BAB 2
LANDASAN TEORI

Bab ini menguraikan landasan teori dan tinjauan pustaka yang menjadi dasar penelitian. Pembahasan diawali dari domain permasalahan, yaitu tanaman jagung beserta tiga penyakit daun jagung. Selanjutnya diuraikan konsep *Convolutional Neural Network* (CNN) sebagai fondasi pengenalan citra beserta perkembangannya menuju arsitektur modern, yang mengantar pada dua arsitektur fokus penelitian, yaitu ConvNeXt-Tiny dan EfficientNetV2-S. Kemudian dibahas ImageNet, model pra-latih, dan *transfer learning* sebagai pendekatan untuk memanfaatkan pengetahuan model pra-latih pada dataset yang relatif terbatas, serta ditutup dengan metrik evaluasi yang digunakan.

2.1 Tanaman Jagung

Tergolong tumbuhan monokotil, jagung (*Zea mays* L.) merupakan tanaman pangan semusim anggota suku rumput-rumputan *(Poaceae)*. [13]. Secara taksonomi, jagung diklasifikasikan ke dalam kingdom *Plantae*, divisi *Magnoliophyta*, kelas *Liliopsida* (monokotil), ordo *Poales*, famili *Poaceae*, genus *Zea*, dan spesies *Zea mays* L. [13, 14]. Sebagai tumbuhan dengan jalur fotosintesis C4, jagung mampu memanfaatkan cahaya matahari secara efisien sehingga dapat tumbuh optimal di wilayah beriklim hangat dengan curah hujan yang memadai [14].

Tabel 2.1. Klasifikasi Ilmiah Tanaman Jagung

Tingkatan Takson	**Nama**
Kingdom	*Plantae*
Divisi	*Magnoliophyta*
Kelas	*Liliopsida* (Monokotil)
Ordo	*Poales*
Famili	*Poaceae*
Genus	*Zea*
Spesies	*Zea mays* L.

Struktur morfologi tanaman jagung mencakup akar, batang, daun, dan bunga. [15]. Daun jagung berbentuk memanjang menyerupai pita, tersusun dalam dua baris yang berseling pada sisi batang yang berlawanan, serta memiliki tulang daun sejajar dengan ibu tulang daun (*mid-rib*) yang menonjol [14]. Permukaan atas helai daun umumnya berbulu, sedangkan jumlah daun bervariasi antarvarietas,

yaitu rata-rata sekitar 15 helai pada hibrida iklim sedang hingga mencapai 48 helai pada hibrida tropis [14]. Jagung tergolong tumbuhan *monoecious* (berumah satu), karena bunga jantan dan betina berada pada individu yang sama meski letaknya terpisah, bunga jantannya berupa malai (*tassel*), sedangkan tongkol(*ear*) terbentuk dari bunga betina [14].

Bagi Indonesia, jagung memiliki peran strategis sebagai komoditas pangan penghasil karbohidrat terpenting kedua setelah padi, sekaligus sebagai bahan baku utama industri pakan ternak [2]. Daun merupakan organ vital bagi tanaman jagung karena menjadi tempat utama berlangsungnya proses fotosintesis yang menentukan pembentukan biomassa dan hasil panen. Oleh karena itu, gangguan pada daun, khususnya akibat serangan penyakit, dapat menurunkan kapasitas fotosintesis dan berdampak langsung terhadap produktivitas tanaman. Kondisi inilah yang menjadikan kesehatan daun jagung sebagai aspek penting yang akan diuraikan lebih rinci di subbab setelah ini.

2.2 Penyakit Daun Jagung

Daun jagung rentan terhadap berbagai penyakit yang disebabkan oleh jamur patogen. Di antara penyakit yang paling umum dan merusak adalah karat daun (*Common Rust*), hawar daun utara (*Northern Leaf Blight*), dan bercak daun abu-abu (*Gray Leaf Spot*). Berikut penjelasan lebih lengkap mengenai ketiga penyakit tersebut.

2.2.1 *Common Rust* (Karat Daun)

Karat daun atau *Common Rust* adalah gangguan pada daun jagung yang bersumber dari jamur *Puccinia sorghi* [16, 17]. Gejala awal berupa flek-flek klorotik (bintik kuning pucat) pada permukaan daun, yang kemudian menjadi pustula (bintil) bertepung seiring pecahnya spora menembus epidermis daun [16]. Pustula *Common Rust* berbentuk oval hingga memanjang dengan panjang sekitar 3 mm, berwarna cokelat kemerahan hingga cokelat kayu manis (*brick-red*), serta muncul pada kedua permukaan daun[16, 17]. Seiring bertambahnya umur infeksi, spora di dalam pustula berubah warna menjadi cokelat kehitaman akibat pembentukan teliospora [17]. Jaringan daun di sekitar pustula umumnya menguning dan mati; apabila infeksi parah, seluruh helai daun dapat mengering [16]. Perkembangan penyakit ini dipicu oleh suhu yang relatif sejuk (sekitar 16–

23°C) dan kelembapan relatif yang tinggi [17]. Secara visual, ciri khas yang dapat dikenali adalah sebaran pustula cokelat kemerahan yang memberikan tekstur berbintil pada permukaan daun, sehingga menurunkan kapasitas fotosintesis dan berujung pada penurunan hasil panen [7]. Berikut contoh visual dari penyakit *Common Rust*

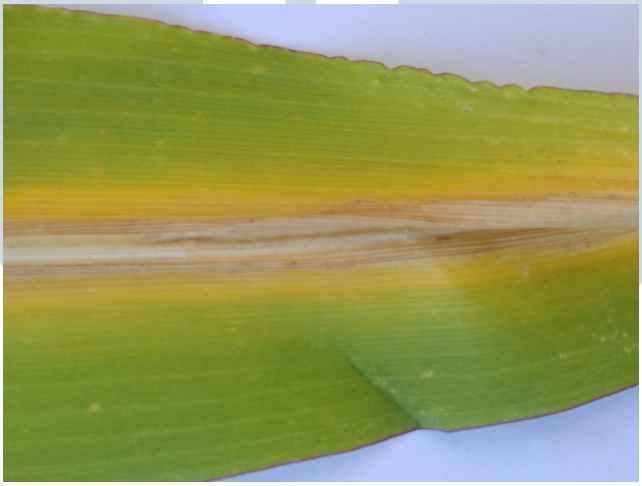
Gambar 2.1. Contoh penyakit *Leaf Rust* pada daun jagung

Sumber: [18]

2.2.2 *Northern Leaf Blight* (Hawar Daun)

Penyakit hawar daun utara (*Northern Leaf Blight*) adalah sebuah penyakit daun jagung yang diakibatkan oleh jamur *Exserohilum turcicum* (sinonim *Helminthosporium turcicum*) dan jamur tersebut dapat terus bertahan hidup pada sisa-sisa tanaman jagung yang terinfeksi[19, 20]. Gejala khas dari penyakit ini adalah lesi memanjang berbentuk elips menyerupai cerutu (*cigar-shaped*) dengan panjang berkisar antara 2–15 cm (sekitar 1–6 inci) [20, 19]. Pada tahap awal penyakit, lesi berwarna hijau keabu-abuan, kemudian berubah menjadi cokelat kekuningan (*tan*) seiring berkembangnya penyakit [20]. Lesi umumnya muncul pertama kali pada daun bagian bawah, lalu menyebar ke kanopi atas [20, 19]. ketika jamur bersporulasi, lesi tampak ”kotor” akibat spora berwarna gelap [20, 19].

