BAB IV

HASIL DAN PEMBAHASAN

A. Requirement Analysis

1. Requirement analysis

   Pada Requirement analysis akan dilakukan pengambilan dokumentasi TBS (Tandan buah segar) di TPH (Tempat pengumpulan buah). Kemudian hasil foto TBS itu akan di akan dilakukan perhitungan menggunakan algoritma Yolov8. Pada bagian ini di lakukan identifikasi kebutuhan sistem adapun yang di amati ada 3 bagian sistem :

   a). Hardware & Sofware

   1. Hardware (*Client-Server Hosting*) Menggunakan model arsitektur client-server untuk hosting sistem. Hardware server, dapat menjalankan aplikasi web seperti platform e-commerce, sistem manajemen konten, serta dapat menggunakan hosting untuk menyimpan data secara aman dan dapat diakses dari mana saja.
   2. Software adalah serangkaian instruksi atau program yang digunakan untuk mengoperasikan komputer dan perangkat lainnya. Software ini dapat berupa sistem operasi yang mengelola sumber daya perangkat keras, aplikasi yang memungkinkan pengguna melakukan tugas tertentu, atau alat-alat lain yang mendukung pengembangan dan pemeliharaan sistem komputer. Contohnya termasuk aplikasi seperti pengolah kata, program desain grafis, dan perangkat lunak pengelola basis data.
   3. Integrasi adalah proses menggabungkan kemampuan deteksi objek berbasis algoritma YOLOv8 ke dalam aplikasi atau platform berbasis web. Integrasi ini memungkinkan deteksi objek dilakukan secara real-time atau melalui pengunggahan gambar langsung di antarmuka web, yang kemudian diproses oleh model YOLOv8 untuk mendeteksi dan menghitung objek secara otomatis. Dalam implementasinya, YOLOv8, yang dilatih untuk mendeteksi objek tertentu, dihubungkan dengan backend web melalui API atau layanan cloud. Gambar atau video yang diunggah pengguna melalui antarmuka web kemudian dikirim ke

server, di mana YOLOv8 melakukan analisis dan deteksi objek. Hasilnya, seperti jumlah objek yang terdeteksi atau lokasi objek, dikirim kembali ke frontend web dan ditampilkan kepada pengguna dalam bentuk yang mudah dipahami, seperti kotak pembatas di gambar atau laporan deteksi. Proses integrasi ini memanfaatkan berbagai teknologi web, seperti Python untuk menghubungkan YOLOv8, Flask atau Django untuk backend, dan HTML, CSS, serta JavaScript untuk antarmuka web. Hasil akhirnya adalah sebuah aplikasi web interaktif yang mampu memanfaatkan kekuatan YOLOv8 untuk mendeteksi objek secara efisien.

b). Data

Dara yang digunakan adalah 100 foto hasil dokumentasi TBS di TPH dengan hasil foto yang baik. Data dokumentasi Tandan Buah Segar (TBS) di Tempat Penumpukan Hasil (TPH) dalam bentuk foto merupakan cara modern untuk mengumpulkan dan memantau data panen sawit. Foto-foto ini diambil di lokasi TPH dan berfungsi sebagai sumber data visual yang mendokumentasikan jumlah dan kondisi tandan buah segar yang dipanen. Metode ini memanfaatkan teknologi pengolahan gambar dan kecerdasan buatan untuk mendeteksi, menghitung, dan mengklasifikasikan TBS secara otomatis

Foto yang diambil di TPH memberikan gambaran visual yang akurat tentang jumlah TBS yang ada pada waktu tertentu. Setiap gambar dapat dianalisis menggunakan algoritma pengolahan citra, seperti YOLO (*You Only Look Once*) atau model deteksi objek lainnya, untuk mengidentifikasi dan menghitung TBS. Selain itu, kualitas dan kematangan buah juga bisa dianalisis dari gambar tersebut. Keuntungan menggunakan data foto adalah kecepatan dan efisiensinya, mengurangi ketergantungan pada penghitungan manual yang memerlukan waktu lebih lama dan berisiko kesalahan.

Proses pengumpulan data dimulai dengan pengambilan foto dari TBS yang sudah dipanen dan ditumpuk di TPH. Pengambilan gambar bisa dilakukan secara manual dengan kamera atau menggunakan drone untuk cakupan area yang lebih luas. Gambar-gambar ini kemudian diunggah ke sistem yang dilengkapi dengan perangkat lunak analisis

gambar. Setelah diunggah, algoritma akan mendeteksi jumlah TBS, menganalisis ukuran, dan memvalidasi hasil panen. Foto-foto ini juga berfungsi sebagai bukti visual yang bisa dilacak kembali untuk audit atau laporan hasil panen.

c). Integrasi

Data dokumentasi Tandan Buah Segar (TBS) di Tempat Penumpukan Hasil (TPH) dalam bentuk foto adalah metode yang efisien dan akurat untuk mengumpulkan data panen sawit. Foto-foto ini diambil secara langsung di lokasi TPH dan dimanfaatkan sebagai sumber data visual, yang memungkinkan deteksi, penghitungan, serta klasifikasi TBS secara otomatis menggunakan teknologi pengolahan gambar seperti YOLO. Dengan mengandalkan gambar sebagai sumber data, proses dokumentasi menjadi lebih cepat dan minim kesalahan dibandingkan dengan penghitungan manual.

Proses pengumpulan data melalui foto melibatkan pengambilan gambar TBS di TPH menggunakan kamera atau drone. Gambar tersebut kemudian diunggah ke sistem analisis yang dilengkapi dengan algoritma pendeteksi objek, yang secara otomatis menghitung jumlah TBS dan mengidentifikasi kualitas tandan. Pendekatan ini memberikan akurasi yang lebih tinggi dalam pemantauan hasil panen, serta menyediakan bukti visual yang dapat digunakan untuk audit atau verifikasi data.

Meskipun penggunaan dokumentasi foto ini sangat efektif, beberapa tantangan tetap ada, seperti perubahan kondisi pencahayaan dan sudut pengambilan gambar yang tidak ideal. Namun, dengan pengaturan yang tepat dan penggunaan perangkat keras yang sesuai, tantangan ini dapat diatasi. Penggunaan data foto dalam pengambilan keputusan juga memberikan keuntungan signifikan, termasuk kemampuan untuk mengukur hasil panen dengan lebih baik, mengoptimalkan manajemen sumber daya, dan menyimpan data sebagai arsip digital untuk keperluan pelacakan jangka panjang.

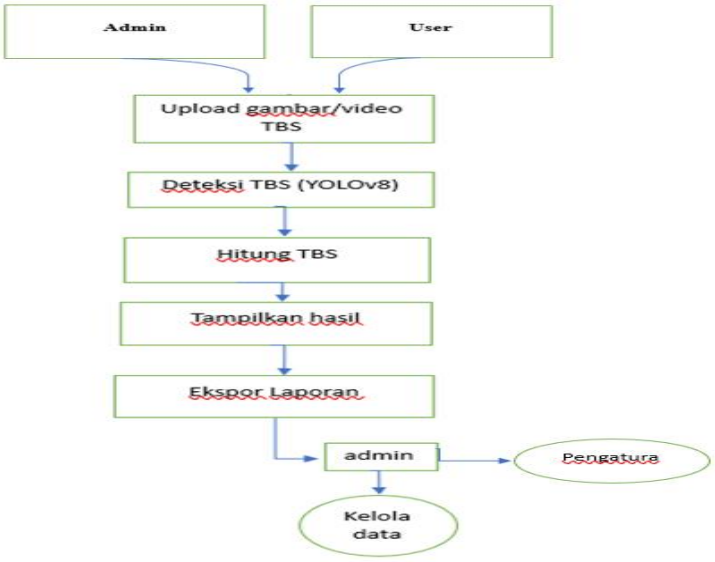
Analisi kebutuhan web ini, dimulai dengan mengambil dokumentas TBS (Tandan buah segar) di TPH (Tempat pengumpulan hasil) dalam bentuk foto untuk mengidentifikasi kebutuhan. Hasilnya menunjukkan efisiensi akses dan pengelolaan

web penghitung cacah kelapa sawit, memungkinkan pemantauan kelapa sawit yang lebih efektif. Teridentifikasi dua (2) informasi utama di TPH, yaitu buah sawit dan brondolan.

