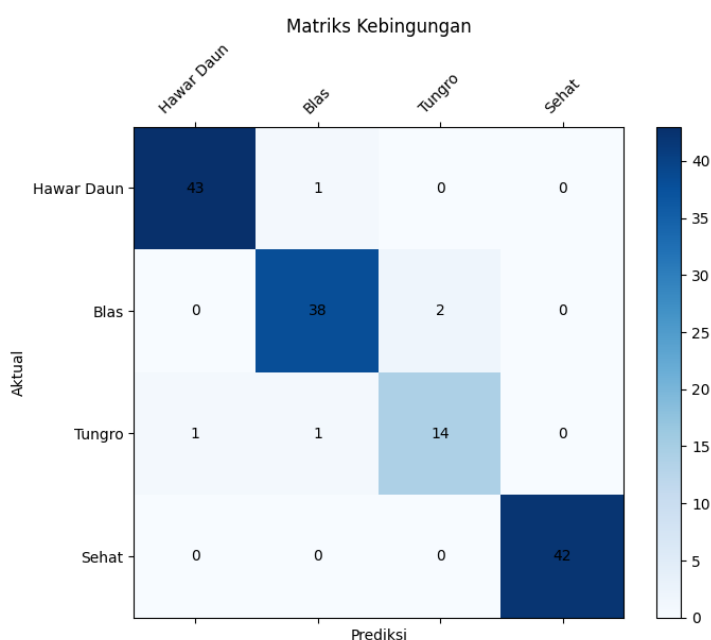
Berdasarkan Gambar 4, menunjukkan nilai yang didapatkan oleh presisi, *recall*, dan *f1-score* adalah sama. Ini berarti model mampu mengklasifikasikan data dengan sangat baik, tanpa adanya bias terhadap kelas tertentu. Presisi tinggi, artinya model jarang mengklasifikasikan daun yang sehat sebagai daun yang tidak sehat. *Recall* tinggi, artinya model sebagian besar mampu mendeteksi daun yang terinfeksi penyakit. Dan, *F1-Score* yang tinggi artinya model bekerja dengan sangat baik dalam menghindari kesalahan prediksi baik itu daun yang sehat dan daun yang terinfeksi penyakit. Kesamaan nilai akurasi, presisi, *recall*, dan *f1-score* ini menunjukkan model mampu memberikan performa yang optimal untuk semua kelas.



Gambar 4. Hasil Evaluation Metrics B3



Gambar 6. Perbandingan Dengan Model CNN Yang Sudah Ada



Gambar 5. Hasil Confusion Matrix B3

Selain itu untuk mengetahui distribusi prediksi secara detail ditampilkan dalam confusion matrix pada Gambar 5. Nilai diagonal menunjukkan klasifikasi yang benar, dengan 43 sampel hawar daun, 38 sampel blas, 14 sampel tungro, dan 42 sampel sehat diprediksi dengan benar. Namun beberapa kesalahan terjadi, seperti 1 sampel hawar daun diklasifikasikan sebagai blas, 2 sampel blas sebagai tungro, 1 sampel tungro sebagai hawar daun, dan 1 sampel tungro sebagai blas. Intensitas warna dalam matriks secara visual membedakan antara nilai tinggi dan rendah, membantu menilai kinerja model dalam membedakan antara berbagai penyakit padi dan kondisi sehat.

Terdapat beberapa penelitian sebelumnya yang dapat dijadikan perbandingan dalam penelitian ini, dengan menggunakan varian CNN yang ada seperti yang dirangkum pada Gambar 6. Pertama, Inception-ResNet-V2 dengan pendekatan *transfer learning* menggunakan model *pre-trained* dan melakukan penyesuaian pada lapisan terakhir. Penggunaan ini memberikan durasi pelatihan yang singkat dan hasil akurasi pengujian mencapai 92,68% [24]. Kedua, VGG-16 dengan

*transfer learning* memanfaatkan bobot *pre-trained* dari ImageNet menghasilkan akurasi sebesar 92,46% [25]. Ketiga, VGG-19 merupakan versi VGG-16 yang disempurnakan mencapai performa lebih baik dengan akurasi 95,24% [26]. Keempat, Xception dengan penggunaan teknik *depthwise separable convolution*, menawarkan komputasi ringan dan akurasi yang baik dengan mencapai 89% [27]. Sementara metode yang diusulkan dengan mencapai angka 96,47% ini, menawarkan performa model yang dapat bersaing tetapi tetap dengan kesederhanaan.