Penyakit ini berkembang optimal pada suhu sedang (sekitar 18–27°C) dengan periode kelembapan atau embun yang berkepanjangan, dan umumnya muncul pada saat atau setelah fase pembungaan [19]. Serangan *Northern Leaf Blight* yang berat dapat menyebabkan kehilangan hasil panen hingga 50–70% [21, 20]. Secara visual, lesi memanjang berukuran besar yang sejajar dengan tulang daun menjadi penanda utama, namun pada beberapa kasus gejalanya mirip dengan *Gray Leaf Spot*, sehingga kerap menyulitkan identifikasi secara manual. Berikut contoh gambar dari penyakit *Northern Leaf Blight*.



Gambar 2.2. Contoh penyakit *Leaf Blight* pada daun jagung

Sumber: [18]

2.2.3 Gray Leaf Spot (Bercak Daun Abu-abu)

Penyakit bercak daun abu-abu (*Gray Leaf Spot*) adalah penyakit daun jagung yang disebabkan oleh jamur *Cercospora zeae-maydis*, yang menyebar melalui spora aseksual (konidia) [22, 23]. Gejala awal penyakit ini berupa bintik-bintik nekrotik kecil yang dikelilingi halo klorotik (lingkaran kuning), umumnya muncul pertama kali pada daun bagian bawah sekitar dua hingga tiga minggu sebelum fase pembungaan [22, 23]. Seiring perkembangan penyakit, bintik tersebut membesar menjadi lesi berbentuk persegi panjang yang sempit dengan panjang mencapai sekitar 5 cm (hingga 2 inci) dan lebar sekitar 3 mm [22, 23]. Pada tahap awal lesi berwarna cokelat muda (*tan*), kemudian berubah menjadi keabu-abuan seiring berlangsungnya sporulasi [22]. Serangan *Gray Leaf Spot* dapat menurunkan hasil panen secara signifikan, terutama apabila lesi berkembang pada daun di atas tongkol pada fase awal pertumbuhan [6, 24].

Pada tahap awal, gejala *Gray Leaf Spot* kerap sulit dibedakan dari *Northern Leaf Blight* karena keduanya sama sama muncul pada daun bagian bawah dan menampilkan lesi berwarna cokelat hingga keabu-abuan [23]. Perbedaan utama terletak pada bentuk dan batas lesi: lesi *Gray Leaf Spot* berbentuk persegi panjang dengan tepi lurus yang dibatasi oleh tulang daun, sedangkan lesi *Northern Leaf Blight* berkembang memanjang menyerupai cerutu (*cigar-shaped*) dan tidak dibatasi oleh tulang daun [25]. Kemiripan visual pada fase awal inilah yang menyebabkan identifikasi secara manual mudah mengalami kesalahan klasifikasi antara kedua penyakit tersebut, sehingga dibutuhkan solusi yang mampu mengenali perbedaan ciri visual secara lebih akurat. Berikut contoh visual dari penyakit *Gray Leaf Spot*.

Gambar 2.3. Contoh penyakit *Leaf Spot* pada daun jagung

Sumber: [18]

2.3 Convolutional Neural Network (CNN)

Convolutional Neural Network (CNN) merupakan model *deep learning* yang dirancang khusus untuk memproses data bertipe grid, seperti citra digital yang tersusun atas piksel dalam susunan dua dimensi [26]. Konsep dasar CNN adalah mempelajari fitur secara otomatis dan bertingkat langsung dari data mentah melalui operasi konvolusi, sehingga tahap ekstraksi fitur yang pada metode konvensional dilakukan secara manual dapat digantikan oleh proses pembelajaran end-to-end [27]. Berbeda dengan jaringan saraf tiruan konvensional yang menghubungkan setiap neuron ke seluruh neuron pada lapisan berikutnya (fully connected), CNN

memanfaatkan korelasi spasial lokal pada citra sehingga jauh lebih efisien dalam mengekstraksi fitur visual [26].

Terdapat tiga karakteristik utama yang menjadikan CNN sangat sesuai untuk tugas klasifikasi citra. Pertama, konektivitas lokal (*local receptive field*), yaitu setiap neuron hanya terhubung pada wilayah kecil citra, sehingga model dapat menangkap pola lokal seperti tepi dan tekstur secara efisien tanpa perlu memproses seluruh piksel sekaligus [26]. Kedua, pembagian bobot (*weight sharing*), yaitu filter yang sama digunakan di seluruh posisi citra, sehingga jumlah parameter yang harus dipelajari jauh lebih sedikit dan model mampu mengenali suatu fitur di mana pun posisinya (invariansi terhadap translasi) [26]. Ketiga, pembelajaran fitur secara hierarkis, yaitu lapisan awal menangkap fitur tingkat rendah seperti tepi dan tekstur, sedangkan lapisan yang lebih dalam menyusunnya menjadi representasi semantik tingkat tinggi yang lebih abstrak [27]. Kombinasi ketiga karakteristik inilah yang membuat CNN mampu menangani data citra berdimensi tinggi secara efisien sekaligus tangguh terhadap variasi posisi objek, sehingga menjadi pendekatan yang unggul untuk klasifikasi citra [27].

Secara arsitektural, CNN pada umumnya dibentuk oleh tiga komponen lapisan inti, yakni *convolutional layer*, *pooling layer*, dan *fully connected layer* [26]. Convolutional layer adalah lapisan inti CNN yang bertugas mengekstraksi fitur dari citra masukan. Pada lapisan ini, sebuah filter atau kernel berukuran kecil (misalnya 3×3 atau 5×5) digeser secara sistematis di atas seluruh citra untuk menghitung respons antara nilai piksel dan bobot filter. Proses ini menghasilkan sebuah *feature map* yang merepresentasikan respons filter terhadap setiap posisi spasial pada citra [26].

Setelah operasi konvolusi, umumnya diterapkan fungsi aktivasi ReLU (*Rectified Linear Unit*) untuk memperkenalkan sifat non-linieritas pada model. Fungsi ini mempertahankan nilai positif apa adanya dan mengubah seluruh nilai negatif menjadi nol, sehingga mempercepat proses pelatihan dan membantu mencegah masalah *vanishing gradient* [26].

