BAB I
PENDAHULUAN

1.1 Latar Belakang

Basal Stem Rot (BSR) atau dalam bahasa Indonesia merupakan penyakit busuk pangkal batang adalah penyakit yang menyerang tanaman sawit yang disebabkan oleh jamur *Ganoderma*. Terdapat beberapa spesies *Ganoderma*, namun yang paling umum dijumpai adalah spesies *Ganoderma boninense (G. boninense)*. *G.boninense* menyebar pada tanaman kelapa sawit lain melalui kontak pada akar, tunas, batang, dan jaringan batang yang membusuk sehingga menjadikan jamur ini sebagai patogen tular tanah. Jamur ini memiliki sifat nekrotofik sehingga sulit untuk dikenali dan dideteksi ketika infeksi awal (Haw, Hum, et al., 2023).

Gambar 1.1 Jamur Ganoderma

Pada saat infeksi awal, jamur mulai untuk menginfeksi dan merusak dinding sel pada tanaman yang diinfeksinya. Meskipun demikian, pohon belum menunjukkan gejala secara visual (Haw, Hum, et al., 2023). Namun ketika infeksi berlanjut, gejala pada pohon akan mulai terlihat pada batang, yaitu munculnya jamur Ganoderma dan pada daun seperti daun nekrotik, daun yang tidak terbuka, mahkota pohon yang kecil, dan mahkota pohon yang melengkung kebawah seperti rok (Kurihara et al., 2022).

Penyebaran penyakit ini juga sangat cepat, dan belum ada pengobatan dan perawatan yang efektif untuk menangani pohon yang terkena penyakit ini. Sekali terinfeksi, maka jamur ini hanya membutuhkan 2-3 untuk membunuuh tanaman kelapa sawit dewasa. Selain itu, telah diprediksi bahwa ketika umur tanaman kelapa sawit telah mencapai setengah dari rata-rata umur ekonomisnya, maka BSR dapat menyebabkan kematian 4 dari 5 pohon jika tidak segera dilakukan penangangan. Di Indonesia, penyakit ini dapat mengakibatkan risiko kematian pada pohon kelapa sawit yang berumur lebih dari 15 tahun diatas 80% (Haw, Lai, et al., 2023; Kurihara et al., 2022).

Gambar 1.2 Pohon Kelapa Sawit mati akibat BSR

Akibat dari infeksi jamur ini dapat berdampak sangat serius pada perekonomian kelapa sawit. Hal ini dikarenakan ketika pohon terinfeksi, terjadi penurunan berat pada buah sehingga diperkirakan akan mengalami penurunan hasil panen sebesar 50 hingga 80% (Haw, Hum, et al., 2023). Oleh karena itu, sangat penting untuk dapat menangani penyebaran dari penyakit ini dengan cara melakukan deteksi secara dini penyakit BSR pada tanaman kelapa sawit.

Terdapat beberapa metode yang digunakan untuk mendeteksi penyakit ini, seperti metode berbasis fisik (manual), berbasis lab, biokontrol, dan berbasis penginderaan jarak jauh (*remote sensing*), yaitu penggunaan sensor hyperspectral, sensor multispectral, tomografi, radar, *electric nose*, *Terrestrial Laser Scanning* (TLS), dan properti listrik yang dipadukan juga dengan beberapa metode kecerdasan buatan seperti *Machine Learning* dan *Deep Learning* (Haw, Lai, et al., 2023; Tee et al., 2021). Beberapa metode berhasil memberikan performa yang baik dalam mendeteksi penyakit BSR pada tanaman kelapa sawit, namun kurang efektif jika diaplikasikan pada lahan kelapa sawit yang luas karena membutuhkan banyak waktu, tenaga ahli, dan biaya serta faktor keselamatan dan kesehatan lingkungan sekitar.

Unmanned Aerial Vehicle (UAV) merupakan pesawat kecil yang dikontrol oleh mikrokontroller yang terpasang atau operator yang berada di darat. Pada saat ini, penggunaan UAV terjadi peningkatan dan menjadi platform akusisi baru yang efektif pada teknologi *remote sensing*. *Unmanned Aerial Vehicle* (UAV) banyak diaplikasikan pada bidang pengawasan, pelacakan, penanganan bencana, teknologi parkir cerdas, sistem transportasi cerdas, militer hingga agrikultur. Terdapat beberapa kelebihan dalam penggunaan *Unmanned Aerial Vehicle (*UAV)seperti biaya yang rendah, mobilitas tinggi, fleksibel, aman digunakan, dapat dikustomisasi dan dapat melakukan analisis gambar atau video secara *real time*. Meskipun citra satelit mampu memberikan pandangan yang luas tetapi UAV dapat melakukan observasi dan monitoring lebih mudah karena dapat menjangkau area yang diinginkan dengan cepat dan gambar yang dihasilkan lebih terbaru dibandingkan dengan menggunakan citra satelit. Dengan kemampuan tersebut, penggunaan

metode berbasis UAV dapat diterapkan sebagai pengganti yang praktis untuk dapat mendeteksi penyakit BSR pada lahan kelapa sawit terutama pada lahan yang berskala luas (Ahmadi et al., 2022; Ammar et al., 2019; Fan et al., 2018).

Untuk mendukung kemampuan UAV dalam mendeteksi penyakit BSR, diperlukan metode deteksi objek berbasis *real time* untuk mendeteksi pohon yang terkena penyakit BSR. Deteksi objek merupakan salah satu dari kemampuan unik dari computer vision yang mampu mendeteksi objek dalam gambar/video dengan mengandalkan kemampuan dari *Machine Learning* dan *Deep Learning*. Deteksi objek banyak diaplikasikan di berbagai bidang seperti monitoring keamanan, *autonomous driving*, pengawasan lalu lintas, analisis pemandangan drone, dan penglihatan robot (Joiya, 2022; Reswara et al., 2023).

Saat ini, perkembangan dari *Deep Learning* telah menjadi peran penting dalam pengembangan dari deteksi objek. *Convolutional Neural Network* (CNN) merupakan teknologi *Deep Learning* yang umum digunakan untuk melakukan deteksi cepat dan akurat dengan mengaplikasikan banyak lapisan konvolusi dan komputasi konvolusi. Terdapat dua pendekatan dalam algoritma deteksi objek, yaitu *one-step detection algorithm* seperti *You Only Look Once* (YOLO) (Redmon et al., 2016) dan *two-step detection algorithm* seperti *Faster Region Convolutional Neural Network* (Faster R-CNN) (Ren et al., 2017). Kedua pendekatan ini melakukan proses yang sama dalam mendeteksi objek, yaitu melakukan lokalisasi dan klasifikasi objek dengan membuat sebuah *bounding box* pada objek yang terdeteksi (Joiya, 2022).

YOLO dan Faster R-CNN menggunakan CNN sebagai arsitektur utama dalam proses deteksinya. Meskipun demikian, kedua algoritma ini memiliki kekurangan dan kelebihannya karena perbedaan pendekatan dalam melakukan proses deteksi (Joiya, 2022; Redmon et al., 2016; Ren et al., 2017).

Program Grant Riset Sawit merupakan program penelitian dan pengembangan Perkebunan kelapa sawit dari aspek hulu hingga hilir yang dikembangkan oleh BPDPKS sebagai upaya untuk melakukan penguatan, pengembangan, dan peningkatan pemberdayaan perkebunan dan industri kelapa sawit nasional (Grant Riset Sawit BPDPKS, n.d.). Pada tahun ini, Universitas Hasanuddin bekerja sama dengan Politeknik ATI Makassar mengikuti projek Grant Riset Sawit untuk mengembangkan prototipe drone yang mampu mendeteksi penyakit BSR pada tanaman kelapa sawit untuk mitigasi penyebaran skala besar.

Oleh karena itu, sebagai bagian dari projek tersebut, peneliti tertarik untuk melakukan penelitian untuk membandingkan performa dari Faster R-CNN dan YOLO dalam melakukan deteksi penyakit BSR pada tanaman kelapa sawit berbasis UAV.

1.2 Rumusan Masalah

Berdasarkan pada uraian latar belakang masalah , maka peneliti tertarik dan ingin melakukan penelitian tentang perbandingan faster r-cnn dan yolo dalam deteksi penyakit bsr pada tanaman kelapa sawit berbasis uav dengan rumusan masalah penelitian yaitu:

1. Bagaimana membuat model menggunakan algoritma Faster R-CNN dan YOLO dalam mendeteksi penyakit BSR pada tanaman kelapa sawit?

2. Bagaimana perbandingan performa Faster R-CNN dan YOLO dalam mendeteksi penyakit BSR pada tanaman kelapa sawit?

1.3 Tujuan, Manfaat dan Ruang Lingkup Penelitian

1.3.1 Tujuan

1. Membuat model menggunakan Faster R-CNN dan YOLO yang dapat mendeteksi penyakit BSR pada tanaman kelapa sawit.
2. Menunjukkan perbandingan performa Faster R-CNN dan YOLO dalam mendeteksi penyakit BSR pada tanaman kelapa sawit.

1.3.2 Manfaat

1. Membantu dalam menentukan metode yang optimal dalam kasus deteksi penyakit BSR pada tanaman kelapa sawit sehingga dapat dilakukan penanganan secara cepat dan tepat.
2. Sebagai rujukan untuk penelitian selanjutnya terhadap deteksi penyakit pada kelapa sawit, terkhusus pada tanaman kelapa sawit.
3. Sebagai media informasi mengenai penggunaan framework Faster R-CNN dan YOLO dalam deteksi penyakit pada tanaman.

1.3.3 Ruang Lingkup

1. Kelas objek yang akan dideteksi adalah tanaman kelapa sawit sehat dan tanaman kelapa sawit yang terinfeksi penyakit BSR.
2. Penelitian hanya berfokus pada pengembangan model Computer Vision.
3. Deteksi penyakit berfokus pada melihat gejala pada daun/mahkota kelapa sawit.
4. Pengambilan data menggunakan kamera drone.

BAB II
METODE PENELITIAN

2.1 Waktu dan Lokasi Penelitian

Penelitian dilakukan mulai sejak disetujuinya proposal penelitian pada April – November 2024. Penulis melakukan penelitian ini pada 5 lokasi kelapa sawit yang berbeda pada dua desa yang berbeda yaitu, desa Padangloang dan desa Malimpung, kecamatan Patampanua, kabupaten Pinrang, Provinsi Sulawesi Selatan.

Gambar 2.1 Lokasi Penelitian 1

Gambar diatas menunjukkan lokasi penelitian pertama yaitu berada di Desa Padangloang, Kecamatan Patampanuan, Kabupaten Pinrang, Provinsi Sulawesi Selatan. Karakteristik lokasi ini yaitu tumbuh di padang datar yang merupakan lahan bekas tanam tanaman cabai. Umur tanaman kelapa sawit pada lahan ini yaitu 2 tahun atau kurang dengan ketinggian berkisar 1.5 – 2.0 m (Kafrawi et al., 2023) dengan diameter mahkota berkisar 3 – 4 m (Ismi Intara et al., 2018). Selain itu, beberapa daerah pada lahan ini juga ditanami tanaman jagung di sekitar pohon kelapa sawit.

Gambar 2.2 Lokasi Penelitian 2

Gambar diatas menunjukkan lokasi penelitian kedua yaitu berada di Desa Padangloang, Kecamatan Patampanua, Kabupaten Pinrang, Provinsi Sulawesi Selatan. Lokasi ini merupakan salah satu lahan kelapa sawit tertua di desa Padangloang dengan umur tanaman lebih dari 15 tahun. Adapun tinggi kelapa sawit pada lahan ini yaitu 15 – 18 m (Ismi Intara et al., 2018) dengan diameter mahkota lebih dari 10 m (Carolita et al., 2017).

Gambar 2.3 Lokasi Penelitian 3

Gambar diatas menunjukkan lokasi penelitian ketiga yaitu berada di Desa Malimpung, Kecamatan Patampanua, Kabupaten Pinrang, Provinsi Sulawesi Selatan. Lahan kelapa sawit ini merupakan lahan tidak terawat dengan umur tanaman yaitu berada diantara 9 hingga 12 tahun. Adapun untuk tinggi kelapa sawit pada lahan ini yaitu berkisar 7 – 12 m (Wahyu Krisdiarto & Sutiarso, 2016) dengan diameter mahkota kurang lebih 9 m (Kurihara et al., 2022).

Gambar 2.4 Lokasi Penelitian 4

Gambar diatas lokasi penelitian keempat yaitu berada di Desa Malimpung, Kecamatan Patampanua, Kabupaten Pinrang, Provinsi Sulawesi Selatan. Lahan kelapa sawit ini terdiri atas dua bagian dengan umur tanaman yang berbeda. Untuk bagian pertama, merupakan tanaman kelapa sawit yang berumur lebih dari 15 tahun dengan tinggi kelapa sawit 15 – 18 m (Ismi Intara et al., 2018) dengan diameter mahkota lebih dari 10 m (Carolita et al., 2017). Sedangkan untuk bagian kedua merupakan tanaman kelapa sawit dengan umur tanaman berada diantara 9 hingga 12 tahun dengan dengan diameter mahkota kurang lebih 9 m (Kurihara et al., 2022).

Gambar 2.5 Lokasi Penelitian 5

Gambar diatas menunjukkan lokasi penelitian kelima yaitu berada di Desa Malimpung, Kecamatan Patampanua, Kabupaten Pinrang, Provinsi Sulawesi Selatan. Lahan kelapa sawit ini merupakan lahan terbengkalai/tidak terawat dengan umur kelapa sawit berada diantara 10 hingga 15 tahun dengan tinggi kelapa sawit berada diantara 8 – 15 m (Ismi Intara et al., 2018; Wahyu Krisdiarto & Sutiarso, 2016) dengan diameter mahkota kurang lebih 9 – 11 m (Carolita et al., 2017; Kurihara et al., 2022).

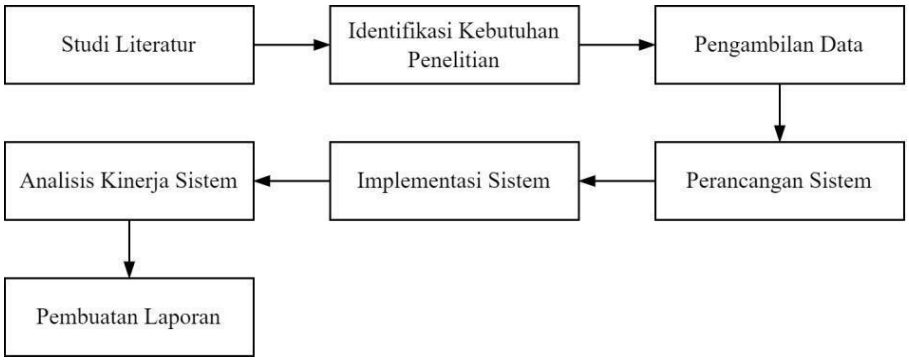
2.2 Benda Uji dan Alat

Benda maupun alat yang digunakan dalam penelitian ini meliputi perangkat lunak dan keras sebagai berikut:

1. Perangkat lunak
   a. Sistem operasi Windows 11
   b. Google Colab
   c. Python 3.10
   d. Microsoft Excel
   e. Microsoft Word
   f. Roboflow
   g. DJI GO 4
   h. Google Drive
   i. GPU T4 16 GB GPU RAM
2. Perangkat keras
   a. Laptop Legion 5 Pro (2022) RAM 16 GB
   b. DJI Mavic Air
   c. HP Huawei Nova 5T
   d. SD Card 128 GB
   e. Redmi Pad RAM 6 GB

2.3 Teknik Pengumpulan Data

Penelitian dilakukan dengan beberapa tahapan seperti yang ditunjukkan pada gambar 2.6 berikut:



Gambar 2.6 Tahapan Penelitian

Tahapan-tahapan yang dilakukan pada gambar 3.6 dapat dijelaskan sebagai berikut:

1. Studi literatur

Studi literatur merupakan langkah awal untuk mengidentifikasi teknologi yang relevan dan dapat digunakan untuk menyelesaikan latar belakang masalah. Referensi yang akan dicari melibatkan jurnal,artikel, buku dan sumber lain yang membahas terkait deteksi penyakit BSR pada kelapa sawit, penggunaan drone, dan deteksi objek.

2. Identifikasi kebutuhan penelitian

   Identifikasi kebutuhan penelitian merupakan tahap persiapan yang dilakukan sebelum memulai pengambilan data dan melakukan eksperimen penelitian. Tahap ini mencakup semua alat dan bahan yang dibutuhkan dalam penelitian, serta faktor-faktor yang akan mendukung kelancaran dalam proses pengambilan data dan ekperimen penelitian.

3. Pengambilan data

   Pengambilan data dilakukan di lokasi lahan sawit yang berada di Desa Padangloang dan Desa Malimpung Kecamatan Patampanua Kabupaten Pinrang,Sulawesi Selatan Pengambilan data dilakukan menggunakan drone yang merekam daun/mahkota tanaman kelapa sawit dari atas berdasarkan rute yang telah ditentukan dengan tujuan untuk melihat ciri-ciri daun/mahkota seperti daun nekrotik, daun yang tidak terbuka, mahkota pohon yang kecil (Kurihara et al., 2022) yang ada pada tanaman kelapa sawit. Pada penelitian ini, drone yang akan digunakan adalah DJI Mavic Air

4. Rancangan Bangun Sistem

   Perancangan sistem dilakukan untuk merancang sistem yang akan digunakan. Dalam penelitian ini, akan menggunakan metode beberapa metode preprocessing. Metode yang digunakan dalam mendeteksi objek adalah frame work YOLO dan Algoritme Faster R-CNN

5. Implementasi sistem

   Pada tahapan ini terbagi menjadi 4 tahapan utama yaitu data preparation, preprocessing, training, validation dan testing. Pada tahap ini beberapa metode pada perancangan sistem akan diimplementasikan dalam mengestimasi berat telur.