Secara keseluruhan model CNN yang diusulkan menunjukkan peningkatan akurasi sekitar 1,23%. Salah satu faktor yang berkontribusi dalam peningkatan ini adalah pembentukan arsitektur yang tepat dengan karakteristik data yang digunakan. Dalam penelitian ini, arsitektur CNN kustom yang diusulkan, dirancang dengan beberapa pertimbangan seperti jumlah *hidden layer*, penggunaan *padding* atau tidak pada *convolutional layer*, penentuan ukuran kernel dan jumlahnya, penentuan jenis aktivasi, penentuan teknik pada lapisan *pooling*, penentuan jenis optimasi, penentuan teknik perhitungan *loss*, penentuan metode untuk mengatasi *overfitting*, jumlah *epoch*, nilai *learning rate*, dan jumlah *batch size*. Dengan penggunaan arsitektur yang lebih tepat, model dapat mempelajari fitur-fitur kompleks pada data dengan lebih maksimal, mampu mengatasi masalah *overfitting/underfitting*, dan meningkatkan akurasi. Selain itu, *preprocessing* juga berperan penting dalam meningkatkan akurasi model, sehingga kinerja yang dihasilkan lebih baik daripada model CNN lainnya yang dilakukan oleh peneliti sebelumnya yang mungkin tidak dioptimalkan.

#### IV. KESIMPULAN

Berdasarkan penelitian yang telah dilakukan, “Klasifikasi Penyakit Padi Menggunakan *Convolutional Neural Network* (CNN) Berbasis Citra Daun”, model CNN yang digunakan mampu mengklasifikasikan daun yang terinfeksi dan tidak terinfeksi secara akurat. Dengan akurasi pengujian, presisi, *recall*, dan *f1-score* diperoleh nilai yang sama yaitu 0,96. Dan kesalahan prediksi juga menunjukkan angka yang minimum dengan total 5 kesalahan. Sementara itu, pada kelas *healthy* semua data berhasil diprediksi dengan benar. Hasil tersebut



menunjukkan bahwa arsitektur CNN yang digunakan dalam pendekatan ini mampu mencapai peningkatan akurasi dan kinerja yang bersaing dengan model klasifikasi CNN yang ada, meskipun dengan kesederhanaan.

Saran untuk ke depannya, dikarenakan pada penelitian ini hanya difokuskan terhadap pembangunan model CNN secara dasar, ada beberapa kemungkinan untuk dilakukan perbaikan yang lebih optimal di penelitian yang mendatang. Pertama, disarankan untuk mengeksplorasi integrasi teknik tambahan seperti *transfer learning*, *fine tuning* atau metode lainnya yang dapat meningkatkan kinerja model. Karena penelitian ini memiliki keterbatasan hanya menggunakan arsitektur CNN yang sederhana dan tidak menggunakan teknik tambahan tersebut. Pendekatan ini dipilih untuk fokus pada kemampuan dasar CNN dalam mengenali pola visual dari citra daun padi secara langsung, sehingga hasil yang diperoleh lebih merefleksikan efektivitas dasar arsitektur CNN itu sendiri tanpa pengaruh dari teknik eksternal.

Selain itu, penelitian ini menggunakan kumpulan data yang terbatas yang diperoleh dari sumber yang tersedia untuk umum. Memperluas kumpulan data dengan gambar yang beragam dan berkualitas tinggi, dapat membantu kemampuan generalisasi model. Lebih jauh, mengeksplorasi pendekatan hibrida dengan menggabungkan metode CNN dengan metode klasifikasi lainnya, dapat memberikan ekstraksi fitur dan kinerja lebih baik.

#### KONFLIK KEPENTINGAN

Penulis menegaskan bahwa tidak ada potensi konflik kepentingan dalam penelitian ini.

#### KONTRIBUSI PENULIS

Konseptualisasi, Moh. Heri Susanto dan Irwan Budi Santoso; metodologi, Moh. Heri Susanto; perangkat lunak, Moh. Heri Susanto; validasi, Moh. Heri Susanto, Irwan Budi Santoso, Suhartono, dan Ahmad Fahmi Karami; analisis formal, Moh. Heri Susanto, Irwan Budi Santoso, dan Suhartono; Investigasi, Moh. Heri Susanto; kurasi data, Moh. Heri Susanto; penulisan—penyusunan draf asli, Moh. Heri Susanto, dan Irwan Budi Santoso; penulisan—peninjauan dan penyuntingan, Suhartono, dan Ahmad Fahmi Karami.

#### UCAPAN TERIMA KASIH

Terima kasih disampaikan kepada Jurusan Teknik Informatika, Universitas Islam Negeri Maulana Malik Ibrahim Malang atas dukungan dan bimbingan yang diberikan selama penelitian ini.