Pooling layer berfungsi mereduksi dimensi spasial *feature map* untuk mengurangi jumlah parameter dan beban komputasi, sekaligus membuat model lebih toleran terhadap pergeseran posisi objek dalam citra [26]. Teknik yang paling umum digunakan adalah *max pooling*, yaitu mengambil nilai maksimum dari setiap jendela kecil pada *feature map* sehingga ukuran *feature map* menjadi lebih ringkas tanpa kehilangan informasi penting [26].

Fully connected layer merupakan lapisan akhir CNN yang menghubungkan

seluruh neuron dari lapisan sebelumnya ke setiap neuron pada lapisan berikutnya. Sebelum masuk ke lapisan ini, *feature map* yang dihasilkan oleh blok konvolusi–pooling diratakan (*flatten*) menjadi vektor satu dimensi [26]. Lapisan ini bertugas mengintegrasikan seluruh fitur yang telah diekstraksi dan memetakannya ke kelas-kelas keluaran menggunakan fungsi aktivasi *softmax* untuk menghasilkan probabilitas setiap kelas [26]. Dengan demikian, secara garis besar CNN bekerja melalui rangkaian tahapan ekstraksi fitur pada blok konvolusi dan pooling yang dilakukan secara berulang, dilanjutkan dengan proses klasifikasi pada fully connected layer hingga menghasilkan prediksi kelas akhir [26].

Dalam konteks pertanian, keunggulan tersebut membuat CNN terbukti menjadi metode yang andal untuk klasifikasi penyakit tanaman berbasis citra daun [27]. Kemampuan CNN dalam mengekstraksi fitur visual secara otomatis, seperti perubahan warna, tekstur bercak, hingga bentuk lesi, menjadikannya jauh lebih andal dibandingkan metode *machine learning* konvensional yang masih mengandalkan ekstraksi fitur manual [26]. Berbagai arsitektur CNN seperti VGG, ResNet, MobileNet, dan EfficientNet telah banyak diaplikasikan dan menghasilkan akurasi klasifikasi penyakit daun yang tinggi, umumnya di atas 90% [27].

2.3.1 Arsitektur CNN Tradisional dan Modern

Seiring perkembangannya, arsitektur CNN dapat dikelompokkan secara garis besar menjadi arsitektur tradisional dan arsitektur modern. Arsitektur CNN tradisional, seperti AlexNet, VGG, GoogLeNet, dan ResNet, umumnya dirancang secara manual berdasarkan intuisi dan eksperimen, dengan strategi utama menumpuk konvolusi berukuran kecil (umumnya 3×3) secara berlapis dan memperdalam jaringan untuk meningkatkan akurasi [27]. Arsitektur jenis ini umumnya menggunakan blok penyusun berupa konvolusi standar yang diikuti *Batch Normalization*, fungsi aktivasi ReLU, dan operasi *max pooling*, serta mengandalkan *inductive bias* lokal yang kuat dari operasi konvolusi [28].

Sementara itu, arsitektur CNN modern, seperti EfficientNet, EfficientNetV2, dan ConvNeXt, dikembangkan dengan pendekatan yang lebih sistematis. Sebagian dirancang menggunakan *neural architecture search* (NAS) dan penskalaan majemuk (*compound scaling*) untuk mengoptimalkan akurasi sekaligus efisiensi [12, 29], sedangkan sebagian lain mengadopsi prinsip desain *Vision Transformer* ke dalam kerangka konvolusional, seperti penggunaan kernel *depthwise* berukuran besar, *Layer Normalization*, fungsi aktivasi GELU, dan struktur *inverted bottleneck*

[1]. Perbandingan karakteristik kedua kelompok arsitektur ini dirangkum pada Tabel 2.2, sedangkan kelebihan dan kekurangan masing-masing disajikan pada Tabel 2.3.

Tabel 2.2. Perbandingan karakteristik CNN tradisional dan modern

Aspek	**CNN Tradisional**	**CNN Modern**
Contoh arsitektur	AlexNet, VGG, ResNet	EfficientNetV2, ConvNeXt
Metode desain	Manual/heuristik, memperdalam jaringan	NAS, *compound scaling*, adaptasi desain Transformer
Blok penyusun	Konvolusi 3×3 + BatchNorm + ReLU + *max pooling*	MBConv/Fused-MBConv, konvolusi *depthwise* kernel besar, LayerNorm, GELU
Efisiensi	Relatif rendah	Tinggi (akurasi tinggi dengan parameter/FLOPs lebih sedikit)
Konteks fitur	Lokal dan bertahap	Luas, meniru atensi global

Tabel 2.3. Kelebihan dan kekurangan CNN tradisional dan modern

Kelompok	**Kelebihan**	**Kekurangan**
CNN Tradisional	Sederhana dan matang, mudah diimplementasikan, dukungan pustaka serta bobot pra-latih melimpah, *inductive bias* kuat sehingga relatif hemat data	Kurang efisien secara parameter/komputasi, perlu jaringan sangat dalam untuk akurasi tinggi, konteks global terbatas, akurasi cenderung menemui titik jenuh
CNN Modern	Akurasi tinggi dengan efisiensi lebih baik, representasi fitur lebih kaya, pelatihan lebih cepat (mis. EfficientNetV2), kompetitif dengan Transformer	Arsitektur lebih kompleks, resep pelatihan lebih sensitif dan sering menuntut pra-pelatihan berskala besar, konsumsi memori lebih tinggi

Berdasarkan Tabel 2.3, penelitian ini memilih arsitektur CNN modern, yaitu *ConvNeXt Tiny* dan *EfficientNetV2-S*, karena keduanya mampu memberikan akurasi yang tinggi. Selain itu, *EfficientNetV2-S* menawarkan efisiensi komputasi yang baik, sedangkan *ConvNeXt Tiny* menyediakan representasi fitur yang kuat sehingga keduanya sesuai untuk diterapkan pada klasifikasi penyakit daun jagung berbasis *transfer learning*.

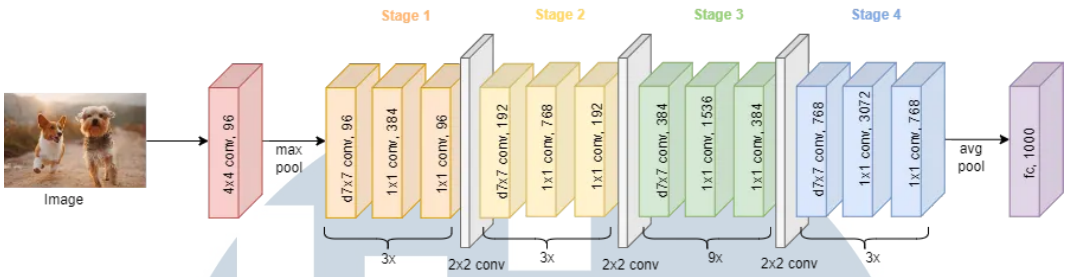
2.3.2 Vision Transformer (ViT)

Vision Transformer (ViT) merupakan arsitektur *deep learning* yang menerapkan mekanisme *Transformer*, yang semula dikembangkan untuk pemrosesan bahasa alami, secara langsung pada tugas klasifikasi citra [30]. Berbeda dengan CNN yang mengekstraksi fitur melalui operasi konvolusi, ViT terlebih dahulu membagi citra masukan menjadi sejumlah *patch* berukuran tetap (misalnya 16×16 piksel), lalu setiap *patch* diubah menjadi representasi vektor (*embedding*) yang dilengkapi informasi posisi. Rangkaian *patch* tersebut kemudian diproses sebagai sebuah urutan menggunakan mekanisme *self-attention*, yang memungkinkan model menangkap keterkaitan antarbagian citra secara global [30].

