BAB 2
LANDASAN TEORI

2.1 Tinjauan Teori

2.1.1 Deteksi Tanaman Melon

Berdasarkan penelitian sebelumnya, identifikasi penyakit *Powdery Mildew* pada tanaman melon telah dilakukan menggunakan metode *YOLO* dengan bantuan *Deep Learning, Faster R-CNN, dan Single Shot Multibox Detection*. Penelitian tersebut berfokus pada pendeteksian abnormalitas pada daun melon. Selain itu, identifikasi penyakit daun melon dalam *greenhouse* telah diimplementasikan menggunakan model *Pruned-YOLO v5s+Shuffle (PYSS)*, yang menggabungkan *YOLO v5s* dengan *ShuffleNet V2* untuk deteksi secara *real-time*. Model ini memiliki tingkat akurasi dan kecepatan lebih tinggi dibandingkan *YOLO v3, Faster R-CNN, dan YOLO v5s*. Namun, hingga saat ini, belum ada sistem yang mengidentifikasi penyakit *Powdery Mildew, Cucumber Mosaic Virus*, dan *Alternaria* pada tanaman melon dengan pendekatan berbasis *CNN*.

2.1.2 *Alternaria Leaf Blight*

Alternaria Leaf Blight merupakan genus jamur patogen yang sering menginfeksi berbagai jenis tanaman, termasuk melon (*Cucumis melo*). Infeksi yang disebabkan oleh jamur ini dikenal sebagai *alternaria leaf spot* atau *alternaria blight*. Penyakit ini dapat berdampak pada penurunan kualitas dan hasil panen tanaman melon, yang berpotensi menyebabkan kerugian ekonomi yang cukup besar di sektor pertanian[3].

A Penyebab *Alternaria Leaf Blight* pada Tanaman Melon

Penyakit *Alternaria Leaf Blight* pada tanaman melon disebabkan oleh jamur *alternaria alternata*, yang merupakan patogen utama dari genus *alternaria*. Jamur ini menghasilkan konidia (spora aseksual) yang dapat menyebar melalui angin, percikan air hujan, atau kontak langsung antara tanaman [4]. Spora yang mendarat pada permukaan daun melon akan berkecambah dan menembus jaringan tanaman, sehingga memicu infeksi.

B Gejala *Alternaria Leaf Blight* Tanaman Melon

Tanaman melon yang terinfeksi *alternaria* umumnya menunjukkan gejala pada bagian daunnya. Pada tahap awal infeksi, muncul bercak kecil berwarna coklat hingga kehitaman dengan tepi yang tidak beraturan. Seiring waktu, bercak tersebut membesar dan membentuk pola cincin konsentris yang menyerupai target. Jika infeksi semakin parah, daun yang terkena akan menguning, mengering, dan akhirnya rontok. Pada kasus yang lebih berat, penyakit ini dapat menyebabkan kematian tanaman secara keseluruhan[4].

2.1.3 *Cucumber Mosaic Virus*

Cucumber Mosaic Virus (CMV) merupakan salah satu virus yang sering menginfeksi tanaman melon (*cucumis melo*) serta berbagai jenis tanaman lainnya. Virus ini termasuk dalam kelompok *cucumovirus* dari famili *cucumoviridae* dan dapat menyebabkan gangguan pertumbuhan yang signifikan, sehingga berdampak pada penurunan hasil panen. Infeksi *cucumber mosaic virus* pada tanaman melon umumnya mempengaruhi kualitas daun dan buah, dengan gejala yang bervariasi tergantung pada tingkat keparahan infeksi serta kondisi lingkungan sekitar[5].

A Penyebab *Cucumber Mosaic Virus* pada Tanaman Melon

CMV disebabkan oleh virus RNA dari genus *cucumovirus*, dengan *cucumber mosaic virus* sebagai spesies utamanya. Penyebaran virus ini terjadi melalui vektor serangga, khususnya kutu daun dari famili *aphididae* [5]. Kutu daun yang mengisap getah dari tanaman yang terinfeksi dapat membawa partikel virus dan menularkannya ke tanaman sehat melalui gigitan berikutnya. Selain melalui serangga vektor, *cucumber mosaic virus* juga dapat menyebar melalui kontak langsung dengan tanaman yang terinfeksi atau melalui peralatan pertanian yang telah terkontaminasi virus.

B Gejala Infeksi *Cucumber Mosaic Virus* pada Tanaman Melon

Tingkat infeksi *cucumber mosaic virus* pada tanaman melon dapat menimbulkan berbagai gejala yang berbeda, tergantung pada kondisi tanaman dan faktor lingkungan. Gejala yang umum terjadi meliputi perubahan warna daun yang tidak merata, seperti munculnya bercak kuning atau hijau pucat, serta pola

mosaik hijau terang dan gelap [6]. Selain itu, daun dapat mengalami deformasi atau keriting, sementara pertumbuhan tanaman menjadi terhambat. Dalam kondisi infeksi yang lebih parah, buah yang dihasilkan bisa berukuran lebih kecil, memiliki bentuk tidak normal, atau bahkan gagal berkembang. Jika infeksi terjadi sejak dini, tanaman melon yang terinfeksi *cucumber mosaic virus* berisiko mengalami kematian lebih cepat.

