

PLASTIC BOTTLE WASTE DETECTION WITH ROBOFLOW AND PYTORCH

Prepared by

Febrina Elisabeth Sihombing	11323060
-----------------------------	----------

Ize Ronauli Sitorus	11323044
---------------------	----------

Wika Romauli Siregar	11323015
----------------------	----------

Nokatri Sitinjak	11323039
------------------	----------

BACKGROUND

Botol plastik adalah salah satu jenis sampah yang paling banyak ditemukan di lingkungan, terutama dari kemasan minuman. Setiap tahun, miliaran botol plastik diproduksi, tetapi hanya sebagian kecil yang didaur ulang, sementara sisanya mencemari tempat pembuangan, sungai, dan laut, merusak ekosistem. Masalah utama dalam pengelolaan sampah botol plastik adalah kurangnya sistem yang efektif untuk mengenali dan memilahnya. Saat ini, proses pemilahan masih sering dilakukan secara manual di fasilitas daur ulang, yang memakan waktu, tenaga, dan biaya. Selain itu, tidak semua botol plastik dapat dikenali dengan mudah oleh sistem pemilahan konvensional, terutama ketika botol-botol tersebut bercampur dengan jenis sampah lainnya.

OBJECTIVE

Tujuan proyek ini adalah mengembangkan sistem AI berbasis Roboflow dan PyTorch untuk mendeteksi sampah botol plastik untuk meningkatkan efisiensi dalam proses pemilahan sampah dengan sistem otomatis yang cepat dan akurat, sehingga mendukung proses daur ulang botol plastik menjadi produk baru seperti pakaian, furnitur, atau botol plastik baru.