Dengan struktur ini, hubungan yang terorganisir antar tabel memungkinkan integrasi data dan analisis komprehensif terkait buah sawit sawit di TPH. pada lampiran data dokumentasi foto di TPH yang menjadi dasar dari analisis kebutuhan ini.

2. Use Case Diagram

Penelitian berjudul "Penghitung Cacah TBS dengan Menggunakan Algoritma YOLOv8" dapat dijelaskan melalui beberapa interaksi sistem dalam bentuk use case diagram. Pengguna memiliki kemampuan untukmengunggah gambar atau video tandan buah segar (TBS). Setelah itu, sistem akan menjalankan fungsi deteksi TBS dengan menggunakan algoritma YOLOv8, yang secara otomatis mendeteksi dan menghitung jumlah TBS yang ada dalam gambar atau video tersebut. Setelah proses deteksi selesai, sistem akan menampilkan hasil perhitungan TBS kepada pengguna. Pengguna kemudian dapat mengekspor hasil deteksi tersebut dalam bentuk laporan, baik dalam format PDF atau Excel. Selain itu, admin memiliki hak istimewa untuk mengelola data dalam sistem, seperti menghapus data lama atau memperbarui informasi pengguna dan sistem. Sistem juga akan menyimpan log aktivitas untuk m



Gambar 4.1 Use Case Diagram

B. Desain

Setelah tahap analisis kebutuhan, langkah berikutnya adalah menyusun desain sistem, seperti desain antarmuka pengguna (UI) dan desain database. Pada tahap ini, penulis mengonversi hasil identifikasi masalah ke dalam bentuk rancangan sistem, yang mencakup struktur data, arsitektur sistem, serta prosedur pengkodean yang akan diimplementasikan. Bahasa pemrograman yang digunakan yaitu Python.

1. Model arsitektur

Desain sistem ini memanfaatkan model arsitektur client-server. Pada sisi server, web ini diimplementasikan menggunakan fraks sebagai back-end, yang menyediakan layanan web dan berkomunikasi dengan basis data. YOLOv8 (*You Only Look Once version* 8) adalah model deep learning yang dirancang untuk deteksi objek secara real-time dengan akurasi tinggi dan latensi rendah. Arsitektur YOLOv8 lebih ringan dan efisien dibandingkan versi sebelumnya, dengan berbagai peningkatan pada komponen inti. YOLOv8 terdiri dari backbone, neck, dan head. Backbone bertanggung jawab untuk mengekstraksi fitur penting dari gambar menggunakan convolutional layers, sedangkan neck memproses fitur-fitur ini melalui layer seperti PANet (*Path Aggregation Network*) atau FPN *(Feature Pyramid Network*), memungkinkan deteksi multi-skala. Pada head, prediksi dilakukan pada berbagai skala, di mana model mendeteksi objek dan menentukan kotak pembatas serta klasifikasinya. Dengan desain ini, YOLOv8 mampu melakukan deteksi cepat dengan presisi tinggi.

Salah satu keunggulan utama YOLOv8 adalah kemampuannya untuk melakukan deteksi objek dalam satu tahap, yang menghemat waktu komputasi dibandingkan model lain yang memerlukan beberapa tahap deteksi. Model ini juga dilengkapi dengan peningkatan pada penyesuaian anchor boxes, yang memungkinkan model lebih fleksibel dalam mendeteksi objek dari berbagai ukuran dan posisi. Algoritma optimasi seperti Mosaic augmentation dan data augmentation lainnya juga digunakan untuk meningkatkan performa pada gambar yang lebih bervariasi. YOLOv8 mendukung deteksi objek dalam kondisi pencahayaan yang sulit atau dengan sudut yang kompleks, menjadikannya ideal untuk aplikasi industri, termasuk penghitungan TBS dalam perkebunan atau sistem pengawasan otomatis.

2. Deteksi objek

Dalam melakukan deteksi opjek, kita harus memastikan semua library yang dibutuhkan terinstal. Anda dapat menggunakan yolov8 melalui pip labelimg, ultralytics. Kemudian dapat memuat model YOLOv8 yang sudah dilatih (pretrained) atau melatih model yang kita ingginkan menggunakan anaconda.

Bagian menu deteteksi yang di gunakan :

a. *Backbone* YOLOv8

Bagian pertama dari desain arsitektur YOLOv8 adalah *backbone*, yang berfungsi untuk mengekstraksi fitur-fitur penting dari gambar input. Backbone pada YOLOv8 menggunakan serangkaian convolutional layers untuk mendeteksi pola, tepi, tekstur, dan fitur visual lainnya dari gambar. Proses ini dikenal sebagai feature extraction, di mana informasi dari gambar diolah secara bertahap melalui layer yang semakin dalam, menghasilkan representasi fitur yang lebih abstrak. YOLOv8 menggunakan teknik modern seperti CSP (*Cross Stage Partial networks*) dan residual connections yang meningkatkan efisiensi pemrosesan data dan mengurangi redundansi informasi, sehingga mempercepat proses deteksi objek sambil menjaga akurasi.

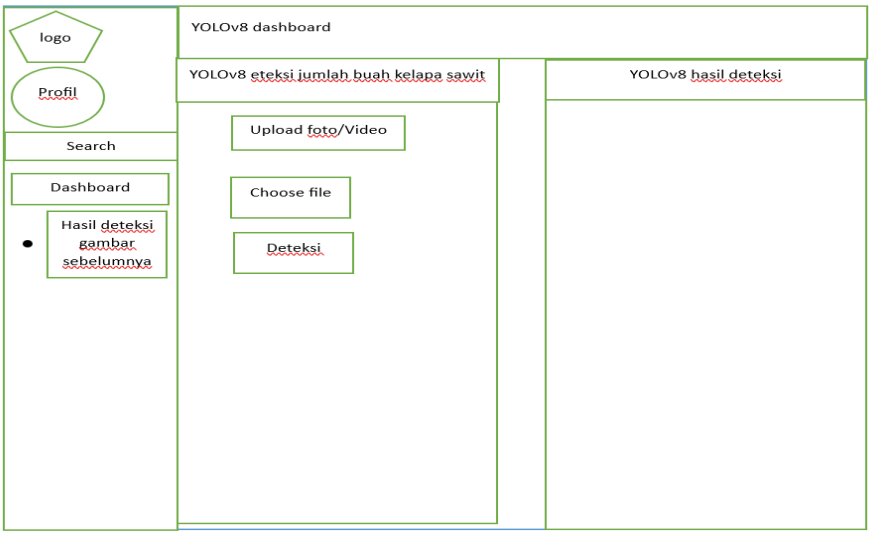
b. Neck YOLOv8

Setelah fitur diekstraksi oleh backbone, bagian neck mengambil peran penting dalam memproses fitur tersebut untuk deteksi multi-skala. Neck pada YOLOv8 menggunakan arsitektur seperti PANet (*Path Aggregation Network*) atau *Feature Pyramid Network* (FPN) yang dirancang untuk menggabungkan fitur dari berbagai level kedalaman (*shallow dan deep layers*). Dengan deteksi multi-skala ini, YOLOv8 mampu mendeteksi objek dengan ukuran yang bervariasi, baik objek kecil maupun besar, dalam satu gambar. Bagian neck ini memainkan peran vital karena memungkinkan model untuk mempertahankan informasi resolusi tinggi pada fitur penting yang membantu meningkatkan kemampuan deteksi objek, terutama dalam skenario kompleks di mana objek mungkin tumpang tindih atau tersembunyi sebagian.