6. Analisis kinerja sistem

   Pada tahap ini penulis akan menganalisis kinerja System berdasarkan hasil metrik evaluasi precision, recall, confusion matrix, mean IoU, speed inference. Analisis ini bertujuan untuk mengetahui seberapa baik setiap sistem dapat mendeteksi penyakit BSR dengan akurat dan tepat.

7. Pembuatan laporan

   Pada tahap akhir ini, penulis menyusun dan mendokumentasikan secara sistematis seluruh proses dan temuan penelitian dalam bentuk skripsi

2.4 Teknik Pengambilan Data

Pengambilan data dilakukan di 5 lokasi lahan sawit yang berbeda pada 2 kelurahan yang berbeda, yaitu desa Padangloang dan desa Malimpung, pada kecamatan Patampanua, kabupaten Pinrang, provinsi Sulawesi Selatan.

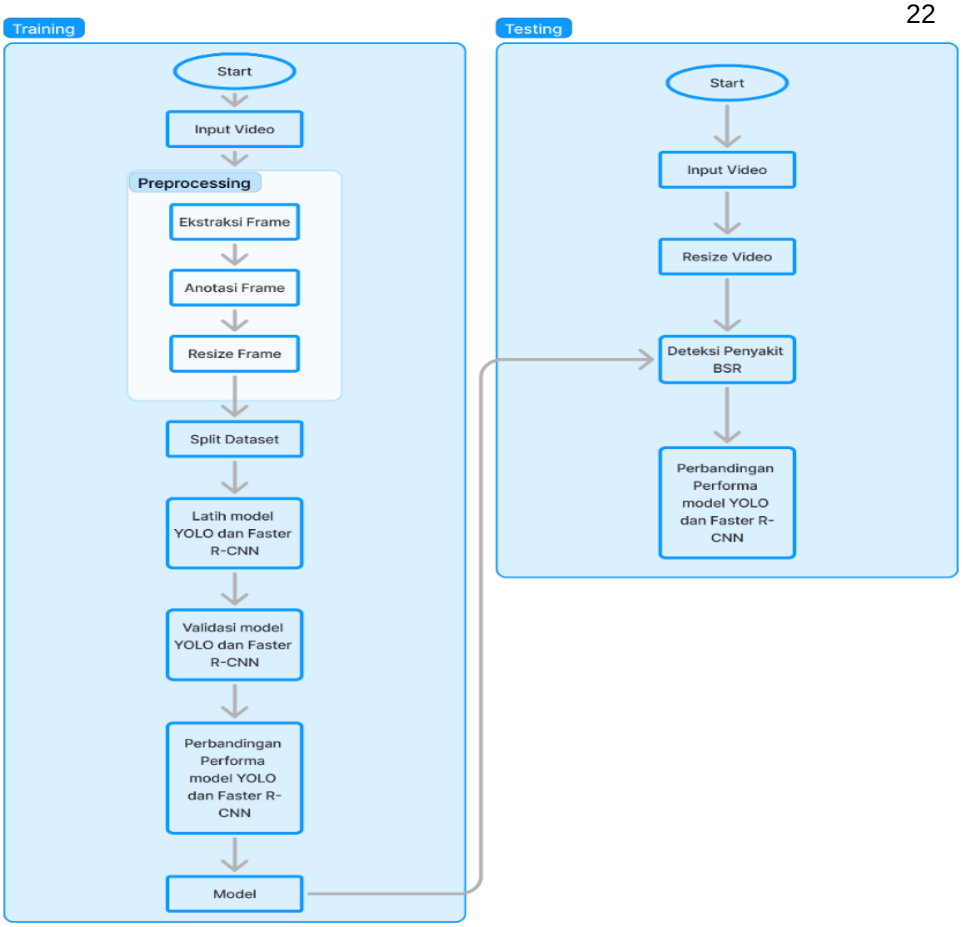
Pengambilan data dilakukan menggunakan drone yang merekam daun/mahkota tanaman kelapa sawit dari atas berdasarkan rute yang telah ditentukan dengan tujuan untuk melihat ciri-ciri yang ada pada daun/mahkota yang berkaitan dengan penyakit BSR.

Terdapat beberapa skenario pengambilan data yang berbeda pada setiap lokasi pengambilan data dikarenakan karakteristik lingkungan yang berbeda-beda pada setiap lahannya yang meliputi ketinggian terbang drone dari pucuk tanaman kelapa sawit, kecepatan terbang drone dan waktu pengambilan data. Adapun pada penelitian ini, drone yang akan digunakan adalah DJI Mavic Air.

1. Pada lokasi pertama, skenario pengambilan data dilakukan dengan waktu pengambilan data pada pagi hari, yaitu pada pukul 06.30 – 07.00 WITA. Adapun untuk ketinggian terbang drone yaitu berkisar antara 3 – 10 m dari pucuk kelapa sawit dengan kecepatan terbang drone yaitu berkisar antara 0,5 – 2,0 m/s.
2. Pada lokasi kedua, skenario pengambilan data dilakukan dengan waktu pengambilan data pada sore hari, yaitu pada pukul 16.30 – 17.00 WITA. Adapun untuk ketinggian drone yaitu berkisar antara 5 – 15 m dari pucuk kelapa sawit dengan kecepatan terbang drone yaitu berkisar antara 1,0 – 3,5 m/s.
3. Pada lokasi ketiga, skenario pengambilan data dilakukan dengan waktu pengambilan data pada pagi hari, yaitu pada pukul 07.00 – 07.30 WITA. Adapun untuk ketinggian drone yaitu berkisar antara 2 – 9 m dari pucuk kelapa sawit dengan kecepatan terbang drone yaitu 1,0 – 2,5 m/s.
4. Pada lokasi keempat, skenario pengambilan data dilakukan dengan waktu pengambilan data pada pagi, yaitu pada pukul 09.00 – 09.30 WITA. Untuk ketinggian terbang dan kecepatan drone pada lokasi ini dibedakan berdasarkan 2 bagian karena perbedaan umur tanaman pada setiap bagiannya. Untuk bagian pertama, yang merupakan daerah pohon kelapa sawit yang berumur lebih dari 15 tahun, maka ketinggian terbang drone berkisar antara 5 – 10 m dari pucuk kelapa sawit dengan kecepatan terbang drone 1,5 – 4,5 m/s. Adapun untuk bagian kedua, yang merupakan daerah pohon kelapa sawit yang berumur pada kisaran 9 – 12 tahun, ketinggian terbang drone berkisar antara 2 – 10 m dari pucuk kelapa sawit dengan kecepatan terbang drone 1,0 – 2,5 m/s.
5. Pada lokasi kelima, skenario pengambilan data dilakukan dengan waktu pengambilan data pada siang hari, yaitu pada pukul 14.30 – 15.00 WITA. Adapun untuk ketinggian terbang drone yaitu berkisar antara 2 – 10 m dari pucuk kelapa sawit dengan kecepatan terbang drone 0,5 – 2,0 m/s

2.5 Rancangan dan Implementasi Sistem

Rancangan sistem bertujuan untuk memberikan gambaran perancangan sistem yang akan dibangun dan dikembangkan, serta untuk memahami alur proses dalam sistem. Adapun rancangan sistem estimasi dan klasifikasi berat telur ditunjukkan pada gambar 2.7



Gambar 2.7 Alur Perancangan Sistem

Berdasarkan gambar 3.7 perancangan system terdiri dari dua tahapan yaitu tahapan pelatihan dan pengujian yang diuraikan sebagai berikut :

2.5.1. Input Video

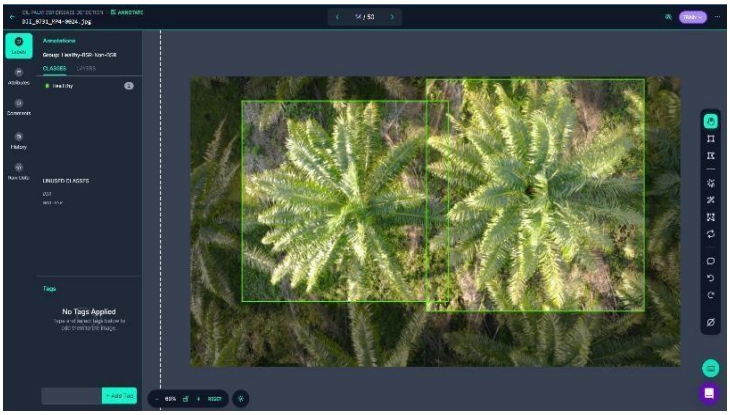
Langkah awal dalam penelitian ini adalah mempersiapkan data yang akan digunakan pada proses pelatihan dan pengujian sistem. Data ini berupa video yang diambil pada masing-masing lokasi untuk selanjutnya akan dilakukan proses ekstraksi frame pada tahap selanjutnya

2.5.2. Preprocessing

2.5.2.1. Ekstraksi Frame

Video yang telah diambil sebelumnya kemudian akan dilakukan proses ekstraksi frame. Proses ekstraksi frame ini menggunakan rasio 1:1, yaitu 1 detik untuk 1 frame. Proses ini dilakukan menggunakan platform roboflow

2.5.2.2. Anotasi Frame



Gambar 2.8 Anotasi Frame

Setelah proses ekstraksi frame, kemudian dilakukan proses anotasi frame pada data. Proses anotasi dilakukan pada platform roboflow untuk melabeli kelapa sawit dalam 3 kelas, yaitu BSR, Healthy, dan Non-BSR. Dengan pembagian ketiga kelas tersebut menggunakan ciri– ciri dari tiap kelas sebagai berikut :

a. BSR : Tombak daun 3 atau lebih, daun berwarna kuning pada ujung pelepah bagian bawah (tua) atau pada keseluruhan pelepah, daun nekrosis berwarna hij, daun mati, bentuk mahkota tidak sempurna

Bentuk mahkota tidak sempurna

Daun nekrosis pada ujung pelepah

Daun Mati Sepenuhnya

Daun berwarna coklat sebagian atau seluruhnya

Daun berwarna kuning sebagian atau seluruhnya

b. Healthy : Daun berwarna hijau tua, bentuk mahkota sempurna atau lebat

Daun lebat: mahkota sempurna Daun berwarna hijau tua

c. Non – BSR : Semua ciri-ciri kelapa sawit yang tidak memenuhi dua kelas diatas

Daun Hijau Mahkota tidak beraturan Pelepah daun rusak/patah

Dalam proses anotasi frame, tidak semua frame yang diambil, hanya frame yang memiliki objek kelapa sawit diatas 70% dari proporsi frame dan objek yang memiliki blur yang parah (hingga objek yang ada didalamnya tidak terlihat dengan jelas).

2.5.2.3. Resize Frame

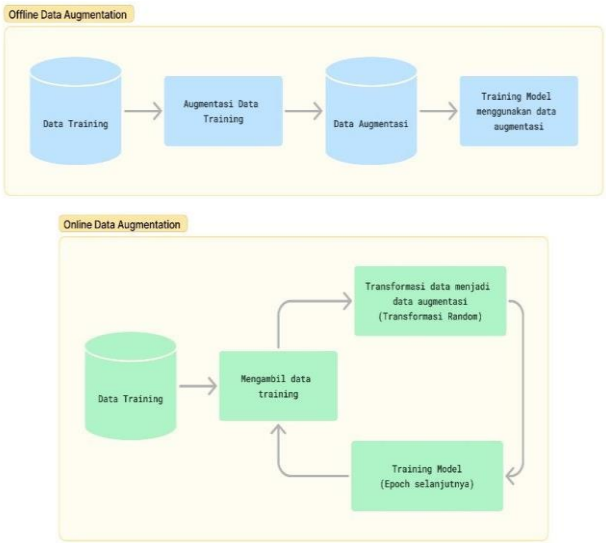
Proses setelah setiap frame dilabeli, model kemudian di-resize dari ukuran sebenarnya menjadi ukuran yang diinginkan agar memastikan bahwa semua input konsisten dan mendukung kinerja yang baik bagi model YOLO dan Faster R-CNN dalam melakukan pelatihan dan prediksi. Adapun ukuran resize frame yang digunakan yaitu 480 x 480 (480p), 640 x 640 (640p), dan 720 x 720 (720p).

Pemilihan ketiga ukuran frame ini mengacu pada skenario pelatihan yang akan dibahas pada bagian selanjutnya

2.5.3. Split Dataset

Pembagian dataset disini bertujuan untuk membagi dataset yang dilakukan untuk pelatihan (train) dan untuk memvalidasi kemampuan dari model (valid) dalam melakukan deteksi penyakit BSR pada kelapa sawit. Pembagian ini menggunakan rasio 80:20 dengan tujuan agar kemampuan model pada data latih dapat divalidasi pada data yang beragam

2.5.4. Augmentasi Data



Gambar 2.9 Offline Data Augmentation vs Online Data Augmentation

Pada penelitian ini, metode augmentasi yang digunakan adalah Online Data Augmentation. Metode ini merupakan teknik augmentasi yang dilakukan secara real time selama proses pelatihan model. Online Data Augmentation bekerja dengan cara melakukan transformasi data secara dinamis pada setiap batch atau iterasi training. Proses ini dilakukan secara on-the-fly yang dimana data mentah ditransformasi sebelum dijadikan sebagai input pelatihan model. Metode ini berbeda dengan metode data augmentasi konvensional yang melakukan augmentasi data sebelum proses pelatihan model dimulai. Hal ini memberikan keuntungan pada Online Data Augmentation dimana proses ini menghemat penggunaan memori karena sistem tidak perlu untuk menyimpan hasil augmentasi data secara permanen kedalam penyimpanan yang dimana ini akan sangat menguntungkan jika jumlah dataset cukup besar. Selain itu,metode ini memungkinkan variasi data yang lebih dinamis sehingga mampu meningkatkan generalisasi dan mencegah overfitting karena variasi data yang lebih beragam digunakan selama proses pelatihan (Z. Tang et al., 2020). Adapun metode transformasi yang dilakukan pada penelitian ini adalah:

a. Median Blur

Median blur merupakan transformasi citra yang menggunakan teknik filter non-linear yang bertujuan meningkatkan ketahanan model dalam melakukan deteksi pada citra yang mengalami blur dan memungkinkan model terlatih dalam mendeteksi fitur penting pada citra yang mengalami degradasi visual (T. S. Huang et al., 1979). Adapun proses dari median blur ini adalah:

a. Pemilihan ukuran kernel $k\ x\ k$ dimana k adalah bilangan ganjil (k = 3, 5, 7, …)
b. Ekstrak semua nilai intensitas piksel dalam kernel $k\ x\ k$ di sekitar piksel $P(x, y)$.
c. Urutkan nilai-nilai intensitas piksel secara menaik (Ascending).
d. Gunakan nilai piksel $P(x, y)$ dengan nilai median dari piksel tersebut

Secara matematis, rumus dari median blur adalah

$$P'(x, y) = Median(\{I(i, j) \mid i, j \in kernel\})$$

yang dimana $P'(x, y)$ adalah nilai piksel baru dan $I(i, j)$ adalah intensitas piksel pada jendela kernel

b. Contrast Limited Adaptive Histogram Equalization (CLAHE)

Contrast Limited Adaptive Histogram Equalization atau lebih umum disingkat sebagai CLAHE merupakan metode perbaikan kontras adaptif yang membagi citra menjadi blok- blok kecil dan melakukan proses ekualisasi histogram secara lokal. Metode ini mampu mengatasi kelemahan dari metode global histogram equalization dengan membatasi kontras pada setiap blok untuk mencegah terjadinya over-enhancement. Penggunaan CLAHE dalam teknik augmentasi data bertujuan untuk meningkatkan kemampuan model dalam melakukan deteksi pada objek pada berbagai kondisi pencahayaan terutama ketika citra memiliki pencahayaan tidak merata (Pizer et al., 1987). Adapun untuk proses dari CLAHE adalah:

dalam teknik augmentasi data bertujuan untuk meningkatkan kemampuan model dalam melakukan deteksi pada objek pada berbagai kondisi pencahayaan terutama ketika citra memiliki pencahayaan tidak merata (Pizer et al., 1987). Adapun untuk proses dari CLAHE adalah:

1. Membagi citra $I(x,\ y)$ ke dalam beberapa blok kecil atau tiles dengan ukuran $M\ x\ M$.
2. Untuk setiap blok Tij, hitung histogram Hk, dimana k merupakan nilai intensitas piksel. Adapun untuk rumusnya adalah:

$$Hk = \sum_{x,y \in Tij} \delta(I(x, y) - k)$$

dimana δ adalah fungsi delta Kronecker

3. Membatasi tinggi maksimum histogram dengan nilai ambang batas ClipLimit. Histogram Hk yang melebihi batas ini akan diratakan kembali:

$$H'k = \min(Hk, ClipLimit)$$

Setelah proses ini, piksel yang diratakan akan didistribusikan secara merata ke seluruh tingkat intensitas.

4. Equalisasi histogram diterapkan pada histogram yang telah dibatasi:

$$I'(x, y) = CDF(I(x, y)) \cdot (L - 1)$$

Dimana CDF merupakan fungsi distribusi kumulatif (Cumulative Distribution Function) dan L adalah jumlah tingkat intensitas

5. Hasil dari setiap blok diinterpolasi untuk menghasilkan transisi halus antara blok-blok yang berdekatan, menghasilkan citra akhir $I'(x, y)$.

c. Random Brightness Contrast

Metode transformasi ini secara acak mengubah kecerahan dan kontras pada citra untuk meningkatkan variasi pada dataset. Teknik ini berguna dalam metode augmentasi data untuk melatih model agar lebih tahan terhadap perubahan pencahayaan atau kontras di dunia nyata (Ma et al., 2023). Proses ini dirumuskan sebagai berikut:

1. Mengubah tingkat kecerahan yang ada pada citra yang dirumuskan sebagai berikut:

$$I'(x, y) = I(x, y) + \beta$$

$I(x, y)$ = nilai intensitas piksel asli pada kordinat (x, y).

β = nilai pergeseran kecerahan citra yang diambil secara acak dalam rentang tertentu $[\beta_{min}, \beta_{max}]$.

