BAB II

LANDASAN TEORI

2.1 Penelitian Terdahulu

Tabel 2.1 menunjukan penelitian mengenai klasifikasi penyakit tanaman berbasis citra telah berkembang pesat seiring dengan kemajuan teknologi *deep learning*. Berbagai arsitektur, mulai dari Convolutional Neural Network (CNN), Vision Transformer (ViT), hingga model *hybrid* CNN–Transformer, telah diusulkan untuk meningkatkan akurasi dan kemampuan generalisasi model pada *dataset* penyakit tanaman. Subbab ini membahas beberapa penelitian terdahulu yang relevan dengan topik penelitian.

EfficientNet merupakan salah satu arsitektur CNN yang banyak digunakan dan menunjukkan performa tinggi dalam klasifikasi penyakit tanaman [12], [13]. Penelitian [12] mengevaluasi beberapa arsitektur CNN menggunakan *dataset PlantVillage* dan menunjukkan bahwa EfficientNet-B0 mencapai akurasi tertinggi sebesar 98,51% dibandingkan Resnet dan Densenet. Penelitian [13] juga melaporkan bahwa EfficientNet-B4 dan EfficientNet-B5 mampu mencapai akurasi di atas 99% pada *dataset* yang telah diaugmentasi. Penelitian lain mengenai klasifikasi penyakit daun tomat menggunakan DenseNet dan citra sintetis berbasis Conditional Generative Adversarial Network (C-GAN) menunjukkan bahwa kombinasi augmentasi geometris dan citra sintetis mampu meningkatkan performa klasifikasi melalui peningkatan variasi data pelatihan. Namun, augmentasi diterapkan secara gabungan sehingga pengaruh masing-masing jenis augmentasi terhadap robustness model belum dianalisis secara individual[11]. Namun, evaluasi pada kedua penelitian tersebut dilakukan pada *dataset* dengan distribusi yang serupa antara data latih dan data uji, sehingga kemampuan model dalam menghadapi variasi kondisi visual nyata belum dianalisis secara eksplisit.

Keterbatasan generalisasi model terhadap perbedaan domain juga menjadi perhatian dalam penelitian [15], yang menunjukkan bahwa baik CNN maupun Transformer mengalami penurunan performa ketika diuji pada distribusi data yang berbeda dari data latih. Temuan ini mengindikasikan bahwa akurasi tinggi pada

dataset terkontrol belum tentu mencerminkan tingkat *robustness* model pada kondisi visual nyata di lapangan. Penelitian [19] menunjukkan bahwa penerapan augmentasi citra dapat meningkatkan kemampuan generalisasi model klasifikasi penyakit tanaman melalui peningkatan variasi data pelatihan. Penelitian lain oleh Huang dkk. memperkenalkan Data Augmented Loss Invariant Regularization (DAIR) yang mengombinasikan random data augmentation dengan consistency regularization untuk meningkatkan robustness model terhadap distribution shift. Meskipun menunjukkan peningkatan robustness, augmentasi pada penelitian tersebut diterapkan secara gabungan sehingga kontribusi masing-masing jenis augmentasi terhadap performa model belum dievaluasi secara individual [10].

Sebagai alternatif dari CNN, *Vision Transformer* (ViT) diperkenalkan dengan mekanisme *self-attention* yang mampu menangkap hubungan global antar bagian citra [14]. Arsitektur ini menunjukkan performa tinggi pada tugas klasifikasi citra berskala besar, terutama ketika dilatih menggunakan *dataset* berukuran besar. Namun, ViT murni cenderung membutuhkan data pelatihan yang besar dan dapat mengalami penurunan performa pada *dataset* yang lebih kecil atau spesifik domain.

Untuk mengatasi keterbatasan tersebut, pendekatan hybrid CNN–*Vision Transformer* mulai dikembangkan. Penelitian [16] memperkenalkan model hybrid CNN–ViT yang menunjukkan performa baik pada *dataset* dengan latar belakang kompleks serta tetap efisien dalam jumlah parameter. Selain itu, penelitian [17] membahas klasifikasi penyakit tanaman pada kondisi lapangan (in-the-wild) dengan mengombinasikan *Vision Transformer* dan Mixture of Experts, yang menunjukkan peningkatan kemampuan lintas domain dibandingkan CNN konvensional.

FastViT diperkenalkan sebagai arsitektur *hybrid* CNN–Transformer yang dirancang untuk mencapai keseimbangan antara akurasi dan efisiensi komputasi [11]. Penelitian tersebut menunjukkan bahwa FastViT memiliki kecepatan inferensi yang lebih tinggi dibandingkan beberapa model CNN konvensional dengan performa klasifikasi yang tetap kompetitif. Meskipun demikian, evaluasi FastViT dalam penelitian tersebut masih dilakukan pada benchmark klasifikasi citra umum

dan belum diterapkan secara khusus pada klasifikasi penyakit daun tanaman maupun pengujian robustness menggunakan augmentasi individual.

Selain aspek akurasi, efisiensi komputasi juga menjadi pertimbangan penting dalam implementasi sistem klasifikasi penyakit tanaman. Penelitian [8] menunjukkan bahwa model CNN ringan memiliki efisiensi komputasi yang baik dan berpotensi digunakan pada lingkungan dengan sumber daya komputasi terbatas. Hal ini menunjukkan bahwa pemilihan arsitektur model tidak hanya bergantung pada akurasi, tetapi juga pada efisiensi dan stabilitas performa dalam berbagai kondisi.

Penelitian *Deployable Deep Learning for Cross-Domain Plant Leaf Disease Detection* menunjukkan bahwa model EfficientNet, DenseNet, dan ResNet mengalami penurunan performa ketika berpindah dari dataset laboratorium ke citra lapangan (*cross-domain*). Meskipun EfficientNet menunjukkan performa terbaik dibandingkan model lainnya, penelitian tersebut belum menganalisis pengaruh masing-masing gangguan visual secara spesifik terhadap robustness model[18].

Berdasarkan kajian penelitian terdahulu, sebagian besar penelitian berfokus pada peningkatan akurasi melalui augmentasi gabungan, peningkatan kapasitas model, maupun peningkatan generalisasi lintas domain. Namun, belum terdapat penelitian yang secara sistematis membandingkan EfficientNet-B0 dan FastViT menggunakan pendekatan evaluasi augmentasi individual untuk menganalisis pengaruh masing-masing gangguan visual terhadap akurasi, *precision*, *recall*, *F1-score*, *robustness score*, AURC, stabilitas performa, dan efisiensi komputasi pada klasifikasi penyakit daun tomat.

