Министерство образования Республики Беларусь

Учреждение образования

«Брестский Государственный технический университет»

Кафедра ИИТ

Лабораторная работа №3

По дисциплине «ОИвИС»

Тема: “**Обучение детекторов объектов**”

Выполнил:

Студент 4 курса

Группы ИИ-23

Копач А. В.

Проверила:

Андренко К.В.

Брест 2025

Цель: осуществлять обучение нейросетевого детектора для решения задачи обнаружения заданных объектов

Вариант 7.

В-т	Детектор	Датасет
7	YOLOv11n	**Люди:** https://universe.roboflow.com/leo-ueno/people-detection-o4rdr/dataset/10

Код программы:

```
import os
import sys
import shutil
import numpy as np
import pandas as pd
import yaml
import cv2
import matplotlib.pyplot as plt
from ultralytics import YOLO
import torch
import glob
from datetime import datetime
import json
import gc


class CSVToYOLOConverter:
    def __init__(self, base_path):
        self.base_path = base_path
        self.class_name = "person"
        self.class_id = 0

    def convert_annotations(self, csv_path, images_dir, output_dir):
        """Конвертация CSV аннотаций в YOLO формат"""
        labels_dir = os.path.join(output_dir, 'labels')
        images_output_dir = os.path.join(output_dir, 'images')

        os.makedirs(labels_dir, exist_ok=True)
        os.makedirs(images_output_dir, exist_ok=True)

        print(f"Чтение CSV: {csv_path}")

        try:
            df = pd.read_csv(csv_path)
            print(f"Загружено {len(df)} аннотаций из {csv_path}")
        except Exception as e:
            print(f"Ошибка чтения CSV: {e}")
            return 0
```

```
    processed_files = 0
    missing_images = 0

    for filename, group in df.groupby('filename'):
        src_image_path = os.path.join(images_dir, filename)

        if not os.path.exists(src_image_path):
            missing_images += 1
            continue

        dst_image_path = os.path.join(images_output_dir, filename)
        shutil.copy2(src_image_path, dst_image_path)

        label_filename = os.path.splitext(filename)[0] + '.txt'
        label_path = os.path.join(labels_dir, label_filename)

        try:
            with open(label_path, 'w') as f:
                for _, row in group.iterrows():
                    x_center, y_center, width, height = self.convert_to_yolo_format(
                        row['xmin'], row['ymin'], row['xmax'], row['ymax'],
                        row['width'], row['height']
                    )
                    f.write(f"{self.class_id} {x_center:.6f} {y_center:.6f} {width:.6f} {height:.6f}\n")

            processed_files += 1
        except Exception as e:
            print(f"Ошибка создания аннотации для {filename}: {e}")

    if missing_images > 0:
        print(f"Всего отсутствующих изображений: {missing_images}")

    print(f"Обработано {processed_files} файлов для {output_dir}")
    return processed_files

def convert_to_yolo_format(self, xmin, ymin, xmax, ymax, img_width, img_height):
    """Конвертация координат в YOLO формат"""
    x_center = (xmin + xmax) / 2 / img_width
    y_center = (ymin + ymax) / 2 / img_height
    width = (xmax - xmin) / img_width
    height = (ymax - ymin) / img_height

    x_center = max(0, min(1, x_center))
    y_center = max(0, min(1, y_center))
    width = max(0, min(1, width))
    height = max(0, min(1, height))

    return x_center, y_center, width, height
```

```
class PeopleDetectorTrainer:
    def __init__(self, dataset_path='lab3'):
        self.script_dir = os.path.dirname(os.path.abspath(__file__))
        self.dataset_path = os.path.join(self.script_dir, dataset_path)
        self.yolo_dataset_path = os.path.join(self.script_dir, 'yolo_dataset')
        self.model = None
        self.results = None
        self.timestamp = datetime.now().strftime("%Y%m%d_%H%M%S")
        self.output_dir = os.path.join(self.script_dir, 'training_results', self.timestamp)

    def setup_environment(self):
        """Проверка и настройка окружения"""
        print("Настройка окружения...")

        torch.backends.cudnn.benchmark = True
        if torch.cuda.is_available():
            gpu_name = torch.cuda.get_device_name(0)
            gpu_memory = torch.cuda.get_device_properties(0).total_memory / 1024 ** 3
            print(f"GPU доступен: {gpu_name} ({gpu_memory:.1f} GB)")
            self.device = 0
            # Очистка памяти
            torch.cuda.empty_cache()
            # Установка лимита памяти
            torch.cuda.set_per_process_memory_fraction(0.7)  # Используем только 70% памяти
        else:
            print("GPU не доступен, используется CPU")
            self.device = None

        os.makedirs(self.output_dir, exist_ok=True)
        os.makedirs(os.path.join(self.output_dir, 'detections'), exist_ok=True)
        os.makedirs(os.path.join(self.output_dir, 'graphs'), exist_ok=True)
        os.makedirs(os.path.join(self.output_dir, 'models'), exist_ok=True)

    def prepare_dataset(self):
        """Подготовка датасета в YOLO формате"""
        print("Подготовка датасета в YOLO формате...")