2. Penyesuaian nilai kontras pada citra yang mengacu pada perbedaan antara area terang dan gelap dalam citra. Penyesuaian ini dirumuskan sebagai:

$$I''(x, y) = \alpha \cdot (I'(x, y) - \overline{I}) + \overline{I}$$

atau lebih sederhana

$$I''(x, y) = \alpha \cdot I'(x, y)$$

dimana:

α = faktor skala kontras yang diambil secara acak dalam rentang tertentu $[\alpha_{min}, \alpha_{max}]$.

$\bar{I}$ = nilai rata-rata intensitas piksel pada citra.

3. Nilai piksel $I''(x, y)$ dikontrol agar tetap berada dalam rentang yang valid, yaitu [0, $2k - 1$] untuk citra k -bit.

$$Ifinal(x, y) = \min(\max(I''(x, y), 0) , 2k - 1)$$

d. Vertical dan Horizontal Flip

Flip vertikal dan horizontal merupakan metode transformasi geometris sederhana yang bertujuan untuk mengubah orientasi citra dengan cara membalik citra dengan vertikal atau horizontal. Metode ini bertujuan untuk meningkatkan variasi pada dataset dan membuat model mampu mengenali objek dalam berbagai posisi dan orientasi yang berbeda (Shorten & Khoshgoftaar, 2019). Adapun untuk rumusnya yaitu:

$$I'_{vertical}(x, y) = I(H - 1 - x, y)$$

$$I'_{horizontal}(x, y) = I(x, H - 1 - y)$$

Dimana

- x = kordinat piksel pada sumbu vertikal
- y = kordinat piksel pada sumbu horizontal
- W = lebar citra
- H = tinggi citra

e. Mosaic Data Augmentation

Mosaic data augmentation merupakan metode augmentasi yang menggambungkan 4 citra menjadi 1 citra baru dengan cara mengambil beberapa bagian dari masing-masing citra secara acak untuk digabungkan menjadi 1 citra yang baru. Metode ini pertama kali diperkenalkan pada YOLOv4 sebagai inovasi dalam pelatihan model objek deteksi dan masih digunakan hingga versi YOLO terbaru saat ini untuk membuat model tahan terhadap berbagai variasi dan bentuk model, meningkatkan kemampuan model dalam mendeteksi objek kecil, mengurangi ketergantungan pada lokasi objek, dan membantu model dalam memahami latar belakang yang bervariasi. Metode ini sering dikombinasikan dengan teknik transformasi seperti random scaling, cropping, atau padding (Bochkovskiy et al., 2020). Adapun proses dari mosaic data augmentation:

- Empat citra $I1$, $I2$, $I3$, $I4$yang diambil secara acak dari dataset
- Setiap citra dapat diubah melalui proses seperti:

$$I'_i = Scale(I_i, s_i) \; dan \; I'_i = Transform(I'_i, T_i)$$

Dimana si merupakan faktor skala dan Ti merupakatransformasi (misalnya rotasi, flip, atau pemangkasan).

- Keempat citra $I'1$, $I'2$, $I'3$, $I'4$ digabungkan menjadi satu citra

$Imosaic$ dengan resolusi baru $W\ x\ H$. Proses penggabungan dilakukan dengan menentukan titik tengah acak (cx, cy), dimana:

- $cx \in [\alpha W, (1-\alpha)W]$,
- $cy \in [\alpha H, (1-\alpha)H]$,

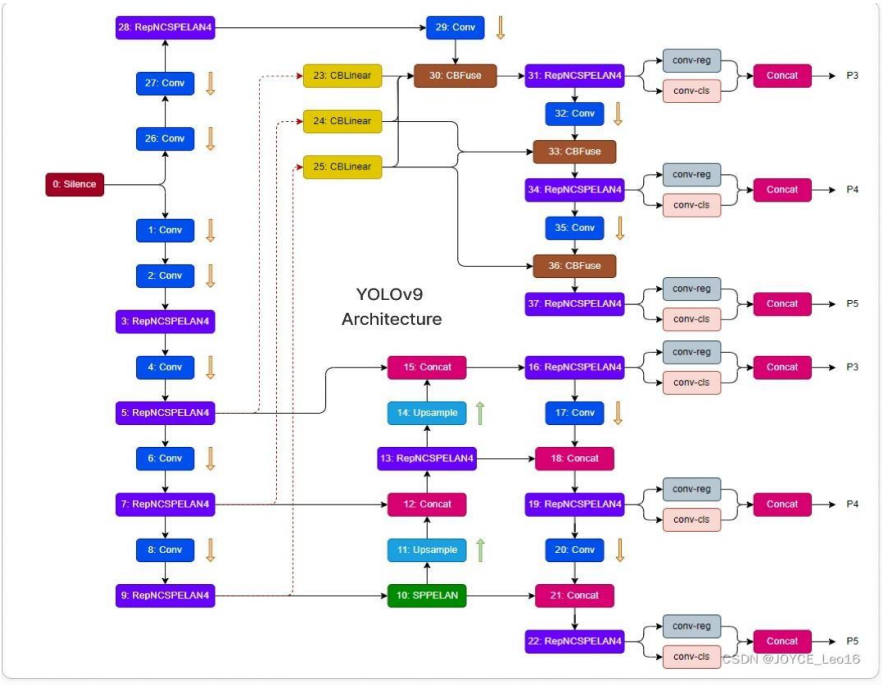
dengan α merupakan faktor yang menentukan proporsi ukuran setiap kuadran. Adapun untuk pembagian kuadran dilakukan sebagai berikut:

$$I'1(x, y), \quad untuk\ x < cx, y < cy$$
$$I'2(x - cx, y), \quad untuk\ x \geq cx, y < cy$$
$$Imosaic(x, y) = \begin{cases} I'3(x, y - cy), & untuk\ x < cx, y \geq cy \\ I'4(x - cx, y - cy), & untuk\ x \geq cx, y \geq cy \end{cases}$$

2.5.5. Pelatihan Model YOLO dan Faster R-CNN

Adapun pada pelatihan ini, model YOLO yang digunakan adalah YOLOv9 dan untuk model Faster R-CNN adalah model awal dari Faster R-CNN.

a. YOLO v9



Gambar 2.10 Arsitektur YOLOv9 (C.-Y. Wang et al., 2024)

YOLOv9 merupakan versi YOLO resmi yang dikembangkan oleh Wang et.al pada tahun Februari 2024, model ini merupakan pengembangan dari arsitektur YOLOv7. Adapun arsitekturnya sebagai berikut:

a. Backbone

Pada YOLOv9, backbone yang digunakan adalah GELAN (Generalized ELAN) yang merupakan modifikasi dari arsitektur ELAN pada YOLOv7 dengan menambahkan CSPNet, Re-Paremeterized untuk meningkatkan akurasi dan kecepatan deteksi dari model. Selain itu, terdapat metode baru, yaitu Programmable Gradient Information (PGI) yang mengatasi pengurangan informasi pada jaringan yang dalam sehingga meningkatkan akurasi dan memungkinkan model dengan jaringan yang dangkal memiliki performa yang sama dengan jaringan yang dalam.

b. Neck

Pada bagian neck, penambahan arsitektur SPPNet dilakukan agar model mampu mempelajari fitur dari berbagai resolusi yang diberikan. Selain itu, feature yang dihasilkan pada backbone di-upsample untuk agar model dapat mempelajari informasi detail pada input citra. Setelah itu, model di- downsampling untuk kemudian dijadikan sebagai input pada bagian head.

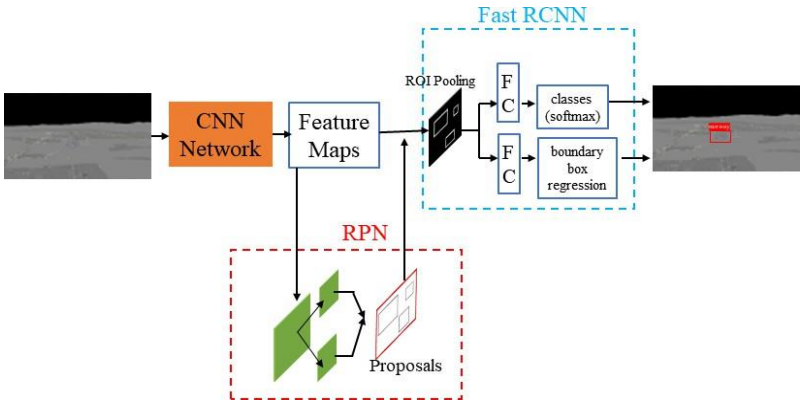
c. Auxiliary Head

Auxiliary merupakan arsitektur tambahan yang berjalan secara paralel dengan arsitektur utama (backbone dan neck). Auxiliary memiliki tujuan untuk membantu model dalam meningkatkan performa deteksi dengan cara membantu model untuk dapat mempelajari representasi fitur yang lebih baik dan mengatasi masalah yang mungkin terjadi selama pelatihan.

d. Head

Pada bagian head, merupakan jaringan Fully Connected (FC Layer) yang berfungsi untuk melakukan prediksi bounding box dan melakukan klasifikasi. Pada YOLOv9 terdapat dua jaringan head, yaitu pada jaringan utama dan pada jaringan auxiliary. Fungsi jaringan head pada jaringan auxiliary adalah untuk membantu jaringan head pada jaringan utama dalam melakukan deteksi dan klasifikasi objek yang lebih baik

b. Faster R-CNN



Gambar 2.11 Arsitektur Faster R-CNN (Ren et al., 2017)

Faster R-CNN merupakan model deteksi objek yang dikembangkan oleh Ren et.al pada tahun 2016 dan merupakan perkembangan dari model R-CNN dan Fast R-CNN dengan menambahkan Region Proposal Network dalam proses region proposals-nya. Adapun arsiterktu dari Faster R-CNN adalah sebagai berikut

a) Backbone
Pada Faster R-CNN, backbone yang digunakan adalah VGG16. VGG16 merupakan backbone yang mampu mempelajari feature yang kompleks secara efektif. Output dari backbone ini berupa feature map yang akan digunakan pada jaringan RPN dan lapisan RoI Pool menggunakan sharing computation.

b) Region Proposal Network
Region Proposal Network (RPN) merupakan jaringan fully connected yang berfungsi untuk menentukan region pada input citra yang memiliki objek yang ada didalamnya dengan menggunakan feature map yang dihasilkan pada backbone. Output dari RPN merupakan set dari kordinat region yang dihasilkan dan objectness score yaitu seberapa yakin region tersebut terdapat objek didalamnya.

c) RoI Pool Layer
Jaringan RoI Pool berfungsi untuk memilih region yang dihasilkan pada RPN yang memiliki objek yang didalamnya. RoI Pool kemudian mengambil feature yang ada pada region menggunakan feature map yang dihasilkan pada backbone untuk kemudian direpresentasikan ulang dalam bentuk 7 x 7 dengan menggunakan max pooling untuk kemudian digunakan sebagai input pada jaringan fully connected.

d) Fully Connected Layer
Jaringan fully connected pada Faster R-CNN berfungsi untuk memprediksi kordinat objek dan mengklasifikasikan objek tersebut

menggunakan feature map yang dihasilkan pada jaringan RoI Pool.Skenario Pelatihan

2.5.6. Skenario Pelatihan

Pada penelitian ini, dilakukan skenario pelatihan untuk menentukan hyperparameter terbaik pada masing-masing model sebelum dilakukan perbandingan performa. Adapun skenario pelatihannya adalah sebagai berikut:

a. Kombinasi Optimizer dan Learning Rate

Skenario ini bertujuan untuk menentukan kombinasi optimizer dan learning rate terbaik pada masing-masing model untuk kemudian akan digunakan pada skenario pelatihan selanjutnya. Model yang digunakan sebagai awal pelatihan ini adalah model Faster R-CNN dengan backbone VGG-16 dan YOLOv9m (medium) dengan ukuran input citra 640 x 640 piksel.

1. Learning Rate

Learning rate merupakan salah satu hyperparameter yang paling kritis dalam proses pelatihan deep learning. Parameter ini menentukan seberapa besar langkah yang diambil dalam melakukan proses optimasi gradien untuk memperbarui jaringan.

Learning rate yang terlalu besar dapat berakibat pada proses pelatihan model yang tidak stabik atau dapat berakibat pada gagal konvergen karena “melewati” solusi yang opitmal, sementara learning rate yang terlalu kecil dapat mengakibatkan proses pelatihan yang lambat dan berisiko terjebak dalam minimum lokal. Proses pemilihan learning rate tepat sangat tidak menentu karena dapat dipengaruhi oleh kompleksitas model, dataset, dan tujuan pelatihan model itu sendiri. Pada penelitian ini, learning rate yang akan digunakan yaitu 0.01, 0.001, dan 0.0001.

2. Batch Size

Batch size merupakan jumlah sampel data pelatihan yang diproses secara simultan dalam satu iterasi pada proses optimasi gradien. Dalam kerangka teoritis pada proses pembelajaran mesin, batch size berperan krusial dalam mengoptimalkan proses konvergensi dan efisiensi komputasi. Batch size memiliki pengaruh terhadap kecepatan pelatihan model dimana batch size tinggi memiliki kecepatan latihan yang tinggi namun berisiko menyebarkan konvergensi model ke titik optimal lokal yang berbeda, sedangkan batch size rendah meningkatkan potensi untuk terhindar dari overfitting namun memiliki kecepatan latihan yang lambat (Keskar et al., 2017; Rumelhart et al., 1986). Pada penelitian ini akan menggunakan batch size 10 untuk model Faster R-CNN dan 16 untuk YOLOv9. Pemilihan ini dilakukan karena ukuran

kedua model yang berbeda dan penggunaan GPU yang terbatas, yaitu GPU T4 dengan GPU RAM 16 GB.

3. Stochastic Gradient Descent (SGD)

Stochastic Gradient Descent (SGD) merupakan metode optimizer yang fundamental dalam deep learning. SGD bekerja dengan memperbarui parameter model menggunakan gradien yang dihitung dari subset (mini-batch) pada data latih yang dipilih secara acak, bukan menggunakan seluruh dataset sekaligus. Rumus dasar dari SGD adalah

$$\theta(t + 1) = \theta(t) - \mu\nabla J(\theta(t))$$

Namun, dalam perkembangan SGD, terdapat beberapa metode yang ditambahkan pada SGD untuk meningkatkan kemampuannya, yaitu momentum dan weight decay. Momentum merupakan teknik yang digunakan untuk mempercepat proses konvergensi dan mengurangi osilasi. Momentum bekerja dengan menambahkan fraksi dari perubahan parameter sebelumnya pada parameter sekarang. Dengan penambahan momentum, model mampu melewati lokal minimum yang sering terjadi ketika learning rate rendah, serta meningkatkan stabilitas dalam proses pelatihan model.

Adapun weight decay merupakan teknik yang digunakan untuk mencegah overfitting dengan meambahkan penalty term ke dalam fungsi loss. Dengan penambahan weight decay, SGD mampu meningkatkan generalisasi pada model, membuat model menjadi lebih tahan terhadap noise dalam data hingga membantu konvergensi dengan membuat landscape optimisasi yang lebih halus. Dengan penambahan momentum dan weight decay pada SGD, rumus awal SGD berubah menjadi:

$$v(t + 1) = \gamma v(t) + \mu\,(\nabla J(\theta(t)) + \lambda\theta(t))$$

$$\theta(t + 1) = \theta(t) - v(t + 1)$$

$\theta(t + 1)$ = parameter model pada iterasi t + 1

$\theta(t)$ = parameter model pada iterasi t

$v(t + 1)$ = velocity pada waktu t + 1

$v(t)$ = velocity pada waktu t

γ = momentum = 0.9 (default) (Krizhevsky et al., 2017; Simonyan & Zisserman, 2015)

λ = weight decay = 0.0005 (default) (Krizhevsky et al., 2017; Simonyan & Zisserman, 2015)

4. Adaptive Moment Estimation (Adam)

Adaptive Moment Estimation (Adam) merupakan algoritma optimasi yang menggabungkan ide dari RMSprop dan momentum yang dimana Adam menggunakan estimasi momen pertama (mean) dan momen kedua (variance) dai gradien untuk melakukan adaptasi learning rate untuk setiap parameter. Rumus dari Adam yaitu:

$$mt = \beta 1 mt{-}1 + (1 - \beta 1) gt$$

$$vt = \beta 2 vt{-}1 + (1 - \beta 2) g2$$

mt = Estimasi momen pertama (mean)

vt = Estimasi momen kedua (variance)

gt = Gradien pada waktu t

$\beta 1$, $\beta 2$ = hyperparameter decay rates = (0.937, 0.999) (Reddi et al., 2018)

Karena bias awal mendekati nol, maka dilakukan bias correction:

$$\widehat{m} = \frac{m_t}{1 - \beta_1^t}, \hat{v} = \frac{v_t}{1 - \beta_2^t}$$

Dengan perubahan parameter:

$$\theta_t = \theta_{t-1} - \alpha \frac{\widehat{m}}{\sqrt{\hat{v}} + \epsilon}$$

α = *learning rate*

ϵ = konstanta untuk mencegah pembagian nol

Tabel 2.1 Skenario Pelatihan Learning Rate dan Optimizer

Model	Learning Rate	Optimizer	Ukuran Citra	Batch Size	Epoch
VGG-16 Faster R-CNN	0.01	SGD	640 x 640 piksel	10	100
VGG-16 Faster R-CNN	0.001	SGD	640 x 640 piksel	10	100
VGG-16 Faster R-CNN	0.0001	SGD	640 x 640 piksel	10	100
VGG-16 Faster R-CNN	0.01	Adam	640 x 640 piksel	10	100
VGG-16 Faster R-CNN	0.001	Adam	640 x 640 piksel	10	100
VGG-16 Faster R-CNN	0.0001	Adam	640 x 640 piksel	10	100
YOLOv9m (medium)	0.01	SGD	640 x 640 piksel	16	100
YOLOv9m (medium)	0.001	SGD	640 x 640 piksel	16	100
YOLOv9m (medium)	0.0001	SGD	640 x 640 piksel	16	100
YOLOv9m (medium)	0.01	Adam	640 x 640 piksel	16	100
YOLOv9m (medium)	0.001	Adam	640 x 640 piksel	16	100
YOLOv9m (medium)	0.0001	Adam	640 x 640 piksel	16	100

b. Ukuran Input Citra

Skenario ini bertujuan untuk menentukan ukuran input citra yang sesuai yang dapat menigkatkan performa model secara maksimal pada beberapa input citra yang diberikan dan melihat pengaruh ukuran input citra performa model dengan menggunakan model pada skenario pelatihan sebelumnya. Adapun untuk ukuran input citra yang digunakan yaitu 480 x 480, 640 x 640, dan 720 x 720. Model terbaik pada pelatihan ini akan digunakan untuk skenario penelitian selanjutnya.

