НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО”
КАФЕДРА ІНФОРМАТИКИ ТА ПРОГРАМНОЇ ІНЖЕНЕРІЇ

Проектування та дослідження систем із штучним інтелектом

ЗВІТ

до модульної контрольної роботи

Виконав ІП-41мн Пашковський Євгеній Сергійович

Київ 2025

Завдання

Завдання на МКР: "Реалізувати детекцію логотипу з лаб6, або улюбленця з 5-ї лабораторної на відео".

1. Створити набір даних для НМ типу YOLO.

2. Розмножити набір даних за допомогою бібліотеки https://albumentations.ai/docs/getting_started/bounding_boxes_augmentation/

3. Створити на навчити нейронну мережу YoloV8 за допомогою бібліотеки https://www.ultralytics.com/ru/yolo

4. Опрацювати відео навченою НМ і знайдений об'єкт взяти в рамку.

Хід роботи

prepare.ipynb

```
import albumentations as A
import cv2
from glob import glob
import os
from uuid import uuid4
import random
import math
import shutil

old_images_path = "./datasets/old-images/Train/toyota"
old_bboxes_path = "./datasets/old_bboxes"

new_images_path = "./datasets/images"
new_bboxes_path = "./datasets/bboxes"
final_dataset_path = "./datasets/final"

augmentations_count = 8
```

```
transform = A.Compose([
  A.HorizontalFlip(),
  A.RandomBrightnessContrast(0.2, 0.2),
  A.Rotate((10, 10)),
  A.GaussNoise(p=0.1),
  A.Blur(p=0.3),
  A.HueSaturationValue(p=0.1),
  A.Resize(640, 640, cv2.INTER_LANCZOS4, cv2.INTER_LANCZOS4)
```

```
    ], bbox_params=A.BboxParams(format="yolo"))
```

```
    os.makedirs(new_images_path, exist_ok=True)
    os.makedirs(new_bboxes_path, exist_ok=True)

    missed_count = 0
    processed_count = 0

    for image_path in glob(os.path.join(old_images_path, "*.jpg") ):
      img = cv2.imread(image_path)

                        bbox_path          =          os.path.join(old_bboxes_path,
os.path.basename(image_path).replace(".jpg", ".txt"))

      bboxes = []
      class_labels = []
      try:
        with open(bbox_path, "r") as file:
          for line in file:
            cols = line.strip().split()
            class_labels.append(int(cols[0]))
            bboxes.append(list(map(float, cols[1:])))
      except:
        print(f"Missed {image_path} because it has no bboxes")
        missed_count += 1
        continue


      for _ in range(augmentations_count):
                        aug_result     =     transform(image=img,     bboxes=bboxes,
class_labels=class_labels)
        aug_image = aug_result["image"]
        aug_bboxes = aug_result["bboxes"]

        uuid = uuid4().hex

        cv2.imwrite(os.path.join(new_images_path, f"{uuid}.jpg"), aug_image)

        with open(os.path.join(new_bboxes_path, f"{uuid}.txt"), "w") as file:
          for bbox, class_id in zip(aug_bboxes, class_labels):
            file.write(f"{class_id} {' '.join(map(str, bbox))}\n")
```

```
        processed_count += 1

    print(f"Processed {processed_count} images, missed {missed_count} images")
```

```
    dataset_subsets_names = ["train", "val", "test"]
    dataset_subsets_ratio = [0.7, 0.2, 0.1]

    for name in dataset_subsets_names:
            os.makedirs(os.path.join(final_dataset_path, "images", name),
exist_ok=True)
            os.makedirs(os.path.join(final_dataset_path, "labels", name),
exist_ok=True)
```

```
    images = os.listdir(new_images_path)

    random.shuffle(images)

    dataset_subsets_count = list(map(lambda x: math.floor(len(images) * x),
dataset_subsets_ratio[:len(dataset_subsets_ratio) - 1]))

    dataset_subsets_count.append(len(images) - sum(dataset_subsets_count))

    assert sum(dataset_subsets_count) == len(images)

    subsets = []
    cursor = 0
    for count in dataset_subsets_count:
        subsets.append(images[cursor:cursor+count])
        cursor += count
```

```
    def copy(subset, subset_name):
        for image_name in subset:
            label_name = image_name.rsplit(".", 1)[0] + ".txt"

            src_image = os.path.join(new_images_path, image_name)
            src_label = os.path.join(new_bboxes_path, label_name)
```

```
            dst_image = os.path.join(final_dataset_path, "images", subset_name,
image_name)
            dst_label = os.path.join(final_dataset_path, "labels", subset_name,
label_name)

          shutil.copyfile(src_image, dst_image)
          if(os.path.exists(src_label)):
            shutil.copyfile(src_label, dst_label)

      for i in range(len(dataset_subsets_names)):
        subset = subsets[i]
        subset_name = dataset_subsets_names[i]
        copy(subset, subset_name)
        print(f"{subset_name}: {len(subset)}")
```