UNIVERSITAS MULTIMEDIA NUSANTARA

Tabel 2.1 Tabel Penelitian Terdahulu

No	Penelitian	Dataset	Metode / Model	Hasil Utama	Kelebihan	Kekurangan	*Research Gap* terhadap Penelitian Ini
[10]	*Robustness through Data Augmentation Loss Consistency*	ImageNet	CNN	Meningkatkan robustness model terhadap distribution shift menggunakan kombinasi augmentasi random.	Mengevaluasi robustness dengan pendekatan random augmentation dan consistency regularization.	Augmentasi diterapkan secara gabungan (random augmentation), sehingga pengaruh masing-masing augmentasi tidak dianalisis secara terpisah.	Belum mengevaluasi robustness berdasarkan setiap jenis augmentasi (individual augmentation) pada tugas klasifikasi penyakit daun tanaman.
[11]	*Tomato plant disease detection using transfer learning with C-GAN synthetic images*	Plant Village Dataset (Tomato Only)	CNN (EfficientNet dan Densnet	Kombinasi citra sintetis C-GAN dan augmentasi geometris meningkatkan performa klasifikasi penyakit daun tomat.	Memanfaatkan kombinasi augmentasi geometris dan citra sintetis untuk meningkatkan variasi data pelatihan.	Evaluasi dilakukan menggunakan kombinasi augmentasi sehingga kontribusi masing-masing augmentasi tidak dianalisis secara individual.	Belum mengevaluasi robustness terhadap berbagai jenis augmentasi individual (*brightness*, *contrast*, *saturation*, *hue*, *affine*, *perspective*, dan *blur*).
[12]	*A Comprehensive Analysis of Various Deep Learning Based Multi-Class Plant Disease Classification Techniques*	Plant Village Dataset (54.000 Gambar Tomato, Bell Pepper, Potatoes)	CNN (EfficientNet, ResNet, DenseNet, MobileNet)	EfficientNet-B0 mencapai akurasi 98,51%	Membandingkan berbagai arsitektur *deep learning* pada tugas klasifikasi penyakit tanaman	Fokus pada akurasi klasifikasi	Belum mengevaluasi *robustness* terhadap variasi visual
[13]	*Plant Leaf Disease Classification Using EfficientNet Deep Learning Model*	Plant Village Dataset (54.000 Gambar Tomato,	EfficientNet-B4 dan EfficientNet-B5	Akurasi lebih dari 99,9% dengan *precision* tinggi	Menunjukkan efektivitas EfficientNet pada	Dataset menggunakan kondisi citra yang relatif terkontrol	Belum menguji ketahanan model terhadap gangguan visual individual

No	Penelitian	Dataset	Metode / Model	Hasil Utama	Kelebihan	Kekurangan	*Research Gap* terhadap Penelitian Ini
		Bell Pepper, Potatoes)			klasifikasi penyakit daun		
[19]	*An Image is Worth 16×16 Words: Transformers for Image Recognition at Scale*	ImageNet	Vision Transformer (ViT)	ViT mencapai performa kompetitif terhadap CNN pada klasifikasi citra	Memanfaatkan *self-attention* untuk menangkap hubungan global	Membutuhkan dataset besar atau *pretraining* yang kuat	Belum diterapkan pada klasifikasi penyakit tanaman dan evaluasi *robustness*
[9]	*FastViT: A Fast Hybrid Vision Transformer Using Structural Reparameterization*	ImageNet 1k (1.000 Kelas)	FastViT	FastViT lebih cepat dan efisien dengan akurasi kompetitif	Menggabungkan efisiensi CNN dan kemampuan representasi global Transformer	Evaluasi berfokus pada benchmark umum komputer visi	Belum diterapkan pada klasifikasi penyakit daun tomat
[20]	*Enhancing Plant Disease Classification with Transfer Learning and Data Augmentation*	Plant Village Dataset (54.000 Gambar Tomato, Bell Pepper, Potatoes)	DenseNet, ResNet, EfficientNet	DenseNet mencapai akurasi sekitar 99,7%	Menunjukkan manfaat *transfer learning* dan augmentasi	Augmentasi digunakan untuk meningkatkan data pelatihan	Tidak mengevaluasi pengaruh tiap augmentasi secara terpisah
[16]	*Vision Transformer Meets CNN for Plant Disease Classification*	Plant Village Dataset (54.000 Gambar Tomato, Bell Pepper, Potatoes)	Hybrid CNN–ViT	Akurasi 98,86% pada PlantVillage dan 89,24% pada Embrapa	Mampu bekerja pada latar belakang yang lebih kompleks	Fokus pada performa klasifikasi	Belum menguji *robustness* terhadap variasi visual individual
[17]	*Plant Disease Classification in the Wild Using Vision*	PlantDoc Dataset (2,569 Gambar Tomat)	Vision Transformer +	Meningkatkan kemampuan	Menggunakan citra yang lebih	Kompleksitas model relatif tinggi	Tidak membandingkan CNN dan Transformer secara langsung

No	Penelitian	Dataset	Metode / Model	Hasil Utama	Kelebihan	Kekurangan	*Research Gap* terhadap Penelitian Ini
	Transformers and Mixture of Experts		Mixture of Experts	klasifikasi pada kondisi lapangan	mendekati kondisi nyata		
[8]	*Plant Disease Detection Based on Lightweight CNN Model*	Plant Village Dataset (54.000 Gambar Tomato, Bell Pepper, Potatoes)	Lightweight CNN	Model ringan dengan latensi rendah	Cocok untuk implementasi pada perangkat dengan sumber daya terbatas	Fokus pada efisiensi komputasi	Tidak mengevaluasi *robustness* model
[18]	*Deployable Deep Learning for Cross-Domain Plant Leaf Disease Detection via Ensemble Learning, Knowledge Distillation, and Quantization*	Plant Village Dataset (Tomato Only) & *Real World Dataset*	EffciientNet, Densenet, Resnet	Terjadi penurunan performa saat berpindah *domain*	Efficient unggul walau ada penuruan signifikan saat berpindah domain.	Tidak menganalisis jenis gangguan visual secara spesifik	Belum mengevaluasi *robustness* berdasarkan augmentasi individual
[21]	*Data Augmentation Method for Plant Leaf Disease Recognition*	Plant Village Dataset (54.000 Gambar Tomato, Bell Pepper, Potatoes)	CNN + Data Augmentation	Augmentasi meningkatkan kemampuan generalisasi model	Menunjukkan pentingnya variasi data pelatihan	Fokus pada peningkatan performa pelatihan	Tidak mengevaluasi ketahanan model terhadap tiap jenis gangguan visual

2.2 Teori yang berkaitan

2.2.1 Penyakit Pada Daun Tomat

Penyakit pada daun tomat merupakan salah satu faktor utama yang menyebabkan penurunan produktivitas dan kualitas hasil panen. Berbagai patogen seperti jamur, bakteri, virus, serta hama dapat menyerang jaringan daun dan menimbulkan gejala berupa bercak, perubahan warna, nekrosis, hingga deformasi daun. Infeksi yang tidak terdeteksi secara dini dapat mengganggu proses fisiologis tanaman, terutama fotosintesis, sehingga berdampak pada pertumbuhan dan hasil produksi. Studi terbaru dalam bidang patologi tanaman menunjukkan bahwa penyakit daun tomat bersifat kompleks dan dipengaruhi oleh faktor lingkungan seperti kelembapan, suhu, dan kepadatan tanam [22], [23]. Berikut adalah penjelasan singkat masing-masing kelas penyakit yang digunakan dalam penelitian ini:

a. Healthy

Gambar 2.1 menunjukkan kelas *healthy* yang merepresentasikan daun tomat dalam kondisi normal tanpa gejala infeksi. Daun sehat umumnya memiliki warna hijau merata, tekstur utuh, serta tidak menunjukkan bercak atau deformasi. Kelas ini digunakan sebagai pembanding untuk membedakan karakteristik daun sehat dan daun yang terinfeksi penyakit [22].

Gambar 2.1 Contoh Daun *Tomat Sehat*

b. Bacterial Spot

Bacterial spot, pada Gambar 2.2, merupakan penyakit yang disebabkan oleh bakteri dari genus *Xanthomonas* yang menginfeksi daun, batang, dan buah tomat. Gejala utama berupa bercak kecil berwarna gelap yang dapat berkembang menjadi

nekrosis dan menyebabkan gugurnya daun pada infeksi berat [22]. Penyakit ini dapat menurunkan kualitas tanaman serta menghambat pertumbuhan apabila tidak ditangani sejak dini.

Gambar 2.2 Contoh Daun Tomat *Bacterial Spot*

c. Early Blight

Gambar 2.3 menunjukkan penyakit *early blight* yang disebabkan oleh jamur *Alternaria solani* dan ditandai dengan bercak cokelat berbentuk konsentris (*target spot*) pada daun yang lebih tua. Penyakit ini dapat menyebabkan defoliasi dini dan menurunkan hasil panen secara signifikan [22]. Pada kondisi infeksi yang parah, penyebaran penyakit dapat mengurangi produktivitas tanaman secara keseluruhan.