c. *Head* YOLOv8

Bagian terakhir dari desain YOLOv8 adalah head, yang bertanggung jawab untuk melakukan prediksi akhir. Pada tahap ini, YOLOv8 menghasilkan kotak pembatas (*bounding boxes*) yang mengelilingi objek yang terdeteksi, serta memberikan label klasifikasi untuk setiap objek tersebut. Head juga menggunakan teknik anchor boxes, yang dirancang untuk mencocokkan berbagai ukuran dan bentuk objek dalam gambar. Prediksi dilakukan pada beberapa skala untuk memastikan bahwa objek dari berbagai ukuran dapat terdeteksi dengan akurat. Pada output ini, YOLOv8 tidak hanya memberikan koordinat kotak pembatas, tetapi juga nilai probabilitas untuk setiap kelas objek yang memungkinkan model menghasilkan hasil deteksi yang lebih presisi. Sistem ini mendukung deteksi real-time dengan latensi rendah, yang menjadikannya ideal untuk berbagai aplikasi praktis, termasuk pengawasan, penghitungan hasil panen, atau aplikasi industri lainnya. Head YOLOv8 adalah bagian akhir dari arsitektur model YOLOv8 (*You Only Look Once version* 8), yang bertanggung jawab untuk menghasilkan prediksi akhir dari deteksi objek. Bagian ini terdiri dari beberapa lapisan konvolusi dan output yang berfungsi untuk memprediksi koordinat bounding box, kelas objek, dan confidence score dari setiap objek yang terdeteksi. YOLOv8 menggunakan pendekatan *anchor-free* pada bagian head-nya, berbeda dengan versi sebelumnya yang menggunakan *anchor-based*. Hal ini membuat deteksi lebih cepat dan sederhana karena model tidak perlu mencari *anchor* yang sesuai. Selain itu, head YOLOv8 dirancang untuk memproses keluaran dari fitur yang diekstraksi oleh *backbone* dan *neck*, sehingga dapat mendeteksi objek pada berbagai skala dengan presisi yang lebih tinggi.

3. Tampilan awal



Gambar 4.2 Tampilan awal desain web

a. Bagian Logo (Pojok kiri atas):
   1) Ada tempat untuk logo.
   2) Di bawah logo, terdapat ikon profil dengan label "Profil."

   Sidebar (Panel kiri):

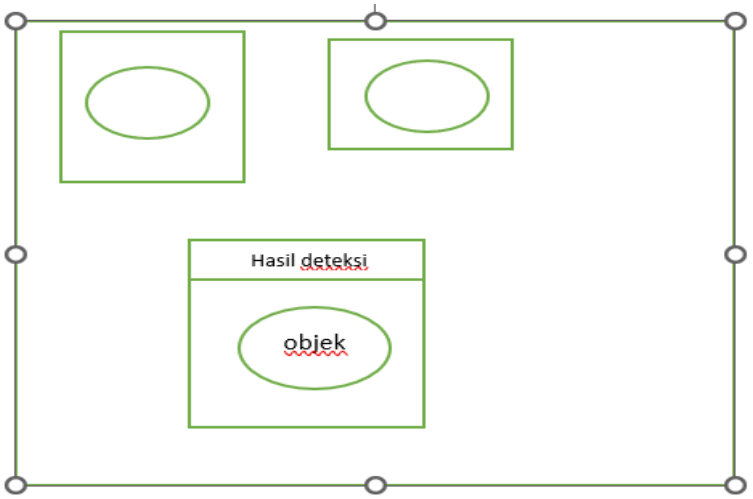
b. Terdapat kolom pencarian (Search).
   1) Ada tombol "Dashboard" untuk navigasi.
   2) Di bawah tombol dashboard, terdapat pilihan lain yang dapat diklik untuk melihat hasil deteksi gambar sebelumnya, dengan label "Hasil deteksi gambar sebelumnya."

c. Bagian Utama (Tengah):
   a) Di bagian header, tertulis "YOLOv8 dashboard."
   b) Di bawahnya terdapat dua bagian:
      a). Bagian kiri: "YOLOv8 deteksi jumlah buah kelapa sawit."
      b). Bagian kanan: "YOLOv8 hasil deteksi."
      c). Bagian kiri menampilkan tombol untuk mengunggah foto atau video, memilih file, dan ada tombol "Deteksi" untuk memulai proses deteksi.

d. Bagian Kanan:

   Area ini digunakan untuk menampilkan hasil deteksi, dengan label "YOLOv8 hasil deteksi."

1. Hasil dekteksi



Gambar 4.3 Hasil deteksi sawit menggunakan YOLOv8

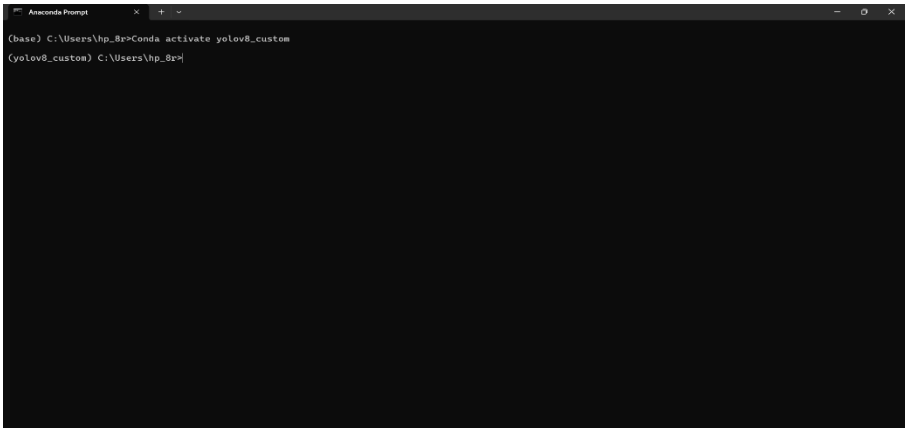
Gambar 4.3 Ini merupakan tempat di mana hasil deteksi dari objek akan ditampilkan setelah proses deteksi dilakukan**.** Hasil deteksi buah sawit menggunakan YOLOv8 menunjukkan tingkat akurasi yang tinggi dalam identifikasi Tandan Buah Segar (TBS) dan brondolan sawit di Tempat Penampungan Hasil (TPH). Sistem berbasis web ini mampu mendeteksi buah sawit dengan tingkat kepercayaan lebih dari 0,85, yang berarti sistem memiliki keandalan dalam membedakan antara buah sawit dan objek lain di sekitarnya. Confusion matrix digunakan untuk mengukur tingkat kepercayaan deteksi, menunjukkan bahwa pengembangan sistem menggunakan YOLOv8 efektif dalam meningkatkan efisiensi proses perhitungan buah sawit yang sebelumnya dilakukan secara manual.