        if not os.path.exists(self.dataset_path):
            print(f"Директория датасета не найдена: {self.dataset_path}")
            return False

        converter = CSVToYOLOConverter(self.dataset_path)

        yolo_dirs = ['train', 'val', 'test']
        for dir_name in yolo_dirs:
            os.makedirs(os.path.join(self.yolo_dataset_path, dir_name, 'images'), exist_ok=True)
```

```
        os.makedirs(os.path.join(self.yolo_dataset_path, dir_name, 'labels'), exist_ok=True)

    train_csv = os.path.join(self.dataset_path, 'train', '_annotations.csv')
    train_images = os.path.join(self.dataset_path, 'train')
    print("Конвертация тренировочных данных...")
    converter.convert_annotations(train_csv, train_images,
                                  os.path.join(self.yolo_dataset_path, 'train'))

    val_csv = os.path.join(self.dataset_path, 'valid', '_annotations.csv')
    val_images = os.path.join(self.dataset_path, 'valid')
    print("Конвертация валидационных данных...")
    converter.convert_annotations(val_csv, val_images,
                                  os.path.join(self.yolo_dataset_path, 'val'))

    test_csv = os.path.join(self.dataset_path, 'test', '_annotations.csv')
    test_images = os.path.join(self.dataset_path, 'test')
    print("Конвертация тестовых данных...")
    converter.convert_annotations(test_csv, test_images,
                                  os.path.join(self.yolo_dataset_path, 'test'))

    self.create_data_yaml()
    print("Датасет подготовлен в YOLO формате")
    return True

def create_data_yaml(self):
    """Создание data.yaml файла"""
    data = {
        'path': os.path.abspath(self.yolo_dataset_path),
        'train': 'train/images',
        'val': 'val/images',
        'test': 'test/images',
        'nc': 1,
        'names': ['person']
    }

    yaml_path = os.path.join(self.yolo_dataset_path, 'data.yaml')
    with open(yaml_path, 'w') as f:
        yaml.dump(data, f, default_flow_style=False)

    print(f"data.yaml создан: {yaml_path}")

def analyze_dataset(self):
    """Анализ датасета"""
    print("Анализ датасета:")

    splits = ['train', 'val', 'test']
    for split in splits:
        images_dir = os.path.join(self.yolo_dataset_path, split, 'images')
```

```
        labels_dir = os.path.join(self.yolo_dataset_path, split, 'labels')

        if not os.path.exists(images_dir):
            print(f"Директория {images_dir} не найдена")
            continue

        num_images = len([f for f in os.listdir(images_dir) if f.endswith(('.jpg', '.png', '.jpeg'))])
        num_labels = len([f for f in os.listdir(labels_dir) if f.endswith('.txt')]) if os.path.exists(
            labels_dir) else 0

        total_objects = 0
        if os.path.exists(labels_dir):
            for label_file in os.listdir(labels_dir):
                if label_file.endswith('.txt'):
                    try:
                        with open(os.path.join(labels_dir, label_file), 'r') as f:
                            total_objects += len(f.readlines())
                    except:
                        pass

        print(f"  {split.capitalize()}: {num_images} изображений, {num_labels} аннотаций, {total_objects} объектов")

    def setup_model(self):
        """Инициализация модели YOLOv11n"""
        print("Инициализация модели YOLOv11n...")

        try:
            self.model = YOLO('yolo11n.pt')
            print("Модель успешно загружена")
            return True
        except Exception as e:
            print(f"Ошибка загрузки модели: {e}")
            return False

    def train_model(self, epochs=50, imgsz=640, batch=12, patience=15):
        """
        Обучение модели с безопасными параметрами
        """
        print("Запуск обучения модели...")

        if self.model is None:
            print("Модель не инициализирована")
            return False

        try:
            # Безопасные параметры для избежания "Killed"
            self.results = self.model.train(
```

```
            data=os.path.join(self.yolo_dataset_path, 'data.yaml'),
            epochs=epochs,
            imgsz=imgsz,
            batch=batch,  # Уменьшили batch size
            device=self.device,
            workers=4,  # Уменьшили workers
            lr0=0.01,
            lrf=0.01,
            momentum=0.937,
            weight_decay=0.0005,
            warmup_epochs=3.0,
            warmup_momentum=0.8,
            warmup_bias_lr=0.1,
            box=7.5,
            cls=0.5,
            dfl=1.5,
            patience=patience,
            save=True,
            save_period=10,
            cache=False,  # Отключили кэш для экономии памяти
            name=f'people_detection_{self.timestamp}',
            exist_ok=True,
            amp=True,  # Mixed precision обязательно
            # Минимальные аугментации
            hsv_h=0.015,
            hsv_s=0.7,
            hsv_v=0.4,
            degrees=0.0,
            translate=0.1,
            scale=0.5,
            shear=0.0,