Gambar 2.3 Contoh Daun Tomat *Early Blight*

d. Late Blight

Gambar 2.4 menunjukkan penyakit late blight yang disebabkan oleh *Phytophthora infestans*, patogen yang sangat agresif terutama pada kondisi lembap. Gejala meliputi bercak berwarna gelap yang cepat meluas dan sering disertai

pertumbuhan miselium putih pada bagian bawah daun . Penyakit ini dikenal sebagai salah satu penyakit tomat yang paling merusak karena mampu menyebar dengan cepat[22].

Gambar 2.4 Contoh Daun Tomat *Late Blight*

e. Leaf Mold

Penyakit *leaf mold*, yang ditunjukkan pada Gambar 2.5, disebabkan oleh jamur *Passalora fulva* (*Cladosporium fulvum*). Penyakit ini ditandai dengan bercak kuning pada permukaan atas daun dan lapisan jamur berwarna hijau keabu-abuan pada permukaan bawah daun. Infeksi yang berkelanjutan dapat menurunkan kemampuan fotosintesis tanaman dan memengaruhi hasil panen.[22].

Gambar 2.5 Contoh Daun Tomat *Leaf Mold*

f. Septoria Leaf Spot

Gambar 2.6 merupakan *Septoria Leaf Spot* yang disebabkan oleh *Septoria lycopersici*, penyakit ini menimbulkan bercak kecil berbentuk bulat dengan pusat berwarna abu-abu dan tepi gelap. Infeksi berat dapat menyebabkan daun menguning dan rontok [22]. Karakteristik bercak yang berukuran kecil dan tersebar pada permukaan daun menjadikan pola visual penyakit ini sebagai salah satu fitur penting dalam proses identifikasi menggunakan metode berbasis pengolahan citra.

Gambar 2.6 Contoh Daun Tomat *Septoria Leaf Spot*

g. Spider Mites

Spider mites, yang dapat dilihat pada Gambar 2.7, merupakan hama tungau (*Tetranychus urticae*) yang menyerang daun dengan cara mengisap cairan sel. Gejala berupa bintik-bintik kuning kecil dan jaring halus pada permukaan daun yang dapat menyebabkan daun mengering. Kerusakan yang ditimbulkan dapat mengganggu pertumbuhan tanaman dan menurunkan kualitas daun.[23].

Gambar 2.7 Contoh Daun Tomat *Spider Mites*

h. Target Spot

Target spot, yang dapat dilihat pada Gambar 2.8, disebabkan oleh *Corynespora cassiicola* dan ditandai dengan bercak melingkar besar berpola konsentris menyerupai target. Penyakit ini dapat berkembang cepat pada kondisi kelembapan tinggi. Penyebaran yang tidak terkendali dapat menyebabkan kerusakan daun dalam jumlah besar.[23]

Gambar 2.8 Contoh Daun Tomat *Target Spot*

i. Tomato Mosaic Virus (ToMV)

Gambar 2.9 menunjukkan penyakit *Tomato Mosaic Virus* (*ToMV*) yang menyebabkan pola mosaik berwarna hijau muda dan hijau tua pada daun. Virus ini juga menyebabkan distorsi bentuk daun dan pertumbuhan tanaman yang terhambat serta menyebar melalui kontak mekanis dan bahan tanam yang terkontaminasi. Infeksi virus dapat menurunkan kualitas dan produktivitas tanaman secara signifikan.[23].

Gambar 2.9 Contoh Daun Tomat *Mosaic Virus*

j. *Tomato Yellow Leaf Curl Virus* (TYLCV)

Gambar 2.10 menunjukkan penyakit *Tomato Yellow Leaf Curl Virus* (*TYLCV*) yang ditularkan oleh serangga vektor *Bemisia tabaci* (*whitefly*). Gejala khas berupa daun menguning, menggulung ke atas, serta pertumbuhan tanaman yang kerdil dan penurunan hasil panen. Penyakit ini menjadi salah satu ancaman utama pada budidaya tomat di berbagai wilayah produksi. [23].

Gambar 2.10 Contoh Daun Tomat *Yellow Leaf Curl Virus*

2.2.2 Klasifikasi Citra

Klasifikasi citra merupakan salah satu tugas fundamental dalam bidang *computer vision* yang bertujuan untuk mengelompokkan suatu citra digital ke dalam kelas atau kategori tertentu berdasarkan karakteristik visualnya. Secara umum, proses klasifikasi citra melibatkan pemetaan input berupa data piksel dua atau tiga dimensi menjadi label kelas melalui suatu fungsi prediktif. Tahapan utama dalam klasifikasi citra meliputi akuisisi data, *preprocessing*, ekstraksi fitur, dan proses klasifikasi. Pada pendekatan tradisional, fitur diekstraksi secara manual menggunakan metode statistik atau tekstural seperti histogram warna, Gray Level Co-occurrence Matrix (GLCM), Scale-Invariant Feature Transform (SIFT), dan Histogram of Oriented Gradients (HOG), yang kemudian diproses menggunakan algoritma klasifikasi seperti Support Vector Machine (SVM), K-Nearest Neighbor (KNN), atau Decision Tree. Pendekatan ini banyak digunakan sebelum

berkembangnya metode pembelajaran mendalam dan masih relevan untuk *dataset* berukuran kecil hingga menengah [24].

2.2.3 Deteksi Penyakit Tanaman

Deteksi penyakit tanaman merupakan proses identifikasi dini terhadap gejala infeksi patogen pada tanaman dengan tujuan meminimalkan penyebaran penyakit dan mengurangi kerugian hasil panen. Secara konvensional, deteksi dilakukan melalui observasi visual oleh petani atau pakar patologi tanaman. Namun, metode ini memiliki keterbatasan karena bersifat subjektif, membutuhkan keahlian khusus, serta kurang efisien untuk skala lahan yang luas. Perkembangan teknologi pengolahan citra dan kecerdasan buatan telah mendorong penggunaan pendekatan otomatis berbasis citra digital untuk meningkatkan akurasi dan konsistensi diagnosis penyakit tanaman [25]. Studi terkini menunjukkan bahwa sistem berbasis visi komputer mampu mengidentifikasi gejala penyakit dari pola bercak, perubahan warna, tekstur, hingga deformasi daun dengan tingkat akurasi yang tinggi pada berbagai jenis tanaman [26].

Dalam implementasinya, deteksi penyakit tanaman berbasis citra umumnya melibatkan tahapan akuisisi gambar, *preprocessing*, ekstraksi fitur, serta klasifikasi menggunakan algoritma *machine learning* atau *deep learning*. Pendekatan modern tidak hanya berfokus pada peningkatan akurasi, tetapi juga pada aspek *robustness* terhadap variasi pencahayaan, sudut pengambilan gambar, dan latar belakang yang kompleks. Selain itu, integrasi sistem deteksi dengan perangkat mobile, drone, dan platform pertanian digital semakin memperluas penerapannya dalam mendukung konsep *precision agriculture* [25], [26]. Dengan demikian, deteksi penyakit tanaman berbasis kecerdasan buatan berperan penting dalam meningkatkan efisiensi pengendalian penyakit, mendukung pengambilan keputusan yang lebih cepat, serta menjaga stabilitas produksi pertanian secara berkelanjutan.

2.2.4 Transfer Learning

Transfer learning merupakan salah satu pendekatan dalam *deep learning* yang memanfaatkan model yang telah dilatih sebelumnya (*pre-trained model*) pada dataset berskala besar untuk menyelesaikan tugas klasifikasi pada dataset yang

berbeda. Pendekatan ini memungkinkan model menggunakan pengetahuan yang telah dipelajari sebelumnya, seperti kemampuan mengenali pola dasar berupa tepi, tekstur, bentuk, maupun karakteristik visual lainnya, sehingga proses pelatihan menjadi lebih efisien dibandingkan membangun model dari awal (*training from scratch*) [20].