C. Implementasi

1. Melabeling data

a. Install Aplikasi LabelImg

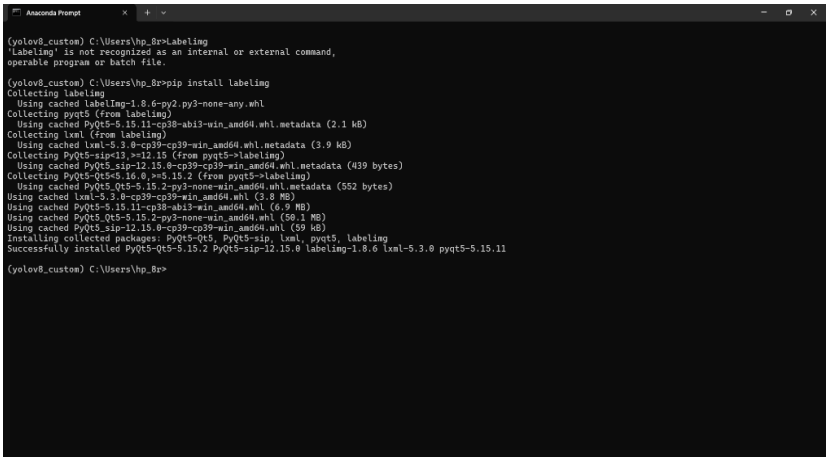
- Unduh dan instal aplikasi LabelImg di komputer Anda. LabelImg tersedia untuk berbagai platform, termasuk Windows, macOS, dan Linux.
- Setelah diunduh, instal sesuai dengan sistem operasi yang Anda gunakan



Gambar 4.4 mengaktifkan lingkungan virtual dengan perintah "Conda activate yolov8_custom"

b. Buka Aplikasi dan Muat Gambar

- Buka aplikasi LabelImg.
- Klik tombol Open atau Open Dir untuk memuat gambar atau folder yang berisi gambar yang ingin Anda beri label.



Gambar 4.5 Untuk membuka aplikasi labelimg

c. Pilih Format Label (PascalVOC atau YOLO)

- Sebelum memulai pelabelan, pilih format output file yang ingin Anda gunakan. LabelImg mendukung dua format utama:
    1. PascalVOC (XML): Format default untuk dataset VOC.
    2. YOLO (TXT): Format yang digunakan untuk deteksi objek dengan model YOLO.

d. Mulai Pelabelan Objek

- Klik tombol Create RectBox atau tekan tombol W pada keyboard untuk memulai pelabelan.
- Pilih objek pada gambar dengan menarik kotak pembatas (bounding box) di sekitar objek yang ingin dilabeli.
- Setelah kotak dibuat, sebuah jendela pop-up akan muncul untuk meminta Anda memasukkan nama label atau kelas objek (misalnya, “TBS”, “Mobil”, “Orang”, dll.).
- Klik OK setelah memberi label pada objek tersebut.

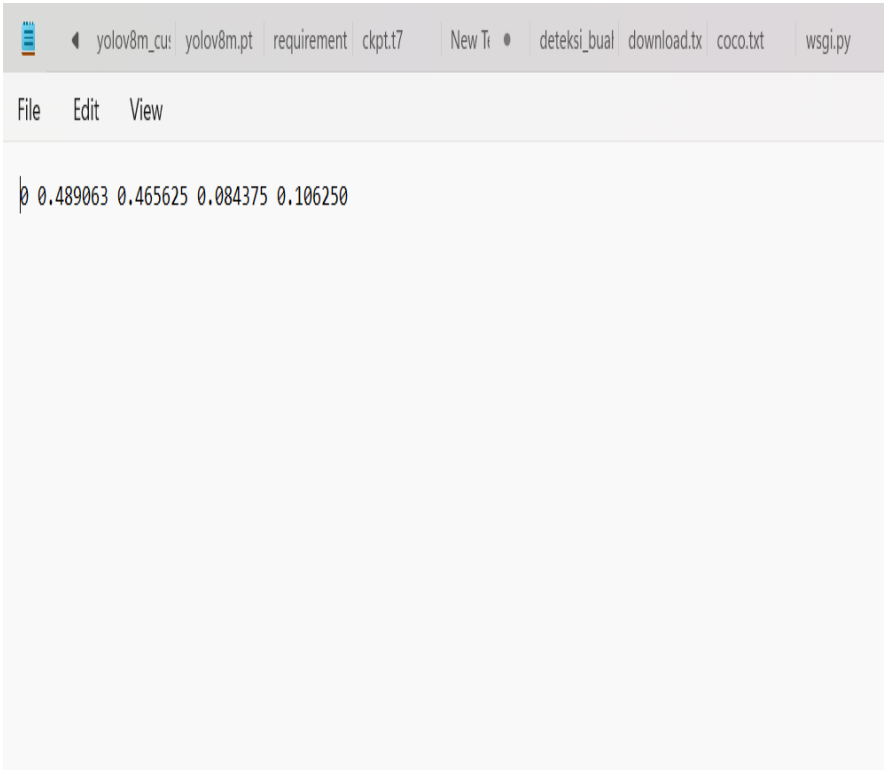
Gambar 4.6 proses untuk memberikan lebelan

e. Simpan Label

- Setelah selesai memberi label pada gambar, simpan file label tersebut dengan menekan tombol Save atau tekan Ctrl+S.
- File label akan disimpan dalam format yang dipilih sebelumnya (XML untuk PascalVOC atau TXT untuk YOLO) di folder yang sama dengan gambar.

f. Melanjutkan Pelabelan Gambar Lainnya

- Untuk melabeli gambar lainnya, gunakan tombol panah Next Image atau Prev Image untuk berpindah ke gambar berikutnya atau sebelumnya dalam folder.
- Ulangi proses pelabelan hingga semua gambar yang diinginkan selesai.

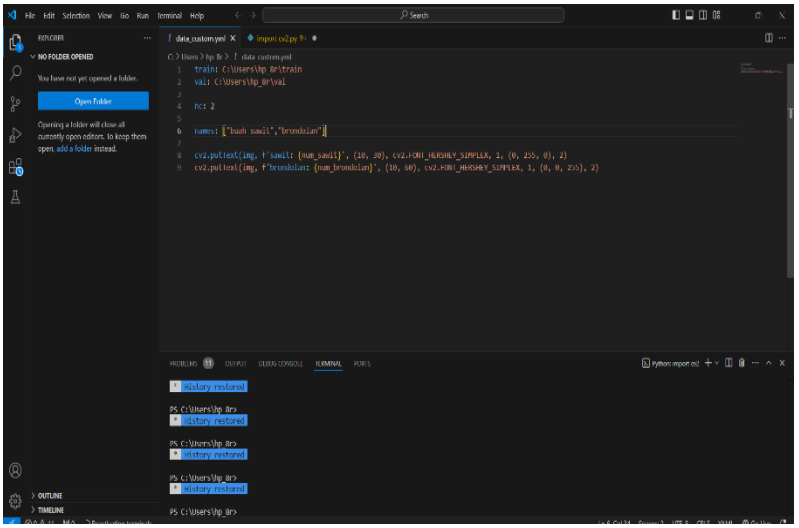


Gambar 4.7 Hasil dari pelebelan

2. Pelatihan algoritma YOLOv8

a. Persiapkan Dataset:

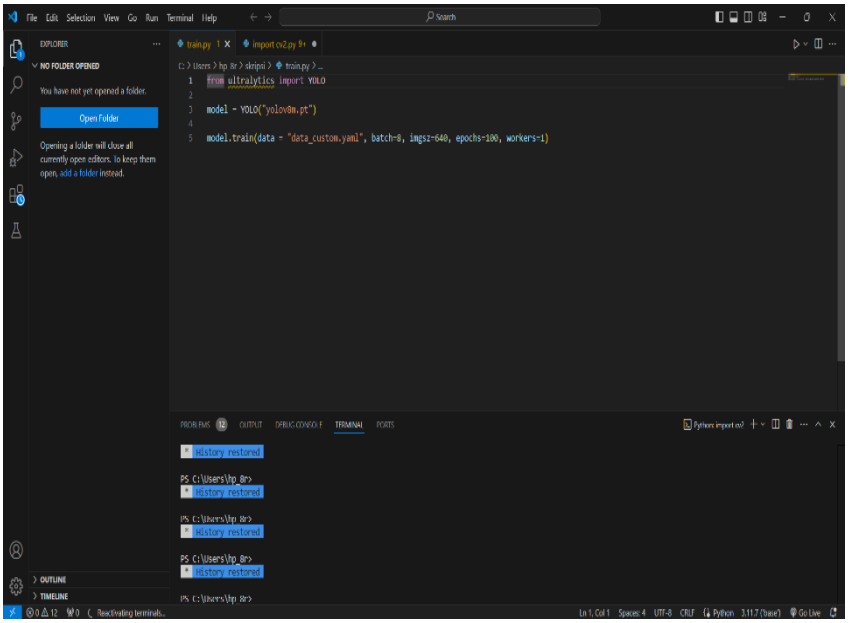
Siapkan dataset Anda dalam format yang kompatibel dengan YOLOv8. Biasanya, data pelatihan diorganisasi dalam folder dengan gambar dan file label yang sesuai.