Perbedaan mendasar antara ViT dan CNN terletak pada *inductive bias*. CNN memiliki *inductive bias* berupa konektivitas lokal dan invariansi terhadap translasi yang melekat pada operasi konvolusi, sedangkan ViT mengandalkan *self-attention* global sehingga membutuhkan data pelatihan berskala sangat besar untuk mencapai performa optimal [30]. Keberhasilan ViT mendorong lahirnya arsitektur CNN modern seperti ConvNeXt, yang mengadopsi sejumlah prinsip desain ViT, seperti penggunaan kernel berukuran besar, normalisasi *Layer Normalization*, fungsi aktivasi GELU, serta struktur *inverted bottleneck*, ke dalam kerangka konvolusional murni [1].

2.4 ConvNeXt

ConvNeXt adalah arsitektur jaringan saraf konvolusional murni (pure ConvNet) yang diperkenalkan oleh Liu et al. dari Facebook AI Research (FAIR) dan UC Berkeley pada tahun 2022 melalui paper ”A ConvNet for the 2020s” yang dipresentasikan di CVPR 2022 [1]. Latar belakang kemunculannya adalah dominasi Vision Transformer (ViT) yang sejak 2020 berhasil melampaui CNN konvensional dalam berbagai benchmark klasifikasi citra. Liu et al. berargumen bahwa keunggulan tersebut bukan semata karena mekanisme self-attention pada Transformer, melainkan karena pilihan desain arsitektur yang lebih modern. Dengan ”memodernisasi” ResNet secara bertahap mengadopsi prinsip-prinsip desain dari Swin Transformer tanpa menggunakan modul attention sama sekali, mereka menghasilkan ConvNeXt, sebuah CNN murni yang mampu menyaingi bahkan melampaui Swin Transformer dalam akurasi dan skalabilitas [1].



Gambar 2.4. Arsitektur ConvNeXt

Sumber: [31]

2.4.1 Arsitektur dan Cara Kerja ConvNeXt

Salah satu modernisasi awal yang dilakukan ConvNeXt terletak pada bagian *stem*, yaitu lapisan paling awal yang memproses citra masukan. Jika ResNet konvensional menggunakan konvolusi 7×7 dengan *stride* 2 yang diikuti *max pooling* sehingga area yang diproses saling tumpang-tindih, ConvNeXt menggantinya dengan strategi *patchify*, yaitu konvolusi 4×4 dengan *stride* 4 yang bersifat non-*overlapping* [1]. Dengan demikian, citra masukan langsung dipecah menjadi *patch-patch* yang tidak saling bertindih, meniru mekanisme *patch embedding* pada Vision Transformer. Desain ini menjadikan tahap awal ekstraksi fitur lebih efisien sekaligus lebih selaras dengan paradigma *Transformer*.

Inti dari ConvNeXt adalah desain bloknya yang memiliki susunan sebagai berikut [1]:

$$\text{Input} \xrightarrow{\text{DWConv } 7\times7} \xrightarrow{\text{LayerNorm}} \xrightarrow{\text{Conv } 1\times1} \xrightarrow{\text{GELU}} \xrightarrow{\text{Conv } 1\times1} +\text{Shortcut} \tag{2.1}$$

Setiap komponen memiliki peran spesifik yang dijelaskan sebagai berikut:

A *Depthwise Convolution* 7×7

Kernel konvolusi yang digunakan adalah 7×7, jauh lebih besar dari kernel 3×3 yang lazim pada CNN konvensional. Ukuran kernel yang lebih besar ini memperluas receptive field setiap neuron sehingga model mampu menangkap konteks spasial yang lebih luas dalam sekali operasi [1]. Dipilih ukuran 7×7 karena Liu et al. secara empiris menemukan bahwa ukuran ini memberikan keseimbangan optimal antara peningkatan akurasi dan biaya komputasi. Penggunaan depthwise

memastikan bahwa meski kernel besar, jumlah parameter tetap efisien [1].

B *Inverted Bottleneck*

Berbeda dengan **bottleneck** konvensional pada ResNet yang menyempitkan dimensi kanal di tengah blok (misalnya 256 → 64 → 256), ConvNeXt menggunakan desain **inverted bottleneck** yang justru melebarkan dimensi kanal di bagian tengah sebesar 4× sebelum dikembalikan ke dimensi semula [1]. Jika dimensi masukan adalah CC C, maka konvolusi 1×1 pertama mengekspansi ke 4C4C 4C, kemudian konvolusi 1×1 kedua mengompresi kembali ke CC C. Desain ini diadopsi dari *Transformer FFN (Feed-Forward Network)* dan memungkinkan model mempelajari representasi yang lebih kaya di ruang dimensi yang lebih tinggi [1].

C *LayerNorm* (LN) menggantikan *BatchNorm* (BN)

CNN konvensional secara umum menggunakan *Batch Normalization* (BN) yang menormalisasi aktivasi berdasarkan statistik seluruh mini-batch. ConvNeXt menggantikannya dengan *Layer Normalization* (LN) yang menormalisasi berdasarkan statistik individual per sampel, konsisten dengan praktik yang digunakan pada Transformer [1]. Secara matematis, LN dinyatakan sebagai:

$$\mathrm{LN}(x) = \gamma \cdot \frac{x - \mu}{\sqrt{\sigma^2 + \varepsilon}} + \beta \tag{2.2}$$

dengan μ dan σ^2 masing-masing adalah rata-rata dan varians yang dihitung per sampel (berbeda dengan BN yang menghitungnya per mini-batch), ε adalah konstanta kecil untuk menjaga kestabilan numerik, serta γ dan β merupakan parameter skala dan pergeseran yang dapat dipelajari (*learnable*). Selain itu, jumlah lapisan normalisasi dalam satu blok dikurangi dari tiga menjadi hanya satu, yaitu tepat setelah *depthwise convolution*. Perubahan ini terbukti meningkatkan stabilitas pelatihan [1].