#### REFERENSI

- [1] Dinas Pertanian Kabupaten Buleleng, "Penyakit Hawar Daun Bakteri Pada Tanaman Padi," Dinas Pertanian Pemerintah Kabupaten Buleleng. Accessed: Mar. 15, 2025. [Online]. Available: [https://distan.bulelengkab.go.id/informasi/detail/artikel/55\\_penyakit-hawar-daun-bakteri-pada-tanaman-padi?](https://distan.bulelengkab.go.id/informasi/detail/artikel/55_penyakit-hawar-daun-bakteri-pada-tanaman-padi?)
- [2] Dinas Pertanian Kabupaten Buleleng, "Mengenal Penyakit Blas Pada Tanaman Padi." Accessed: Mar. 15, 2025. [Online]. Available: [https://distan.bulelengkab.go.id/informasi/detail/artikel/57\\_mengenal-penyakit-blas-pada-tanaman-padi?](https://distan.bulelengkab.go.id/informasi/detail/artikel/57_mengenal-penyakit-blas-pada-tanaman-padi?)
- [3] S. Sipi and Subiadi, "Penyakit Tungro Dan Keracunan Fe Pada Tanaman Padi," Manokwari, 2016. Accessed: Mar. 15, 2025. [Online]. Available: <https://repository.pertanian.go.id/server/api/core/bitstreams/6eb9a1de-0c26-4d49-9534-203a469b8f24/content>
- [4] S. Chaudhary and U. Kumar, "Automated Detection and Classification of Rice Crop Diseases Using Advanced Image Processing and Machine Learning Techniques," *Traitement du Signal*, vol. 41, no. 2, pp. 739–752, Apr. 2024, doi: 10.18280/ts.410216.
- [5] N. Sunandar and J. Sutopo, "Utilization of Artificial Neural Network in Rice Plant Disease Classification Using Leaf Image," *International Journal of Research In Science & Engineering*, no. 42, pp. 1–10, Feb. 2024, doi: 10.55529/ijrise.42.1.10.
- [6] U. N. Oktaviana, R. Hendrawan, A. D. K. Annas, and G. W. Wicaksono, "Klasifikasi Penyakit Padi berdasarkan Citra Daun Menggunakan Model Terlatih Resnet101," *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi)*, vol. 5, no. 6, pp. 1216–1222, Dec. 2021, doi: 10.29207/resti.v5i6.3607.
- [7] R. A. Saputra, Suharyanto, S. Wasiyanti, D. F. Saefudin, A. Supriyatna, and A. Wibowo, "Rice Leaf Disease Image Classifications Using KNN Based on GLCM Feature Extraction," in *Journal of Physics: Conference Series*, IOP Publishing Ltd, Nov. 2020. doi: 10.1088/1742-6596/1641/1/012080.
- [8] A. D. P. Alwy, M. S. N. Wahid, B. N. N. Ag, and M. M. Fakhri, "Klasifikasi Penyakit Padi Dengan Ekstraksi Fitur LBP dan GLCM," *Journal of Deep Learning, Computer Vision and Digital Image Processing*, vol. 1, no. 1, 2023, [Online]. Available: <https://journal.diginus.id/index.php/decoding>
- [9] R. Rahmi and R. A. Saputra, "Klasifikasi Penyakit Padi Melalui Citra Daun Menggunakan Metode Naive Bayes," *Jurnal Informatika dan Teknik Elektro Terapan*, vol. 12, no. 2, Apr. 2024, doi: 10.23960/jitet.v12i2.4012.
- [10] A. K. Abasi, S. N. Makhadmeh, O. A. Alomari, M. Tubishat, and H. J. Mohammed, "Enhancing Rice Leaf Disease Classification: A Customized Convolutional Neural Network Approach," *Sustainability (Switzerland)*, vol. 15, no. 20, Oct. 2023, doi: 10.3390/su152015039.
- [11] K. N. L. V. Narasimha Prasad, C. S. Pavan Kumar, B. Subedi, H. B. Abrahma, and V. E. Sathishkumar, "Rice leaf diseases prediction using deep neural networks with transfer learning," *Environ Res*, vol. 198, Jul. 2021, doi: 10.1016/j.envres.2021.111275.
- [12] I. W. Suartika E. P, A. Y. Wijaya, and R. Soelaiman, "Klasifikasi Citra Menggunakan Convolutional Neural Network (CNN) pada Caltech 101," *JURNAL TEKNIK ITS*, vol. 5, no. 1, 2016.
- [13] R. Jain, P. Nagrath, G. Kataria, V. Sirish Kaushik, and D. Jude Hemanth, "Pneumonia detection in chest X-ray images using convolutional neural networks and transfer learning," *Measurement (Lond)*, vol. 165, Dec. 2020, doi: 10.1016/j.measurement.2020.108046.
- [14] K. Azmi, S. Defit, and U. Putra Indonesia YPTK Padang Jl Raya Lubuk Begalung-Padang-Sumatera Barat, "Implementasi Convolutional Neural Network (CNN) Untuk Klasifikasi Batik Tanah Liat Sumatera Barat," *Jurnal Unitek*, vol. 16, no. 1, 2023.
- [15] A. Batool and Y. C. Byun, "A Lightweight Multi-path Convolutional Neural Network Architecture Using Optimal Features Selection for Multiclass Classification of Brain Tumor Using Magnetic Resonance Images," *Results in Engineering*, vol. 25, Mar. 2025, doi: 10.1016/j.rineng.2025.104327.
- [16] S. S. Bhat, A. Ananth, and P. S. Venugopala, "Design and Evolution of Deep Convolutional Neural Networks in Image Classification - A Review," *International Journal of Integrated Engineering*, vol. 15, no. 1, pp. 213–225, 2023, doi: 10.30880/ijie.2023.15.01.019.
- [17] I. B. Santoso and I. K. E. Purnama, "Epileptic EEG signal classification using convolutional neural networks based on optimum window length and FFT's length," in *ACM International Conference Proceeding Series*, Association for Computing Machinery, Jul. 2020, pp. 87–91. doi: 10.1145/3411174.3411179.
- [18] M. A. Rahman, M. R. Islam, M. A. H. Rafath, and S. Mhejabin, "CNN Based Covid-19 Detection from Image Processing," *Journal of ICT Research and Applications*, vol. 17, no. 1, pp. 99–113, Apr. 2023, doi: 10.5614/itbj.ict.res.appl.2023.17.1.7.
- [19] A. Ghosh, A. Sufian, F. Sultana, A. Chakrabarti, and D. De, "Fundamental concepts of convolutional neural network," in *Intelligent Systems Reference Library*, vol. 172, Springer, 2019, pp. 519–567. doi: 10.1007/978-3-030-32644-9\_36.
- [20] S. B. Shrestha, L. Zhu, and P. Sun, "Spikemax: Spike-based Loss Methods for Classification," *IEEE International Joint Conference on Neural Network*, May 2022, Accessed: Jul. 31, 2024. [Online]. Available: <http://arxiv.org/abs/2205.09845>
- [21] C. R. Rahman *et al.*, "Identification and recognition of rice diseases and pests using convolutional neural networks," *Biosyst Eng*, vol. 194, pp. 112–120, Jun. 2020, doi: 10.1016/j.biosystemseng.2020.03.020.
- [22] Wardianto, Farikhin, and D. M. K. Nugraheni, "Analisis Sentimen Berbasis Aspek Ulasan Pelanggan Restoran Menggunakan LSTM Dengan Adam Optimizer," *Journal of Information Technology and Computer Science*, vol. 8, no. 2, pp. 67–76, 2023.
- [23] A. Bhattacharjee, A. A. Popov, A. Sarshar, and A. Sandu, "Improving the Adaptive Moment Estimation (ADAM) stochastic optimizer through an Implicit-Explicit (IMEX) time-stepping approach," Mar. 2024, [Online]. Available: <http://arxiv.org/abs/2403.13704>