Tabel 2.2 Skenario Pelatihan Ukuran Input Citra

Model	Learning Rate	Optimizer	Ukuran Citra	Batch Size	Epoch
VGG-16 Faster R-CNN	Learning Rate Terbaik	Optimizer Terbaik	480 x 480 piksel	10	100
VGG-16 Faster R-CNN	Learning Rate Terbaik	Optimizer Terbaik	640 x 640 piksel	10	100

VGG-16 Faster R-CNN	Learning Rate Terbaik	Optimizer Terbaik	720 x 720 piksel	10	100
YOLOv9m (medium)	Learning Rate Terbaik	Optimizer Terbaik	480 x 480 piksel	16	100
YOLOv9m (medium)	Learning Rate Terbaik	Optimizer Terbaik	640 x 640 piksel	16	100
YOLOv9m (medium)	Learning Rate Terbaik	Optimizer Terbaik	720 x 720 piksel	16	100

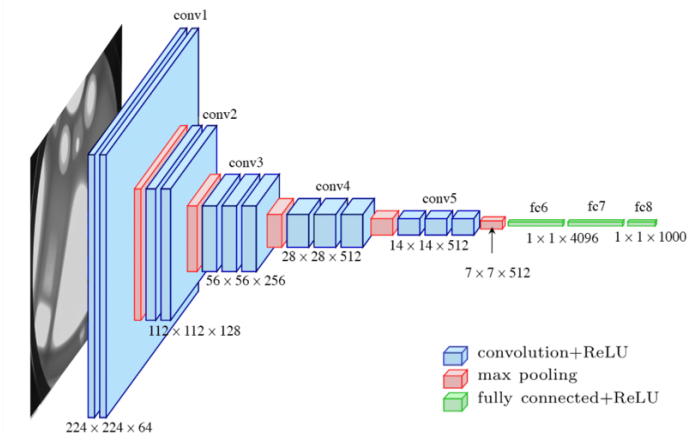
c. Backbone (Untuk Faster R-CNN)

Tabel 1.3 Skenario Pelatihan Arsitektur Backbone (Faster R-CNN)

Model	Learning Rate	Optimizer	Ukuran Citra	Batch Size	Epoch
VGG-16 Faster R-CNN	Learning Rate Terbaik	Optimizer Terbaik	Ukuran Citra Terbaik	10	100
ResNet50+FPN Faster R-CNN	Learning Rate Terbaik	Optimizer Terbaik	Ukuran Citra Terbaik	10	100
CSPDarknet53 Faster R-CNN	Learning Rate Terbaik	Optimizer Terbaik	Ukuran Citra Terbaik	10	100
EfficientNetv2 Faster R-CNN	Learning Rate Terbaik	Optimizer Terbaik	Ukuran Citra Terbaik	10	100

Skenario ini dikhususkan untuk model Faster R-CNN, dimana melihat pengaruh model dengan menggunakan berbagai arsitektur CNN terhadap performanya dalam mendeteksi BSR pada kelapa sawit dengan menggunakan hyperparameter dan ukuran input citra terbaik pada skenario pelatihan sebelumnya. Adapun arsitektur CNN yang digunakan yaitu:

a. VGG16



VGG16 (Visual Geometry Group 16-layer) merupakan arsitektur jaringan konvolusi yang diperkenalkan oleh Simonyan dan Zisserman yang memberikan paradigma baru dalam mendesain arsitektur deep learning dalam pengenalan citra. Arsitektur ini menggunakan filter konvolusi 3 x 3 secara konsisten pada seluruh lapisannya yang memungkinkan model untuk mengekstraksi fitur kompleks melalui tumpukan lapisan konvolusi yang dalam dengan kedalaman totalnya 16 (Simonyan & Zisserman, 2015).

Meskipun memiliki jumlah parameter yang besar, VGG16 tetap menjadi arsitektur yang efektif dalam berbagai tugas termasuk dalam deteksi objek. Backbone ini juga yang digunakan ketika Faster R-CNN pertama kali diperkenalkan (Ren et al., 2017).

b. ResNet50FPN

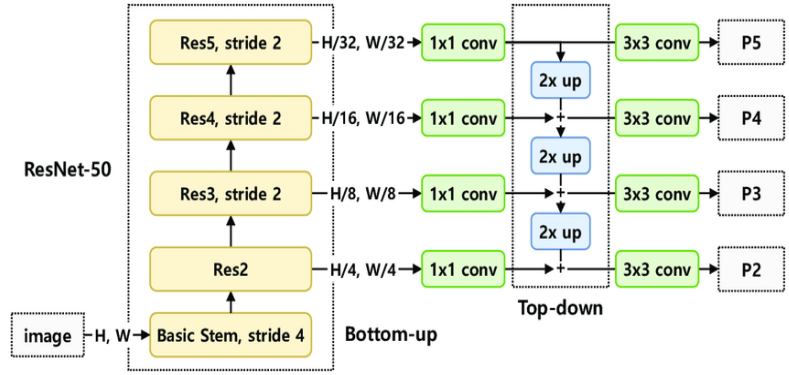
Residual Network (ResNet) merupakan inovasi arsitektur jaringan CNN yang diperkenalkan oleh He et al. pada tahun 2015. Arsitektur ini mengatasi permasalahan vanishing gradient dan gradient explotion yang menjadi tantangan utama dalam pelatihan jaringan deep learning yang dalam dengan memperkenalkan konsep skip connection atau residual block. Pada setiap blok, input tidak hanya diproses melalui jaringan konvolusi dan aktivasi, tetapi juga secara langsung input dialirkan melalu jalur tambahan yang disebut sebagai shortcut connection memungkinkan gradien untuk mengalir lebih muda h pada jaringan yang sangat dalam (He et al., 2016).

Mekanisme ini memungkinkan pelatihan jaringan dengan kedalaman hingga ratusan layer yang sebelumnya tidak mungkin untuk dilakukan dengan cara menambahkan identity mapping yang memungkinkan jaringan belajar untuk belajar representasi yang lebih kompleks.

Feature Pyramid Network (FPN) merupakan arsitektur yang dikembangkan dengan tujuan untuk mengatasi permasalahan pada deteksi objek dalam mendeteksi objek pada berbagai skala. Arsitektur ini diperkenalkan oleh Lin et al. pada tahun 2017 yang dimana arsitektur ini mengintegrasikan representasi fitur pada berbagai tingkat resolusi dalam jaringan konvolusi. Arsitektur ini membangun piramida fitur dengan cara mengkombinasikan informasi semantik dari lapisan terakhir detail spasial pada lapisan yang lebih rendah melalui operasi upsampling dan lateral connection (X. Li et al., 2019).

Pendekatan ini memungkinkan jaringan untuk mendeteksi objek pada berbagai skala dengan akurasi tinggi dengan memanfaatkan hierarki fitur yang dihasilkan pada jaringan konvolusi.

Resnet50 mempunyai keunggulan yang signifikan dibandingkan berbagai variasi ResNet lainnya, seperti ResNet18 atau ResNet101. Hal ini dikarenakan ResNet50 menawarkan keseimbangan optimal antara kompleksitas komputasi dan kapasitas representasi (He et al., 2016).



Gambar 2.12 Arsitektur ResNet50FPN

Dengan 50 layer, arsitektur ini mampu menghasilkan representasi fitur yang lebih kompleks dibandingkan ResNet18 dan ResNet34, namun lebih efisien dibandingkan ResNet101 dan ResNet152. Integrasi FPN dengan ResNet50 memungkinkan jaringan untuk mengekstraksi fitur multi- skala dengan presisi tinggi, dan mampu meningkatkan kemampuan deteksi objek pada berbagai ukuran dan kondisi (X. Li et al., 2019).

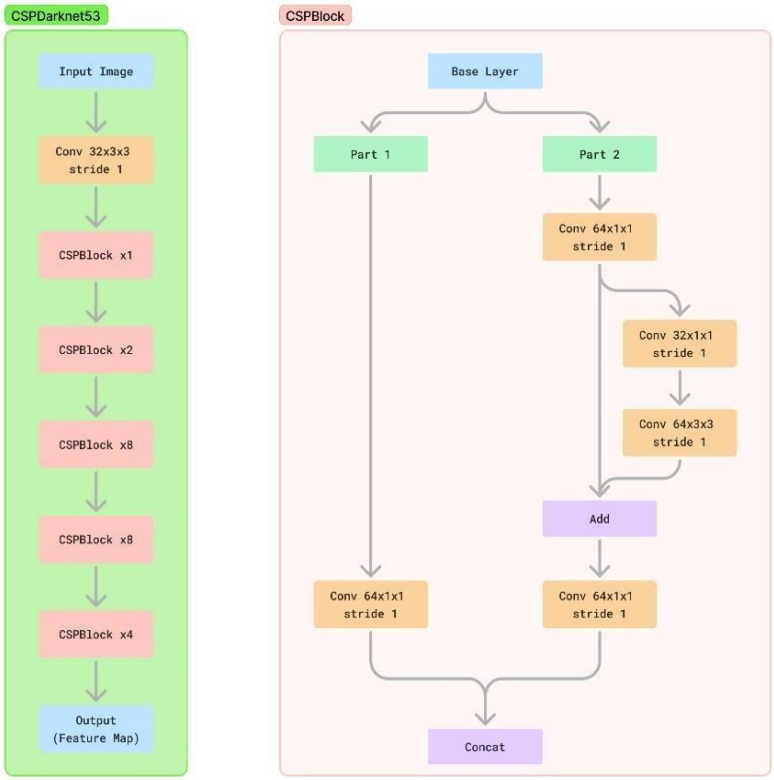
c. CSPDarknet53

Darknet53 merupakan arsitektur jaringan konvolusi yang diperkenalkan oleh Redmon pada arsitektur YOLOv3. Arsitektur ini dirancang khusus untuk deteksi objek dengan efisiensi komputasi yang tinggi.

Darknet53 dikembangkan dengaan menggabungkan konsep dari residual block pada ResNet dengan mendesain arsitektur sedemikian rupa agar optimal pada kecepatan dan akurasi. Struktur jaringan ini terdiri dari serangkaikan blok konvolusi dengan skip connection, serta menggunakan global average pooling (Redmon & Farhadi, 2018). Arsitektur ini dirancang untuk mencapai keseimbangan antara kompleksitas komputasi dan kemampuan ekstraksi fitur, menjadikan arsitektur ini cepat namun akurat dalam deteksi objek.

Cross Stage Partial Network (CSPNet) merupakan inovasi arsitektur jaringan deep learning yang dikembangkan untuk mengurangi kompleksitas komputasi dan meningkatkan efisiensi dalam merepresentasikan fitur. CSPDarknet53 merupakan arsitektur yang dikembangkan dengan mengintegrasikan CSPNet dan Darknet53 dan dikembangkan pertama kali dalam arsitektur

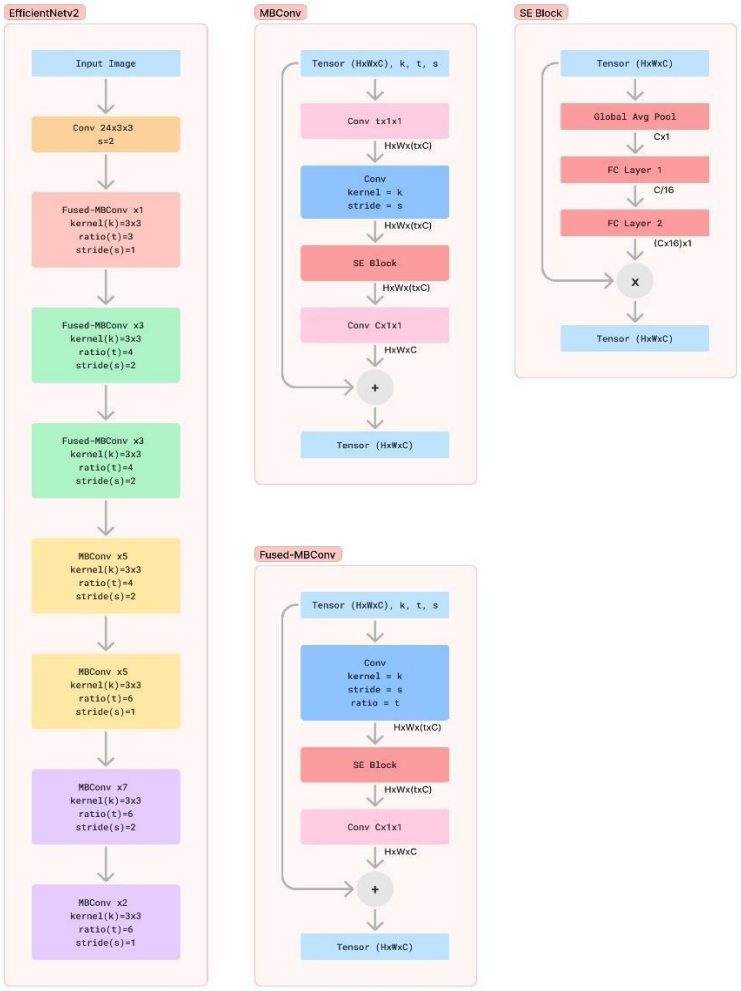
YOLOv4. Dengan pengintegrasian kedua jaringan ini, mampu menghasilkan arsitektur yang secara signifikan meningkatkan performa deteksi objek (C. Y. Wang et al., 2020).



Gambar 2.13 arsitektur CSPDarkenet53

CSP membagi feature map menjadi dua bagian yang diproses secara terpisah untuk kemudian digabungkan sehingga memungkinkan jaringan untuk mengektraksi fitur yang lebih kaya dengan beban komputasi yang rendah. Keuntungan utama dari CSPDarknet53 adalah kemampuannya untuk mengurangi jumlah parameter komputasi, meningkatkan kecepatan inference, namun tetap menjaga akurasi yang tinggi. Hal ini menjadikan arsitektur ini banyak digunakan termasuk pada YOLOv5 dan YOLOv8 dengan membentuk jaringan baru dengan CSPDarknet53 sebagai baseline backbone (Bochkovskiy et al., 2020).

d. EfficientNetv2



Gambar 2.14 Arsitektur EfficientNetv2

EfficientNetv2 merupakan arsitektur jaringan saraf konvolusional yang diperkenalkan oleh Tan dan Le pada tahun 2021. Arsitektur ini menghadirkan sebuah terobosan yang baru dalam mendesain arsitektur deep learning dengan memberikan fokus utama pada skalabilitas, efisiensi komputasi, dan performa. EfficientNetv2 merupakan pengembangan dari versi sebelumnya dengan mengimplementasikan strategi metode scaling yang lebih efektif dengan memperkenalkan konsep progressive learning dan adaptive regularization, yang memungkinkan model dapat dilatih dengan lebih cepat dan efisien pada berbagai ukuran dataset.

Arsitektur ini juga memperkenalkan inovasi terbaru seperti blok MBConv dan blok Fused-MBConv, adaptive batch normalization, dan dipadukan dengan metode progressive training strategy yang

membuat model mampu untuk mencapai akurasi yang tinggi dengan kompleksitas komputasi yang rendah. Dengan pendekatan ini, EfficientNetv2 mampu mencapai trade-off optimal antara akurasi, kecepatan pelatihan, dan efisiensi komputasi (Tan & Le, 2021).

d. Model Size (untuk YOLOv9)

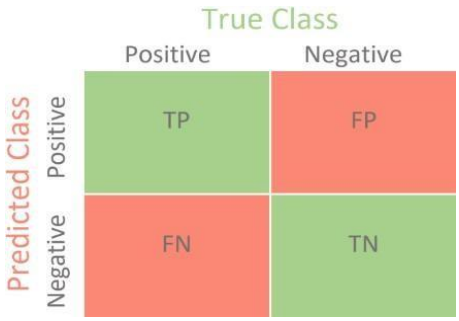
Skenario ini dikhususkan untuk YOLOv9, dimana melihat pengaruh model ketika ukuran dari YOLOv9 ditingkatkan atau diturunkan. Adapun *hyperparameter* dan ukuran input citra yang digunakan yaitu *hyperparameter* dan ukuran input citra terbaik dari skenario pelatihan sebelumnya. Ukuran model YOLOv9 yang digunakan pada penelitian ini yaitu YOLOv9t (tiny), YOLOv9s (small), YOLOv9m (medium), YOLOv9c (compact) dan YOLOv9e (extensive).

Tabel 1.4 Skenario Pelatihan Model Size YOLOv9

Model	Learning Rate	Optimizer	Ukuran Citra	Batch Size	Epoch
YOLOv9t (tiny)	Learning Rate Terbaik	Optimizer Terbaik	Ukuran Citra Terbaik	10	100
YOLOv9s (small)	Learning Rate Terbaik	Optimizer Terbaik	Ukuran Citra Terbaik	10	100
YOLOv9m (medium)	Learning Rate Terbaik	Optimizer Terbaik	Ukuran Citra Terbaik	10	100
YOLOv9c (compact)	Learning Rate Terbaik	Optimizer Terbaik	Ukuran Citra Terbaik	10	100
YOLOv9e (extensive)	Learning Rate Terbaik	Optimizer Terbaik	Ukuran Citra Terbaik	10	100

2.5.7. Evaluasi Model

a. Confusion Matrix



Gambar 2.15 Confusion Matrix

Confusin matrix merupakan salah satu alat evaluasi untuk mengukur kinerja model klasifikasi. Matriks ini menyajikan representasi visual yang komprehensif tentang prediksi benar atau salah yang dihasilkan oleh

model klasifikasi, dengan mengorganisasikan hasil prediksi ke empat kategori utama, yaitu *True Positive* (TP), *True Negative* (TN), *False Negative* (FN), dan *False Positive* (FP).