D Fungsi Aktivasi GELU

Fungsi aktivasi ReLU yang standar digantikan oleh GELU (*Gaussian Error Linear Unit*), yaitu fungsi aktivasi berbasis distribusi kumulatif Gaussian [1]. Secara matematis, GELU didefinisikan sebagai berikut:

$$\text{GELU}(x) = x \cdot \Phi(x) = x \cdot \frac{1}{2}\left[1 + \text{erf}\left(\frac{x}{\sqrt{2}}\right)\right] \tag{2.3}$$

dengan $\Phi(x)$ merupakan fungsi distribusi kumulatif (*cumulative distribution function*) dari distribusi normal standar, dan $\text{erf}(\cdot)$ adalah *error function*. GELU menghasilkan kurva transisi yang lebih halus dibandingkan ReLU karena tidak memiliki titik transisi yang tajam, sehingga gradien dapat mengalir lebih mulus selama proses pelatihan. Selain itu, jumlah fungsi aktivasi dalam satu blok dikurangi menjadi hanya satu, ditempatkan di antara dua konvolusi 1×1, meniru desain blok MLP pada *Transformer* [1].

E Keunggulan ConvNeXt dan Penerapannya di Klasifikasi Penyakit Tanaman

Dibandingkan CNN konvensional, ConvNeXt unggul dalam tiga aspek utama. Pertama, akurasi yang jauh lebih tinggi dalam kompleksitas komputasi yang sebanding, ConvNeXt-B mencapai 83,8% di ImageNet-1K dibandingkan ResNet-50 yang hanya 76,1%. Kedua, efisiensi parameter berkat penggunaan depthwise convolution. Ketiga, kesederhanaan implementasi dibandingkan Transformer karena dibangun sepenuhnya dari modul konvolusi standar tanpa mekanisme attention khusus [1]. Dalam bidang klasifikasi penyakit tanaman, ConvNeXt telah terbukti efektif di berbagai penelitian. Wu et al. [32] menerapkan ConvNeXt pada klasifikasi penyakit daun kedelai dan menunjukkan performanya melampaui ResNet50 dan Swin Transformer. Dong et al. [33] mengintegrasikan ConvNeXt-T dengan modul perhatian untuk deteksi penyakit gandum dan memperoleh akurasi rata-rata 88,05%, tertinggi di antara semua model pembanding. Kemampuan ConvNeXt dalam menangkap pola visual seperti perubahan warna, tekstur bercak, dan bentuk lesi menjadikannya pilihan yang kuat untuk klasifikasi penyakit daun jagung pada penelitian ini.

Secara keseluruhan, arsitektur ConvNeXt tersedia dalam beberapa varian berdasarkan kapasitas model, sebagaimana ditunjukkan pada Tabel berikut:

Tabel 2.4. Varian ConvNeXt dan Akurasi ImageNet-1K [1]

Varian	Parameter	Akurasi ImageNet-1K
ConvNeXt-T (*Tiny*)	28M	82,1%
ConvNeXt-S (*Small*)	50M	83,1%
ConvNeXt-B (*Base*)	89M	83,8%
ConvNeXt-L (*Large*)	198M	84,3%
ConvNeXt-XL (*XLarge*)	350M	84,4%

2.5 EfficientNetV2-S

EfficientNetV2 adalah keluarga arsitektur *Convolutional Neural Network* (CNN) yang diperkenalkan oleh Tan dan Le pada tahun 2021 sebagai penyempurnaan dari arsitektur EfficientNet sebelumnya [12, 29]. Pengembangan keluarga model ini bertujuan untuk memperoleh kecepatan pelatihan yang lebih tinggi sekaligus efisiensi parameter yang lebih baik dibandingkan model-model terdahulu. Tujuan tersebut dicapai melalui kombinasi *training-aware neural architecture search* (NAS) dan penskalaan (*scaling*) yang dioptimalkan secara bersamaan untuk kecepatan pelatihan dan efisiensi parameter [12].

EfficientNetV2-S merupakan varian terkecil (*small*) dari keluarga EfficientNetV2, dengan jumlah parameter yang relatif sedikit, yaitu sekitar 21 juta parameter [12]. Karena ukurannya yang ringkas namun tetap memiliki akurasi yang tinggi, varian ini banyak digunakan sebagai model dasar (*backbone*) yang telah dilatih sebelumnya (*pre-trained*) pada dataset ImageNet untuk diterapkan pada tugas klasifikasi citra melalui pendekatan *transfer learning*. Pada penelitian ini, EfficientNetV2-S digunakan sebagai model pembanding terhadap ConvNeXt-Tiny.

2.5.1 Arsitektur dan Cara Kerja EfficientNetV2-S

Arsitektur EfficientNetV2 dibangun di atas prinsip penskalaan majemuk (*compound scaling*) yang diwariskan dari EfficientNet. Berbeda dengan pendekatan konvensional yang hanya menskalakan satu dimensi jaringan, penskalaan majemuk menskalakan kedalaman (*depth*), lebar (*width*), dan resolusi (*resolution*) jaringan secara seimbang dan proporsional menggunakan sebuah koefisien majemuk tunggal [29]. Koefisien ini mengatur besarnya sumber daya komputasi yang dialokasikan untuk penskalaan model, sedangkan proporsi antar ketiga dimensi

tersebut ditentukan melalui *grid search*, sehingga peningkatan kapasitas model tetap seimbang dan efisien [29].

Blok penyusun utama arsitektur ini adalah *Mobile Inverted Bottleneck Convolution* (MBConv), yang pertama kali diperkenalkan pada arsitektur MobileNetV2 [34]. Blok MBConv menggunakan struktur *inverted bottleneck*, yaitu melakukan ekspansi jumlah kanal melalui konvolusi 1×1, diikuti operasi *depthwise convolution* untuk penyaringan spasial, lalu memproyeksikan kembali jumlah kanal melalui konvolusi 1×1, dengan tambahan koneksi residual. Pada blok MBConv juga disisipkan mekanisme *Squeeze-and-Excitation* (SE), yaitu sebuah mekanisme atensi berbasis kanal yang secara adaptif mengkalibrasi ulang bobot tiap kanal fitur [35].

Inovasi utama EfficientNetV2 terletak pada penambahan blok *Fused-MBConv*. Blok ini menggantikan kombinasi konvolusi ekspansi 1×1 dan *depthwise convolution* pada MBConv dengan sebuah konvolusi standar 3×3 [12]. Meskipun blok *Fused-MBConv* memiliki jumlah parameter dan operasi (FLOPs) yang lebih banyak, blok ini dapat memanfaatkan akselerator perangkat keras modern secara lebih optimal sehingga berjalan lebih cepat, terutama pada lapisan-lapisan awal jaringan [12]. Melalui proses *neural architecture search*, EfficientNetV2 secara otomatis menentukan kombinasi terbaik antara *Fused-MBConv* pada lapisan awal dan MBConv pada lapisan akhir [12].

Selain itu, EfficientNetV2 menerapkan strategi penskalaan non-uniform dengan menambahkan lebih banyak lapisan secara bertahap pada tahap-tahap akhir jaringan serta membatasi ukuran citra maksimum untuk menekan beban memori [12]. Pada proses pelatihannya, EfficientNetV2 menggunakan metode *progressive learning*, yaitu meningkatkan ukuran citra secara bertahap selama pelatihan sambil menyesuaikan kekuatan regularisasi (seperti *dropout* dan augmentasi data) agar diperoleh pelatihan yang cepat tanpa menurunkan akurasi [12].