- [24] M. A. Islam, M. N. Rahman Shuvo, M. Shamsojjaman, S. Hasan, M. S. Hossain, and T. Khatun, "An Automated Convolutional Neural Network Based Approach for Paddy Leaf Disease Detection," (*IJACSA International Journal of Advanced Computer Science and Applications*, vol. 12, no. 1, 2021, [Online]. Available: [www.ijacsa.thesai.org](http://www.ijacsa.thesai.org)
- [25] S. Ghosal and K. Sarkar, "Rice Leaf Diseases Classification Using CNN With Transfer Learning," in *Proceedings of 2020 IEEE Calcutta Conference (CALCON)*, IEEE, 2020.
- [26] A. A. J. V. Priyanka and I. M. S. Kumara, "Classification Of Rice Plant Diseases Using the Convolutional Neural Network Method," *Lontar Komputer : Jurnal Ilmiah Teknologi Informasi*, vol. 12, no. 2, p. 123, Aug. 2021, doi: 10.24843/lkjiti.2021.v12.i02.p06.
- [27] A. R. Muslikh, D. R. I. M. Setiadi, and A. A. Ojugo, "Rice Disease Recognition Using Transfer Learning Xception Convolutional Neural Network," *Jurnal Teknik Informatika (Jutif)*, vol. 4, no. 6, pp. 1535–1540, Dec. 2023, doi: 10.52436/1.jutif.2023.4.6.1529.