Tools



PROCESS FLOW

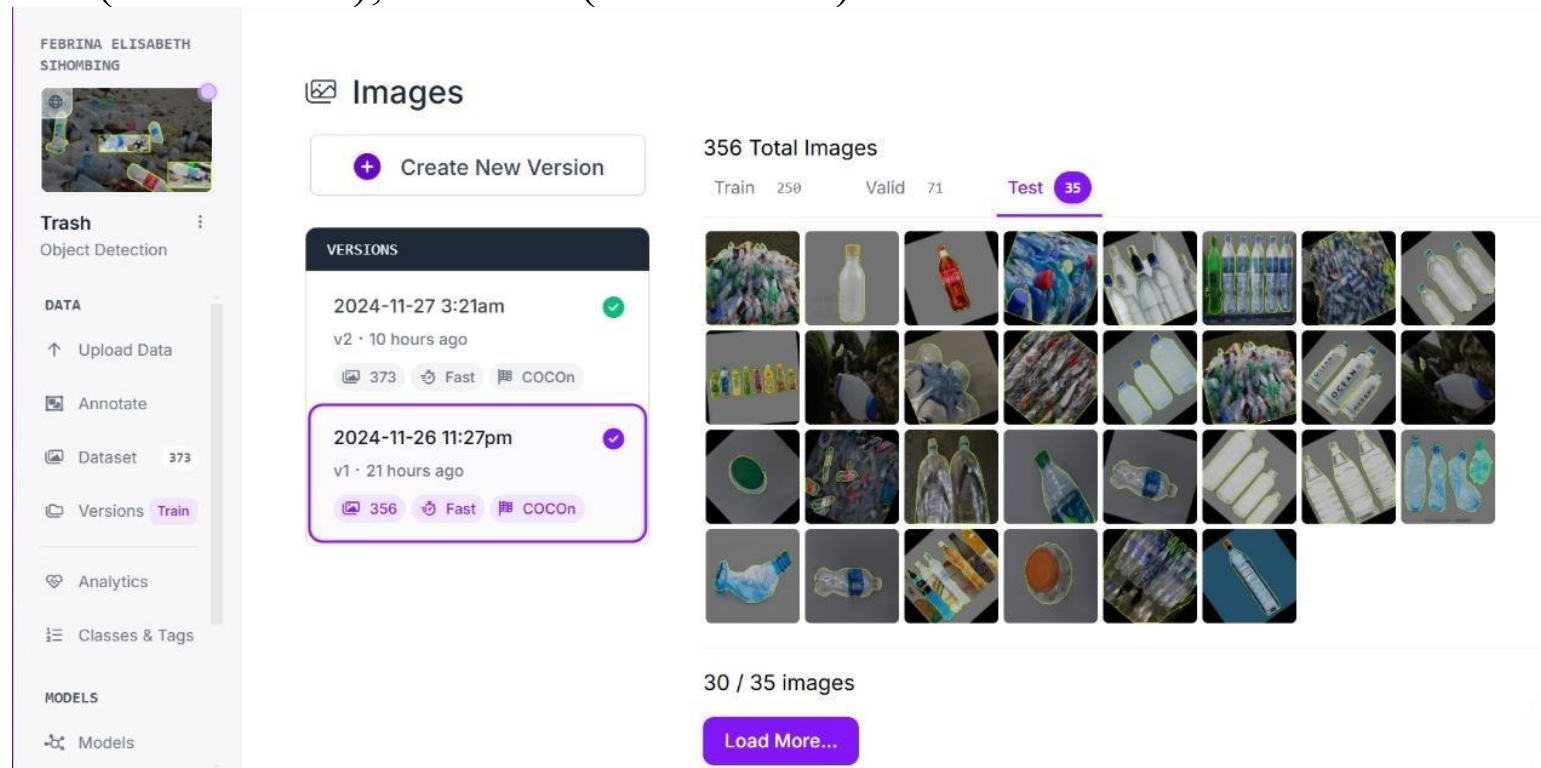
Sistem deteksi sampah botol plastik bekerja dengan menerima input berupa gambar sekumpulan sampah yang diproses menggunakan Roboflow untuk melatih model AI. Setelah model terlatih dihasilkan, PyTorch digunakan untuk mengakses model tersebut melalui API Roboflow guna mendeteksi dan mengklasifikasikan sampah yang tergolong botol plastik. Sistem ini kemudian menghasilkan hasil klasifikasi secara otomatis, menggantikan proses manual dalam menentukan sampah botol plastik.

SISTEM KERJA ROBOFLOW

Roboflow mempermudah alur kerja computer vision dengan mengelola dataset, melakukan anotasi gambar, dan menyediakan augmentasi untuk meningkatkan kualitas data. Gambar relevan diunggah dan diberi label untuk mendukung pelatihan model AI menggunakan framework seperti PyTorch. Roboflow juga membersihkan data, mengonversinya ke format yang kompatibel, dan menawarkan opsi pelatihan langsung di platform. Setelah pelatihan, model dapat dihosting dan diakses melalui API untuk deteksi objek secara real-time, dengan fitur monitoring untuk evaluasi dan pengembangan berkelanjutan.

ROBOFLOW TRAIN MODEL

Di *Roboflow* kami mempunyai dataset sebanyak 356 Gambar sampah total untuk dilatih. Yang dimana dataset ini akan displit atau dibagi menjadi 3 bagian yaitu Train Set (250Gambar), Valid Set (71 Gambar), Test Set (35 Gambar).