2.5.2 Perbandingan ConvNeXt-Tiny dan EfficientNetV2-S

Meskipun ConvNeXt-Tiny dan EfficientNetV2-S sama-sama merupakan arsitektur jaringan saraf konvolusional murni (*pure ConvNet*), keduanya lahir dari filosofi desain yang berbeda. ConvNeXt dikembangkan dengan cara "memodernisasi" arsitektur ResNet secara bertahap dengan mengadopsi prinsip-prinsip desain dari Swin Transformer, seperti penggunaan kernel *depthwise convolution* berukuran besar (7×7), struktur *inverted bottleneck*, *Layer*

Normalization, dan fungsi aktivasi GELU [1]. Sebaliknya, EfficientNetV2 dirancang melalui pendekatan *training-aware neural architecture search* (NAS) yang dikombinasikan dengan penskalaan majemuk, dengan blok penyusun berupa MBConv dan Fused-MBConv yang berfokus pada kecepatan pelatihan dan efisiensi parameter [12, 29]. Perbedaan utama kedua arsitektur tersebut dirangkum pada Tabel 2.5.

Tabel 2.5. Perbandingan ConvNeXt-Tiny dan EfficientNetV2-S

Aspek	**ConvNeXt-Tiny**	**EfficientNetV2-S**
Tahun (sumber)	2022 (Liu et al.)	2021 (Tan & Le)
Filosofi desain	Modernisasi ResNet	NAS + compound scaling
Blok inti	Blok ConvNeXt (DWConv 7×7 + Conv 1×1)	MBConv & Fused-MBConv (+ Squeeze-and-Excitation)
Normalisasi	Layer Normalization	Batch Normalization
Fungsi aktivasi	GELU	SiLU (Swish)
Jumlah parameter	$\sim$28 juta	$\sim$21 juta
Akurasi ImageNet-1K	82,1%	$\sim$84%

Perbedaan filosofi desain tersebut sekaligus menjelaskan mengapa kedua arsitektur mampu menghasilkan kinerja yang lebih baik dibandingkan CNN konvensional. Pada ConvNeXt-Tiny, penggunaan kernel *depthwise* berukuran besar (7×7) memperluas *receptive field* sehingga model dapat menangkap konteks spasial yang lebih luas dan menghasilkan representasi fitur yang lebih kaya, sementara *Layer Normalization* dan fungsi aktivasi GELU membuat proses optimasi lebih stabil dengan aliran gradien yang lebih halus. Kombinasi ini memungkinkan ConvNeXt mencapai akurasi tinggi tanpa penambahan beban komputasi yang berarti [1]. Sementara itu, pada EfficientNetV2-S, strategi *compound scaling* menyeimbangkan kedalaman, lebar, dan resolusi jaringan sehingga kapasitas model dimanfaatkan secara optimal, sedangkan blok *Fused-MBConv* pada lapisan awal mengatasi keterbatasan kecepatan *depthwise convolution* sehingga pelatihan menjadi lebih efisien [12]. Mekanisme *Squeeze-and-Excitation* pada blok MBConv turut memperkuat fitur antar-kanal yang paling informatif, misalnya pola lesi dan perubahan warna pada daun, sehingga kemampuan diskriminatif model semakin meningkat [35].

Dengan demikian, keunggulan kinerja kedua arsitektur tidak diperoleh dengan sekadar memperdalam jaringan, melainkan melalui desain yang lebih efisien dalam memanfaatkan parameter sekaligus lebih efektif dalam mengekstraksi fitur visual. Efisiensi parameter yang tinggi ini menjadikan keduanya mampu mencapai

akurasi kompetitif pada jumlah parameter yang relatif kecil, sehingga sangat sesuai untuk diterapkan pada tugas klasifikasi penyakit daun jagung berbasis *transfer learning* dengan sumber daya komputasi yang terbatas.

Sebelum EfficientNetV2-S ditetapkan sebagai model pembanding, penulis terlebih dahulu menimbang beberapa kandidat arsitektur. Proses penyaringan ini didasarkan pada tiga kriteria utama, yaitu: (1) kapasitas model yang sebanding dengan ConvNeXt-Tiny agar perbandingan bersifat adil; (2) merupakan arsitektur CNN murni, bukan berbasis *Transformer*, agar tetap berada dalam ranah perbandingan yang sama; serta (3) ketersediaan bobot pra-latih pada ImageNet.

Beberapa kandidat kemudian dievaluasi berdasarkan kriteria tersebut. MobileNetV2 memiliki jumlah parameter yang jauh lebih kecil, yaitu sekitar 3,4 juta, sehingga kapasitasnya tidak setara dengan ConvNeXt-Tiny [34]. Arsitektur berbasis *Transformer* seperti *Vision Transformer* dan Swin Transformer tidak dipilih karena bukan merupakan CNN murni, sehingga berada di luar cakupan perbandingan penelitian ini [30]. Kandidat yang cukup menjanjikan adalah ResNet-50 karena memiliki jumlah parameter yang sebanding, yaitu sekitar 25,6 juta [28]. Akan tetapi, ResNet-50 merupakan arsitektur generasi lama, sehingga membandingkannya dengan ConvNeXt-Tiny akan lebih mencerminkan perbandingan antara arsitektur tradisional dan modern, bukan perbandingan antar-paradigma modern yang menjadi fokus penelitian ini. Terlebih lagi, ConvNeXt pada dasarnya merupakan hasil modernisasi dari ResNet, sehingga membandingkan keduanya menjadi kurang tepat untuk menilai keunggulan antararsitektur modern [1]. EfficientNet generasi pertama juga sempat dipertimbangkan, namun arsitektur tersebut telah disempurnakan oleh generasi penerusnya, yaitu EfficientNetV2 [29, 12].

Setelah menimbang kelebihan dan kekurangan seluruh kandidat, EfficientNetV2-S dipilih sebagai model pembanding karena paling memenuhi ketiga kriteria tersebut. Sebagai arsitektur CNN murni yang sama-sama modern dengan ConvNeXt-Tiny, EfficientNetV2-S berada pada kelas kapasitas yang sebanding, yaitu sekitar 21 juta parameter berbanding sekitar 28 juta parameter pada ConvNeXt-Tiny [1, 12]. Kesetaraan kapasitas ini penting agar perbandingan bersifat adil (*apples-to-apples*), sehingga perbedaan kinerja yang teramati lebih mencerminkan perbedaan paradigma desain arsitektur, bukan semata-mata akibat perbedaan ukuran model.