Adapun ukuran dari matriks ini adalah $m\ x\ m$ dengan m adalah jumlah kelas. Dalam deteksi objek, setiap kelas akan dianggap sebagai kelas positif independen dan background akan dianggap sebagai kelas negatif sehingga ukuran confusin matrix pada deteksi objek adalah (m + 1) x (m + 1). Pada penelitian ini, jumlah kelas yang harus dideteksi adalah 3 sehingga ukuran confusion matrix setiap model adalah 4 x 4.

b. Precision

Precision mengukur ketepatan model dalam melakukan prediksi positif yang didefinisikan sebagai rasio antara jumlah prediksi positif yang benar (True Positive) dengan total prediksi positif yang dihasilkan oleh model. Secara umum, rumus dari precision adalah:

$$Precision\ (P) = \frac{True\ Positive\ (TP)}{True\ Positve\ (TP) + False\ Positive\ (FP)}$$

Namun dalam kasus deteksi objek, perhitungan precision padamodel sedikit berbeda dimana setiap kelas kecuali background dipandang sebagai kelas positif independen sehingga precision merupakan rata-rata dari nilai precision masing-masing kelas pada suatu model.

$$P_{tot} = \frac{1}{n}\sum_{i=0}^{n-1} P_i$$

$$P_i = \frac{TP_i}{TP_i + FP_i}$$

$Ptot$ = Precision pada keseluruhan kelas
Pi = Precision pada kelas ke-i
TPi = Jumlah True Positive pada kelas ke-i
FPi = Jumlah False Positive pada kelas ke-i
n = Jumlah kelas

c. Recall

Recall atau yang kadang disebut juga sebagai sensitivitas atau true positive rate, merupakan metrik evaluasi yang mengukur kemampuan model dalam mengidentifikasi semua instance positif yang sebenarnya pada keseluruhan instance positif yang ada. Secara umum, rumus recall adalah:

$$Recall\ (R) = \frac{True\ Positive\ (TP)}{True\ Positive\ (TP) + False\ Negative\ (FN)}$$

Dalam deteksi objek, pengukuran recall pada model deteksi objek sama dengan pengukuran precision.

$$R_{tot} = \frac{1}{n}\sum_{i=0}^{n-1} R_i$$

$$R_i = \frac{TP_i}{TP_i + FN_i}$$

$Rtot$ = Recall pada keseluruhan kelas

Ri = Recall pada kelas ke-i

TPi = Jumlah True Positive pada kelas ke-i

FNi = Jumlah False Negative pada kelas ke-i

n = Jumlah kelas

d. Mean Average Precision (mAP)

Intersection over Union (IoU) merupakan metrik fundamental dalam mengevaluasi model deteksi objek yang mengukur tingkat tumpang tindih antara ground truth bouding box dan prediksi bounding box pada citra. Adapun rumus dari IoU adalah:

$$IoU = \frac{Area\ of\ Overlap}{Area\ of\ Union}$$

Rentang nilai IoU adalah 0 sampai 1, dimana 1 menandakan prediksi bounding box tumpang tindah secara sempurna pada ground truth bounding box, sedangkan 0 menandakan prediksi bounding box tidak tumpang tidak sama sekali pada ground truth bounding box. Penggunaan IoU memungkinkan peneliti untuk mengukur presisi lokalisasi objek secara kuantitatif.

Average Precision (AP) dan mean average precision (mAP) merupakan metrik evaluasi yang lebih komprehensif dibandingkan akurasi sederhana dalam konteks deteksi objek. AP diukur sebagai luas dibawah kurva precision-recall pada setiap kelas yang dirumuskan sebagai berikut:

$$AP_i = \int_0^1 P(r)_i dr$$

APi = Average Precision

$P(r)i$ = Precision pada tingkat recall r pada kelas ke-i

Adapun mean average precision merupakan nilai rata-rata AP pada keseluruhan kelas yang dirumuskan sebagai berikut:

$$mAP = \frac{1}{n} \sum_{i=0}^{n-1} AP_i$$

$$mAP = AP \leftrightarrow n = 1$$

mAP = Mean Average Precision

APi = Average Precision pada kelas ke-i

mAP menjadi metriks utama dalam menentukan performa suatu model deteksi objek dibandingkan akurasi karena mampu menangkap performa model secara menyeluruh, mempertimbangkan trade-off antara precision dan recall, serta memberikan evaluasi yang lebih

komprehensif pada dataset dengan distribusi kelas yang tidak seimbang.

Dalam perkembangannya, mAP memiliki dua varian sebagai standar utama dalam bencmark deteksi objek, yaitu mAP@50 dari Pascal VOC Challenge dan mAP@(50-95) dari MS COCO Challenge. mAP@50 menggunakan threshold iou

0.5 sebagai standar untuk menentukan prediksi benar. Sedangkan pada mAP@(50-95) evaluasi yang diberikan lebih ketat dan komprehensif karena mengukur rata-rata dari mAP pada berbagai threshold iou (0.5, 0.55, 0.6, …, 0.90, 0.95).

$$mAP@50 = \frac{1}{n} \sum_{i=0}^{n-1} AP_i$$

$$mAP@(50-95) = \frac{1}{10\ x\ n} \sum_{t=0.5}^{0.95} \sum_{i=0}^{n-1} AP_{i,t}$$

e. Inference Speed

Inference Speed atau kecepatan inferensi merupakan metrik yang digunakan untuk mengukur waktu komputasi yang dibutuhkan model untuk melakukan prediksi pada satu atau sekelompok data input. Pada konteks pengenalan gambar atau video, satuan yang digunakan untuk mengukur inference speed ada frame per second (FPS).

Adapun proses perhitungannya dilakukan dengan mengukur jumlah frame atau sampel yang dapat diproses dalam satu detik atau rasio antara total frame/citra dengan total waktu yang dibutuhkan untuk memprediksi semua frame/citra

$$FPS = \frac{n}{\left(\frac{t}{1000}\right)} = \frac{n}{T}$$

FPS = Frame Per Second
t = total inference (ms)
T = total inference (s)
n = total frame/citra

2.5.8. Pengujian Sistem

Setelah memilih model terbaik pada masing-masing pada model Faster R-CNN dan YOLOv9, pada tahap selanjutnya akan dilakukan pengujian sistem. Pengujian sistem dilakukan menggunakan input video yang diambil pada salah satu lokasi tetapi pada daerah yang tidak masuk kedalam data latih maupun validasi. Tujuan dari pengujian sistem ini untuk melihat bagaimana ketahanan model ketika dihadapkan dalam memprediksi penyakit BSR pada kelapa sawit ketika input berupa video dibandingkan hanya menggunakan citra.

BAB III
HASIL DAN PEMBAHASAN

3.1 Hasil Data Preparation

3.1.1. Pengambilan Data

Hasil dari pengambilan data yaitu berupa data video yang akan dilakukan pada proses. Adapun untuk rinciannya pada Tabel.

Tabel 3.1 Hasil Pengambilan Data

Lokasi	Kecepatan Drone (m/s)	Ketinggian Drone (m)	Panjang Video	Resolusi (piksel)	FPS	Ukuran File (GB)
Lokasi 1	0,5 - 2,0	5,8 - 12	Video 1: 10 menit 41 detik Video 2: 9menit 52 detik	1920 x1080	30	Video 1: 7,739 Video 2: 5,2
Lokasi 2	1,0 - 3,5	20 - 30	5 menit 56 detik	1920 x1080	30	2,94
Lokasi 3	1,0 - 2,5	9 - 16	10 menit	1920 x1080	30	4,5
Lokasi 4	1,0 - 4,5	9 - 25	10 menit 11 detik	1920 x1080	30	4,58
Lokasi 5	0,5 - 2,0	10 - 20	7 menit 3 detik	1920 x1080	30	3,963

3.1.2. Ekstraksi Frame

Hasil dari proses pengubahan data video menjadi data citra (frame) ditunjukkan pada tabel

Tabel 3.2 Hasil Ekstraksi Frame

Lokasi	Jumlah frame hasil ekstraksi	Jumlah frame setelah diseleksi	Contoh Frame
Lokasi 1	121 2	792	

Lokasi 2	356	325	
Lokasi 3	600	514	

Contoh Frame yang diambil

Contoh Frame yang tidak diambil

Gambar 3.1 contoh frame yang digunakan dan tidak digunakan

Gambar diatas merupakan contoh frame yang digunakan dan tidak digunakan. frame yang digunakan adalah frame yang memiliki kelapa sawit minimal 1 pohon dengan 70% bagiannya atau lebih terlihat, sedangkan frame yang tidak diambil adalah frame yang tidak memiliki pohon kelapa sawit di dalamnya atau terdapat 1 pohon kelapa sawit namun bagian yang terlihat kurang dari 70%

Contoh Frame yang diambil pada ketinggian 30 m dan kecepatan 4,5 m/s

Contoh Frame yang diambil pada ketinggian 30 m dan kecepatan 2 m/s

Gambar 3.2 contoh frame ketinggian sama

Gambar diatas merupakan contoh frame yang diambil pada ketinggian drone 30 m dari atas tanah namun dalam kecepatan drone yang berbeda, yaitu 4.5m/s dan 2m/s. Dari dua frame tersebut, terlihat bahwa frame dengan kecepatan lebih rendah menghasilkan visual yang lebih jelas dan detail serta efek blur yang minim dibandingkan frame yang diambil pada kecepatan yang lebih tinggi.

Contoh Frame yang diambil pada ketinggian 7 m dan kecepatan 0.5 m/s

Contoh Frame yang diambil pada ketinggian 5 m dan kecepatan 0.5 m/s

Gambar 3.3 contoh frame pada ketinggian sama dengan kecepatan bebeda

Gambar diatas merupakan frame yang diambil pada kecepatan drone 0.5m/s namun dalam ketinggian drone yang berbeda, yaitu 7 m dan 5 m dari atas tanah. Dari kedua frame tersebut, terlihat bahwa model dengan ketinggian drone yang lebih rendah memberikan detail visual yang lebih jelas dibandingkan dengan ketinggian drone yang lebih tinggi

Contoh Frame yang diambil pada jam 9 pagi

Contoh Frame yang diambil pada jam 7 pagi

Gambar 3.4 contoh frame pada ketinggian sama dengan kecepatan berbeda

Gambar diatas merupakan contoh frame yang diambil pada pukul 9 pagi dan pada pukul 7 pagi. Dari kedua frame tersebut, terlihat bahwa frame yang diambil pada pukul 7 pagi, sinar matahari yang lebih rendah menyebabkan bayangan yang lebih tegas pada kelapa sawit, sedangkan frame yang diambil pada pukul 9 pagi memiliki tingkat pencahayaan lebih merata

Contoh Frame yang diambil pada jam 14:30 siang

Contoh Frame yang diambil pada jam 4:30 sore

Gambar 3.5 contoh frame pada waktu yang berbeda

Gambar diatas merupakan contoh frame yang diambil pada pukul 14.30 dan pada pukul 16.30. Dari kedua frame tersebut, frame yang diambil pada pukul 14.30, sinar matahari terlihat lebih terang dan menyinari area dengan lebih intens sehingga dedaunan terlihat lebih jelas dan kontras. Sedangkan pada frame yang diambil pada pukul 16.30, sinar matahari sudah mulai melemah karena posisinya semakin rendah di ufuk barat sehingga intensitas cahaya pada lahan tersebut menjadi lebih redup dan berakibat warna daun hijau terlihat sedikit lebih gelap. Selain itu, frame yang diambil pada pukul 16.30 memberikan efek pada seolah-olah daun tersebut berwarna kuning. Ini dapat mengakibatkan kesulitan model dalam melakukan deteksi karena model akan ambigu dalam menentukan apakah pohon tersebut benar-benar sakit atau tidak.

Contoh Frame yang diambil pada jam 9 pagi kondisi cuaca cerah

Contoh Frame yang diambil pada jam 9 kondisi cuaca mendung dan gerimis

Gambar 3.6 contoh frame pada cuaca yang berbeda

Gambar diatas merupakan contoh frame yang diambil pada pukul 9 pagi pada kondisi cuaca cerah dan pada kondisi cuaca mendung dan gerimis. Dari kedua frame tersebut, tidak tampak perbedaan yang signifikan yang dapat

berakibat pada kesulitan model dalam melakukan deteksi BSR pada kelapa sawit.

Contoh Frame yang diambil pada jam 7 pagi kondisi cuaca cerah

Contoh Frame yang diambil pada jam 7 kondisi cuaca mendung

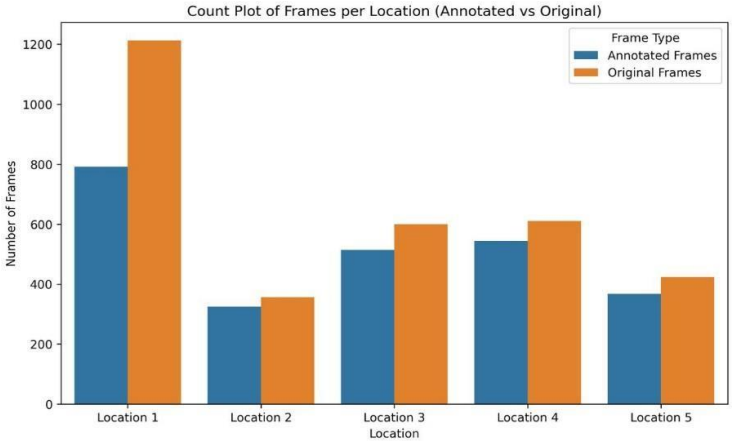
Gambar 3.7 contoh frame pada cuaca yang berbeda

Gambar diatas merupakan contoh frame ketika pengambilan data dilakukan pada pukul 7 pagi pada cuaca cerah dan pada cuaca mendung. Dari kedua frame tersebut, tidak tampak perbedaan yang terlalu signifikan yang dapat berakibat pada kesulitan model dalam melakukan deteksi penyakit BSR pada kelapa sawit.

Contoh perubahan objek akibat terpaan angin

Gambar 3.8 contoh frame pada cuaca yang berbeda

Gambar diatas merupakan contoh frame ketika kelapa sawit terkena terpaan angin. Kecepatan angin yang kencang menyebabkan pelepah kelapa sawit terangkat ke arah angin sehingga mengakibatkan bentuk kelapa sawit menjadi “tidak normal”. Kondisi akan menyulitkan model dalam melakukan deteksi terutama pada tanaman kelapa sawit sehat karena model akan ambigu dalam menentukan apakah pohon kelapa sawit tersebut sehat atau tidak.

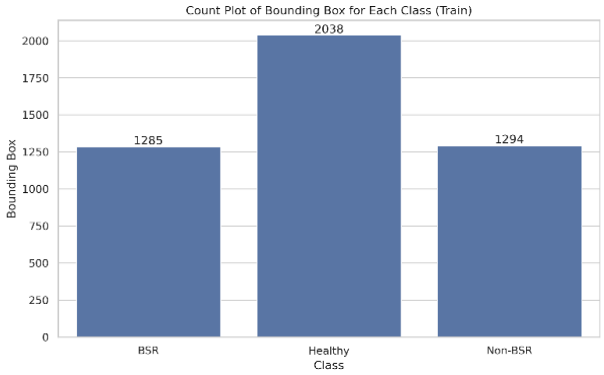


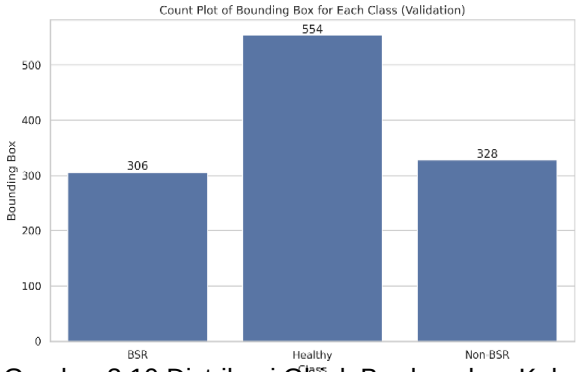
Gambar 3.9 Perbandingan jumlah frame yang akan digunakan pada tahap anotasi dengan jumlah frame sebenarnya pada masing-masing lokasi

Gambar diatas merupakan grafik distribusi frame yang akan diambil untuk melakukan proses anotasi dengan jumlah frame seluruhnya pada masing-masing lokasi. Terlihat bahwa lokasi pertama memiliki jumlah frame yang lebih banyak dibandingkan lokasi yang lain namun selisih antara jumlah frame yang digunakan untuk proses anotasi dengan jumlah frame sebenarnya sangat jauh. Hal ini dikarenakan ukuran kelapa sawitnya yang masih kecil, jarak antar pohon yang cukup jauh, serta kecepatan drone yang lambat mengakibatkan banyak frame yang tidak memiliki atau hanya menampilkan sebagian saja (kurang dari 70%) kelapa sawit yang terlihat dalam frame. Hal ini berbeda dengan lokasi lain, meskipun jarak tanam nya cukup jauh, akan tetapi karena diameter mahkotanya membuat pohon kelapa sawit tersebut tampak berdekatan satu sama lain sehingga hanya sedikit frame yang tidak digunakan pada proses anotasi pada lokasi-lokasi tersebut.