2.1.4 *Downy Mildew*

Downy mildew adalah penyakit tanaman yang disebabkan oleh jamur yang bernama *pseudoperonospora cubensis*. Penyakit ini terutama menyerang tanaman dari famili *cucurbitaceae*, seperti melon, mentimun, dan semangka. Infeksi *downy mildew* terjadi pada bagian daun, dengan gejala khas berupa bercak kuning (klorosis) yang berkembang menjadi nekrosis. Penyakit ini menyebar dengan cepat melalui spora yang terbawa angin, terutama pada kondisi lingkungan yang lembab. *Downy mildew* dapat menurunkan kapasitas fotosintesis tanaman secara signifikan dan menyebabkan penurunan hasil panen, bahkan kegagalan produksi apabila tidak dikendalikan dengan baik [7] .

A Penyebab *Downy Mildew* pada Tanaman Melon

Penyakit ini disebabkan jamur *pseudoperonospora cubensis*, yang berkembang biak melalui spora yang tersebar lewat udara. Spora ini menyerang bagian daun tanaman dan dapat menyebabkan kerusakan jaringan secara luas. Infeksi umumnya terjadi saat kelembaban daun tinggi, karena kondisi basah mendukung perkecambahan spora dan pembentukan struktur infeksi awal. Tidak seperti p*owdery mildew*, *downy mildew* sangat bergantung pada kelembaban tinggi untuk memulai infeksi.

B Gejala *Downy Mildew* pada Tanaman Melon

Gejala awal ditandai dengan munculnya bercak klorosis berwarna kuning muda pada permukaan daun yang tidak terbatas oleh tulang daun dan berbentuk tidak beraturan. Seiring waktu, bercak ini membesar dan berubah warna menjadi kuning terang, kemudian berkembang menjadi nekrosis berwarna coklat keoranyean hingga coklat tua. Infeksi tingkat lanjut dapat menyebabkan kerusakan

parah pada daun dan menurunkan kemampuan fotosintesis tanaman, sehingga tanaman tampak kerdil dan tidak tumbuh optimal.

2.1.5 *Powdery Mildew*

Powdery mildew merupakan penyakit yang disebabkan oleh berbagai jenis jamur patogen yang menyerang tanaman melon (*cucumis melo*). Infeksi ini dapat berdampak signifikan terhadap pertumbuhan serta hasil panen melon. Penyakit ini umumnya dipicu oleh jamur *erysiphe cichoracearum* dan *sphaerotheca fuliginea*, yang termasuk dalam ordo *erysiphales*[8]. Serangan *powdery mildew* dapat menurunkan kualitas serta kuantitas hasil panen, yang pada akhirnya berpengaruh terhadap nilai ekonomi melon di pasaran.

A Penyebab *Powdery Mildew* pada Tanaman Melon

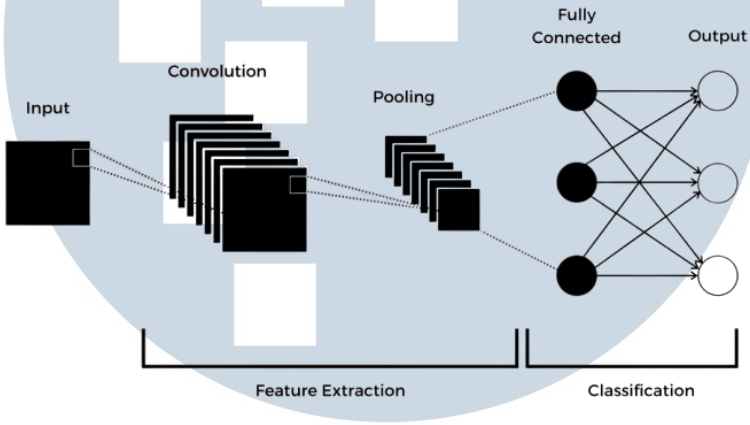
Penyakit ini disebabkan oleh jamur *erysiphe cichoracearum* dan *sphaerotheca fuliginea*, yang berkembang biak melalui spora yang menyebar lewat angin. Spora ini dapat menginfeksi berbagai bagian tanaman, seperti daun, batang, dan tunas. Perkecambahan spora terjadi pada permukaan tanaman dengan tingkat kelembaban yang cukup, meskipun kondisi tersebut tidak harus terlalu basah. Hal ini membedakan *powdery mildew* dari jenis penyakit jamur lainnya yang umumnya berkembang dalam lingkungan yang lebih lembab[9].

B Gejala *Powdery Mildew* pada Tanaman Melon

Gejala awal infeksi *powdery mildew* pada melon ditandai dengan munculnya bercak putih menyerupai tepung pada permukaan daun, baik di bagian atas maupun bawah. Bercak ini merupakan hasil pertumbuhan miselium jamur serta konidia. Jika infeksi semakin parah, bercak tersebut akan menyebar luas, menyebabkan daun menguning dan akhirnya mengering. Infeksi yang cukup berat dapat menghambat proses fotosintesis, memperlambat pertumbuhan tanaman, serta menurunkan kualitas buah melon. Dalam kasus yang lebih ekstrem, infeksi parah dapat menyebabkan kematian tanaman.

2.1.6 *Convolutional Neural Networks (CNN)*

Convolutional Neural Network (CNN) merupakan bentuk khusus dari *Artificial Neural Network (ANN)* yang secara luas dianggap sebagai pendekatan paling efektif dalam pengenalan gambar. CNN memiliki tiga komponen utama dalam arsitekturnya, yaitu lapisan konvolusi, pooling, dan *fully connected*. Struktur CNN yang ditampilkan terdiri dari dua lapisan konvolusi dan pooling, serta dua lapisan *fully connected*[10].