FEBRINA ELISABETH SIHOMBING

Images

Create New Version

356 Total Images

Train 250 Valid 71 Test 35

VERSIONS

2024-11-27 3:21am v2 · 10 hours ago 373 Fast COCO

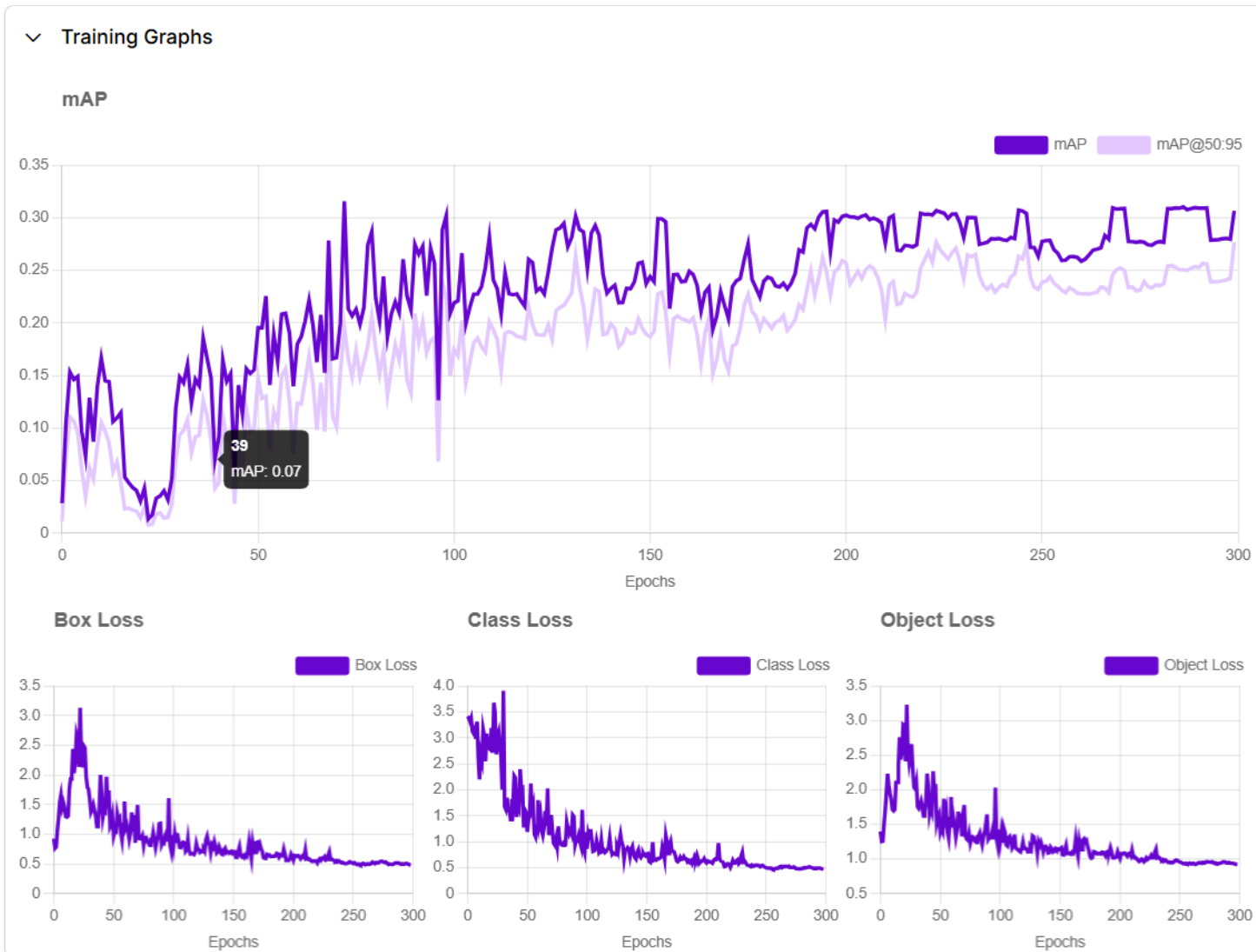
2024-11-26 11:27pm v1 · 21 hours ago 356 Fast COCO

30 / 35 images

Load More...