Gambar 4.8 Persiapan dataset

b. Buat File Konfigurasi Dataset:

Buat file konfigurasi YAML yang mendefinisikan jalur ke data pelatihan dan validasi.



Gambar 4.9 Konfigurasi dataset

2. Mulai Pelatihan:

- Jalankan pelatihan model dengan perintah berikut. Gantilah data.yaml dan model.cfg sesuai dengan konfigurasi Anda: “ yolo train data=data.yaml model=yolov8n.pt epochs=50 batch=16 imgsz=640”
- Dalam perintah ini:

    1. data=data.yaml merujuk ke file konfigurasi dataset.
    2. model=yolov8n.pt adalah model YOLOv8 yang digunakan sebagai titik awal (pretrained model).
    3. epochs=50 menentukan jumlah epoch pelatihan.
    4. batch=16 adalah ukuran batch.
    5. imgsz=640 menentukan ukuran gambar input.

3. Monitoring dan Evaluasi

- Monitor Proses Pelatihan:

  Selama pelatihan, Anda dapat memantau proses melalui log yang ditampilkan di terminal. Ini biasanya mencakup metrik seperti loss dan akurasi.

- Evaluasi Model**:**

  Setelah pelatihan selesai, evaluasi model Anda dengan menggunakan dataset validasi untuk memeriksa performa. Gunakan skrip atau perintah yang disediakan oleh YOLOv8 untuk mengevaluasi hasil model.

4. Simpan dan Gunakan Model

- Simpan Model:

  Model yang telah dilatih akan disimpan di direktori output. Anda dapat menggunakan model ini untuk inferensi atau untuk melanjutkan pelatihan di masa mendatang.

- Gunakan Model untuk Inferensi:

  Setelah pelatihan selesai, gunakan model untuk inferensi pada gambar baru atau video dengan perintah berikut: “yolo predict model=yolov8n.pt

source=/path/to/image.jpg”

Untuk menjalankan pelatihan “Yolo task=detect mode=train epochs=100 data=data_custom.yml model=yolov8m.pt imgsz=640” Perintah ini digunakan untuk melatih model YOLOv8 pada tugas deteksi objek. Berikut adalah rincian setiap bagiannya:

1. yolo

- yolo adalah perintah utama yang digunakan untuk menjalankan YOLOv8. YOLOv8 dilengkapi dengan CLI (Command Line Interface), sehingga perintah ini mengeksekusi berbagai tugas yang terkait dengan model YOLOv8, seperti pelatihan, prediksi, dan evaluasi.

2. task=detect

- task=detect menunjukkan bahwa tugas yang akan dilakukan adalah deteksi objek. YOLOv8 mendukung beberapa tugas lain seperti segmentasi atau klasifikasi, tetapi dalam konteks ini, tugas utamanya adalah mendeteksi objek dalam gambar.

3. mode=train

- mode=train menunjukkan bahwa mode yang dipilih adalah pelatihan. Ini berarti model akan dilatih menggunakan dataset yang disediakan. YOLOv8 mendukung beberapa mode seperti:train: Untuk melatih model.

  a) predict: Untuk melakukan inferensi atau prediksi.

  b) val: Untuk melakukan evaluasi terhadap model.

4. epochs=100

- epochs=100 menentukan jumlah epoch untuk pelatihan, yaitu 100 epoch. Setiap epoch mewakili satu kali iterasi penuh dari pelatihan di seluruh dataset.

  a) Semakin tinggi jumlah epoch, semakin lama waktu pelatihan, tetapi biasanya ini meningkatkan kemampuan model untuk belajar lebih baik dari data.

  b) Anda dapat menyesuaikan jumlah epoch tergantung pada kebutuhan pelatihan dan seberapa baik model berkinerja selama pelatihan.

5. data=data_custom.yml

1) data=data_custom.yml merujuk pada file konfigurasi data (dalam format

YAML) yang mendefinisikan lokasi dataset pelatihan dan validasi.

a) File ini biasanya berisi informasi seperti jalur ke gambar dan label, jumlah kelas, dan nama kelas.

6. model=yolov8m.pt

model=yolov8m.pt menentukan model YOLOv8 versi "m" (medium) yang akan digunakan sebagai titik awal pelatihan. YOLOv8 menyediakan beberapa ukuran model dengan trade-off antara kecepatan dan akurasi :

a) yolov8n.pt (nano): Model kecil, lebih cepat, tetapi dengan akurasi yang lebih rendah.

b) yolov8s.pt (small): Model kecil-menengah, lebih cepat, tetapi dengan akurasi yang sedikit lebih baik.

c) yolov8m.pt (medium): Model ukuran menengah, memberikan keseimbangan antara kecepatan dan akurasi.

d) yolov8l.pt (large): Model besar, lebih lambat tetapi lebih akurat.

e) yolov8x.pt (extra-large): Model terbesar dengan akurasi tertinggi tetapi dengan waktu inferensi yang lebih lambat.

f) Penggunaan pretrained model ini memungkinkan model untuk memulai pelatihan dari titik yang sudah "terlatih" (transfer learning) sehingga lebih cepat mencapai konvergensi dan menghasilkan akurasi yang lebih baik.

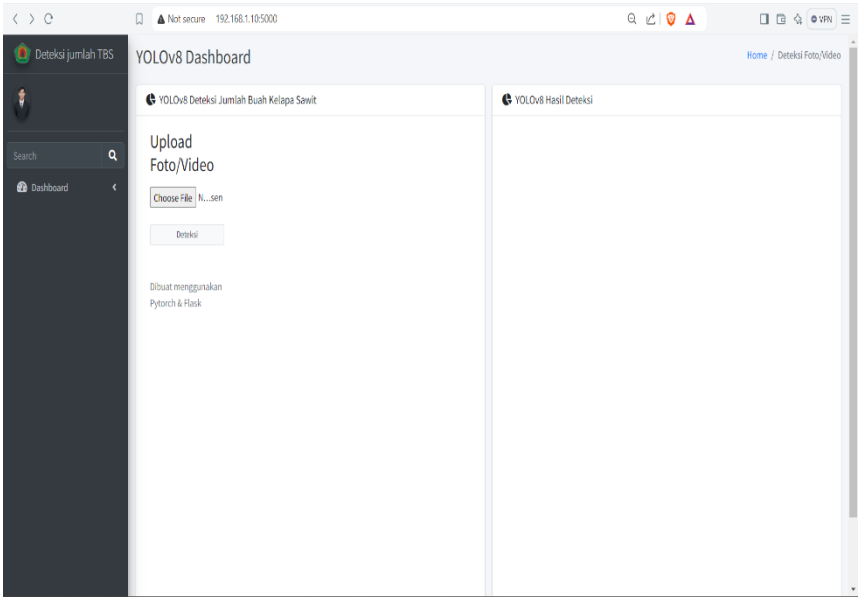
7. imgsz=640

- imgsz=640 menentukan ukuran input gambar yang digunakan dalam pelatihan. Gambar akan diskalakan ke resolusi 640x640 piksel.
    1) Ukuran gambar yang lebih besar memungkinkan deteksi lebih detail, tetapi membutuhkan lebih banyak memori dan waktu komputasi.
    2) Resolusi ini adalah ukuran standar untuk YOLOv8, dan Anda bisa menyesuaikannya tergantung pada kebutuhan dataset dan kapasitas perangkat keras (GPU).