Pada implementasinya, transfer learning dilakukan dengan menggunakan bobot awal (*pre-trained weights*) dari model yang telah dilatih pada dataset umum, seperti ImageNet, kemudian melakukan proses *fine-tuning* menggunakan dataset target. Melalui proses tersebut, model dapat menyesuaikan representasi fitur dengan karakteristik data baru tanpa harus mempelajari seluruh fitur dari awal. Pendekatan ini juga mampu mempercepat proses konvergensi, mengurangi kebutuhan data pelatihan yang besar, serta meningkatkan kemampuan generalisasi model [11].

2.2.5 Early Stopping

Early stopping merupakan teknik yang digunakan untuk mencegah terjadinya *overfitting* selama proses pelatihan model *deep learning*. Prinsip kerjanya adalah menghentikan proses pelatihan secara otomatis apabila performa model pada data validasi tidak menunjukkan peningkatan setelah sejumlah epoch tertentu (*patience*). Dengan menghentikan pelatihan pada saat yang tepat, model dapat mempertahankan kemampuan generalisasi yang lebih baik sekaligus mengurangi waktu komputasi yang tidak diperlukan [27]. Pada penelitian ini, diterapkan *early stopping* dengan nilai *patience* sebesar lima epoch, sehingga proses pelatihan akan dihentikan apabila nilai evaluasi pada data validasi tidak mengalami perbaikan selama lima epoch berturut-turut.

2.2.6 *Deep learning*

Deep learning merupakan cabang dari *machine learning* yang menggunakan jaringan saraf tiruan (*Artificial Neural Network*) dengan banyak lapisan untuk mempelajari representasi data secara otomatis. Berbeda dengan metode tradisional yang memerlukan ekstraksi fitur manual, *deep learning* mampu melakukan pembelajaran fitur secara end-to-end langsung dari data mentah [18].

Pada bidang *computer vision*, *deep learning* terbukti mampu mencapai performa yang sangat tinggi dalam berbagai tugas seperti klasifikasi citra, deteksi objek, dan segmentasi. Kemampuan ini diperoleh melalui proses pelatihan berbasis optimasi fungsi loss menggunakan algoritma backpropagation [20]. Keunggulan utama *deep learning* dalam klasifikasi penyakit tanaman adalah kemampuannya mengenali pola kompleks seperti tekstur bercak, perubahan warna daun, dan variasi bentuk gejala penyakit secara otomatis.

Pada penelitian ini, setiap jenis augmentasi diterapkan menggunakan beberapa tingkat severity yang ditentukan secara bertahap untuk menghasilkan variasi gangguan ringan hingga berat. Pemilihan nilai severity dilakukan secara empiris berdasarkan karakteristik masing-masing transformasi dan mengacu pada praktik umum. Pendekatan ini bertujuan untuk mengamati pola degradasi performa model secara bertahap seiring meningkatnya intensitas gangguan visual.

2.2.7 *Optimizer Adam*

Adam (*Adaptive Moment Estimation*) merupakan salah satu algoritma optimasi yang banyak digunakan dalam pelatihan model deep learning karena mampu mempercepat proses konvergensi dan menghasilkan performa yang stabil. Adam menggabungkan keunggulan metode Momentum dan Root *Mean* Square Propagation (RMSProp) dengan menghitung estimasi momen pertama (first moment) dan momen kedua (second moment) dari gradien untuk menyesuaikan nilai *learning rate* setiap parameter secara adaptif. Pendekatan ini memungkinkan proses pembelajaran menjadi lebih efisien dibandingkan metode optimasi konvensional seperti Stochastic Gradient Descent (SGD), terutama pada permasalahan dengan jumlah parameter yang besar [28]. Dalam penelitian ini, optimizer Adam digunakan pada proses pelatihan EfficientNet-B0 dan FastViT-T8 karena mampu memberikan proses optimasi yang cepat dan stabil.

2.2.8 *Data augmentation*

Data augmentation merupakan teknik untuk memperkaya variasi data pelatihan dengan melakukan transformasi pada citra tanpa mengubah label kelasnya

Dalam penelitian ini, augmentasi dilakukan secara individual meliputi:

a. *Brightness*

Transformasi *brightness* mengubah tingkat kecerahan citra dengan menyesuaikan intensitas piksel. Variasi ini mensimulasikan perbedaan kondisi pencahayaan saat pengambilan gambar di lapangan. Penyesuaian *brightness* terbukti membantu model menjadi lebih adaptif terhadap kondisi cahaya rendah maupun berlebih [29].

b. *Contrast*

Contrast augmentation mengatur perbedaan intensitas antara area terang dan gelap dalam citra. Teknik ini membantu model mengenali pola meskipun terdapat perbedaan distribusi intensitas warna akibat perubahan lingkungan atau kualitas kamera [30].

c. *Saturation*

Saturation mengontrol tingkat kejenuhan warna pada citra. Perubahan saturasi mensimulasikan variasi kondisi sensor kamera dan pencahayaan alami, sehingga model dapat tetap mengenali pola meskipun terjadi perubahan intensitas warna. Dengan melatih ektahanan *Saturation,* model dapat menjadi lebih tahan.

d. *Hue*

Hue augmentation menggeser spektrum warna citra tanpa mengubah struktur objek. Transformasi ini penting untuk meningkatkan ketahanan model terhadap variasi warna akibat faktor lingkungan atau perangkat akuisisi gambar [29]. Perubahan nilai hue dapat menggeser warna citra menjadi lebih kebiruan, kehijauan, atau kemerahan tanpa mengubah bentuk maupun tekstur daun. Oleh karena itu, augmentasi ini digunakan untuk mengevaluasi kemampuan model dalam mempertahankan performa ketika karakteristik warna citra mengalami perubahan.

e. *Affine*

Transformasi *affine* mencakup kombinasi translasi, rotasi, skala, dan shear. Augmentasi ini mensimulasikan perubahan sudut pandang dan jarak kamera terhadap objek sehingga meningkatkan kemampuan generalisasi model terhadap variasi perspektif sederhana. Meskipun bentuk objek mengalami perubahan secara

geometris, hubungan antarbagian objek tetap dipertahankan sehingga karakteristik utama daun masih dapat dikenali.

f. *Perspective*

Perspective transformation mensimulasikan perubahan sudut pandang yang lebih kompleks dibanding *affine*, sehingga objek tampak mengalami distorsi perspektif. Teknik ini penting dalam skenario pengambilan gambar di lapangan yang tidak selalu tegak lurus terhadap objek [29]. Perubahan perspektif dapat mengubah bentuk, ukuran, dan proporsi objek pada citra tanpa menghilangkan informasi visual secara keseluruhan.

g. *Blur*

Blur augmentation menambahkan efek kabur pada citra untuk mensimulasikan gangguan seperti fokus kamera yang kurang tepat atau gerakan saat pengambilan gambar. Penambahan *blur* membantu model tetap mampu mengenali pola utama meskipun detail citra berkurang [30]. Pendekatan augmentasi individual digunakan untuk mengukur dampak masing-masing variasi visual terhadap performa model.

2.2.9 *Robustness* dan Generalisasi Model

Robustness merupakan kemampuan model untuk mempertahankan performa ketika dihadapkan pada variasi data yang berbeda dari kondisi pelatihan. Model yang robust tidak hanya memiliki akurasi tinggi pada *dataset* bersih, tetapi juga stabil terhadap gangguan visual seperti perubahan pencahayaan, rotasi, atau distorsi. Generalisasi berkaitan dengan kemampuan model untuk bekerja dengan baik pada data baru yang belum pernah dilihat sebelumnya. Evaluasi *robustness* penting dalam konteks implementasi sistem klasifikasi penyakit tanaman di lapangan, di mana kondisi visual sering kali tidak ideal.