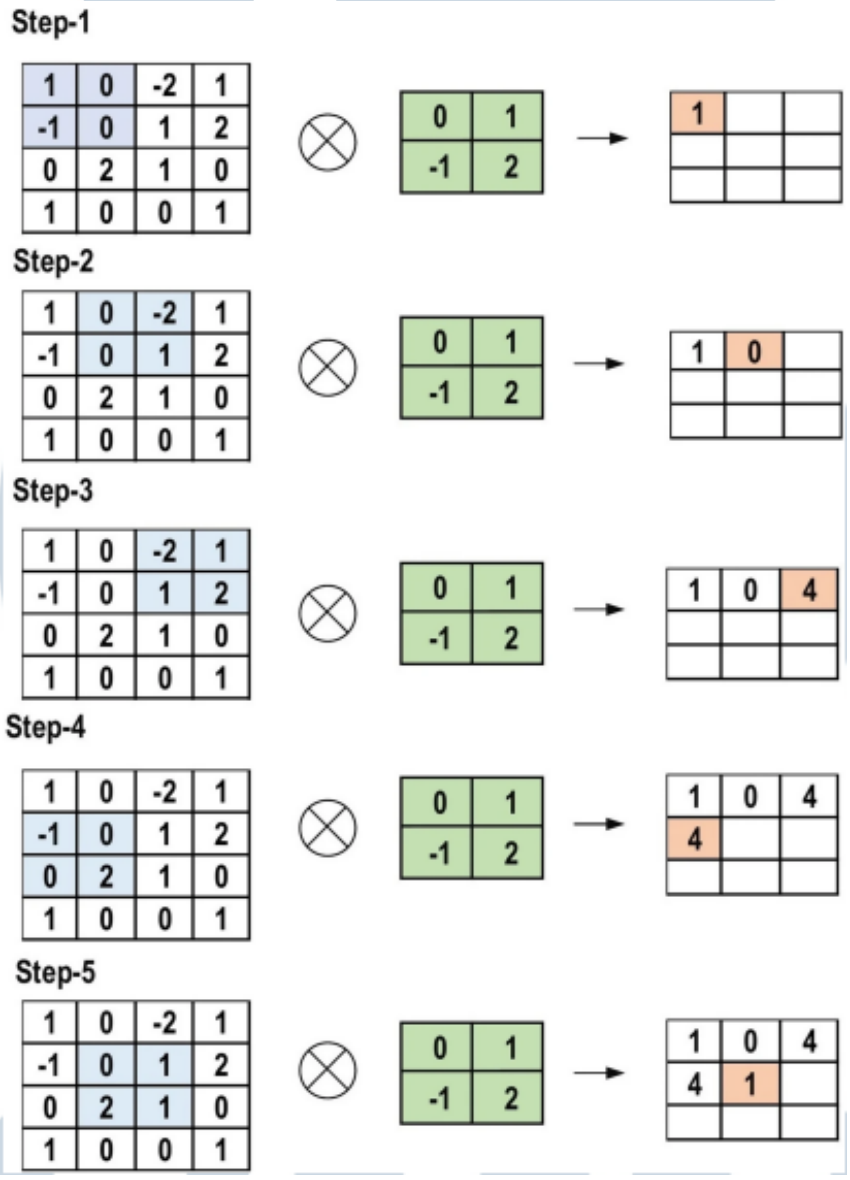
Gambar 2.1. *CNN Architecture*

Seperti yang ditunjukkan pada Gambar 2.1 , terdapat dua bagian utama dalam metode CNN, yaitu proses ekstraksi fitur dan proses klasifikasi. Ekstraksi fitur dilakukan melalui lapisan konvolusi dan *pooling*, sedangkan klasifikasi berlangsung di lapisan *fully connected*. Proses ekstraksi ini memungkinkan jaringan mengenali karakteristik spesifik dari objek, sementara bagian klasifikasi bertugas untuk mempelajari pola-pola tersebut dan menentukan label yang paling sesuai bagi setiap gambar yang diuji.

A *Convolution Layer*

Convolution Layer merupakan komponen inti CNN yang dikembangkan dengan menambahkan *depth-wise convolution*, di mana setiap saluran input diproses dengan pola konvolusi berbeda. Kombinasi konvolusi standar dan depth-wise menghasilkan over-parameter namun dapat disederhanakan menjadi satu lapisan yang disebut DO-Conv.

Eksperimen menunjukkan DO-Conv meningkatkan performa CNN secara signifikan pada tugas klasifikasi, deteksi objek, dan segmentasi. Saat inferensi, *depth-wise convolution* dapat digabungkan dengan konvolusi standar sehingga kompleksitas komputasi setara dengan lapisan konvolusi biasa. DO-Conv mampu meningkatkan performa tanpa menambah beban komputasi, sehingga layak sebagai alternatif pengganti lapisan konvolusi standar dalam CNN.



Gambar 2.2. Proses *Convolution Layer*

Dapat dilihat pada Gambar 2.2 mengenai perhitungan utama yang dilakukan pada setiap langkah. Warna hijau muda pada gambar mewakili kernel 2x2, sedangkan warna biru muda mewakili area dengan ukuran yang sama dari gambar

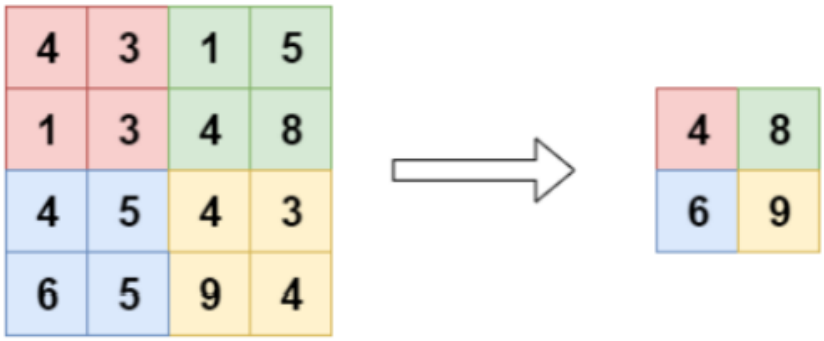
masukkan. Kemudian keduanya dikalikan lalu mengeluarkan hasil yang ditandai dengan warna oranye muda, hasil tersebut mewakili nilai entri ke *output feature map*.