train.ipynb

```
      from ultralytics import YOLO

      dataset_config_path = "./mydataset.yaml"
      model = YOLO("yolo11n.pt")

      history = model.train(data=dataset_config_path, epochs=80, imgsz=640,
batch=16, patience=10, save=True, plots=True, device="cuda",
name="yolo11n_toyota_logo")
```

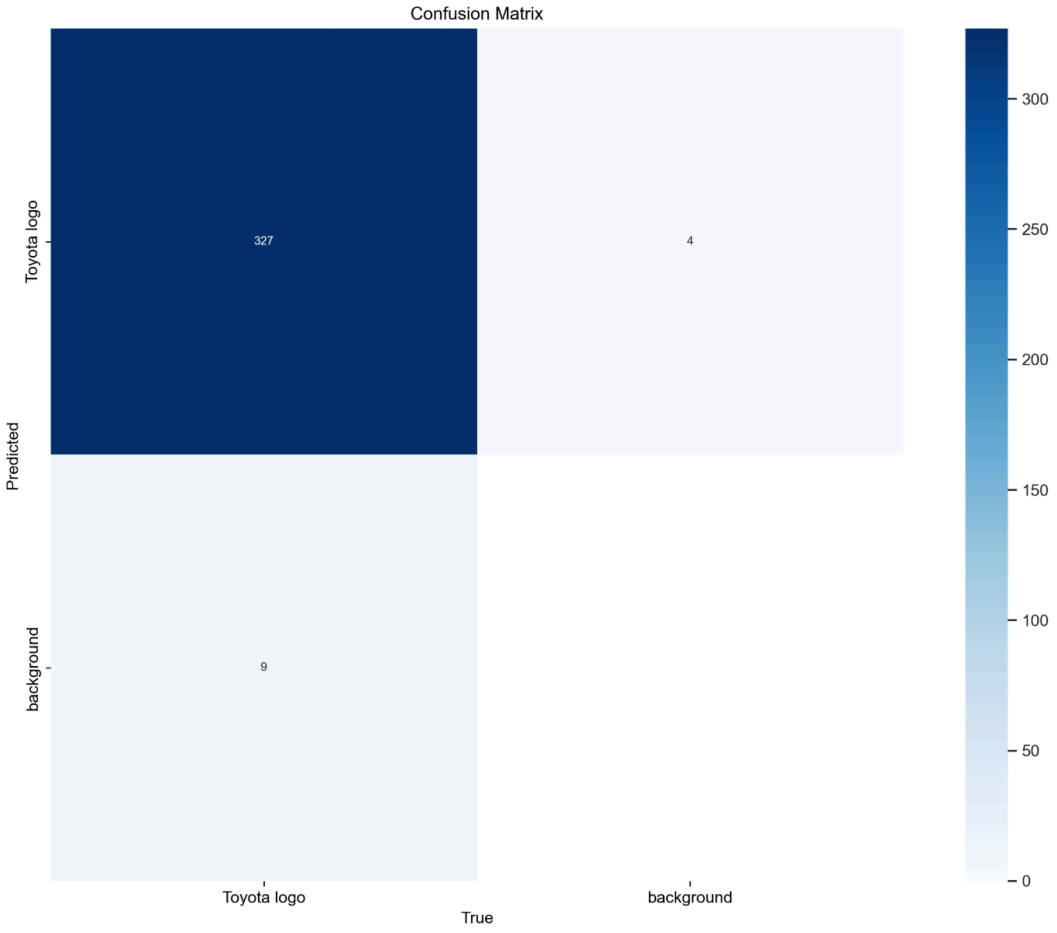
test.ipynb

```
      import cv2
      from ultralytics import YOLO
```