2.2.10 Metrik Evaluasi

Dalam penelitian ini, evaluasi performa model dilakukan tidak hanya berdasarkan metrik klasifikasi konvensional, tetapi juga menggunakan metrik *robustness* untuk mengukur ketahanan model terhadap variasi kondisi visual. Berikut merupakan metrik yang digunakan:

a. *Accuracy*

Accuracy merupakan metrik yang mengukur proporsi jumlah prediksi yang benar dibandingkan dengan keseluruhan data uji. Metrik ini memberikan gambaran umum mengenai tingkat ketepatan model dalam melakukan klasifikasi.

$$Accuracy = Number\ of\ Correct\ Predictions/Total\ Number\ of\ Data \quad (2.1)$$

Pada Rumus 2.1, *Number of Correct Predictions* menyatakan jumlah sampel yang berhasil diklasifikasikan dengan benar oleh model, sedangkan *Total Number of Data* merupakan jumlah seluruh sampel pada data uji. Nilai *accuracy* berada pada rentang 0 hingga 1 atau dapat dinyatakan dalam bentuk persentase, di mana nilai yang semakin tinggi menunjukkan bahwa model memiliki tingkat ketepatan klasifikasi yang semakin baik.

Meskipun mudah dipahami dan sering digunakan, *accuracy* dapat menjadi kurang representatif pada kondisi *dataset* yang tidak seimbang karena tidak mempertimbangkan distribusi kelas secara spesifik [31].

b. *Precision*

Precision mengukur tingkat ketepatan prediksi positif yang dihasilkan oleh model. Nilai *precision* yang tinggi menunjukkan bahwa model memiliki tingkat kesalahan positif palsu (*false positive*) yang rendah. Metrik ini penting dalam sistem diagnosis otomatis untuk menghindari kesalahan identifikasi penyakit pada objek yang sebenarnya sehat.

$$Precision\ =\ True\ Positive\ /\ (True\ Positive + False\ Positive) \quad (2.2)$$

Pada Persamaan 2.2, *True Positive* menyatakan jumlah sampel positif yang berhasil diprediksi dengan benar oleh model, sedangkan *False Negative* merupakan jumlah sampel positif yang salah diprediksi sebagai kelas lain. Nilai *recall* berada pada rentang 0 hingga 1 atau dapat dinyatakan dalam bentuk persentase, di mana nilai yang semakin tinggi menunjukkan bahwa model semakin baik dalam mendeteksi seluruh sampel positif [31]. Pada klasifikasi penyakit daun tomat, nilai recall yang tinggi menunjukkan bahwa model mampu meminimalkan jumlah citra penyakit yang gagal dikenali sehingga potensi kesalahan identifikasi menjadi lebih rendah.

c. Recall

Recall atau sensitivitas mengukur kemampuan model dalam mendeteksi seluruh data yang benar-benar termasuk dalam kelas tertentu.

$$Recall = True\ Positive\ /\ (True\ Positive + False\ Negative) \quad (2.3)$$

Pada Rumus 2.3, *True Positive* menyatakan jumlah sampel positif yang berhasil diprediksi dengan benar oleh model, sedangkan *False Negative* merupakan jumlah sampel positif yang salah diprediksi sebagai kelas lain. Nilai *recall* berada pada rentang 0 hingga 1 atau dapat dinyatakan dalam bentuk persentase, di mana nilai yang semakin tinggi menunjukkan bahwa model semakin baik dalam mendeteksi seluruh sampel positif [31]. Nilai *recall* yang tinggi menunjukkan bahwa model mampu menangkap sebagian besar kasus positif, sehingga risiko kesalahan negatif palsu (false negative) dapat diminimalkan.

d. F1-score

F1-score merupakan kombinasi harmonik antara *precision* dan *recall* yang digunakan untuk memberikan keseimbangan antara keduanya.

$$F1\ Score = 2((Precision\ x\ Recall)/(Precision + Recall) \quad (2.4)$$

Pada Rumus 2.4, *Precision* menunjukkan tingkat ketepatan model dalam memprediksi kelas positif, sedangkan *Recall* menunjukkan kemampuan model dalam mendeteksi seluruh sampel positif. Nilai *F1-score* berada pada rentang 0 hingga 1 atau dapat dinyatakan dalam bentuk persentase, di mana nilai yang semakin tinggi menunjukkan bahwa model memiliki keseimbangan yang semakin baik antara *precision* dan *recall*. Metrik ini sangat relevan pada klasifikasi multi-kelas atau *dataset* yang tidak seimbang karena mampu merepresentasikan performa model secara lebih komprehensif dibandingkan *accuracy* saja [31].

e. *Confusion matrix*

Confusion matrix merupakan tabel evaluasi yang menampilkan distribusi hasil prediksi model terhadap label sebenarnya pada setiap kelas. Melalui *confusion matrix*, dapat dianalisis pola kesalahan klasifikasi secara lebih rinci, termasuk kelas mana yang paling sering tertukar oleh model.

f. *Performance drop* (*Drop Accuracy*)

Performance drop digunakan untuk mengukur penurunan akurasi model akibat penerapan augmentasi dengan tingkat gangguan tertentu. Nilai *drop* dihitung menggunakan rumus berikut:

$$Drop = Accuracy\ Clean - Accuracy\ Severity \quad (2.5)$$

Pada Rumus 2.5, Accuracy Clean *menyatakan akurasi model pada data uji tanpa augmentasi, sedangkan Accuracy Severity* merupakan akurasi model pada data uji setelah diberikan augmentasi pada tingkat *severity* tertentu. Nilai *performance drop* yang semakin kecil menunjukkan bahwa penurunan performa model akibat gangguan visual juga semakin kecil nilai *drop*, semakin stabil model terhadap variasi kondisi visual[32].

g. *Robustness score*

Robustness score digunakan untuk merepresentasikan ketahanan model secara agregat terhadap gangguan visual, yang dihitung dengan rumus:

$$Robustness = 1 - MeanDrop \quad (2.6)$$

Pada Rumus 2.6, *Mean Drop* merupakan nilai rata-rata *performance drop* yang diperoleh dari seluruh tingkat *severity* pada suatu jenis augmentasi. Nilai *robustness score* berada pada rentang 0 hingga 1, di mana nilai yang semakin mendekati 1 menunjukkan bahwa model mengalami penurunan performa yang semakin kecil sehingga memiliki tingkat ketahanan (*robustness*) yang lebih baik terhadap variasi kondisi visual [32].

h. *Area Under Robustness Curve* (AURC)

AURC mengukur luas area di bawah kurva yang menggambarkan hubungan antara tingkat severity gangguan dan performa model. Metrik ini memberikan evaluasi menyeluruh terhadap degradasi performa model seiring meningkatnya tingkat gangguan visual [24].

i. Total Parameters

Total parameters menunjukkan jumlah keseluruhan parameter dalam model. Model dengan parameter lebih sedikit umumnya lebih ringan dan lebih

efisien secara komputasi, sehingga lebih sesuai untuk implementasi pada perangkat dengan keterbatasan sumber daya.

j. Inference Time (Total Time per Dataset)

Inference time mengukur total waktu yang dibutuhkan model untuk melakukan proses prediksi terhadap seluruh data uji. Metrik ini digunakan untuk mengevaluasi efisiensi komputasi model selama tahap inferensi, yaitu ketika model melakukan klasifikasi tanpa melalui proses pelatihan.

$$Inference\ Time\ =\ End\ Time\ -\ Start\ Time \tag{2.7}$$

Pada Rumus 2.7 *Start Time* menyatakan waktu saat proses inferensi dimulai, sedangkan *End Time* merupakan waktu ketika seluruh proses prediksi selesai dilakukan. Nilai *inference time* dinyatakan dalam satuan detik (*second*) atau milidetik (*millisecond*), di mana nilai yang semakin kecil menunjukkan bahwa model mampu melakukan prediksi dengan lebih cepat dan efisien[33]

k. *Latency* per Image (*Mean Latency*)