3.2 Hasil Spill Dataset

Dataset dibagi dengan rasio 80% untuk data latih dan 20% untuk data validasi sehingga total data untuk data latih adalah 2021 frame, sedangkan total data untuk validasi adalah 521 frame. Baik data latih maupun data validasi terdistribusi pada 5 lokasi yang digunakan sebagai tempat pengambilan data. Adapun distribusi jumlah objek berdasarkan kelasnya adalah sebagai berikut**:**



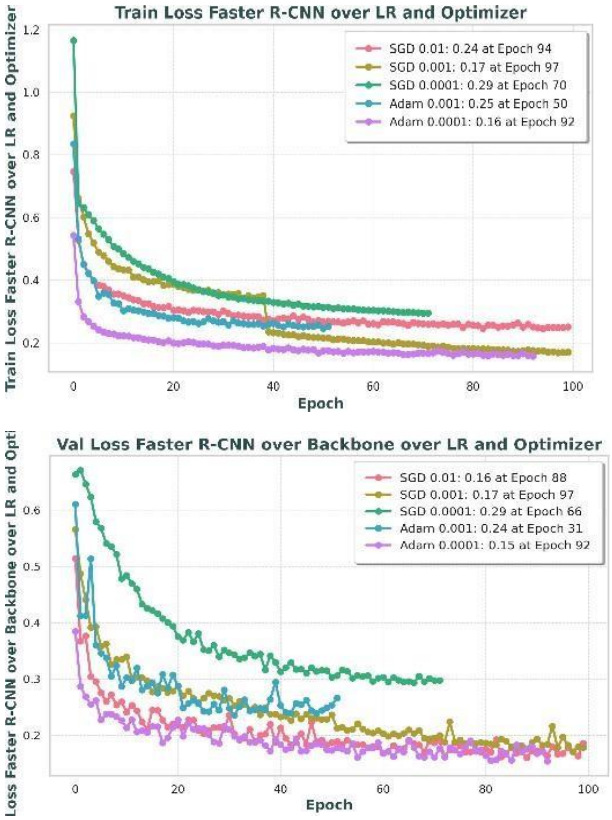


Gambar 3.10 Distribusi Objek Berdasarkan Kelas

3.3 Hasil Training

3.3.1 Faster R-CNN

3.3.1.1 Fine Turning Learning Rate dan Optimizer



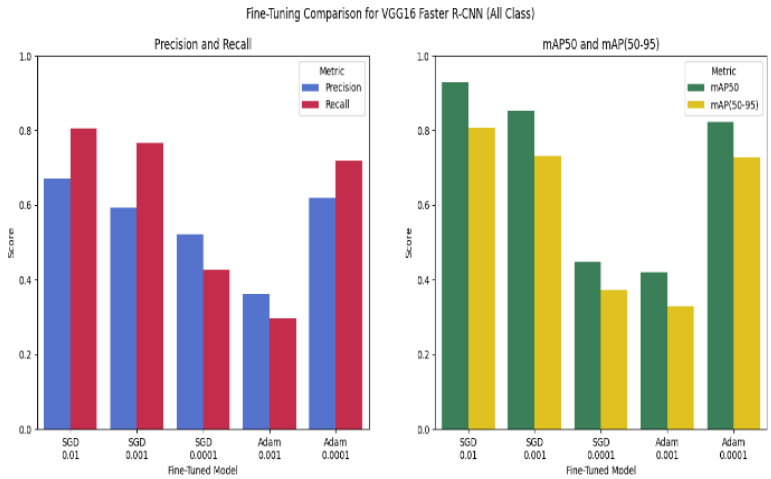
Gambar 3.11 Grafik Fain Los dan Train Loss

Gambar x, y merupakan grafik train loss dan val loss tuning model VGG16 Faster R-CNN skenario pelatihan pada berbagai kombinasi hyperparameter ukuran citra = 640 x 640 piksel, batch size = 10, learning rate = [0.01, 0.001, 0.0001] dan optimizer= [SGD, Adam] di setiap epoch. Model dengan learning rate 0.01

dan optimizer Adam tidak dimasukkan karena train loss nya bernilai NaN, dan val loss nya sangat besar. Adapun untuk kombinasi hyperparameter yang lain, semua model mengalami penurunan train loss meskipun tidak signifikan, kecuali pada hyperparameter learning rate 0.001 dan optimizer SGD yang mengalami penurunan signifikan pada epoch ke-39. Ini menandakan bahwa model belajar dengan baik pada data latih.

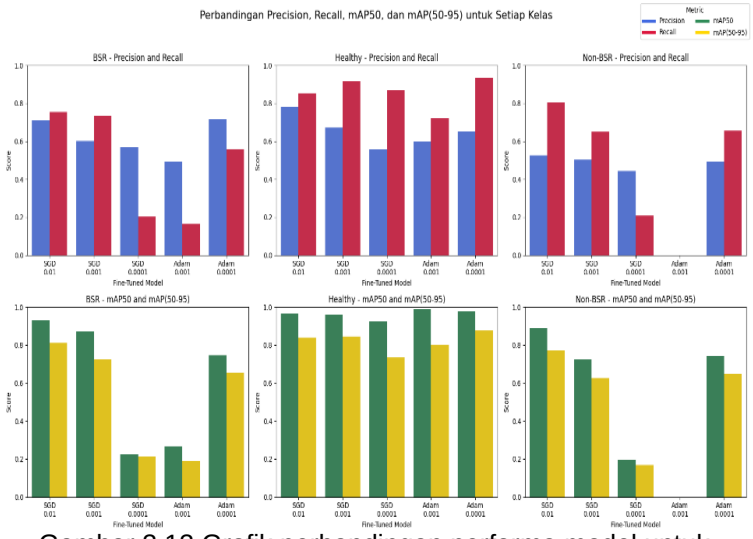
Adapun pada val loss, pada model dengan optimizer SGD, terjadi penurunan nilai loss meskipun tidak signifikan dan fluktuatif kecuali pada awal epoch. Meskipun demikian, model dengan learning rate 0.0001 memiliki nilai val loss tertinggi dibandingkan learning rate 0.01 dan 0.001.

Ini menandakan bahwa model VGG16 Faster R- CNN dengan learning rate tinggi mampu dengan cepat beradaptasi dengan data valid sehingga dapat menurunkan nilai val loss nya pada awal epoch. Adapun untuk optimizer Adam, model terlihat kesulitan mempertahankan nilai val loss nya jika learning rate nya tinggi dibandingkan jika menggunakan learning rate rendah.



Gambar 3.12 Grafik perbandingan performa model 1 untuk seluruh kelas

Adapun untuk perbandingan performa model untuk seluruh kelas, model dengan optimizer SGD terjadi penurunan performa pada semua metrics ketika nilai learning rate dinaikkan. Adapun untuk model dengan optimizer Adam, terjadi peningkatan performa yang signifikan pada semua metrics ketika learning rate diturunkan. Model dengan performa terbaik adalah pada kombinasi optimizer SGD dan learning rate 0.01 dengan precision 0.672, recall 0.805,mAP@50 0.929 dan mAP@(50-95) 0.807.



Gambar 3.13 Grafik perbandingan performa model untuk seluruh kelas

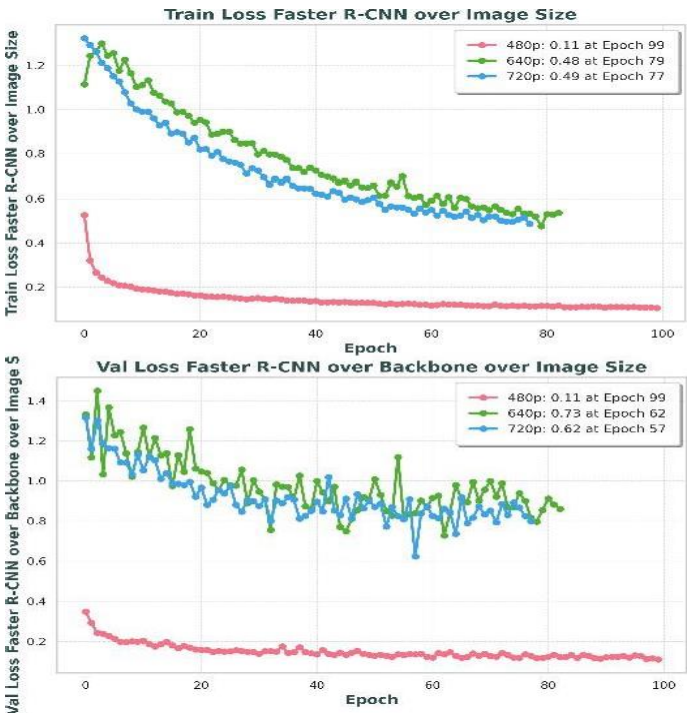
Adapun untuk perbandingan performa model pada masing-masing kelas, pada kelas BSR, terjadi penurunan performa pada semua metrics pada optimizer SGD ketika learning rate dinaikkan dan terjadi peningkatan performa pada semua metrics pada optimizer Adam ketika learning rate diturunkan. Adapun untuk model terbaik untuk kelas BSR adalah model dengan optimizer SGD dan learning rate 0.01 dengan nilai precision 0.711, recall 0.755, mAP@50 0.931 dan mAP@(50-95)0.810.
Untuk kelas Healthy, semua model mampu memberikan performa yang baik pada mAP@50 dengan nilai diatas 0.9. Akan tetapi, terjadi penurunan performa yang sama dengan kelas BSR dalam precision, sedangkan recall nya fluktuatif. Meskipun demikian, secara keseluruhan model dengan performa terbaik adalah model dengan optimizer SGD dan learning rate 0.01 dengan nilai precision 0.779, recall 0.854, mAP@50 0.966 dan mAP@(50-95) 0.839 karena mampu menyeimbangkan antara nilai precision dengan recall, dan nilai mAP@50 dan mAP@(50-95).

Untuk kelas Non-BSR, terjadi penurunan performa yang sama dengan kelas sebelumnya dalam precision, dan recall baik itu dalam optimizer SGD maupun Adam. Namun, model dengan optimizer Adam dan learning rate 0.001 tidak mampu untuk memprediksi objek pada kelas Non- BSR sama sekali. Adapun untuk model dengan performa terbaik adalah model model dengan optimizer SGD dan learning rate 0.01 dengan nilai

precision 0.526, recall 0.805, mAP@50 0.890 dan mAP@(50-95) 0.771.

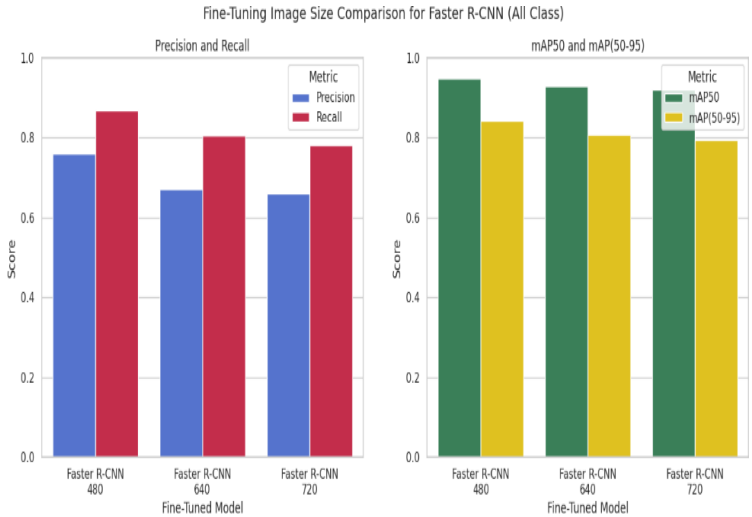
Secara keseluruhan, dapat disimpulkan bahwa model terbaik pada skenario pelatihan ini adalah model dengan optimizer SGD dan learning rate 0.01 karena mampu menurunkan nilai train loss dan val loss dengan baik, dan memberikan performa yang baik pada masing-masing kelas. Berdasarkan hasil tersebut, model Faster R-CNN membutuhkan pemilihan hyperparameter yang tepat karena dapat mempengaruhi performa model.

3.3.1.2 Fine Turning Ukuran Citra



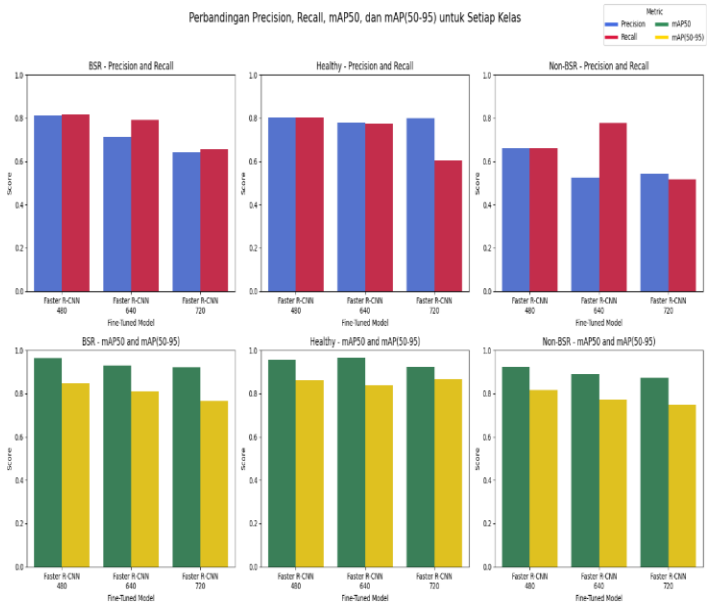
Gambar 3.14 Grafik Loss dan Train

Pada gambar x, y merupakan grafik train loss dan val loss skenario pelatihan model VGG Faster R- CNN berdasarkan ukuran input citra = [480 x 480 piksel (480p), 640 x 640 piksel (640p), 720 x 720 piksel (720p)] dengan menggunakan hyperparameter dengan performa terbaik dari skenario pelatihan sebelumnya. Dari hasil tersebut, model dengan input citra 480p memiliki nilai train loss dan val loss terendah dibandingkan dengan input citra 640p dan 720p dengan perbedaan yang signifikan. Meskipun demikian, baik train loss maupun val loss pada input citra 480p mengalami penurunan yang tidak terlalu signifikan.



Gambar 3.15 grafik performa ketiga ukuran input citra

Adapun untuk performa model dari ketiga ukuran input citra tersebut, terjadi penurunan nilai precision, recall, dan mAP@(50-95) ketika input citra ditingkatkan, namun perbedaan mAP@50 tidak signifikan antara ketiga input citra tersebut. Model dengan performa terbaik adalah model dengan input citra 480p dengan nilai precision 0.759, recall 0.868, mAP@50 0.948 dan mAP@(50-95) 0.843.

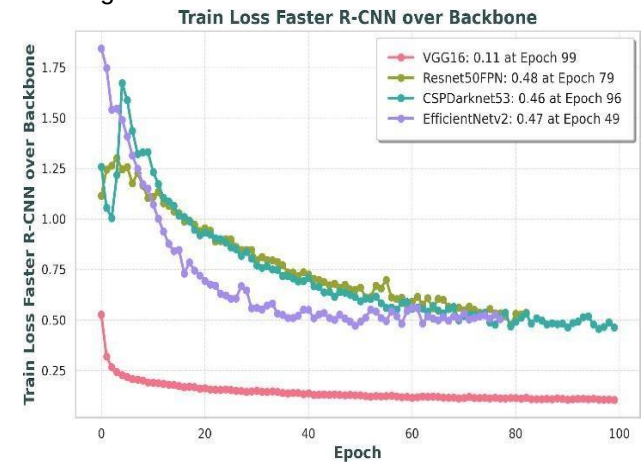


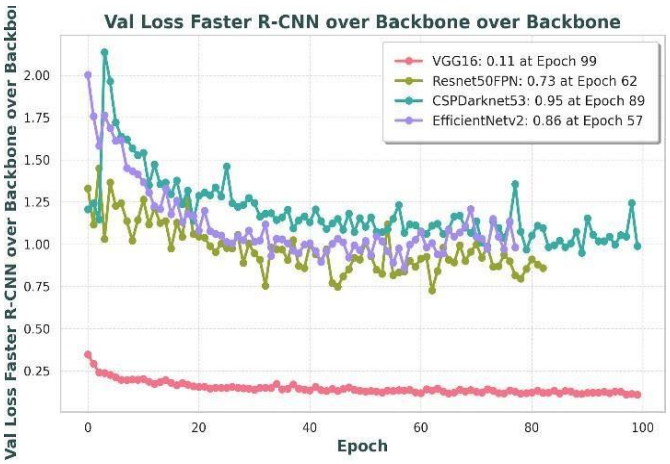
Gambar 3.16 grafik perbandingan performa model

Adapun untuk perbandingan performa model pada masing-masing kelas, model dengan input citra 480p mampu menyeimbangkan antara nilai precision dan recall pada setiap kelasnya meskipun kelas Non-BSR menjadi kelas dengan nilai precision dan recall terendah di antara ketiga kelas tersebut. Pada input citra 640p, model kesulitan untuk mempertahankan keseimbangan nilai precision dan recall pada kelas Non-BSR, sedangkan pada input citra 720p, model kesulitan untuk menyeimbangkan nilai precision dan recall pada kelas Healthy dibandingkan ketiga kelas tersebut.

Meskipun demikian, model dengan input 720p menjadi model dengan nilai precision dan recall terendah diantara ketiga ukuran input citra tersebut pada seluruh kelas. Adapun untuk mAP@50 dan mAP@(50-95), ketiga input citra tersebut tidak memberikan perbedaan signifikan pada masing-masing kelas menandakan bahwa ketiga model tersebut akuradalammemprediksi lokasi dan ukuran bounding box objek. Secara keseluruhan, dapat disimpulkan bahwa model dengan performa terbaik pada skenario pelatihan ini adalah modeldengan ukuran input citra 480p. Peningkatan input citra membuat peningkatan jumlah fitur yang akan dilatih pada model serta meningkatkan detail pada input citra, namun hal itu justru mengurangi performa model dalam mendeteksi objek dengan akurat.