B *Pooling Layer*

Pooling Layer adalah bagian dari struktur jaringan yang berfungsi menjalankan operasi statistik pada feature map yang diterima sebagai input, dengan mengacu pada nilai-nilai piksel di sekitarnya. Dalam arsitektur *Convolutional Neural Network (CNN)*, lapisan ini biasanya disisipkan secara berkala setelah beberapa lapisan konvolusi.

Penempatan *Pooling Layer* di antara lapisan-lapisan konvolusi secara bertahap dapat mengurangi ukuran dimensi feature map, sehingga jumlah parameter dan beban komputasi dalam jaringan juga berkurang. Selain itu, hal ini dapat membantu dalam mengatasi masalah *overfitting*.

Lapisan ini bekerja pada setiap feature map dan secara signifikan mengecilkan ukurannya. Salah satu bentuk *Pooling Layer* yang paling umum adalah penggunaan filter 2x2 dengan *stride 2*, seperti yang ditunjukkan pada Gambar 2.3 . Metode ini dapat mengurangi ukuran feature map hingga sekitar 75% dari ukuran aslinya

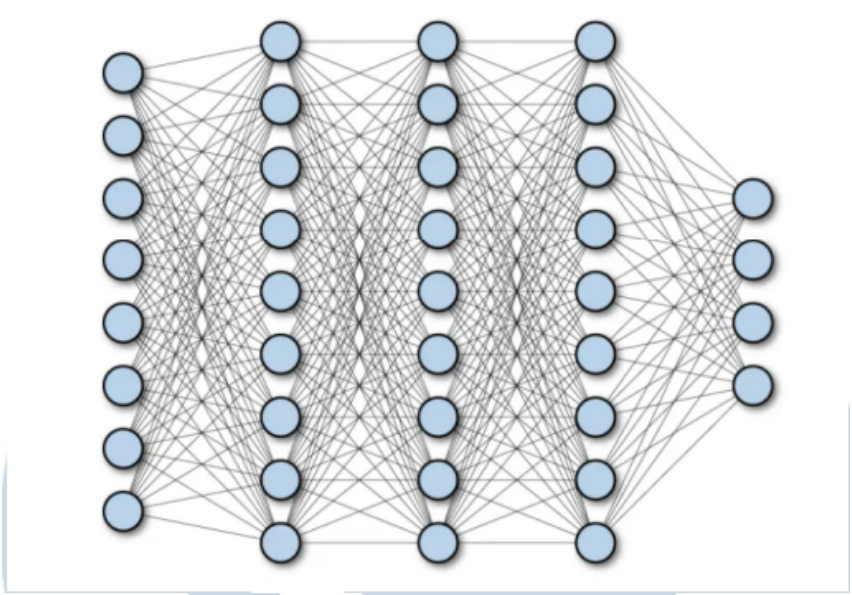


Gambar 2.3. Proses *Pooling Layer*

C *Fully Connected Layer*

Fully Connected Layer adalah lapisan yang biasa digunakan dalam *Multilayer Perceptron* dan berfungsi untuk mentransformasikan setiap dimensi data agar dapat diklasifikasikan secara linear. Seperti ditunjukkan pada Gambar 2.4 , setiap neuron dari lapisan konvolusi perlu diubah terlebih dahulu menjadi bentuk

satu dimensi sebelum diproses oleh *Fully Connected Layer*. Namun, transformasi ini menyebabkan hilangnya informasi spesifik dari data yang tidak dapat dipulihkan, sehingga *Fully Connected Layer* hanya digunakan pada bagian akhir jaringan.