Pemilihan varian terkecil dari masing-masing keluarga, yaitu ConvNeXt-Tiny dan EfficientNetV2-S, turut mempertimbangkan skala dataset dan ketersediaan

sumber daya. ConvNeXt tersedia hingga varian Base dan Large, sementara EfficientNetV2 hingga varian M dan L, yang seluruhnya berkapasitas parameter jauh lebih besar [1, 12]. Namun, dataset yang digunakan pada penelitian ini tergolong berskala terbatas, yaitu 4.000 gambar pada empat kelas, sehingga varian Tiny dan S dinilai sudah memadai untuk menangkap pola penyakit daun jagung. Sebaliknya, varian yang lebih besar justru berisiko mengalami *overfitting* pada dataset berskala kecil dan menuntut biaya komputasi yang jauh lebih tinggi tanpa peningkatan kinerja yang sepadan.

Selain itu, EfficientNetV2-S telah terbukti memberikan kinerja tinggi pada tugas klasifikasi penyakit daun jagung dalam penelitian sebelumnya [9], sehingga relevan dijadikan pembanding untuk mengukur efektivitas ConvNeXt-Tiny. Kesetaraan kapasitas kedua model juga memungkinkan evaluasi dilakukan pada kondisi sumber daya komputasi yang sebanding [36].

2.6 ImageNet dan Model Pra-latih

ImageNet merupakan basis data citra berskala besar yang menjadi tolok ukur standar dalam penelitian klasifikasi citra dan pengembangan arsitektur *deep learning*. Melalui ajang *ImageNet Large Scale Visual Recognition Challenge* (ILSVRC), subset yang paling banyak digunakan memuat lebih dari 14 juta citra beranotasi yang terbagi ke dalam 1.000 kategori objek [37]. Skala dan keragaman inilah yang menjadikan ImageNet sebagai sumber utama untuk pra-pelatihan (*pre-training*) model, karena model yang dilatih pada dataset ini mampu mempelajari representasi fitur visual yang kaya dan bersifat umum, mulai dari tepi dan tekstur hingga bentuk dan pola objek yang kompleks [37].

Kedua arsitektur yang digunakan dalam penelitian ini, yaitu ConvNeXt-Tiny dan EfficientNetV2-S, keduanya menyediakan bobot hasil pra-pelatihan pada ImageNet. Performa kedua model pada tolok ukur ImageNet-1K dirangkum pada Tabel 2.6. ConvNeXt Tiny mencapai akurasi *top-1* sebesar 82,1% dengan sekitar 28,6 juta parameter pada resolusi masukan 224×224 [1]. Sementara itu, EfficientNetV2-S mencapai akurasi *top-1* sebesar 83,9% dengan sekitar 22 juta parameter, dan dapat meningkat hingga 84,9% apabila terlebih dahulu dilakukan pra-pelatihan pada ImageNet-21k sebelum *fine-tuning* pada ImageNet-1K [12].

Perlu dicatat bahwa kedua nilai akurasi tersebut tidak sepenuhnya dapat dibandingkan secara langsung karena perbedaan konfigurasi pelatihan dan resolusi evaluasi antararsitektur. Angka-angka ini disajikan sebagai gambaran kapasitas

Tabel 2.6. Performa pra-latih ConvNeXt-Tiny dan EfficientNetV2-S pada ImageNet-1K

Model	Akurasi Top-1 (%)	Parameter	Sumber
ConvNeXt-Tiny	82,1	∼28,6 jt	[1]
EfficientNetV2-S	83,9	∼22 jt	[12]

awal (*baseline*) masing-masing model pada domain sumber (ImageNet), bukan sebagai kesimpulan mengenai keunggulan salah satu model. Bobot hasil pra-pelatihan inilah yang selanjutnya dijadikan titik awal dalam proses *transfer learning* untuk diadaptasi ke domain target, yaitu klasifikasi penyakit daun jagung, sebagaimana dijelaskan pada subbab berikutnya.

Pada penelitian ini, kedua arsitektur menggunakan bobot pra-latih yang berasal dari **ImageNet-1k**, bukan ImageNet-21k. Pemilihan ini didasarkan pada beberapa pertimbangan. Pertama, demi menjaga kesetaraan perbandingan (*fair comparison*): karena tujuan utama penelitian adalah membandingkan ConvNeXt-Tiny dan EfficientNetV2-S, kedua model harus berangkat dari basis pra-pelatihan yang sama agar perbedaan performa yang teramati benar-benar mencerminkan perbedaan arsitektur, bukan perbedaan skala data pra-pelatihan. Kedua, bobot pra-latih ImageNet-1k tersedia secara resmi dan luas untuk kedua arsitektur pada pustaka standar, sehingga proses penelitian lebih mudah direproduksi. Ketiga, ImageNet-1k dengan 1.000 kelas dan lebih dari 14 juta citra sudah cukup kaya untuk membentuk representasi fitur visual umum yang relevan bagi tugas klasifikasi penyakit daun jagung, sekaligus lebih hemat sumber daya komputasi dibandingkan ImageNet-21k [37].

2.7 Transfer Learning

Transfer learning adalah teknik dalam machine learning di mana pengetahuan yang telah dipelajari oleh sebuah model pada suatu tugas atau domain tertentu (source domain) dimanfaatkan kembali untuk meningkatkan performa model pada tugas atau domain yang berbeda namun berkaitan (target domain) [36]. Pendekatan ini muncul sebagai solusi atas dua keterbatasan utama pelatihan model deep learning dari awal (from scratch): kebutuhan akan data berlabel dalam jumlah besar dan biaya komputasi yang tinggi [38]. Dengan transfer learning, sebuah model yang telah dilatih pada dataset besar seperti ImageNet yang memiliki lebih dari 14 juta gambar dari 1.000 kategori, dapat diadaptasi ke tugas baru menggunakan data yang jauh lebih sedikit, karena bobot-bobot yang

merepresentasikan fitur-fitur visual umum seperti tepi, tekstur, dan bentuk sudah terbentuk selama pra-pelatihan [36].

2.7.1 Jenis-jenis Transfer Learning

Berdasarkan kondisi source domain, target domain, dan ketersediaan data berlabel, transfer learning dikategorikan ke dalam tiga jenis utama [36], [38]:

A Inductive Transfer Learning

Jenis ini diterapkan ketika source domain dan target domain bisa sama atau berbeda, tetapi tugas yang dikerjakan berbeda. Data berlabel tersedia pada target domain. Contoh kasusnya adalah model yang dilatih untuk klasifikasi objek umum di ImageNet lalu diadaptasi untuk mengklasifikasikan penyakit tanaman. Tugas berbeda, tetapi sama-sama berbasis pengenalan citra [36]. Jenis ini paling umum digunakan dalam aplikasi computer vision.

B Transductive Transfer Learning

Jenis ini terjadi ketika tugas antara source dan target sama, namun domainnya berbeda. Data berlabel tidak tersedia pada target domain. Situasi ini sering disebut juga domain adaptation, misalnya mengadaptasi model klasifikasi yang dilatih pada citra di lingkungan laboratorium untuk digunakan pada citra di lapangan yang memiliki kondisi pencahayaan dan latar belakang berbeda [36].