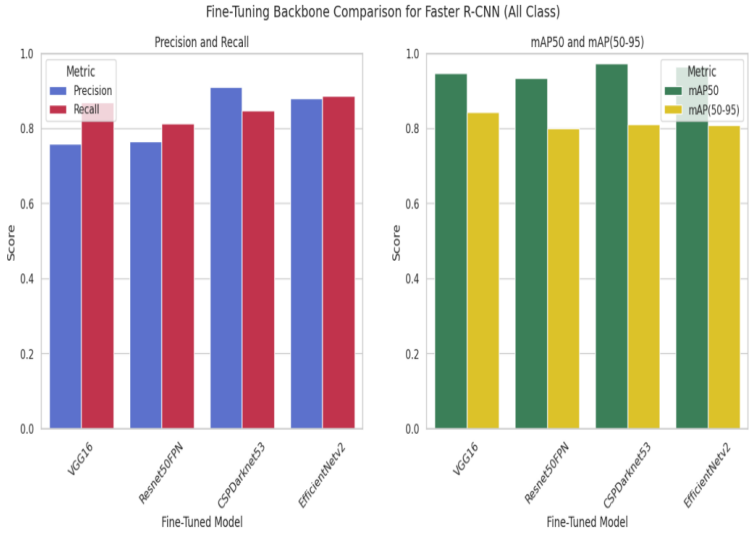
3.3.1.3 Fine Tuning Backbone





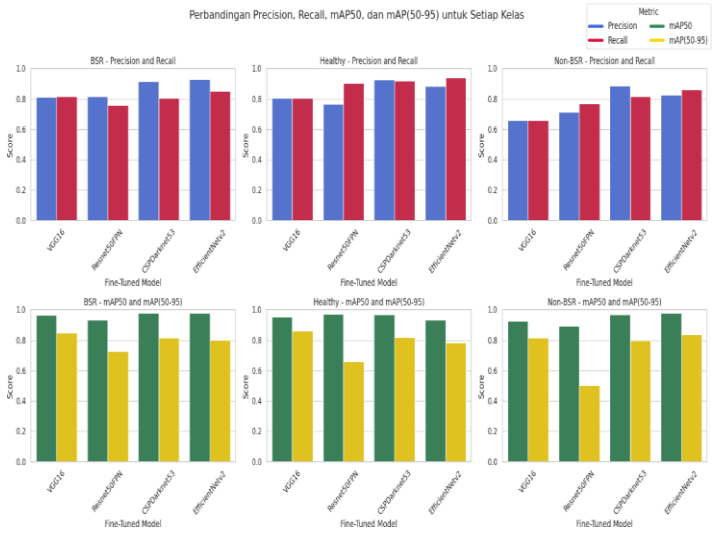
Gambar 3.17 grafik Val Loss dan Train

Pada gambar x, y merupakan grafik train loss dan val loss pada skenario pelatihan model Faster R- CNN dengan menggunakan berbagai arsitektur CNN sebagai backbone Faster R-CNN dengan menggunakan hyperparameter dan input ukuran citra dari skenario pelatihan model sebelumnya. Dari hasil pelatihan tersebut model Faster R-CNN dengan backbone VGG16 menjadi model train loss dan val loss terendah dibandingkan backbone lain yang digunakan. Pada backbone Resnet50FPN tidak memberikan perbedaan signifikan baik pada train loss maupun pada val loss. Selain itu, pada EfficientNetv2, model mampu mempelajari data latih dengan baik tapi kesulitan untuk menurunkan val loss nya sedangkan pada pada CSPDarknet53, model mengalami kesulitan pada epoch awal namun berhasil menurunkan menstabilkan nilai val loss nya.



Gambar 3.18 grafik perbandingan performa setiap Backbone

Adapun untuk perbandingan performa secara keseluruhan dari setiap backbone, tidak ada perbedaan signifikan antara nilai mAP@50 dengan nilai tertinggi pada backbone CSPDarknet53 0.972. Begitupun pada mAP@(50-95), tidak ada perbedaan signifikan diantara setiap backbone dengan nilai tertinggi pada backbone VGG16 0.843. Meskipun demikian, terdapat perbedaan yang sangat signifikan antara mAP@50 dan mAP@(50-95) sehingga semua model kesulitan untuk memprediksi lokasi dan ukuran bounding box-nya sesuai dengan ground truth-nya. Pada precision, terdapat perbedaan signifikan antara backbone CSPDarknet53 dan EfficientNetv2 dengan backbone lain, dimana kedua backbone ini mampu meminimalisir kesalahan prediksi dengan nilai precision tertinggi pada backbone CSPDarknet53 0.910. Pada recall, terjadi penurunan pada backbone Resnet50FPN dibandingkan dengan backbone lain yang mampu meningkatkan nilai recall nya dengan nilai tertinggi pada backbone EfficientNetv2 0.886.



Gambar 3.19 grafik perbandingan performa pada keseluruhan

Pada perbandingan performa model pada masing-masing kelas, pada kelas BSR, precision dan recall dari backbone CSPDarknet53 dan EfficientNetv2 masing- masing nilai precision dan recall nya adalah 0.915, 0.804 dan 0.928, 0.853. Akan tetapi, kedua model ini kesulitan untuk menyeimbangkan nilai precision dan recall nya dibanding dengan backbone VGG16 meskipun nilai precision dan recall nya lebih rendah dari kedua model tersebut yaitu berturut-turut 0.812 dan 0.817. Pada mAP@50, semua model mampu memberikan performa yang baik dengan nilai lebih dari 0.9. Pada mAP@(50-95) performa terbaik terjadi pada backbone VGG16 0.849. Ini menandakan bahwa model dengan VGG16 mampu memprediksi lokasi dan ukuran bounding box untuk kelas BSR lebih baik meskipun akurasi klasifikasinya lebih rendah dibandingkan model dengan backbone CSPDarknet53 dan EfficientNetv2. Pada kelas Healthy, mode CSPDarknet53 mampu menyeimbangkan nilai precision dan recall nya dibandingkan model lain yaitu berturut-turut 0.921 dan 0.926. Hal ini lebih baik dibandingkan EfficientNetv2 yang memiliki recall tinggi yaitu 0.942 dan precision lebih rendah yaitu 0.885. VGG16 juga mampu menyeimbangkan nilai precision dan recall nya tetapi memiliki performa yang lebih rendah dibandingkan CSPDarknet53 yaitu berturut-turut 0.804 dan 0.804. Adapun untuk mAP@50, semua model memiliki performa yang baik dengan nilai diatas dari 0.9.Sedangkan pada mAP@(50-95), performa terbaik terjadi pada backbone VGG16 0.863. Model dengan backbone CSPDarknet53 dan EfficientNetv2 memiliki performa yang lebih

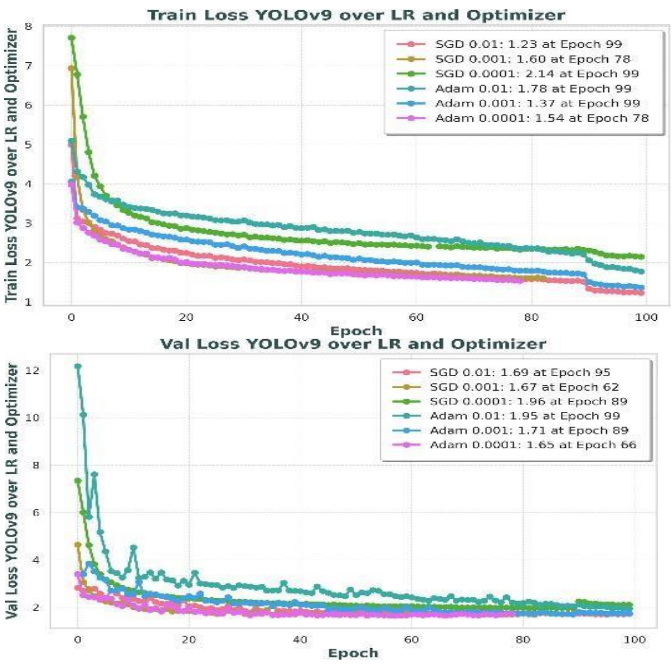
rendah yaitu berturut-turut 0.821 dan 0.785. Ini menandakan bahwa model dengan backbone VGG16 lebih baik dalam memprediksi lokasi dan ukuran bounding box untuk kelas Healthy meskipun akurasi klasifikasinya lebih rendah dibandingkan model dengan backbone CSPDarknet53 dan EfficientNetv.

Pada kelas Non-BSR, precision tertinggi dimiliki oleh backbone CSPDarknet53 dengan nilai 0.887. Sedangkan pada recall, nilai tertinggi dimiliki oleh backbone EfficientNetv2 dengan nilai 0.863. Adapun pada mAP@50, semua model memiliki performa yang baik dengan nilai diatas 0.9. Sedangkan untuk mAP@(50-95), performa terbaik dimiliki oleh model dengan backbone EfficientNetv2 dengan nilai 0.837. Ini menandakan model dengan backbone EfficientNetv2 tidak hanya memprediksi lokasi dan ukuran bounding box, tetapi juga mampu mengklasifikasikan kelas Non-BSR dengan baik.

Model dengan backbone EfficientNetv2 merupakan model dengan performa terbaik karena mampu memprediksi objek dengan baik dan menentukan lokasi dan ukuran bounding box yang sesuai dengan nilai precision 0.880, recall 0.886, mAP@50 0.965 dan mAP@(50-95) 0.808

3.3.2 YOLOv9

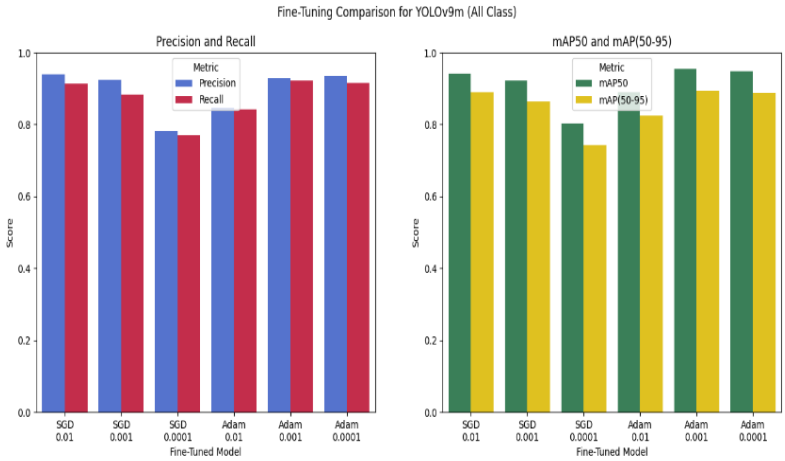
3.3.2.1 Fine Tuning Learning Rate dan Optimizer



Gambar 3.20 grafik val loss dan train

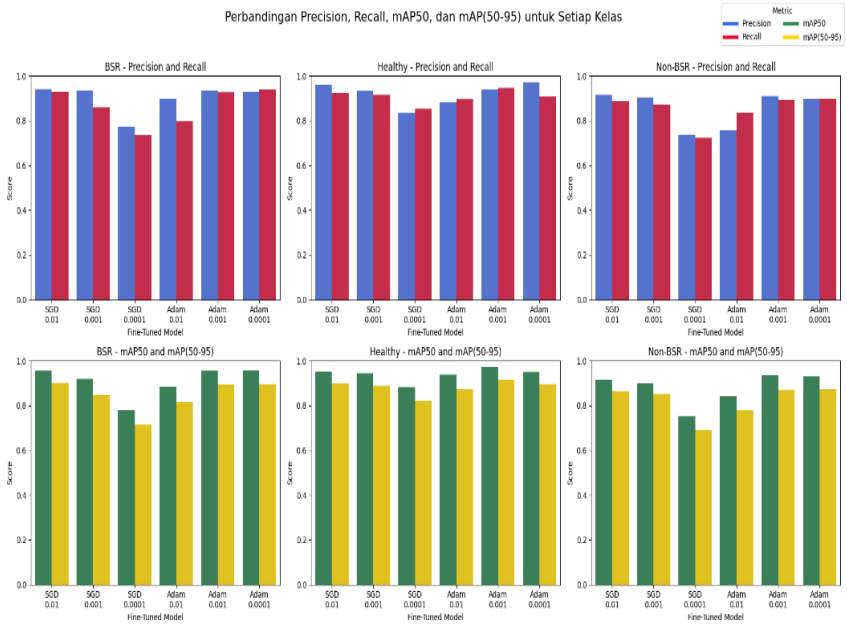
Gambar x, y merupakan grafik train loss dan val loss dari skenario pelatihan YOLOv9m dengan berbagai kombinasi hyperparameter, yaitu batch size = 16, learning rate = [0.01, 0.001, 0.0001], dan optimizer = [SGD, Adam]. Dari hasil training model, semua model menunjukkan pola grafik yang hampir dengan nilai loss yang berbeda.

Model dengan learning rate 0.0001 dan optimizer Adam menjadi model dengan nilai train loss paling rendah. Adapun pada val loss juga demikian, model mengalami penurunan nilai loss tetapi tidak signifikan



Gambar 3.21 grafik perbandingan performa keseluruhan

Adapun untuk perbandingan performa keseluruhan kelas pada model tersebut, terjadi penurunan yang signifikan pada optimizer SGD ketika learning rate diturunkan dari 0.001 ke 0.0001 pada seluruh metrics. Akan tetapi, penurunan performa ketika learning rate diturunkan dari 0.01 ke 0.001 tidak signifikan. Pada optimizer Adam, terjadi peningkatan performa ketika learning rate diturunkan dari 0.01 ke 0.001. Namun, terjadi sedikit penurunan performa pada recall, mAP@50, dan mAP@(50-95) ketika learning rate diturunkan dari 0.001 ke 0.0001. Secara keseluruhan model dengan performa terbaik adalah model dengan optimizer Adam dan learning rate 0.001 yaitu dengan nilai precision 0.928, recall 0.922, mAP@50 0.955, dan mAP@(50-95) 0.894.



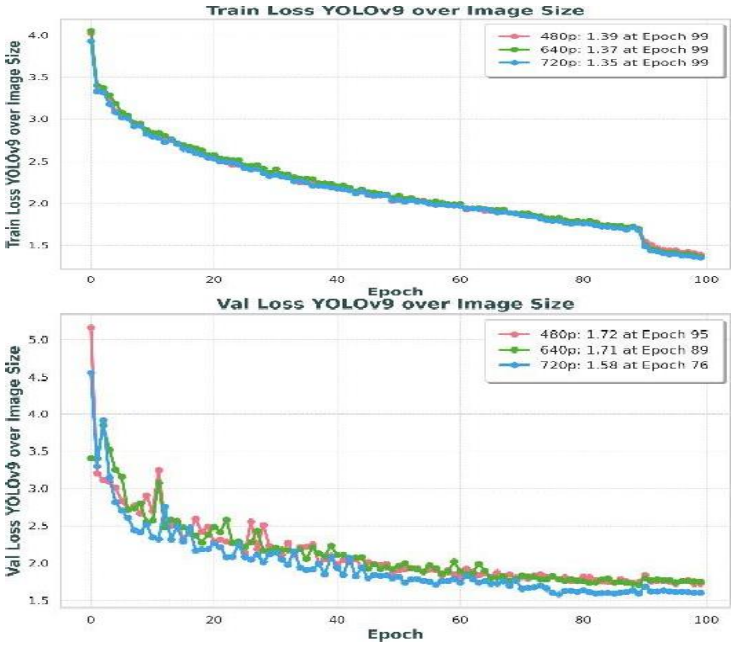
Gambar 3.22 grafik perbandingan performa pada masing – masing kelas

Adapun untuk perbandingan performa pada masing-masing kelas, baik itu pada kelas BSR, Healthy, maupun Non-BSR, menunjukkan pola yang sama, yaitu terjadi penurunan performa pada model dengan optimizer SGD ketika learning rate diturunkan. Adapun pada optimizer Adam, model mengalami peningkatan performa yang signifikan ketika learning rate diturunkan dari 0.01 ke 0.001, namun terjadi sedikit penurunan performa ketika learning rate diturunkan dari 0.001 ke 0.0001.

Secara keseluruhan, model dengan performa terbaik adalah model dengan optimizer Adam merupakan model dengan performa terbaik, yaitu precision 0.928, recall 0.922, mAP@50 0.955, dan mAP@(50-95) 0.894. Meskipun demikian, performanya tidak jauh berbeda dari model dengan optimizer SGD pada learning rate 0.01 dan 0.001 serta pada optimizer Adam pada learning rate 0.0001. Ini menandakan bahwa model tidak terlalu sensitif pada pemilihan hyperparameter

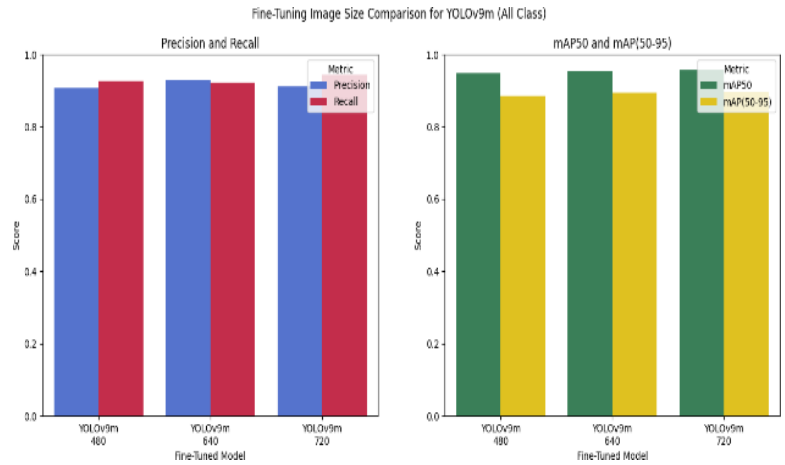
.