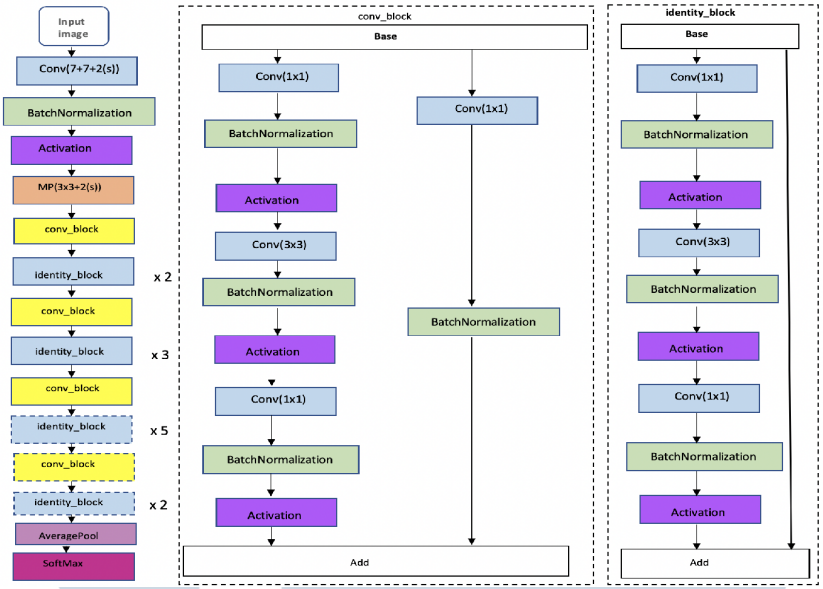


Gambar 2.4. Proses *Fully Connected Layer*

D *Residual Network-50 (ResNet-50)*

ResNet-50 merupakan salah satu arsitektur CNN yang memperkenalkan konsep shortcut connections atau residual connections. Penambahan konsep ini bertujuan untuk mengatasi permasalahan *vanishing gradient* yang sering muncul saat jaringan dibuat semakin dalam. Meningkatkan kedalaman jaringan tidak cukup hanya dengan menambahkan lebih banyak lapisan, karena semakin dalam jaringan dibangun, semakin besar pula kemungkinan munculnya vanishing gradient, yang menyebabkan nilai gradien menjadi sangat kecil dan berdampak pada penurunan akurasi atau performa model.

Shortcut connections dalam ResNet-50 memungkinkan informasi untuk ”melompati” beberapa lapisan dan langsung terhubung ke lapisan yang lebih dalam, sehingga gradien dapat mengalir lebih efektif selama proses backpropagation. Arsitektur ini terdiri dari 50 lapisan yang tersusun dalam beberapa blok residual, di mana setiap blok memiliki koneksi langsung antara input dan output blok tersebut, memungkinkan pelatihan jaringan yang sangat dalam dengan performa yang tetap optimal.



Gambar 2.5. Arsitektur ResNet-50

Pada gambar 2.5 menunjukkan struktur arsitektur ResNet-50 yang terdiri dari 50 lapisan, disusun dalam beberapa blok utama. Setiap blok memuat sejumlah lapisan konvolusi yang diikuti oleh *shortcut connection*. Proses dimulai dari blok awal yang mencakup beberapa lapisan konvolusi serta operasi *max pooling* untuk mengekstraksi fitur awal dari citra input. Setelah itu, terdapat empat blok inti yang masing-masing terdiri dari sejumlah residual block, yang dirancang untuk memperdalam representasi fitur secara bertahap. Dalam setiap blok, shortcut connections memungkinkan aliran informasi untuk melewati lapisan konvolusi, sehingga memperkuat penyebaran informasi dan mencegah terjadinya penurunan gradien secara drastis[11].

Setelah melewati blok utama, jaringan dilanjutkan dengan *global average pooling (GAP)* yang berfungsi mereduksi dimensi spasial dari fitur. Kemudian, hasilnya diteruskan ke *fully connected layer* untuk proses klasifikasi. Di akhir jaringan, fungsi *softmax* digunakan untuk menghasilkan distribusi probabilitas dari masing-masing kelas. Selain itu, setiap lapisan konvolusi dalam ResNet-50 menggunakan fungsi aktivasi *Rectified Linear Unit (ReLU)* untuk menghilangkan nilai negatif.

E *EfficientNet*

EfficientNet adalah arsitektur model CNN yang telah dilatih sebelumnya (pretrained) dan dirancang khusus untuk tugas klasifikasi gambar, yang merupakan hasil pengembangan tim riset Google AI. Arsitektur ini memiliki berbagai varian model yang diberi label dari B0 hingga B7. Beberapa model pada EfficientNet dengan resolusi dan jumlah parameter dapat dilihat pada Tabel 2.1.

Model	**Resolusi**	**Parameter (juta)**
EfficientNetB0	224	5.3
EfficientNetB1	240	7.8
EfficientNetB2	260	9.2
EfficientNetB3	300	12
EfficientNetB4	380	19
EfficientNetB5	456	30
EfficientNetB6	528	43
EfficientNetB7	600	66

Tabel 2.1. Model resolusi dan parameter EfficientNet

Untuk meningkatkan performa akurasi model CNN, terdapat beberapa pendekatan konvensional yang sering diterapkan, seperti menambah kedalaman jaringan melalui penambahan layer, atau memperluas kapasitas model dengan menambah jumlah feature maps. Pendekatan alternatif lainnya adalah dengan menggunakan gambar input beresolusi lebih tinggi. Namun, EfficientNet mengadopsi strategi yang berbeda dengan menerapkan penskalaan terpadu yang mengatur width, depth, dan resolution secara bersamaan menggunakan compound coefficient, yang dikenal sebagai compound scaling method. Pendekatan ini didasarkan pada prinsip bahwa gambar beresolusi tinggi membutuhkan peningkatan proporsional pada width dan depth jaringan untuk mengekstraksi fitur-fitur penting secara efektif dari jumlah piksel yang lebih besar[12].

