Министерство образования Республики Беларусь
Учреждение образования
«Брестский государственный технический университет»
Кафедра ИИТ

Лабораторная работа №3

По дисциплине: «Обработка изображений в ИС»

Тема: «Обучение детекторов объектов»

Выполнил:
Студент 4 курса
Группы ИИ-21
Ясюкевич В.С.
Проверила:
Крощенко А. А.

Брест 2024

Цель: осуществлять обучение НС, сконструированных на базе предобученных архитектур НС.

Ход работы:

Вариант 7

7	YOLOv8n

Осуществлять обучение нейросетевого детектора для решения задачи обнаружения дорожных знаков.

Код программы:

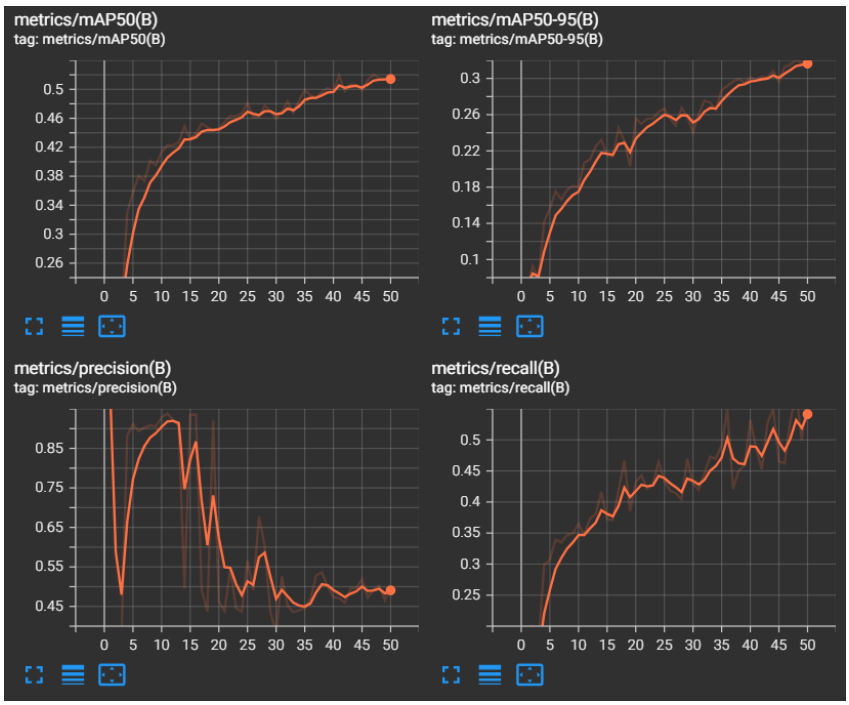
```
import os
import shutil
import pandas as pd
from sklearn.model_selection import
train_test_split
from PIL import Image

annotations_file_train =
r"C:\Users\darac\OneDrive\Рабочий стол\Новая
папка (5)\rtsd-d3-gt\main_road\train_gt.csv"
annotations_file_test =
r"C:\Users\darac\OneDrive\Рабочий стол\Новая
папка (5)\rtsd-d3-gt\main_road\test_gt.csv"
images_dir_train =
r"C:\Users\darac\OneDrive\Рабочий стол\Новая
папка (5)\rtsd-d3-frames\train"
images_dir_test =
r"C:\Users\darac\OneDrive\Рабочий стол\Новая
папка (5)\rtsd-d3-frames\test"
output_dir = "./yolo_data"
classes_file = "classes.txt"

os.makedirs(f"{output_dir}/images/train",
exist_ok=True)
os.makedirs(f"{output_dir}/images/val",
exist_ok=True)
os.makedirs(f"{output_dir}/labels/train",
exist_ok=True)
os.makedirs(f"{output_dir}/labels/val",
exist_ok=True)

df_train = pd.read_csv(annotations_file_train)
df_test = pd.read_csv(annotations_file_test)

all_classes = pd.concat([df_train["sign_class"],
df_test["sign_class"]]).unique()
class_mapping = {cls: idx for idx, cls in
enumerate(sorted(all_classes))}

with open("classes.txt", "w") as f:
    for cls in sorted(class_mapping):
        f.write(f"{cls}\n")

train_files, val_files =
train_test_split(df_train["filename"].unique(),
test_size=0.2, random_state=42)

def copy_files(file_list, subset, df,
images_dir):
    for filename in file_list:
        image_path = os.path.join(images_dir,
filename)
        if not os.path.exists(image_path):
            print(f"Изображение {filename}
отсутствует, пропускаем.")
            continue

        with Image.open(image_path) as image:
            img_width, img_height = image.size

        shutil.copy(image_path,
f"{output_dir}/images/{subset}/{filename}")

        annotation_rows = df[df["filename"] ==
filename]
        label_path = os.path.join(output_dir,
"labels", subset,
f"{os.path.splitext(filename)[0]}.txt")
        with open(label_path, "w") as f:
            for _, row in
annotation_rows.iterrows():
                class_id =
class_mapping[row["sign_class"]]
                x_center = (row["x_from"] +
row["width"] / 2) / img_width
                y_center = (row["y_from"] +
row["height"] / 2) / img_height
                bbox_width = row["width"] /
img_width
                bbox_height = row["height"] /
img_height
                f.write(f"{class_id}
{x_center:.6f} {y_center:.6f} {bbox_width:.6f}
{bbox_height:.6f}\n")
copy_files(train_files, "train", df_train,
images_dir_train)
copy_files(val_files, "val", df_train,
images_dir_train)
copy_files(df_test["filename"].unique(), "val",
df_test, images_dir_test)
data_yaml_content = f"""
train: {output_dir}/images/train
val: {output_dir}/images/val
nc: {len(class_mapping)}  # Количество классов
```

```
names: {list(class_mapping.keys())}  # Список     f.write(data_yaml_content)
классов
"""                                          print("Иерархия успешно построена и данные
with open(f"{output_dir}/data.yaml", "w") as fподготовлены!")
```

Результат:

Пример работы на изображениях:

Метрики обучения:



Вывод: осуществил обучение нейросетевого детектора для решения задачи определения дорожных знаков.