C Unsupervised Transfer Learning

Jenis ini diterapkan ketika tidak tersedia data berlabel baik di source maupun target domain. Fokusnya adalah pada tugas-tugas tidak tersupervisi seperti clustering dan reduksi dimensi. Jenis ini relatif jarang digunakan pada klasifikasi citra [38].

Dalam praktiknya di bidang computer vision, jenis yang paling banyak digunakan adalah inductive transfer learning, yang diimplementasikan melalui dua pendekatan utama: feature extraction dan fine-tuning. Pada pendekatan feature extraction, seluruh lapisan model pra-latih dibekukan (frozen) dan hanya lapisan klasifikasi akhir yang dilatih ulang sesuai dengan kelas target baru. Sementara pada pendekatan fine-tuning, sebagian atau seluruh lapisan model pra-latih dibuka

kembali dan dilatih ulang dengan learning rate yang kecil agar model dapat menyesuaikan representasi fiturnya secara lebih spesifik dengan domain target [38].

2.7.2 Mengapa Transfer Learning Dapat Berhasil

Keberhasilan transfer learning tidak terjadi secara kebetulan, melainkan bertumpu pada beberapa prinsip mendasar. Pertama, transferabilitas fitur secara hierarkis. Model *deep learning* mempelajari fitur secara bertingkat, di mana lapisan-lapisan awal menangkap fitur tingkat rendah yang bersifat generik seperti tepi, tekstur, dan warna. Fitur semacam ini tidak spesifik pada satu tugas tertentu sehingga dapat digunakan kembali pada beragam domain visual, sedangkan hanya lapisan-lapisan akhir yang benar-benar spesifik terhadap tugas tertentu [36]. Karena itu, representasi yang telah dipelajari dari dataset besar seperti ImageNet tetap relevan ketika dipindahkan ke tugas seperti klasifikasi penyakit daun [38].

Kedua, transfer learning menyediakan inisialisasi bobot yang jauh lebih baik dibandingkan inisialisasi acak. Bobot hasil pra-pelatihan menempatkan model pada titik awal yang sudah dekat dengan solusi optimal, sehingga proses pelatihan pada domain target berlangsung lebih cepat, lebih stabil, dan cenderung menghasilkan generalisasi yang lebih baik [38].

Ketiga, pemanfaatan pengetahuan awal ini berperan sebagai bentuk regularisasi implisit yang membantu menekan risiko *overfitting*, khususnya ketika jumlah data berlabel pada domain target terbatas [36]. Ketiga prinsip inilah yang menjelaskan mengapa transfer learning mampu mencapai performa tinggi meskipun dilatih dengan data yang relatif sedikit dan sumber daya komputasi yang terbatas [38].

2.8 Metrik Evaluasi

Evaluasi kinerja model klasifikasi dilakukan untuk mengukur sejauh mana model mampu memprediksi kelas data secara benar. Karena permasalahan pada penelitian ini merupakan klasifikasi multikelas dengan empat kelas (sehat, *Common Rust*, *Northern Leaf Blight*, dan *Gray Leaf Spot*), diperlukan sejumlah metrik yang dapat menggambarkan kinerja model, bukan hanya berdasarkan tingkat ketepatan secara keseluruhan, tetapi juga kinerja pada masing-masing kelas [39]. Metrik evaluasi yang digunakan dalam penelitian ini yaitu akurasi, *confusion matrix*, presisi, *recall*, dan *F1-score*, yang dirangkum melalui rata-rata *macro*.

2.8.1 Confusion Matrix

Confusion matrix merupakan tabel yang digunakan untuk merangkum hasil prediksi sebuah model klasifikasi dengan membandingkan kelas aktual (sebenarnya) terhadap kelas hasil prediksi model [39]. Untuk kasus klasifikasi biner, *confusion matrix* terdiri atas empat komponen dasar, yaitu:

- ***True Positive* (TP)**: jumlah data positif yang diprediksi dengan benar sebagai positif.
- ***True Negative* (TN)**: jumlah data negatif yang diprediksi dengan benar sebagai negatif.
- ***False Positive* (FP)**: jumlah data negatif yang keliru diprediksi sebagai positif.
- ***False Negative* (FN)**: jumlah data positif yang keliru diprediksi sebagai negatif.

Pada klasifikasi multikelas, *confusion matrix* berukuran $n \times n$ (n = jumlah kelas), di mana diagonal utama menunjukkan prediksi yang benar dan elemen luar diagonal menunjukkan kesalahan klasifikasi[39]. Nilai TP, TN, FP, dan FN dihitung dengan pendekatan *one-vs-rest*, yaitu satu kelas diperlakukan sebagai positif dan kelas lainnya sebagai negatif.

2.8.2 Akurasi, Presisi, Recall, dan F1-Score

Berdasarkan komponen pada *confusion matrix*, dapat diturunkan beberapa metrik evaluasi sebagai berikut [39].

Akurasi (*accuracy*) mengukur proporsi prediksi yang benar terhadap keseluruhan data, dirumuskan sebagai:

$$\text{Akurasi} = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.4}$$

Presisi (*precision*) mengukur proporsi prediksi positif yang benar-benar positif, yang menggambarkan ketepatan model dalam memprediksi suatu kelas:

$$\text{Presisi} = \frac{TP}{TP + FP} \tag{2.5}$$

Recall (sensitivitas) mengukur proporsi data positif yang berhasil diprediksi dengan benar oleh model:

$$\text{Recall} = \frac{TP}{TP + FN} \tag{2.6}$$

F1-score merupakan rata-rata harmonik dari presisi dan *recall*, yang memberikan keseimbangan antara kedua metrik tersebut. Metrik ini sangat berguna ketika distribusi data antar kelas tidak seimbang:

$$\text{F1-score} = 2 \times \frac{\text{Presisi} \times \text{Recall}}{\text{Presisi} + \text{Recall}} \tag{2.7}$$

2.8.3 Rata-rata Macro

Pada klasifikasi multikelas, presisi, *recall*, dan *F1-score* dihitung terlebih dahulu untuk masing-masing kelas, kemudian dirangkum menjadi satu nilai melalui teknik perata-rataan. Salah satu teknik perata-rataan yang umum digunakan adalah *macro average*[39].

Macro average dihitung dengan merata-ratakan nilai metrik dari keseluruh kelas tanpa mempertimbangkan jumlah data pada tiap kelas, sehingga setiap kelas diberi bobot yang sama. Untuk n kelas, *macro average* suatu metrik dirumuskan sebagai:

$$\text{Macro Average} = \frac{1}{n} \sum_{i=1}^{n} M_i \tag{2.8}$$

dengan M_i adalah nilai metrik (presisi, *recall*, atau *F1-score*) pada kelas ke-i. Teknik ini memperlakukan setiap kelas secara setara sehingga sensitif terhadap kinerja model pada kelas minoritas [39].