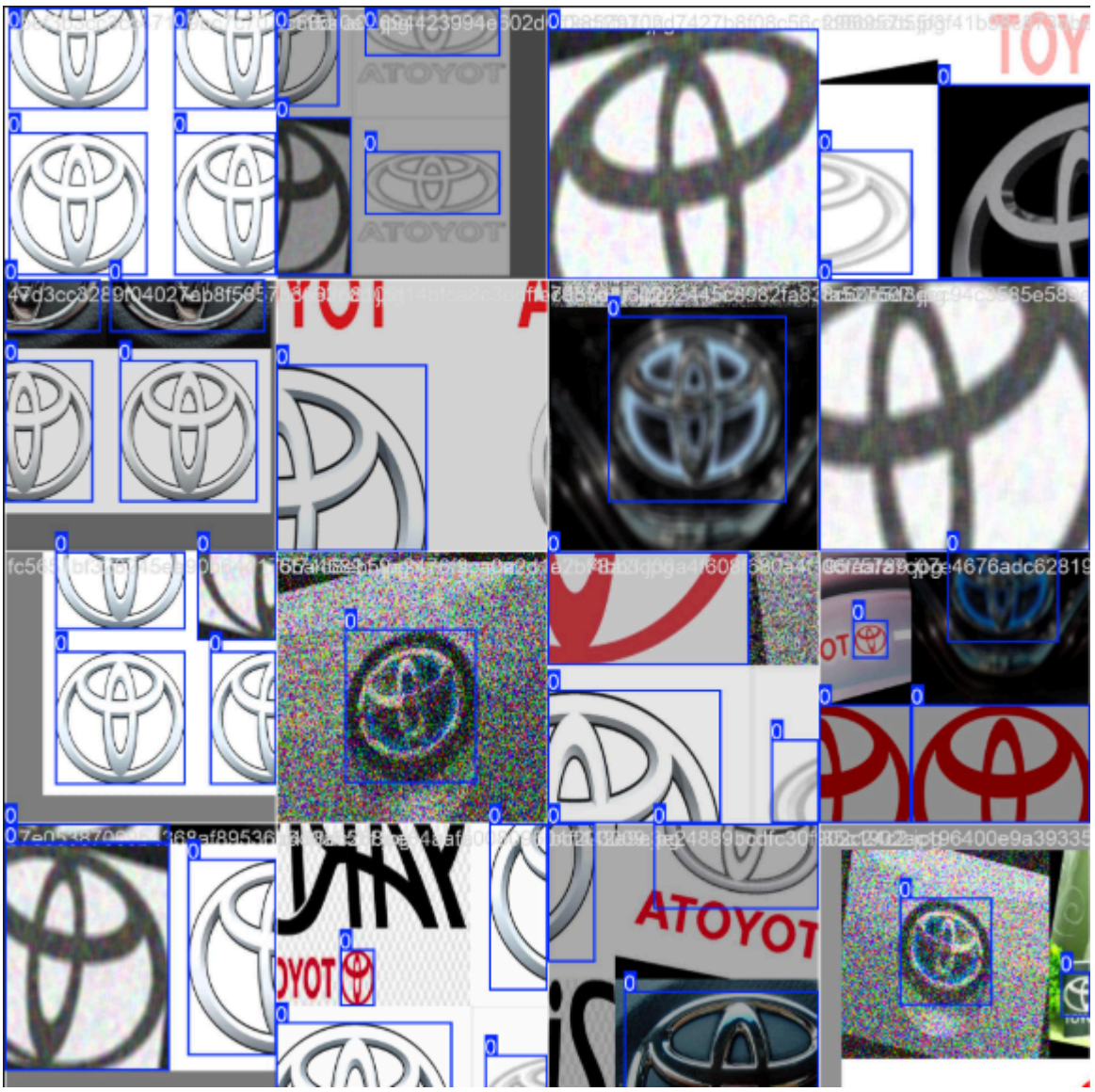
```
      model = YOLO("runs/detect/yolo11n_toyota_logo/weights/best.pt")

      video_path = "./datasets/videos/Toyota підкорила серце персидського котика.mp4"
      output_path = "./datasets/videos/result2.mp4"


      capture = cv2.VideoCapture(video_path)


      fps = capture.get(cv2.CAP_PROP_FPS)
      width = int(capture.get(cv2.CAP_PROP_FRAME_WIDTH))
```

