Logbook 4 : Percobaan Training YOLO Car Damage Detection

Nama : Albi Akhsanul hakim
NPM : 22081010194

Pada logbook keempat ini, saya melakukan percobaan pelatihan model YOLOv8n untuk mendeteksi lokasi kerusakan pada mobil. Berbeda dengan logbook sebelumnya yang menggunakan model *pretrained*, kali ini saya melatih model sendiri menggunakan dataset yang telah saya siapkan dengan format COCO, yang kemudian dikonversi ke format YOLO agar dapat digunakan dalam proses pelatihan.

Berikut adalah cuplikan kode utama yang digunakan dalam proses ini, mulai dari pembacaan anotasi, konversi format dataset, hingga proses training model menggunakan pustaka Ultralytics YOLOv8:

```
from ultralytics import YOLO
import torch
from PIL import Image
from torchvision import transforms
import matplotlib.pyplot as plt
import os
import json
from dotenv import load_dotenv

# Load environment variable
load_dotenv()
datasets_path = os.getenv('DATASET_PATH_YOLO')
datasets_annotations_path =
os.getenv('DATASET_ANNOTATIONS_PATH_YOLO')
datasets_images_path = os.getenv('DATASET_IMAGES_PATH_YOLO')
```

Selanjutnya, file JSON dengan anotasi COCO untuk data training dan validasi dibaca dan dianalisis untuk memastikan integritas data.

```
# Membaca file JSON untuk data train dan val
with open(json_path_train, "r") as f:
    data = json.load(f)
print("Jumlah gambar:", len(data["images"]))
print("Jumlah anotasi:", len(data["annotations"]))

# Proses yang sama diulang untuk data validasi
```

Kemudian dilakukan proses konversi anotasi dari format COCO ke format YOLO:

```
def convert_coco_to_yolo(coco_json_path, output_dir, image_dir):
    ...
    # Loop melalui anotasi dan menulis file .txt sesuai format YOLO
```

Setelah dataset dikonversi, dilakukan validasi terhadap format file .txt untuk memastikan tidak ada file kosong atau format yang salah:

```
# Validasi file label hasil konversi
for file in os.listdir(datasets_path + "/labels/train"):
    ...
```

Setelah dataset dinyatakan siap, proses pelatihan dimulai menggunakan model YOLOv8n:

```
model = YOLO('yolov8n.pt')
model.train(
    data="dataset.yaml",
    epochs=50,
    batch=8,
    imgsz=512,
)
```

Setelah pelatihan, dilakukan proses prediksi pada data validasi dan hasil disimpan:

```
results = model.predict(source=datasets_images_path + "/val",
save=True, conf=0.25)
```

Kode Lengkapnya: https://github.com/AlbiAkhsanul/Car-Damage-Classification-Model

Analisis:

Dari hasil pelatihan, model menunjukkan nilai evaluasi sebagai berikut:

- **Precision:** 45%
- **Recall:** 35%
- **mAP@0.5:** 25%

Hasil ini menunjukkan bahwa model telah mampu mengenali kerusakan pada mobil secara dasar, namun performanya masih belum optimal untuk implementasi nyata. Rendahnya nilai *mean Average Precision* (mAP) dapat disebabkan oleh beberapa faktor, antara lain:

1. **Jumlah dan keragaman data:** Dataset yang digunakan mungkin belum cukup representatif atau masih terlalu kecil untuk melatih model secara mendalam.
2. **Distribusi kelas:** Jika hanya sedikit gambar yang benar-benar mengandung kerusakan atau distribusi label tidak merata, maka model kesulitan mempelajari pola dengan baik.
3. **Resolusi gambar:** Ukuran gambar 512x512 sudah memadai, namun hasil dapat lebih baik jika resolusi dan kualitas gambar ditingkatkan.
4. **Epochs dan batch size:** Dengan hanya 50 epoch dan batch size 8, kemungkinan model belum benar-benar konvergen.

Meskipun begitu, eksperimen ini membuktikan bahwa proses pelatihan YOLOv8n dapat berjalan lancar dari awal hingga akhir, dan menghasilkan model deteksi yang sudah mampu mengenali objek target (dalam hal ini kerusakan mobil) pada gambar validasi. Penerapan seperti ini sangat penting untuk tahap awal proyek berbasis computer vision, dan dapat

ditingkatkan lebih lanjut melalui optimasi hyperparameter, augmentasi data, serta penambahan data latih.

Eksperimen ini membuka peluang untuk mengembangkan sistem deteksi kerusakan kendaraan secara otomatis yang dapat diimplementasikan pada aplikasi bengkel digital, asuransi kendaraan, atau robot inspeksi cerdas.