D. Confusion matrix

1. Proses confusion matrix

a. Tampilan awal

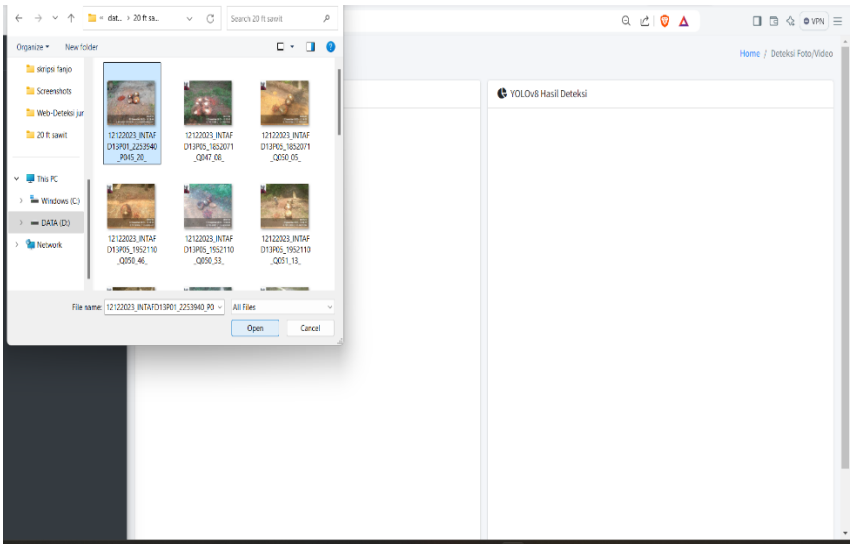


Gambar 4.10 Halaman utama dalam web

Gambar 4.10 Halaman utama web penghitung cacah TBS mengggunakan algoritma YOLOv8, ini menampilkan sebuah antarmuka pengguna (user interface) dari sebuah aplikasi web yang digunakan untuk mendeteksi jumlah Tandan Buah Segar (TBS) pada tanaman kelapa sawit. Aplikasi ini dibangun menggunakan kerangka kerja Flask, yang merupakan teknologi populer untuk pengembangan aplikasi pembelajaran mesin.

Pada bagian kiri, terdapat menu navigasi yang memungkinkan pengguna untuk beralih antara halaman beranda (dashboard) dan halaman untuk mengunggah gambar atau video. Pengguna dapat memilih file gambar atau video yang ingin diproses untuk deteksi jumlah TBS. Setelah file diunggah, aplikasi akan memproses gambar atau video tersebut menggunakan model YOLOv8, yang merupakan salah satu model deteksi objek yang populer. Hasil deteksi kemudian akan ditampilkan pada bagian kanan layar, menunjukkan lokasi dan jumlah TBS yang terdeteksi pada gambar atau video tersebut.

b. Dashboard file



Gambar 4.11 Dashboard adalah tempat file yang pernah digunakan

Gambar ini menampilkan tampilan dashboard yang kemungkinan besar digunakan untuk mengelola dan mengorganisir data gambar. Dashboard ini menunjukkan adanya beberapa folder yang berisi kumpulan gambar, yang ditandai dengan nama-nama file. Folder-folder ini terorganisir dengan baik dalam struktur direktori yang jelas, memudahkan pengguna untuk menemukan gambar yang diinginkan. Fitur pencarian juga tersedia untuk membantu mempercepat proses pencarian.

Dashboard ini juga mengindikasikan adanya proses pengolahan gambar menggunakan model deteksi objek YOLOv8. Hal ini terlihat dari adanya folder "YOLOv8 Hasil Deteksi" yang berisi hasil dari proses deteksi objek pada kumpulan gambar tersebut. Fitur-fitur lain yang terdapat pada dashboard ini antara lain kemampuan untuk memvisualisasikan hasil deteksi. Secara keseluruhan, dashboard ini memberikan tampilan yang terpusat dan interaktif untuk mengelola dan menganalisis data gambar dalam konteks proyek deteksi objek.

C. Deteksi Hasil

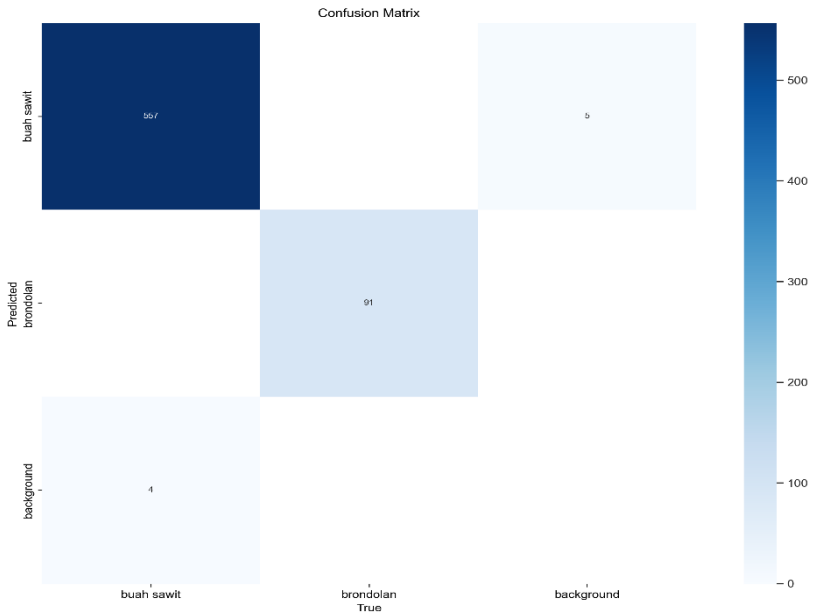
Gambar 4.12 Hasil deteksi menggunakan YOLOv8

Hasil deteksi gambar menunjukkan bahwa sistem telah berhasil mengidentifikasi beberapa objek, yaitu buah sawit dan brondolan. Objek-objek ini ditandai dengan kotak berwarna biru dan disertai dengan label serta tingkat kepercayaan (confidence score). Tingkat kepercayaan ini menunjukkan seberapa yakin sistem bahwa objek yang terdeteksi adalah benar-benar buah sawit atau brondolan. Pada gambar ini, tingkat kepercayaan untuk sebagian besar deteksi buah sawit cukup tinggi, yaitu di atas 0.85, yang mengindikasikan bahwa sistem sangat yakin dengan hasil deteksinya.

2. Hasil confusion matrix

Hasil testing dengan menggunakan Confusion matrix :

1. Confusion matrix



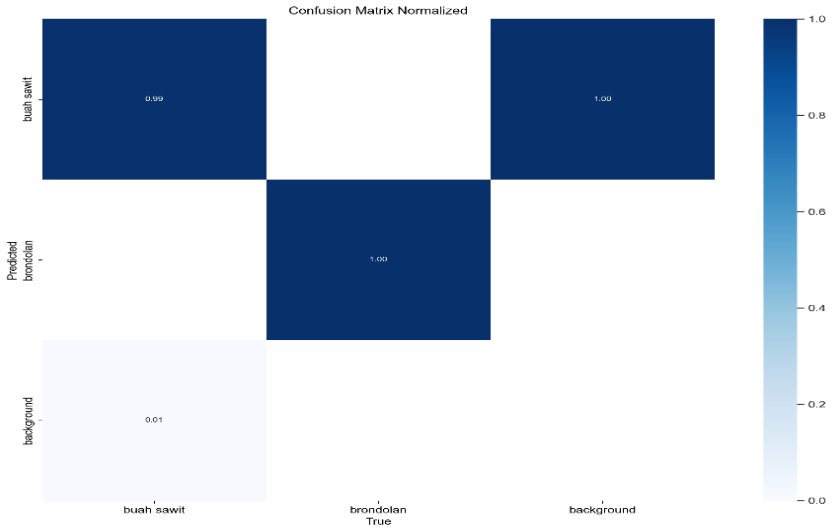
Gambar 4.13 Gambar yang menunjukan tingkat kebenaran suatu objek

a) Sumbu-sumbu:
   1) Sumbu-x (horizontal): Menunjukkan kelas aktual (true label) dari data. Dalam kasus ini, kelas aktualnya adalah "buah sawit", "brondolan", dan "background".
   2) Sumbu-y (vertikal): Menunjukkan kelas yang diprediksi (predicted label) oleh model. Kelas-kelas yang diprediksi juga sama dengan kelas aktual.

b). Angka dalam Sel: Setiap angka dalam sel mewakili jumlah (atau proporsi) data yang seharusnya berada di kelas tertentu (sumbu-x) tetapi diprediksi oleh model berada di kelas lain (sumbu-y).

c). Warna: Warna yang digunakan pada *heat map* ini menunjukkan tingkat kepercayaan model terhadap prediksinya. Warna biru tua menunjukkan tingkat kepercayaan yang tinggi, sedangkan warna putih menunjukkan tingkat kepercayaan yang rendah.