Latency per image atau *Mean latency* merupakan rata-rata waktu yang dibutuhkan model untuk memproses satu citra pada tahap inferensi. Metrik ini digunakan untuk mengevaluasi kecepatan prediksi model terhadap setiap citra secara individual sehingga dapat menggambarkan respons sistem dalam penggunaan sehari-hari.

$$Mean\ Latency\ =\ Total\ Inference\ Time\ /\ Total\ Number\ of\ Images \tag{2.8}$$

Pada Rumus 2.8, *Total Inference Time* menyatakan total waktu yang dibutuhkan model untuk melakukan inferensi terhadap seluruh data uji, sedangkan *Total Number of Images* merupakan jumlah seluruh citra yang diproses. Nilai *Mean latency* umumnya dinyatakan dalam satuan milidetik (*millisecond*) per citra, di mana nilai yang semakin kecil menunjukkan bahwa model mampu menghasilkan prediksi dengan lebih cepat untuk setiap citra [33]

l. *Mean confidence score*

Mean confidence score merupakan rata-rata probabilitas keluaran fungsi softmax terhadap kelas yang diprediksi oleh model. Metrik ini digunakan untuk

mengukur tingkat keyakinan model dalam menghasilkan prediksi pada seluruh data uji.

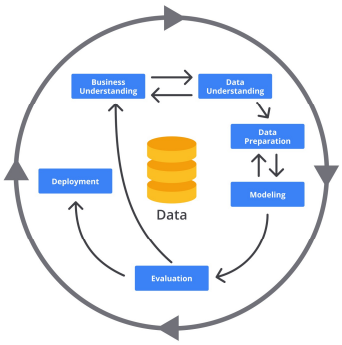
$$Mean\ Confidence\ Score\ =\ \Sigma \frac{Confidence_i}{Sample\ Size} \tag{2.9}$$

Pada Rumus 2.9, *Confidence* menyatakan nilai probabilitas prediksi untuk citra ke-(i) yang dihasilkan oleh fungsi softmax, sedangkan (N) merupakan jumlah seluruh citra (sample size) yang digunakan pada proses evaluasi. Nilai *Mean confidence* score berada pada rentang 0 hingga 1 atau dapat dinyatakan dalam bentuk persentase, di mana nilai yang semakin tinggi menunjukkan bahwa model memiliki tingkat keyakinan yang semakin besar terhadap prediksi yang dihasilkan [32].

2.3 *Framework*/Algoritma yang digunakan

2.3.1 *Framework* CRISP DM

Penelitian ini menggunakan *framework* CRISP-DM (*Cross Industry Standard Process for Data Mining*) yang ditunjukan pada Gambar 2.11, sebagai pendekatan metodologis dalam pengembangan sistem berbasis data dan *machine learning*. CRISP-DM merupakan model proses standar yang banyak digunakan dalam proyek data mining dan analitik karena menyediakan tahapan kerja yang sistematis, terstruktur, dan fleksibel[34].



Gambar 2.11 *Framework* CRISP-DM
Sumber [34]

CRISP-DM terdiri dari enam tahapan utama yang saling berkesinambungan, yaitu:

a. Business Understanding

Tahap *Business Understanding* bertujuan untuk memahami permasalahan yang ingin diselesaikan serta menentukan tujuan dan kebutuhan proyek. Pada tahap ini dilakukan identifikasi ruang lingkup masalah, tujuan analisis, serta kriteria keberhasilan yang ingin dicapai. Tahapan ini menjadi dasar dalam menentukan arah penelitian dan strategi yang akan digunakan.

b. Data Understanding

Tahap *Data Understanding* berfokus pada pengumpulan data dan eksplorasi awal terhadap *dataset* yang tersedia. Proses ini meliputi pemeriksaan struktur data, identifikasi jumlah variabel atau kelas, analisis distribusi data, serta deteksi kemungkinan adanya ketidakseimbangan data atau anomali. Tahap ini penting untuk memahami karakteristik data sebelum dilakukan proses lebih lanjut.

c. Data Preperation

Tahap *Data Preparation* merupakan proses persiapan data agar siap digunakan dalam proses pemodelan. Kegiatan yang dilakukan pada tahap ini umumnya meliputi pembersihan data (*data cleaning*), transformasi data, normalisasi, serta pembagian *dataset* menjadi data latih dan data uji. Pada kasus berbasis citra, tahap ini juga dapat mencakup proses *resizing*, normalisasi piksel, dan augmentasi data. Tahap ini biasanya memakan waktu paling besar dalam keseluruhan proses karena kualitas data sangat memengaruhi hasil pemodelan.

d. Modeling

Tahap *Modeling* merupakan proses pembangunan dan pelatihan model menggunakan algoritma yang telah dipilih. Pada tahap ini dilakukan pemilihan metode atau algoritma, penentuan parameter model, serta proses pelatihan menggunakan data latih. Setiap model yang dibangun kemudian akan menghasilkan output prediksi yang selanjutnya dievaluasi pada tahap berikutnya.

e. *Evaluation*

Tahap *Evaluation* bertujuan untuk menilai kinerja model yang telah dibangun. Evaluasi dilakukan menggunakan metrik yang sesuai dengan jenis permasalahan, seperti akurasi, presisi, *recall*, atau metrik lainnya. Pada tahap ini

juga dilakukan analisis apakah model telah memenuhi tujuan yang ditetapkan pada tahap Business Understanding. Apabila hasil evaluasi belum memenuhi kriteria, proses dapat kembali ke tahap sebelumnya untuk perbaikan model atau penyesuaian data.

f. *Deployment*

Tahap *deployment* merupakan fase akhir dalam kerangka kerja CRISP-DM yang bertujuan untuk mengimplementasikan model yang telah dibangun dan dievaluasi agar dapat digunakan secara nyata oleh pengguna akhir. Setelah model melalui tahapan business understanding, data understanding, data preparation, modeling, dan *evaluation*, tahap *deployment* memastikan bahwa hasil analisis tidak hanya berhenti pada eksperimen, tetapi benar-benar memberikan nilai tambah dalam lingkungan operasional. *Deployment* dapat dilakukan dalam berbagai bentuk, seperti integrasi model ke dalam aplikasi berbasis web atau mobile, sistem monitoring otomatis, pembuatan dashboard analitik, maupun penyusunan laporan keputusan berbasis data. Dalam konteks penelitian klasifikasi citra, *deployment* biasanya berupa integrasi model ke dalam sistem yang mampu menerima input citra baru dan menghasilkan prediksi kelas secara *real-time* atau batch processing.

2.3.2 *Perceptual Hashing*

Pemeriksaan duplikasi citra merupakan salah satu tahapan pada proses data preparation yang bertujuan untuk memastikan kualitas dataset sebelum digunakan dalam proses pelatihan dan evaluasi model. Keberadaan citra yang sama atau sangat mirip pada data latih, validasi, dan data uji (*data leakage*) dapat menyebabkan hasil evaluasi menjadi terlalu optimistis karena model berpotensi mengenali citra yang telah dipelajari sebelumnya. Oleh karena itu, pemeriksaan duplikasi diperlukan untuk memastikan bahwa proses evaluasi benar-benar dilakukan pada data yang independen.

Salah satu metode yang umum digunakan untuk mendeteksi kemiripan citra adalah *Perceptual Hashing* (pHash). Berbeda dengan fungsi hash konvensional yang menghasilkan nilai berbeda meskipun hanya terdapat sedikit perubahan pada citra, *perceptual hashing* menghasilkan representasi hash berdasarkan karakteristik

visual citra. Dengan demikian, citra yang identik atau memiliki kemiripan visual yang tinggi akan menghasilkan nilai *hash* yang sama atau sangat mirip sehingga lebih efektif digunakan untuk mendeteksi duplikasi citra [35].