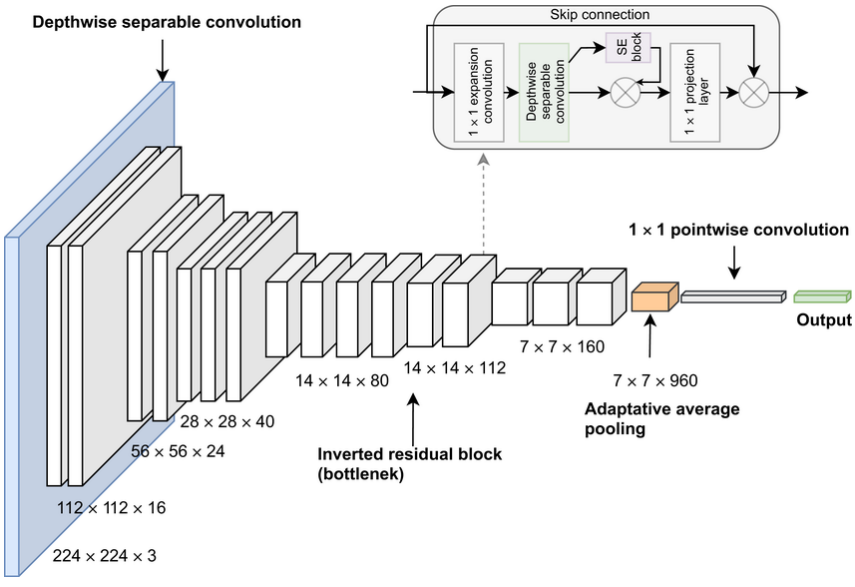
EfficientNetB0 dibangun menggunakan blok fundamental berupa mobile inverted bottleneck convolution (MBConv) yang telah dioptimalkan dengan mekanisme squeeze-and-excitation. EfficientNetB0 arsitektur dapat dilihat pada Tabel 2.2.

Stage	Operator	Resolution	Channels	Layers
1	Conv3x3	224 x 224	32	1
2	MBConv1, k3x3	112 x 112	16	1
3	MBConv6, k3x3	112 x 112	24	2
4	MBConv6, k5x5	56 x 224	40	2
5	MBConv6, k3x3	28 x 28	80	3
6	MBConv6, k5x5	14 x 14	112	3
7	MBConv6, k5x5	14 x 14	192	4
8	MBConv6, k3x3	7 x 7	320	1
9	Conv1x1 & Pooling & Fully-Connected	224 x 224	1280	1

Tabel 2.2. Arsitektur EfficientNetB0

F *MobileNetV3*

MobileNetV3 adalah evolusi dari arsitektur jaringan *lightweight* yang dikembangkan berdasarkan MobileNetV1 dan MobileNetV2, yang mengintegrasikan *inverted residual structure* dan *depthwise separable convolution*. Untuk mengoptimalkan efisiensi komputasi dan performa model, arsitektur ini mengimplementasikan *neural architecture search (NAS)* yang memungkinkan optimasi otomatis terhadap struktur dan parameter jaringan.

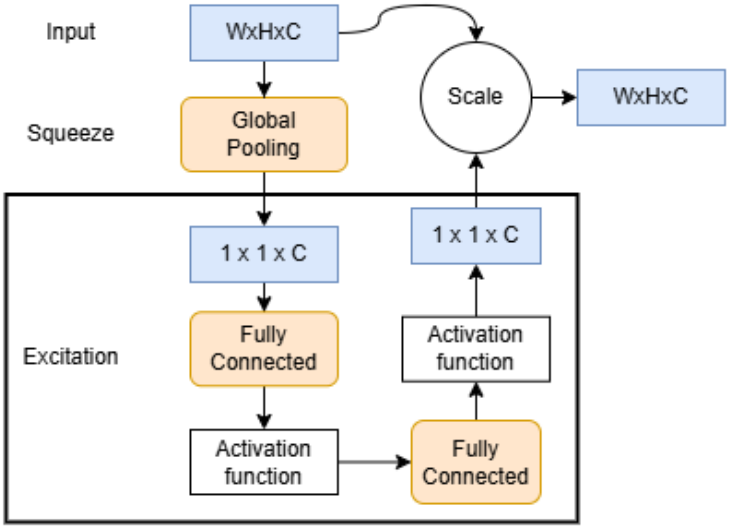


Gambar 2.6. Arsitektur ResNet-50

Berdasarkan penelitian (Nugroho & Baihaqi, 2023), MobileNetV3 menggunakan teknologi Neural Architecture Search (NAS) dalam proses optimasi strukturnya. Selama proses optimasi, beberapa layer yang memiliki beban

komputasi tinggi dimodifikasi untuk efisiensi. Salah satu modifikasi signifikan adalah pengurangan jumlah filter pada layer konvolusi pertama dari 32 menjadi 16, namun perubahan ini tidak menurunkan akurasi model secara keseluruhan.

MobileNetV3 memiliki reputasi sebagai arsitektur yang sangat efisien dengan performa tinggi, menjadikannya ideal untuk implementasi dalam sistem klasifikasi penyakit pada tanaman jagung. Integrasi dengan teknik augmentasi data menjadi krusial untuk memperkuat robustness model dan meningkatkan kemampuan generalisasi melalui penambahan variasi yang disengaja pada dataset training. Fitur penting lainnya adalah implementasi modul Squeeze-and-Excitation (SE), yang berfungsi memberikan pembobotan adaptif pada setiap channel selama proses training, memungkinkan model untuk memprioritaskan fitur-fitur penting dan menekan fitur yang kurang relevan[13].