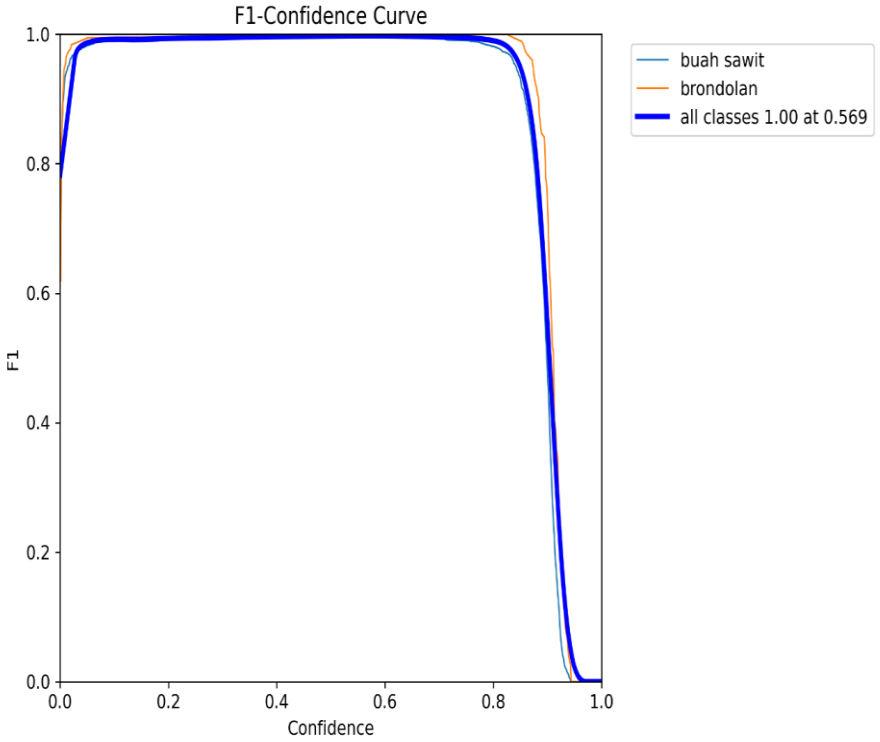
2. Confusion matrix normalized



Gambar 4.14 Confusion matrix normalized

Normalisasi matriks kebingungan adalah teknik yang digunakan untuk menskalakan nilai-nilai dalam matriks kebingungan agar lebih mudah diinterpretasikan dan dibandingkan, terutama saat menghadapi dataset yang tidak seimbang. Matriks kebingungan adalah tabel yang digunakan untuk mengevaluasi kinerja model klasifikasi dengan membandingkan klasifikasi yang diprediksi dan yang sebenarnya. Normalisasi matriks kebingungan dapat memberikan pemahaman yang lebih jelas tentang kinerja model dengan mengubah hitungan mentah menjadi proporsi atau persentase.

3. F1-Confidence curve



Gambar 4.15 . F1-Confidence curve

Gambar 4.15 F1-Confidence Curve ini menggambarkan hubungan antara tingkat kepercayaan (confidence) suatu model dalam mengklasifikasikan suatu objek (dalam hal ini, buah sawit dan brondolan) dengan nilai F1-score yang dihasilkan.

Elemen-elemen penting dalam gambar ini:

a) Sumbu X (Confidence): Menunjukkan tingkat kepercayaan model dalam membuat prediksi. Nilai 0 berarti model sama sekali tidak yakin, sedangkan nilai 1 berarti model sangat yakin dengan prediksinya.

b) Sumbu Y (F1-score): Merupakan metrik evaluasi yang mengukur akurasi dan recall model. Nilai F1-score berkisar antara 0 hingga 1, di mana nilai 1 menunjukkan kinerja sempurna.

c) Kurva:

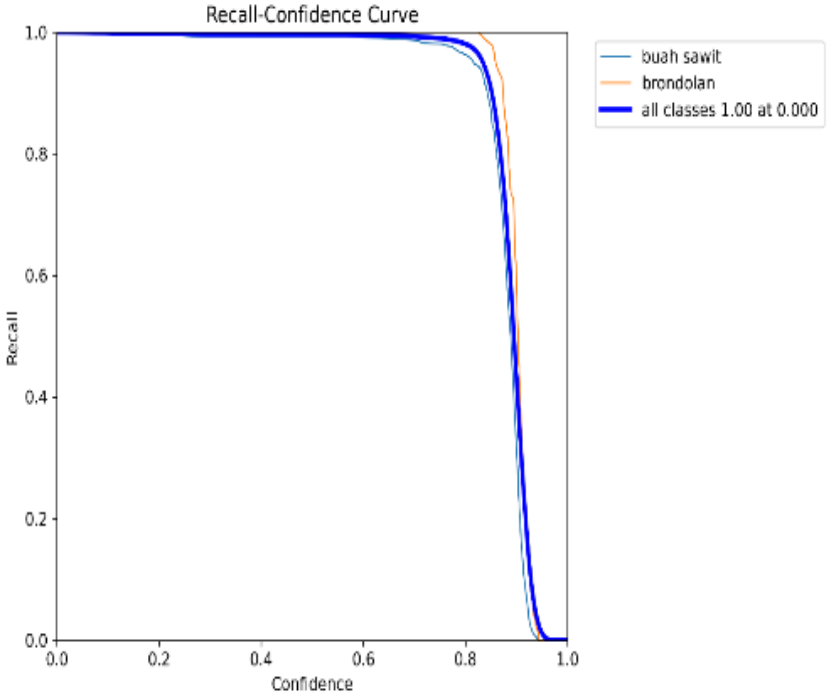
1) Buah Sawit: Kurva ini menunjukkan hubungan antara tingkat kepercayaan model dalam mengklasifikasikan buah sawit dengan nilai F1-score yang dihasilkan.

2) Brondolan: Kurva ini menunjukkan hubungan antara tingkat

kepercayaan model dalam mengklasifikasikan brondolan dengan nilai F1-score yang dihasilkan.

3) All Classes: Kurva rata-rata dari kedua kelas, menunjukkan kinerja model secara keseluruhan. Angka 1.00 menunjukkan F1-score tertinggi yang dicapai model pada tingkat kepercayaan tertentu (0.569).

4. Recall- Confidence Curve



Gambar 4.16 Recall- Confidence Curve

Gambar 4.16 menggambarkan hubungan antara tingkat kepercayaan (confidence) suatu model dalam mengklasifikasikan suatu objek (dalam hal ini, buah sawit dan brondolan) dengan nilai recall yang dihasilkan. Recall adalah metrik yang mengukur kemampuan model dalam mengidentifikasi semua contoh positif (dalam hal ini, semua buah sawit dan brondolan yang sebenarnya).