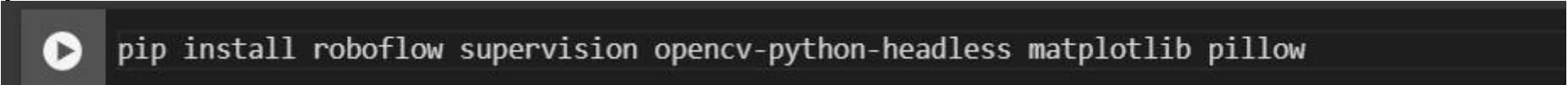
ROBOFLOW TRAIN MODEL

Berikut adalah grafik hasil pelatihan yang kami lakukan. Dapat dilihat, bahwa mAP (rata rata metric yang presisi) meningkat & Grafik Box,Class,Object Loss terlihat menurun dengan percobaan kurang lebih 120 Epoch.



PYTORCH

Dengan model yang sudah dibuat tadi, model tersebut dapat digunakan sebagai otak deplatform manapun, contohnya disini kami menggunakan Google Colab dengan PyTorch untuk melakukan prediksi.



```
pip install roboflow supervision opencv-python-headless matplotlib pillow
```

Langkah yang pertama yang kami lakukan adalah melakukan instalasi library dengan menggunakan pip.

Library yang diinstal yaitu Roboflow dan supervision yang membantu dalam manajemen dan anotasi dataset, OpenCV (headless) memungkinkan pemrosesan gambar dan penerapan algoritma visi komputer tanpa antarmuka grafis, Matplotlib digunakan untuk visualisasi data dan hasil pengolahan gambar dan Pillow menyediakan fungsi untuk manipulasi gambar.

PYTORCH

Disini kami menggunakan Google Colab dengan PyTorch untuk melakukan prediksi.

```
pip install roboflow supervision opencv-python-headless matplotlib pillow
```

Langkah yang pertama yang kami lakukan adalah melakukan instalasi library dengan menggunakan pip. Library yang diinstal yaitu Roboflow dan supervision yang membantu dalam manajemen dan anotasi dataset, OpenCV (headless) memungkinkan pemrosesan gambar dan penerapan algoritma visi komputer tanpa antarmuka grafis, Matplotlib digunakan untuk visualisasi data dan hasil pengolahan gambar dan Pillow menyediakan fungsi untuk manipulasi gambar.

```
[ ] # Import library yang akan digunakan
    from roboflow import Roboflow
    import cv2
    import matplotlib.pyplot as plt
    from google.colab import files
    import numpy as np
```

Lalu kita melakukan import library ke dalam project seperti pengelolaan dataset, pemrosesan gambar, visualisasi, dan interaksi dengan Google Colab.

```
[ ] # Inisialisasi API
    API_KEY = "FjAROWNWlSvKhj3PBvla"
    PROJECT_NAME = "trash-ftkfj"
    MODEL_VERSION = 1
```

Kemudian instalasi API roboflow untuk mengatur parameter penting yang diperlukan untuk berinteraksi dengan API Roboflow. Dengan menginisialisasi API_KEY, PROJECT_NAME, dan MODEL_VERSION, kita dapat dengan mudah mengakses dan menggunakan model pembelajaran mesin serta dataset yang terkait dalam proyek tertentu di Roboflow.

```
▶ rf = Roboflow(api_key=API_KEY)
  project = rf.workspace().project(PROJECT_NAME)
  model = project.version(MODEL_VERSION).model
```

kode ini menginisialisasi objek Roboflow dan mengakses proyek serta model spesifik berdasarkan kunci API, nama proyek, dan versi model yang telah ditentukan. Dengan Langkah ini, kita dapat dengan mudah berinteraksi dengan dataset dan model pembelajaran mesin yang telah disiapkan di Roboflow.

```
def upload_and_predict():
    """
    Meminta pengguna mengunggah gambar, memproses gambar menggunakan model,
    dan menampilkan hasil prediksi.
    """

    def upload_file():
        """Meminta pengguna untuk mengunggah file dan mengembalikan nama file."""
        print("Harap unggah gambar:")
        uploaded_files = files.upload()
        return next(iter(uploaded_files)) # Ambil nama file pertama
```

fungsi ini digunakan untuk mengupload gambar dan melakukan prediksi menggunakan model yang telah diinisialisasi sebelumnya dengan Roboflow.

```
def read_and_predict(image_path, model):
    """Membaca gambar dari path dan menghasilkan prediksi dari model."""
    print("Memproses gambar...")
    img = cv2.imread(image_path)
    predictions = model.predict(image_path, confidence=50).json()
    return img, predictions

def annotate_image(image, predictions):
```