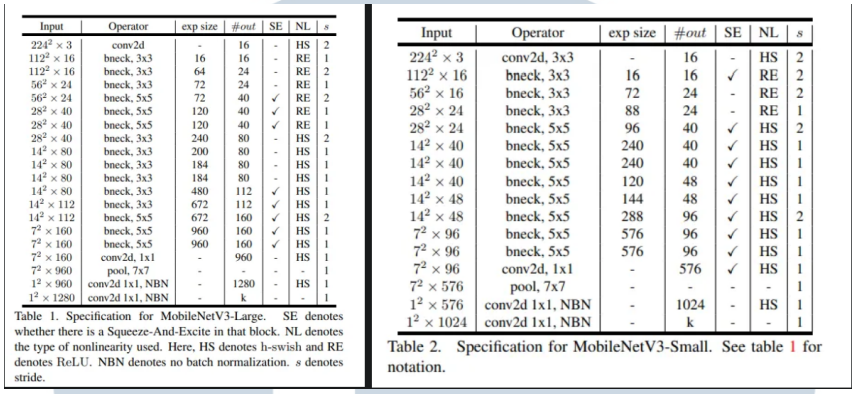


Gambar 2.7. Alur Kerja Modul SE

Berdasarkan studi (Jiang & Tong, 2022), implementasi modul SE memang meningkatkan jumlah parameter dan kompleksitas komputasi jaringan. Walaupun beban komputasi dari fully connected layers dalam modul SE relatif lebih rendah dibandingkan layer konvolusi, penambahan parameter tetap cukup substansial.

Implementasi MobileNetV3 sendiri, terdapat dua jenis yaitu MobileNetV3-Large dan MobileNetV3-Small, yang masing-masing dioptimalkan untuk skenario penggunaan dengan ketersediaan sumber daya komputasi yang berbeda. Varian Large dirancang untuk aplikasi yang memerlukan akurasi tinggi dengan sumber

daya memadai, sementara varian Small ditujukan untuk *deployment* pada perangkat dengan keterbatasan komputasi.



Input	Operator	exp size	#out	SE	NL	s
$224^2 \times 3$	conv2d	-	16	-	HS	2
$112^2 \times 16$	bneck, 3x3	16	16	-	RE	1
$112^2 \times 16$	bneck, 3x3	64	24	-	RE	2
$56^2 \times 24$	bneck, 3x3	72	24	-	RE	1
$56^2 \times 24$	bneck, 5x5	72	40	✓	RE	2
$28^2 \times 40$	bneck, 5x5	120	40	✓	RE	1
$28^2 \times 40$	bneck, 5x5	120	40	✓	RE	1
$28^2 \times 40$	bneck, 3x3	240	80	-	HS	2
$14^2 \times 80$	bneck, 3x3	200	80	-	HS	1
$14^2 \times 80$	bneck, 3x3	184	80	-	HS	1
$14^2 \times 80$	bneck, 3x3	184	80	-	HS	1
$14^2 \times 80$	bneck, 3x3	480	112	✓	HS	1
$14^2 \times 112$	bneck, 3x3	672	112	✓	HS	1
$14^2 \times 112$	bneck, 5x5	672	160	✓	HS	2
$7^2 \times 160$	bneck, 5x5	960	160	✓	HS	1
$7^2 \times 160$	bneck, 5x5	960	160	✓	HS	1
$7^2 \times 160$	conv2d, 1x1	-	960	-	HS	1
$7^2 \times 960$	pool, 7x7	-	-	-	-	1
$1^2 \times 960$	conv2d 1x1, NBN	-	1280	-	HS	1
$1^2 \times 1280$	conv2d 1x1, NBN	-	k	-	-	1

Table 1. Specification for MobileNetV3-Large. SE denotes whether there is a Squeeze-And-Excite in that block. NL denotes the type of nonlinearity used. Here, HS denotes h-swish and RE denotes ReLU. NBN denotes no batch normalization. s denotes stride.

Input	Operator	exp size	#out	SE	NL	s
$224^2 \times 3$	conv2d, 3x3	-	16	-	HS	2
$112^2 \times 16$	bneck, 3x3	16	16	✓	RE	2
$56^2 \times 16$	bneck, 3x3	72	24	-	RE	2
$28^2 \times 24$	bneck, 3x3	88	24	-	RE	1
$28^2 \times 24$	bneck, 5x5	96	40	✓	HS	2
$14^2 \times 40$	bneck, 5x5	240	40	✓	HS	1
$14^2 \times 40$	bneck, 5x5	240	40	✓	HS	1
$14^2 \times 40$	bneck, 5x5	120	48	✓	HS	1
$14^2 \times 48$	bneck, 5x5	144	48	✓	HS	1
$14^2 \times 48$	bneck, 5x5	288	96	✓	HS	2
$7^2 \times 96$	bneck, 5x5	576	96	✓	HS	1
$7^2 \times 96$	bneck, 5x5	576	96	✓	HS	1
$7^2 \times 96$	conv2d, 1x1	-	576	✓	HS	1
$7^2 \times 576$	pool, 7x7	-	-	-	-	1
$1^2 \times 576$	conv2d 1x1, NBN	-	1024	-	HS	1
$1^2 \times 1024$	conv2d 1x1, NBN	-	k	-	-	1

Table 2. Specification for MobileNetV3-Small. See table 1 for notation.

Gambar 2.8. Arsitektur (a).MobileNetV3-Large, (b).MobileNetV3-Small

Perbedaan fundamental antara kedua varian terletak pada kompleksitas arsitektur, di mana MobileNetV3-Large memiliki jumlah layer yang lebih banyak dan struktur yang lebih kompleks, konsekuensinya membutuhkan sumber daya komputasi yang lebih besar dibandingkan dengan MobileNetV3-Small yang lebih ringan dan efisien.