Elemen-elemen penting dalam gambar ini:

a) Sumbu X (Confidence): Sama seperti sebelumnya, sumbu ini menunjukkan tingkat kepercayaan model dalam membuat prediksi. Nilai 0 berarti model sama sekali tidak yakin, sedangkan nilai 1 berarti model sangat yakin dengan prediksinya.

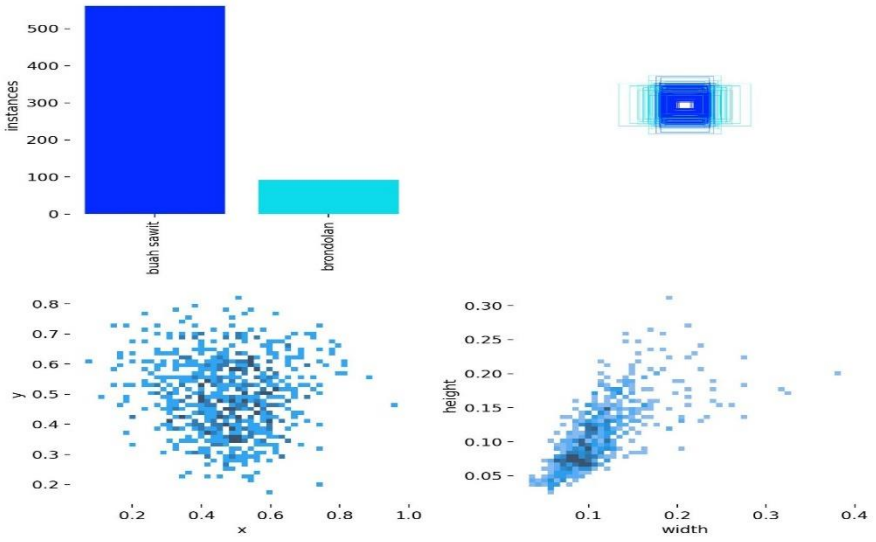
b) Sumbu Y (Recall): Menunjukkan proporsi contoh positif yang

benar-benar diidentifikasi oleh model. Nilai recall yang tinggi berarti model mampu menemukan sebagian besar contoh positif yang ada.

c) Kurva:

1) Buah Sawit: Kurva ini menunjukkan hubungan antara tingkat kepercayaan model dalam mengklasifikasikan buah sawit dengan nilai recall yang dihasilkan.
2) Brondolan: Kurva ini menunjukkan hubungan antara tingkat kepercayaan model dalam mengklasifikasikan brondolan dengan nilai recall yang dihasilkan.
3) All Classes: Kurva rata-rata dari kedua kelas, menunjukkan kinerja model secara keseluruhan. Angka 1.00 menunjukkan recall tertinggi yang dicapai model pada tingkat kepercayaan tertentu (0.000).

5. Visualisasi data



Gambar 4.17 Visualisasi data

Gambar 4. 17 ini sepertinya merupakan visualisasi data yang berkaitan dengan karakteristik fisik dari dua jenis objek, yaitu "buah sawit" dan "brondolan". Visualisasi ini terbagi menjadi beberapa plot yang saling melengkapi untuk memberikan gambaran yang lebih komprehensif.

Penjelasan Setiap Plot

a). Bar Plot:

- Sumbu-X: Kategori objek (buah sawit dan brondolan)
- Sumbu-Y: Jumlah instance atau data titik untuk setiap kategori
- Interpretasi: Plot ini menunjukkan distribusi jumlah data untuk setiap kategori. Tampaknya jumlah data untuk "buah sawit" jauh lebih banyak dibandingkan dengan "brondolan".

b). Scatter Plot dengan Kotak Batas:

- Sumbu-X: Width (lebar)
- Sumbu-Y: Height (tinggi)
- Interpretasi: Plot ini menunjukkan distribusi data berdasarkan lebar dan tinggi objek. Kotak-kotak transparan mewakili batas-batas dari beberapa kelompok data. Ini mungkin menunjukkan adanya beberapa kelompok atau cluster dalam data.

c). Histogram 2D:

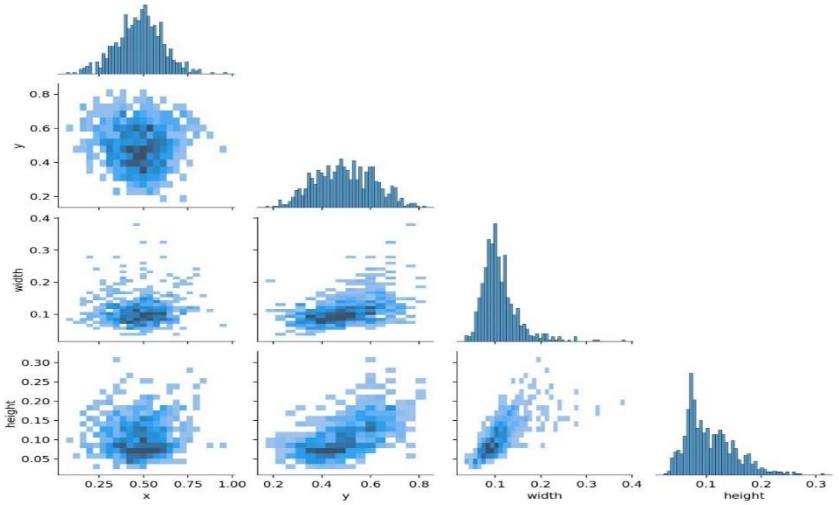
Plot Kiri:

- Sumbu-X: Nilai X (mungkin merupakan fitur lain dari objek)
- Sumbu-Y: Nilai Y (mungkin merupakan fitur lain dari objek)
- Interpretasi: Plot ini menunjukkan distribusi frekuensi gabungan dari dua fitur numerik. Tampaknya ada beberapa pola dalam distribusi data, misalnya konsentrasi data pada rentang nilai X tertentu.

Plot Kanan:

- Sumbu-X: Width (lebar)
- Sumbu-Y: Height (tinggi)
- Interpretasi: Plot ini serupa dengan plot sebelumnya, tetapi fokus pada distribusi lebar dan tinggi.

6. Plot matriks



Gambar 4.18 plot matriks

Gambar 4.18 di atas adalah sebuah plot matriks (matrix plot) yang menunjukkan distribusi dan hubungan antara tiga variabel: *width*, *height*, dan *y*.

Mari kita bahas bagian-bagian dari plot ini:

a) Diagram Histogram:

Setiap diagonal utama plot (kiri atas ke kanan bawah) menampilkan histogram untuk masing-masing variabel. Histogram ini menunjukkan distribusi frekuensi dari nilai-nilai variabel tersebut. Misalnya, histogram untuk variabel *width* menunjukkan berapa banyak data yang memiliki nilai *width* tertentu.

b) Diagram Scatter Plot:

Diagram di luar diagonal utama menunjukkan hubungan antara dua variabel. Misalnya, diagram di baris kedua kolom pertama menunjukkan hubungan antara variabel *y* dan *width*. Semakin banyak titik data yang berkumpul di suatu area, maka semakin kuat korelasi antara kedua variabel tersebut.

Dari plot ini, kita dapat mengamati beberapa hal:

a) Distribusi Variabel:

   Variabel *width* dan *height* memiliki distribusi yang cenderung normal (bentuk lonceng), sementara variabel *y* memiliki distribusi yang sedikit miring ke kanan.

b) Hubungan Antar Variabel:

   1) Terlihat adanya korelasi positif antara *width* dan *height*, artinya semakin besar nilai *width*, maka nilai *height* cenderung juga semakin besar.
   2) Hubungan antara variabel *y* dengan *width* dan *height* juga terlihat cukup kuat, namun pola hubungannya lebih kompleks dan mungkin memerlukan analisis lebih lanjut.