2.3.3 *Penghilangan Citra Duplikat*

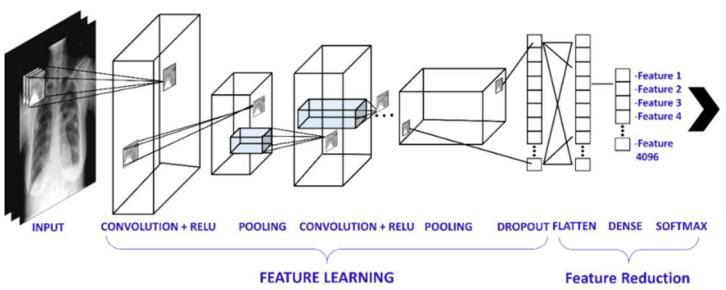
Penghilangan citra duplikat (duplicate image removal) merupakan salah satu tahapan dalam data preparation yang bertujuan untuk meningkatkan kualitas dataset. Keberadaan citra yang identik atau memiliki kemiripan yang sangat tinggi dapat menyebabkan redundansi data sehingga distribusi dataset menjadi kurang representatif. Selain itu, apabila citra duplikat tersebar pada data latih, validasi, dan data uji, kondisi tersebut dapat menimbulkan *data leakage* yang menyebabkan hasil evaluasi model menjadi terlalu optimistis [36].

Proses penghilangan duplikasi umumnya dilakukan setelah tahap deteksi menggunakan metode *perceptual hashing* (pHash). Citra yang memiliki nilai hash identik akan dikelompokkan, kemudian hanya satu citra yang dipertahankan sebagai representasi, sedangkan citra lainnya dihapus. Dengan demikian, dataset yang digunakan pada proses pelatihan dan evaluasi memiliki kualitas yang lebih baik serta mampu menghasilkan pengukuran performa model yang lebih objektif.

2.3.4 *Convolutional Neural Network* (CNN)

Gambar 2.12 memperlihatkan arsitektur Convolutional Neural Network (CNN), yaitu salah satu metode deep learning yang dikembangkan untuk mengolah data berbentuk grid, khususnya citra dua dimensi. CNN melakukan proses pembelajaran dengan menerapkan operasi konvolusi guna mengekstraksi karakteristik penting dari citra secara otomatis. Melalui proses tersebut, model dapat mengenali pola seperti garis tepi, tekstur, hingga bentuk objek secara bertingkat. Secara umum, CNN tersusun atas beberapa bagian utama, antara lain convolution layer yang berperan dalam proses ekstraksi fitur, activation function seperti *Rectified Linear Unit* (ReLU) yang memberikan sifat nonlinier pada jaringan, pooling layer yang digunakan untuk mengurangi ukuran dimensi fitur sekaligus meningkatkan efisiensi komputasi, serta fully connected layer yang

bertugas menghasilkan keluaran atau prediksi akhir berdasarkan fitur yang telah dipelajari sebelumnya.



Gambar 2.12 *Framework* Convolutional Neural Network
Sumber [37]

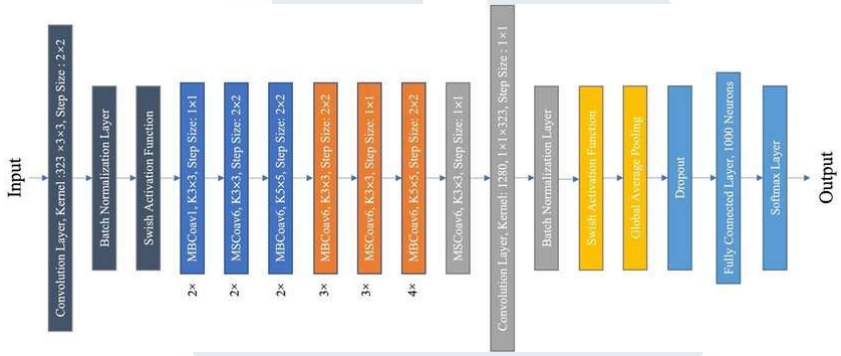
Dengan susunan tersebut, CNN mampu membangun representasi fitur mulai dari tingkat sederhana hingga fitur yang lebih kompleks secara bertahap. Seiring perkembangan penelitian di bidang pengolahan citra, berbagai pendekatan optimasi terhadap arsitektur maupun proses pelatihan CNN terus dilakukan untuk memperoleh performa model yang lebih baik serta efisiensi komputasi yang lebih tinggi dalam berbagai penerapan[37].

CNN dikenal efektif dalam menangkap pola lokal karena mekanisme konvolusinya yang berfokus pada area spasial kecil pada citra. Oleh karena itu, arsitektur ini banyak digunakan dalam tugas klasifikasi penyakit daun berbasis citra, di mana pola bercak, perubahan warna, dan tekstur menjadi indikator utama. Namun, karena sifat konvolusi yang berorientasi lokal, CNN memiliki keterbatasan dalam memodelkan hubungan global antar bagian citra secara langsung. Akibatnya, performa CNN dapat mengalami penurunan ketika dihadapkan pada variasi kondisi visual yang signifikan, seperti perubahan pencahayaan, distorsi perspektif, atau gangguan visual lainnya.

2.3.5 EfficientNet

EfficientNet merupakan salah satu pengembangan arsitektur Convolutional Neural Network (CNN) yang difokuskan pada peningkatan efisiensi model melalui pendekatan *compound scaling*. Metode ini melakukan penskalaan terhadap kedalaman jaringan (*depth*), lebar jaringan (*width*), dan resolusi input secara bersamaan dengan menggunakan satu koefisien skala yang terkontrol. Berbeda

dengan metode konvensional yang umumnya hanya meningkatkan satu aspek arsitektur saja, pendekatan compound scaling pada EfficientNet mampu menghasilkan peningkatan performa yang lebih seimbang dan optimal dengan jumlah parameter yang lebih sedikit. Pendekatan tersebut membuat EfficientNet dapat mencapai akurasi yang tinggi tanpa membutuhkan kompleksitas komputasi yang terlalu besar. Oleh karena itu, EfficientNet banyak dimanfaatkan dalam berbagai penelitian dan aplikasi klasifikasi citra modern karena kemampuannya dalam menjaga keseimbangan antara performa dan efisiensi.



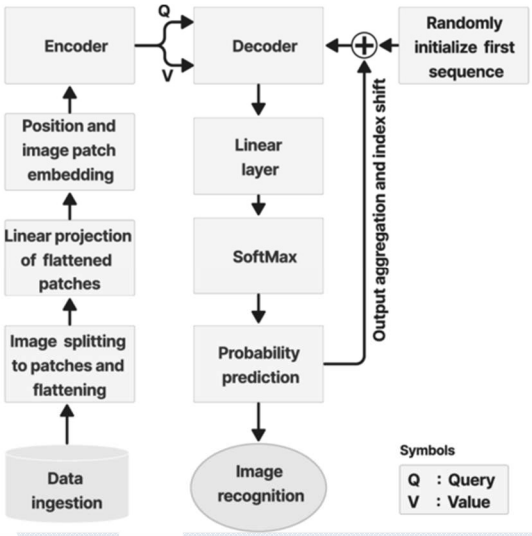
Gambar 2.13 *Framework* EfficientNet
Sumber [33]

EfficientNet-B0 merupakan versi dasar dari keluarga EfficientNet yang menjadi fondasi bagi varian skala lebih besar seperti B1 hingga B7. Meskipun memiliki kompleksitas yang lebih rendah dibandingkan versi lanjutannya, EfficientNet-B0 tetap mampu mempertahankan akurasi yang kompetitif pada berbagai *dataset* klasifikasi citra. Karakteristik ini menjadikannya sesuai untuk implementasi pada perangkat dengan keterbatasan sumber daya komputasi. Selain itu, penelitian terbaru menunjukkan bahwa arsitektur berbasis EfficientNet mampu memberikan performa klasifikasi yang baik pada *dataset* dengan tingkat kemiripan antar kelas yang tinggi [33]. Dalam penelitian ini, EfficientNet-B0 digunakan sebagai representasi arsitektur CNN yang efisien untuk mengevaluasi performa dan *robustness* dalam klasifikasi penyakit daun tomat.