Bagian kode ini membaca gambar yang diupload, menampilkan pesan bahwa gambar sedang diproses, dan melakukan prediksi menggunakan model Roboflow dengan tingkat kepercayaan minimum 50%, menyimpan hasilnya dalam format JSON.

```
Menambahkan anotasi pada gambar berdasarkan prediksi.
Menggambar bounding box, label, dan mask (jika ada).
"""
```

```
for pred in predictions:
    # Koordinat untuk bounding box
    x_start = int(pred['x'] - pred['width'] / 2)
    y_start = int(pred['y'] - pred['height'] / 2)
    x_end = int(pred['x'] + pred['width'] / 2)
    y_end = int(pred['y'] + pred['height'] / 2)
```

Bagian kode ini menghitung koordinat x start, y start, xend, dan y end untuk menggambar bounding box di sekitar objek yang terdeteksi berdasarkan nilai prediksi untuk posisi dan ukuran.

```
# Tambahkan mask ke gambar
if 'mask' in pred:
    mask = np.array(pred['mask'], dtype=np.uint8)
    mask_resized = cv2.resize(mask, (int(pred['width']), int(pred['height'])))
    mask_coords = np.where(mask_resized > 0)
    image[y_start + mask_coords[0], x_start + mask_coords[1]] = (0, 255, 255)
```

Bagian kode ini menggambar mask jika tersedia dengan memeriksa data mask dalam prediksi, mengonversinya menjadi array NumPy, mengubah ukurannya sesuai objek yang terdeteksi, menemukan area mask yang aktif, dan menandai area tersebut dalam gambar asli dengan warna cyan

```
# Tambahkan kotak dan label
cv2.rectangle(image, (x_start, y_start), (x_end, y_end), (0, 255, 0), 2)
label_text = f"{pred['class']} ({pred['confidence'] * 100:.1f}%)"
cv2.putText(image, label_text, (x_start, y_start - 5),
            cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255, 255, 255), 2)
```

Bagian kode ini menggambar bounding box hijau di sekitar objek yang terdeteksi, membuat string label yang mencakup nama kelas dan tingkat kepercayaan, serta menambahkan teks label tersebut ke gambar sedikit di atas bounding box.

```
def show_image_with_annotations(image):  
    """Menampilkan gambar yang telah diberikan anotasi."""  
    plt.figure(figsize=(10, 10))  
    plt.imshow(cv2.cvtColor(image, cv2.COLOR_BGR2RGB))  
    plt.axis("off")  
    plt.title("Prediksi Gambar")  
    plt.show()
```

Bagian kode ini menampilkan hasil prediksi dengan membuat figure berukuran 12x12 inci, menampilkan gambar yang telah diproses dalam format RGB, menyembunyikan sumbu untuk tampilan bersih, menambahkan judul "Hasil Prediksi", dan akhirnya menampilkan plot tersebut.

... Silakan upload gambar:

Choose Files

No file chosen

Cancel upload

Setelah itu kita dapat mengupload gambar dengan menekan button choose files maka gambar yang kita upload akan dideteksi.



WEBSITE

Seperti yang sudah kami jelaskan sebelumnya, model yang telah dibuat diawal dapat digunakan diplatform apa saja. Pengerjaan project kami, kami implementasikan model tersebut di Google Colab.

Kira kira hasilnya akan seperti gambar berikut