3.3.2.2 Fine Tuning Ukuran Citra



Gambar 3.23 grafik val loss dan train

Gambar x, y merupakan grafik train loss dan val loss dari skenario pelatihan YOLOv9m berdasarkan berbagai ukuran input citra = [480 x 480(480p), 640 x 640(640p), 720 x 720(720p)] menggunakan hyperparameter terbaik dari skenario pelatihan sebelumnya. Dari grafik, tidak ada perbedaan yang signifikan baik pada train loss maupun pada val loss.

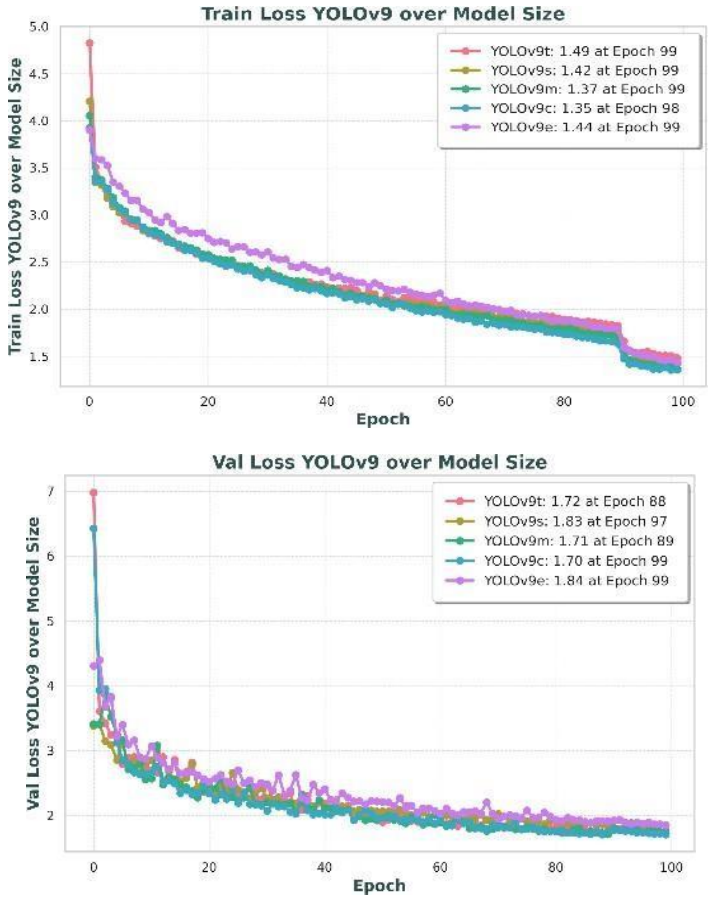


Gambar 3.24 grafik perbandingan performa keseluruhan

Pada perbandingan performa keseluruhan dari ketiga ukuran input citra, tidak terdapat perbedaan yang signifikan

pada seluruh metrics. Begitupun perbandingan pada setiap kelas, tidak ada perbedaan yang signifikan diantara ketiga input citra tersebut. Ini menandakan bahwa ukuran input citra tidak mempengaruhi performa dari model. Adapun model dengan performa terbaik adalah model dengan ukuran input citra 640p, yaitu precision 0.928, recall 0.922, mAP@50 0.955, dan mAP@(50-95) 0.894.

3.3.2.3 Fine Tuning Model Size



Gambar 3.25 grafik val loss dan train

Gambar x, y merupakan grafik train loss dan val loss pada skenario pelatihan model YOLOv9 pada berbagai model size = [tiny, small, medium, compact, extensive] dengan menggunakan hyperparameter dan ukuran input citra pada skenario pelatihan sebelumnya. Tidak ada perbedaan yang signifikan baik pada train loss maupun pada val loss terhadap berbagai model size dari YOLOv9, namun YOLOv9e (extensive) memiliki train loss dan val loss yang paling tinggi diantara model size lainnya.

Gambar 3.26 grafik perbandingan seluruh performa

Adapun pada perbandingan performa model, baik secara keseluruhan maupun masing-masing kelas, menunjukkan tidak perbedaan performa yang signifikan pada setiap model size. Ini menandakan bahwa YOLOv9 dapat dengan fleksibel digunakan sesuai kebutuhan (model size) tanpa harus mempertimbangkan penurunan performa model. Model dengan performa terbaik adalah YOLOv9m (medium) dengan nilai precision 0.928, recall 0.922, mAP@50 0.955, dan mAP@(50-95) 0.894

3.3.3 Perbandingan Faster R-CNN dan YOLO v9

Tabel 3.3 perbandingan Faster R-CNN dan YOLO v9

Model	**Hyper Parameter**	**Image size**	**Precision**	**Recall**	**mAP @50**	**mA@(50-95)**	**Parameter**	**Speed (FPS)**
VGG16 Faster R-CNN	SGD 0.01	480	0.759	0.868	0.948	0.843	43,874,207	28
Resnet 50 FPN Faster R-CNN	SGD 0.01	480	0.766	0.813	0.934	0.800	41,309,411	35
CSPDarknet53 FasterR-CNN	SGD 0.01	480	0.910	0.847	0.972	0.811	88,583,615	52
Efficient Netv2 FasterR-CNN	SGD 0.01	480	0.880	0.886	0.965	0.808	81,359955	35
YOLO v9t	Adam 0.001	640	0.920	0.924	0.943	0.880	2,005,993	132
YOLO v9s	Adam 0.001	640	0.920	0.922	0.946	0.890	7,288,533	90

YOLO v9m	Adam 0.001	640	0.928	0.922	0.955	0.894	20,160,473	57
YOLO v9c	Adam 0.001	640	0.914	0.934	0.954	0.897	25,531,529	47
YOLO v9e	Adam 0.001	640	0.895	0.933	0.948	0.891	58,147,209	20

Tabel diatas merupakan perbanding model Faster R-CNN pada berbagai arsitektur CNN sebagai backbone dengan YOLOv9 pada berbagai model size. Model Faster R-CNN dan YOLOv9 menggunakan hyperparameter dan ukuran input citra terbaik pada masing-masing skenario pelatihannya. Hasil performa ini dibandingkan berdasarkan data validasi yang digunakan. Dari tabel, model YOLO9m memiliki precision tertinggi 0.928 lebih tinggi 1.97% dari precision tertinggi pada model Faster R-CNN dengan backbone CSPDarknet53. Adapun recall, YOLOv9 memiliki nilai tertinggi yaitu 0.922 lebih tinggi 4.1% dari nilai recall tertinggi dari model Faster R-CNN dengan backbone EfficientNetv2. Adapun untuk nilai mAP@50, Faster R-CNN dengan backbone CSPDarknet53 memiliki performa tertinggi, yaitu 0.972 lebih tinggi 1.78% dari model YOLOv9m (medium). Adapun untuk nilai mAP@(50- 95), model dengan nilai tertinggi yaitu YOLOv9c (compact) 0.897 lebih tinggi 6,4% dari model Faster R-CNN dengan backbone VGG16.

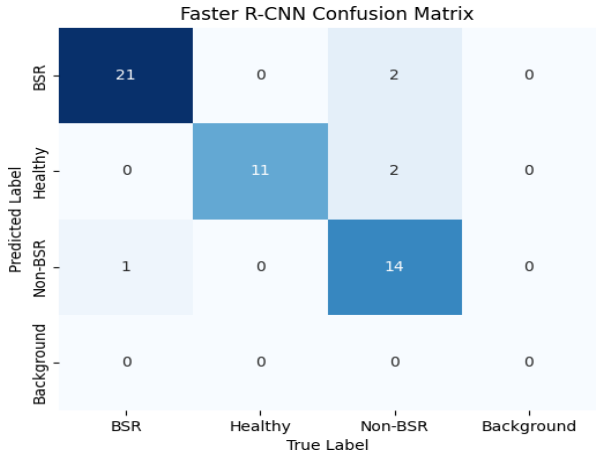
Selain itu, model YOLOv9e menggunakan parameter 34.4% lebih sedikit dari model Faster R-CNN dengan backbone CSPDarknet53 tetapi memiliki performa yang lebih baik. Model Faster R-CNN dengan backbone Resnet50FPN menggunakan paremeter 21x lebih banyak dari model YOLOv9t (tiny) namun YOLOv9t memiliki performa yang lebih baik model Faster R-CNN dengan backbone Resnet50FPN. Adapun model YOLOv9m (medium) menggunakan parameter 4x lebih sedikit dibandingkan model Faster R-CNN dengan backbone EfficientNetv2.

Dari segi kecepatan deteksi, YOLOv9t (tiny) memiliki kecepatan deteksi yang tinggi, yaitu 132 FPS dan performanya masih lebih baik dibandingkan performa seluruh backbone pada Faster R-CNN. YOLOv9e (extensive) memiliki kecepatan FPS 21 lebih rendah 60% dari Faster R- CNN dengan backbone CSPDarknet53. Ini menandakan bahwa meskipun parameternya lebih banyak, kompleksitas arsitektur Darknet53 ditambah CSPNet lebih rendah sehingga memungkinkan proses deteksi terjadi lebih cepat.

Secara keseluruhan, model YOLOv9 menjadi model yang terbaik karena mampu menyeimbangkan antara akurasi dalam klasifikasi dan prediksi lokasi serta ukuran bounding box dalam mendeteksi penyakit BSR pada kelapa sawit. Selain itu, modelnya tidak kompleks sehingga mudah untuk dilatih dan memiliki kecepatan deteksi tinggi.

3.4 Hasil Testing

Pada tahap pengujian, model yang digunakan adalah model Faster R- CNN dengan backbone EfficientNet v2 dan YOLOv9c (compact) yang masing-masing menggunakan hyperparameter dan ukuran input data yang memberikan performa terbaik pada data validasi pada masing-masing model. Adapun data yang digunakan pada pengujian sistem ini merupakan video yang diambil pada lokasi 3. Jumlah pohon kelapa sawit yang jadi bahan pengujian adalah 51 pohon dengan 22 pohon dengan kelas BSR, 11 pohon dengan kelas Healthy, dan 18 pohon dengan kelas Non-BSR.

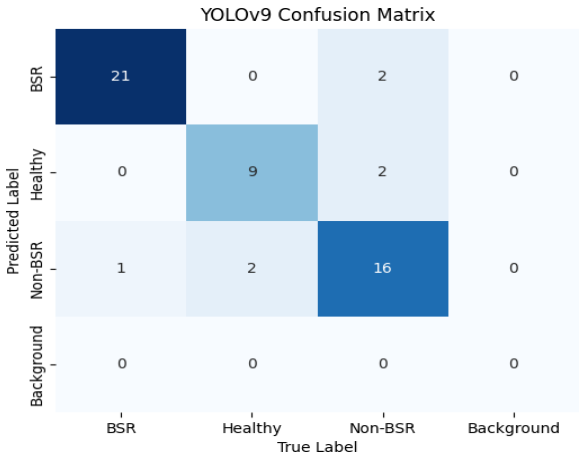


Gambar 3.27 Confusion Matrix EfficientNetv2 Faster R-CNN

Pada pengujian menggunakan model Faster R-CNN dengan backbone EfficientNetv2, model mampu mendeteksi 21 pohon yang terinfeksi BSR dari 22 pohon yang terinfeksi. Namun, model membuat 2 kesalahan dengan mendeteksinya sebagai BSR yang seharusnya dideteksi sebagai Non-BSR sehingga precision dan recall pada kelas BSR berturut- turut 0.913 dan 0.955.

Pada kelas Healthy, model mampu mendeteksi semua pohon sehat, namun membuat 2 kesalahan dengan mendeteksinya sebagai pohon dengan kelas Non- BSR sebagai Healthy sehingga precision dan recall pada kelas Healthy berturut-turut 0.846 dan 1. Pada kelas Non- BSR, model mampu mendeteksi 14 pohon yang termasuk Non-BSR dari 18 pohon yang termasuk Non-BSR, namun model membuat 1 kesalahan dengan mendeteksi pohon yang seharusnya BSR sebagai Non-BSR sehingga precision dan recall pada kelas Non-BSR beruturut-turut 0.933 dan 0.778. Secara keseluruhan, precision dan recall nya adalah 0.897 dan 0.911.

Dari hasil tersebut, model Faster R-CNN dengan backbone EfficientNetv2 mampu mendeteksi BSR dengan akurat, namun kesulitan untuk membedakan kelapa sawit sehat dengan kelapa sawit yang termasuk Non-BSR. Selain itu, model kesulitan untuk mendeteksi dan mempertahankan bounding box-nya pada objek yang berukuran sangat kecil. Ini menandakan model kesulitan mendeteksi objek yang memiliki beragam ukuran. Adapun untuk kecepatan deteksinya adalah 24 FPS.



Gambar 3.28 Confusion Matrix YOLOv9c (compact)

Pada pengujian menggunakan model YOLOv9c (compact), model mampu mendeteksi 21 pohon yang terinfeksi BSR dari 22 pohon yang terinfeksi.

Model hanya membuat satu kesalahan deteksi dengan mendeteksi pohon dengan kelas Non-BSR sebagai BSR sehingga precision dan recall pada kelas BSR berturut-turut 0.955 dan 0.955. Pada kelas Healthy, model hanya mampu mendeteksi 9 pohon dari total 11 pohon sehat. Model hanya membuat satu kesalahan deteksi dengan mendeteksi pohon dengan kelas Non-BSR sebagai Healthy sehingga precision dan recall pada kelas Healthy berturut-turut 0.9 dan 0.818. Pada Non-BSR, model mampu mendeteksi 16 pohon dari total 18 pohon dengan kelas Non-BSR. Namun, model membuat 3 kesalahan, dimana 1 pohon seharusnya dideteksi sebagai BSR, dan 2 pohon seharusnya dideteksi sebagai Healthy sehingga precision dan recall pada kelas Non- BSR berturut-turut 0.842 dan 0.889. Secara keseluruhan, precision dan recall nya adalah 0.899 dan 0.887.

Dari hasil tersebut, model YOLOv9c (compact) mampu mendeteksi BSR dengan akurat, namun kesulitan dalam membedakan pohon yang sehat dengan pohon dengan kelas Non-BSR. Selain itu, model mampu mendeteksi dan mempertahankan bounding box-nya pada objek yang berukuran sangat kecil. Ini menandakan model mampu mendeteksi objek pada berbagai variasi ukuran. Adapun untuk kecepatan deteksinya adalah 31 FPS.

Dari hasil kedua pengujian model tersebut, terdapat beberapa kesamaan yang menjadi kelemahan setiap model. 1) Model kesulitan untuk membedakan pohon kelapa sawit sehat (Healthy) dengan pohon kelapa sawit Non-BSR, hal ini terlihat pada confusion matrix kedua model yang cenderung melakukan kesalahan deteksi pada kedua kelas tersebut. Hal ini dikarenakan variasi dari pohon dengan kelas Non-BSR dan tidak memiliki pola pembeda yang signifikan dan cenderung mirip dengan pohon sehat, sehingga model kesulitan untuk membedakannya. 2) Model sensitif terhadap perubahan posisi, bentuk, orientasi

dari kelapa sawit, hal ini terlihat pada beberapa pohon, kelas model berubah-ubah selaras dengan perubahan posisi dari pohon, selain itu jika pohon kelapa sawit hanya terlihat sebagian atau ketika orientasi pengambilan gambar berubah, model juga cenderung melakukan kesalahan deteksi. Ini menandakan model tidak mampu menangani perubahan posisi, bentuk, dan orientasi dari objek yang bergerak atau ketika posisi pengambilan gambar/video berubah Penggunaan horizontal flip dan vertical flip masih belum mampu untuk menangani masalah ini sehingga masih dibutuhkan metode yang efektif mampu menangani kelemahan kedua model ini.

BAB IV
KESIMPULAN DAN SARAN

4.1 Kesimpulan

1. Implementasi Faster R-CNN dan YOLO dilakukan untuk membandingkan performa model dalam melakukan deteksi penyakit BSR pada tanaman kelapa sawit. Base model yang digunakan pada penelitian ini adalah Faster R-CNN dengan backbone VGG16 dan YOLOv9. Beberapa optimasi dilakukan untuk mencapai performa terbaik pada setiap model, yaitu optimizer, learning rate, ukuran input citra, backbone untuk model Faster R-CNN dan model size untuk YOLOv9. Pada Faster R-CNN, model terbaik adalah model dengan backbone EfficientNetv2, optimizer SGD, learning rate 0.01, batch size 10, epoch 100, momentum 0.9, weight decay 0.0005 dan ukuran input citra 480 x 480 piksel. Adapun untuk YOLOv9, model terbaik adalah YOLOv9c (compact) dengan optimizer Adam, learning rate 0.001, batch size 16, epoch 100, beta 0.937, weight decay 0.0005 dan ukuran input citra 640 x 640 piksel.
2. Pada pengujian sistem, model terbaik adalah YOLOv9c (compact) karena mampu menyeimbangkan kemampuan deteksi pada ketiga kelas dengan precision pada kelas BSR, Healthy, dan Non-BSR berturut-turut 0.955, 0.9, dan 0.842. Adapun untuk nilai recall pada kelas BSR, Healthy, dan Non- BSR berturut-turut 0.955, 0.818, dan 0.889. Hal ini berbeda dengan Faster R-CNN yang sangat baik dalam mendeteksi kelas BSR dan Healthy, namun kurang baik dalam mendeteksi kelas Non-BSR. Selain itu, model YOLOv9c juga lebih cepat dengan kecepatan deteksi 31 FPS dibandingkan Faster R- CNN dengan backbone EfficientNetv2 hanya 24 FPS

4.2 Saran

1. Meningkatkan variasi data yang berkaitan pada perubahan bentuk, posisi, ukuran, dan orientasi agar model mampu mempertahankan deteksinya dari pengaruh eksternal yang dapat mengubah bentuk, posisi, ukuran, dan orientasi objek
2. Melakukan peningkatan kemampuan model agar dapat mendeteksi objek yang memiliki kemiripan atau tidak memiliki perbedaan yang signifikan.
3. Melakukan penelitian lebih lanjut untuk dapat mengimplementasikan sistem pada platform edge computing, sehingga model dapat diaplikasikan pada kasus real time