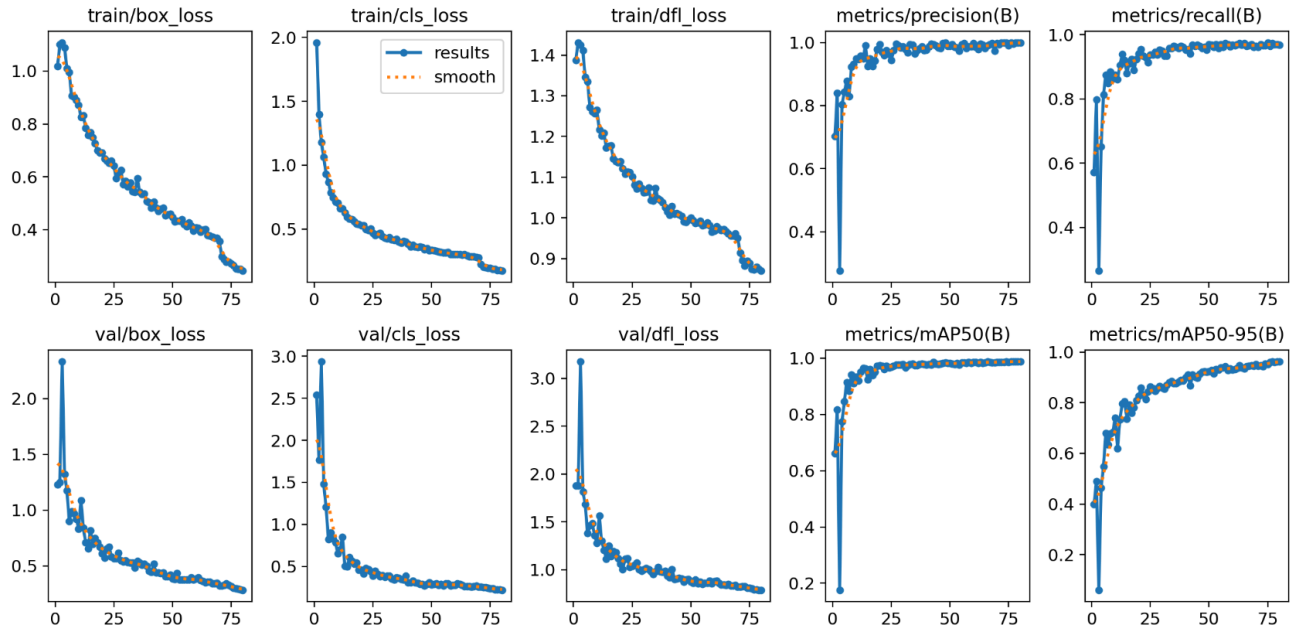
```
    height = int(capture.get(cv2.CAP_PROP_FRAME_HEIGHT))

    video_output = cv2.VideoWriter(output_path, cv2.VideoWriter_fourcc(*'mp4v'),
fps, (width, height))

    while (capture.isOpened()):
      has_more, frame = capture.read()
      if not has_more:
        break

      results = model.predict(frame, imgsz=640, conf=0.2)
      updated_frame = results[0].plot()

      cv2.imshow("Toyota logo detection", updated_frame)

      video_output.write(updated_frame)

      if (cv2.waitKey(1) & 0xFF == ord("q")):
        break

    capture.release()
    video_output.release()
    cv2.destroyAllWindows()
```