G *Confusion Matrix*

Confusion Matrix adalah alat penting dalam klasifikasi yang diawasi dalam pembelajaran mesin untuk mengevaluasi kinerja model klasifikasi. Matriks ini berbentuk tabel persegi dengan baris yang mewakili kelas sebenarnya dan kolom yang menunjukkan kelas yang diprediksi. Dalam klasifikasi biner, Confusion Matrix berukuran 2x2, sedangkan dalam klasifikasi multi-kelas, ukurannya menjadi k x k sesuai dengan jumlah kelas. Matriks ini banyak digunakan untuk mengukur efektivitas model klasifikasi terhadap dataset tertentu. Dalam rekayasa perangkat lunak, Font dkk. menerapkan Confusion Matrix untuk membandingkan nilai prediksi dengan nilai aktual dari elemen model. Empat elemen utama dalam matriks ini adalah *true positive (TP), false positive (FP), true negative (TN), dan false negative (FN)*, yang menjadi dasar perhitungan berbagai metrik performa seperti *precision, recall*, dan *F-score*.

1. Precision adalah ukuran keakuratan asalkan kelas tertentu telah diprediksi

dapat dilihat pada persamaan 2.1.

$$Precision = \frac{TP}{TP + FP} \tag{2.1}$$

2. Recall adalah ukuran kemampuan model prediksi untuk memilih instance kelas tertentu dari kumpulan data dapat dilihat pada Persamaan 2.2.

$$Recall = \frac{TP}{TP + FN} \tag{2.2}$$

3. F1-Scoreadalah rata-rata harmonik antara precision dan recall dapat dilihat pada Persamaan 2.3.

$$F1Score = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{2.3}$$

2.1.7 *mAP(mean Average Precision)*

mAP (mean Average Precision) adalah metrik yang digunakan untuk mengukur akurasi detektor objek di seluruh kelas dalam suatu basis data tertentu. mAP dihitung dengan cara mengambil

rata-rata dari nilai Average Precision (AP) yang didapatkan untuk setiap kelas individual.Average Precision(AP)mengukur area dibawah kurva presisi-recall untuk suatu kelas tertentu, dan dapat dilihat pada Persamaan 2.4.

$$AP_{\text{all}} = \sum_{n} (R_{n+1} - R_n) \cdot P_{\text{interp}}(R_{n+1}) \tag{2.4}$$

Dimana n adalah indeks iterasi saat ini, Rn+1 adalah nilai recall iterasi berikutnya,Rn adalah nilai recall iterasi saat ini, dan Pinterp(Rn+1) adalah presisi interpolasi yang dapat dilihat pada Persamaan 2.5.

$$P_{\text{interp}}(R_{n+1}) = \max_{\tilde{R} \geq R_{n+1}} P(\tilde{R}) \tag{2.5}$$

Dimana R adalah nilai dari recall,R lebih besar atau sama dengan Rn+1 adalah kondisi nilai recall lebih besar atau sama dengan nilai presisi saat ini,dan

P(R) adalah nilai presisi untuk level recall R tertentu. Jadi,mean Average Precision (mAP) didapat dari rata-rata AP pada semua kelas yang dapat dilihat pada Persamaan 2.6.

$$\text{mAP} = \frac{1}{N} \sum_{i=1}^{N} \text{AP}_i \tag{2.6}$$

Di mana APi adalah nilai AP untuk kelas ke-i, dan NNN adalah jumlah total kelas dalam dataset. Dalam konteks CNN yang digunakan untuk klasifikasi gambar, mAP@0.9 dan mAP@0.9:0.99 sering digunakan untuk menilai performa model. mAP@0.9 mengacu pada mAP yang dihitung dengan ambang batas probabilitas klasifikasi sebesar 0.9, yang berarti prediksi dianggap benar jika confidence score model mencapai atau melebihi 0.9. Sementara itu, mAP@0.9:0.99 mengevaluasi mAP dalam rentang confidence score dari 0.9 hingga 0.99 dengan kenaikan 0.01. Dengan menggunakan mAP sebagai metrik evaluasi, CNN dapat dinilai berdasarkan kemampuannya dalam mengklasifikasikan objek dengan akurasi tinggi. Pemilihan ambang batas yang tepat sangat penting agar model tidak hanya mengutamakan presisi tetapi juga mampu menangkap berbagai variasi data dengan recall yang optimal.

2.1.8 *Evaluasi Model*

Evaluasi dapat dilakukan dengan memperhatikan *Losses, Precision, Recall, and F1-score*, dan *mAP (mean Average Precision)* yang dihasilkan dari proses pelatihan menggunakan CNN. Evaluasi dapat dilakukan melalui matriks gambaran dan statistik yang dihasilkan dari proses pelatihan menggunakan CNN. Matriks dan statistik yang diperoleh meliputi:

1. ***Losses***: Menunjukkan kerugian selama pelatihan membantu dalam memahami seberapa baik model belajar. Penurunan kerugian menunjukkan peningkatan kemampuan model dalam mendeteksi dan mengklasifikasikan objek dengan akurat.
2. ***Confusion Matrix***: Digunakan untuk mengukur akurasi model dengan membandingkan nilai yang diprediksi dengan nilai yang sebenarnya.
3. ***Precision, Recall,* and *F1-score***: Metrik-metrik ini memberikan pemahaman yang lebih detail tentang kinerja model Anda pada setiap kelas. Presisi yang tinggi menunjukkan lebih sedikit positif palsu, recall yang tinggi

menunjukkan lebih sedikit negatif palsu, dan nilai F1 yang tinggi menandakan keseimbangan yang baik antara presisi dan recall.

4. ***mAP (mean Average Precision)***: mAP adalah metrik yang banyak digunakan untuk tugas deteksi objek. Ini memberikan ukuran keseluruhan dari kinerja model Anda di semua kelas dan sangat berguna untuk membandingkan model-model berbeda atau melacak kinerja dari waktu ke waktu.