2.3.6 *Vision Transformer* (ViT)

Gambar 2.14 menunjukkan arsitektur Vision Transformer (ViT), yaitu salah satu metode deep learning yang mengadaptasi konsep Transformer dari bidang *Natural Language Processing* (NLP) untuk keperluan pengolahan citra. Tidak

seperti Convolutional Neural Network (CNN) yang mengandalkan operasi konvolusi dalam mengekstraksi fitur lokal, ViT memecah citra menjadi beberapa patch berukuran kecil. Setiap patch kemudian diubah menjadi representasi vektor dan diproses menggunakan mekanisme self-attention.



Gambar 2.14 *Framework Vision Transformer*
Sumber [38]

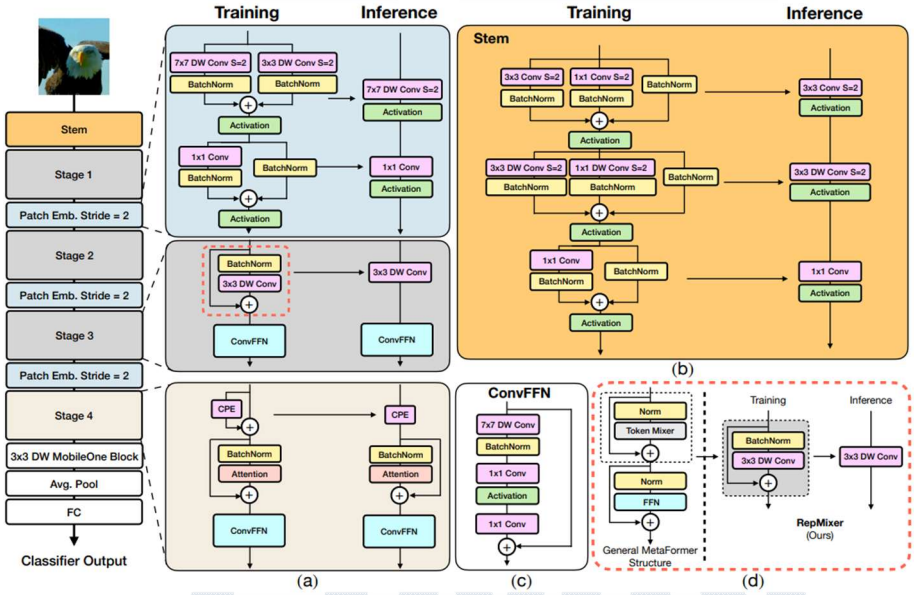
Pendekatan tersebut memungkinkan model memahami keterkaitan antarbagian citra secara global sejak awal proses pemrosesan, tanpa hanya bergantung pada fitur lokal sebagaimana pada CNN. Melalui mekanisme *self-attention*, ViT mampu menangkap hubungan jarak jauh (*long-range dependencies*) dan memahami konteks visual secara lebih menyeluruh dalam satu proses terpadu. Dengan kemampuan tersebut, Vision Transformer dinilai lebih efektif dalam mengenali hubungan global antar objek maupun pola yang terdapat pada citra.

Berbagai perkembangan penelitian terkait Vision Transformer menunjukkan bahwa pendekatan berbasis attention memiliki potensi yang besar dalam meningkatkan performa analisis citra kompleks, termasuk pada penerapan di bidang kesehatan digital dan berbagai aplikasi pengolahan citra lainnya. [38].

2.3.7 FastVit

FastViT merupakan arsitektur hybrid, yang ditunjukan pada Gambar 2.15, yang menggabungkan keunggulan *Convolutional Neural Network* (CNN) dan *Vision Transformer* untuk mencapai keseimbangan antara efisiensi komputasi dan

kemampuan generalisasi model. Berbeda dengan *Vision Transformer* murni yang cenderung memiliki kompleksitas tinggi, FastViT mengintegrasikan komponen konvolusional untuk mempertahankan efisiensi pemrosesan fitur lokal, sekaligus memanfaatkan mekanisme *attention* untuk menangkap hubungan global antar bagian citra. Salah satu kontribusi utama FastViT adalah penggunaan teknik *structural reparameterization*, yaitu strategi yang memungkinkan struktur model disederhanakan pada tahap inferensi tanpa mengurangi kapasitas representasi selama pelatihan. Pendekatan ini secara signifikan meningkatkan kecepatan inferensi dan efisiensi komputasi, sehingga lebih sesuai untuk implementasi pada perangkat dengan keterbatasan sumber daya[9].



Gambar 2.15 *Framework* FastVit
Sumber [9]

Dalam penelitian ini, FastViT digunakan sebagai pembanding EfficientNet-B0 untuk menganalisis perbedaan karakteristik antara arsitektur CNN murni dan arsitektur *hybrid CNN–Transformer*, khususnya dalam aspek *robustness* terhadap variasi kondisi visual. Perbandingan ini penting untuk mengevaluasi sejauh mana kombinasi mekanisme konvolusi dan attention mampu memberikan ketahanan performa yang lebih baik dibandingkan pendekatan konvolusional tradisional.

2.4 *Tools*/software yang Digunakan

2.4.1 Jupyter Notebook

Jupyter Notebook digunakan sebagai lingkungan pengembangan utama dalam penelitian ini. Jupyter Notebook menyediakan platform interaktif yang memungkinkan integrasi antara penulisan kode program, visualisasi hasil, serta dokumentasi eksperimen dalam satu dokumen yang terstruktur.

Lingkungan ini mendukung proses eksplorasi data, pelatihan model, dan analisis hasil secara sistematis, sehingga memudahkan dalam pencatatan serta replikasi eksperimen.

2.4.2 Python

Python digunakan sebagai bahasa pemrograman utama dalam penelitian ini. Python dipilih karena memiliki sintaks yang sederhana serta dukungan pustaka (library) yang luas dalam bidang *machine learning* dan *deep learning*. Dalam penelitian ini, Python digunakan untuk implementasi model, pengolahan data, penerapan augmentasi citra, serta evaluasi performa model klasifikasi.

2.4.3 Gradio

Gradio digunakan sebagai *framework* antarmuka (user *interface*) berbasis web untuk mengimplementasikan model klasifikasi yang telah dilatih ke dalam bentuk aplikasi interaktif. Gradio memungkinkan pembuatan antarmuka sederhana yang dapat menerima input berupa citra, memprosesnya menggunakan model yang telah dikembangkan, serta menampilkan hasil prediksi secara langsung dalam bentuk label kelas dan tingkat kepercayaan (*confidence score*).

Penggunaan Gradio dalam penelitian ini bertujuan untuk mensimulasikan tahap *deployment* secara praktis, sehingga model tidak hanya diuji pada lingkungan eksperimen, tetapi juga dapat digunakan dalam skenario penggunaan nyata. Selain itu, Gradio mendukung integrasi langsung dengan Python dan berbagai *framework deep learning*, sehingga mempermudah proses implementasi tanpa memerlukan pengembangan sistem web yang kompleks. Dengan demikian, Gradio berperan dalam menjembatani proses pengembangan model dan implementasi aplikasi berbasis klasifikasi citra secara lebih aplikatif dan terstruktur.