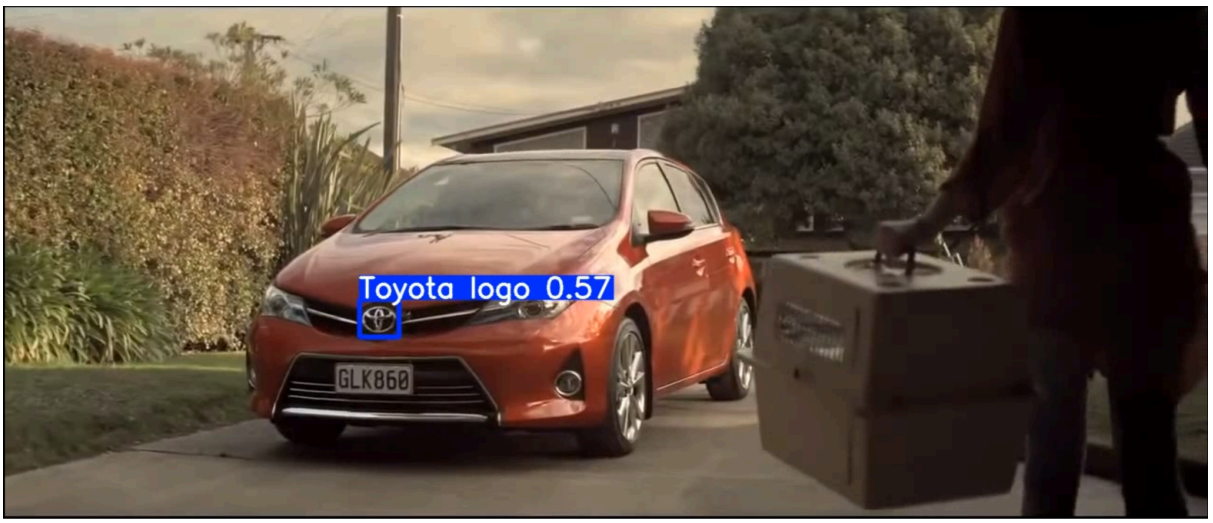
Отриманий результат


Confusion Matrix
Toyota logo
background
Predicted
327
4
9
Toyota logo
background
True
300
250
200
150
100
50
0


Toyota logo 1.0
Toyota logo 1.0
Toyota logo 1.0
Toyota logo 1.0
Toyota logo 1.0
Toyota logo 1.0
Toyota logo 1.0
Toyota logo 1.0
Toyota logo 1.0
Toyota logo 0.9
Toyota logo 1.0
Toyota logo 1.0
Toyota logo 1.0
Toyota logo 1.0
Toyota logo 1.0
Toyota logo 0.9
Toyota logo 1.0
Toyota logo 1.0
Toyota logo 1.0
Toyota logo 0.9
Toyota logo 1.0
Toyota logo 0.9
TOYOTA
Let's Go Places
TOYOTA
TOYOTA
TOYOTA




train/box_loss
train/cls_loss
results
smooth
train/dfl_loss
metrics/precision(B)
metrics/recall(B)
val/box_loss
val/cls_loss
val/dfl_loss
metrics/mAP50(B)
metrics/mAP50-95(B)



Висновки

В рамках цієї контрольної роботи було успішно побудовано, навчено модель на основі нейронної мережі YOLO для детекції певного цільового логотипу машини (toyota) на відео.