            perspective=0.0,
            flipud=0.0,
            fliplr=0.5,
            mosaic=0.0,  # Отключили mosaic для экономии памяти
            mixup=0.0,  # Отключили mixup
            copy_paste=0.0
        )

        print("Обучение успешно завершено!")
        return True

    except Exception as e:
        print(f"Ошибка обучения: {e}")
        return False

def evaluate_model(self):
    """Оценка модели на тестовой выборке"""
```

```
    print("Оценка модели на тестовой выборке...")

    if self.model is None:
        print("Модель не инициализирована")
        return None

    try:
        best_model_path = os.path.join(self.script_dir, 'runs', 'detect',
                                       f'people_detection_{self.timestamp}', 'weights', 'best.pt')
        if os.path.exists(best_model_path):
            self.model = YOLO(best_model_path)
            print("Загружена лучшая модель после обучения")

            # Копируем лучшую модель в папку результатов
            shutil.copy2(best_model_path, os.path.join(self.output_dir, 'models', 'best.pt'))

        # Очистка памяти перед валидацией
        if torch.cuda.is_available():
            torch.cuda.empty_cache()

        metrics = self.model.val(
            data=os.path.join(self.yolo_dataset_path, 'data.yaml'),
            split='test',
            device=self.device,
            batch=8,  # Уменьшили batch для валидации
            workers=2
        )

        print("Результаты оценки:")
        print(f"  mAP50-95: {metrics.box.map:.4f}")
        print(f"  mAP50: {metrics.box.map50:.4f}")
        print(f"  mAP75: {metrics.box.map75:.4f}")
        print(f"  Precision: {metrics.box.precision:.4f}")
        print(f"  Recall: {metrics.box.recall:.4f}")

        self.save_metrics_to_file(metrics)

        return metrics

    except Exception as e:
        print(f"Ошибка оценки: {e}")
        return None

def save_metrics_to_file(self, metrics):
    """Сохранение метрик в файл"""
    metrics_data = {
        'mAP50-95': float(metrics.box.map),
        'mAP50': float(metrics.box.map50),
```

```
        'mAP75': float(metrics.box.map75),
        'precision': float(metrics.box.precision),
        'recall': float(metrics.box.recall),
        'training_time': self.timestamp,
        'model': 'YOLOv11n',
        'dataset': 'People Detection',
        'epochs': 50,
        'image_size': 640,
        'batch_size': 12
    }

    metrics_path = os.path.join(self.output_dir, 'metrics.json')
    with open(metrics_path, 'w', encoding='utf-8') as f:
        json.dump(metrics_data, f, indent=4, ensure_ascii=False)

    print(f"Метрики сохранены: {metrics_path}")

def save_training_plots(self):
    """Сохранение графиков обучения"""
    if self.results is None:
        print("Нет результатов обучения для построения графиков")
        return

    try:
        results_dict = self.results.results_dict

        if results_dict:
            # Графики потерь
            plt.figure(figsize=(15, 10))

            metrics = ['train/box_loss', 'train/cls_loss', 'train/dfl_loss',
                       'val/box_loss', 'val/cls_loss', 'val/dfl_loss']

            for i, metric in enumerate(metrics, 1):
                plt.subplot(2, 3, i)
                if metric in results_dict:
                    plt.plot(results_dict[metric])
                    plt.title(metric.replace('/', ' ').title())
                    plt.xlabel('Эпоха')
                    plt.ylabel('Потери')
                    plt.grid(True, alpha=0.3)

            plt.tight_layout()
            plt.savefig(os.path.join(self.output_dir, 'graphs', 'training_losses.png'),
                        dpi=300, bbox_inches='tight', facecolor='white')
            plt.close()

            # Графики метрик
```

```
            plt.figure(figsize=(15, 5))

            metrics_to_plot = [
                ('metrics/precision(B)', 'Precision'),
                ('metrics/recall(B)', 'Recall'),
                ('metrics/mAP50(B)', 'mAP50'),
                ('metrics/mAP50-95(B)', 'mAP50-95')
            ]

            for i, (metric, title) in enumerate(metrics_to_plot, 1):
                plt.subplot(1, 4, i)
                if metric in results_dict:
                    plt.plot(results_dict[metric])
                    plt.title(title)
                    plt.xlabel('Эпоха')
                    plt.ylabel('Значение')
                    plt.grid(True, alpha=0.3)

            plt.tight_layout()
            plt.savefig(os.path.join(self.output_dir, 'graphs', 'training_metrics.png'),
                        dpi=300, bbox_inches='tight', facecolor='white')
            plt.close()

            print(f"Графики обучения сохранены: {os.path.join(self.output_dir, 'graphs')}")
        else:
            print("Нет данных истории обучения")

    except Exception as e:
        print(f"Ошибка сохранения графиков: {e}")


class DetectionVisualizer:
    def __init__(self, model_path, output_dir):
        self.model = YOLO(model_path)
        self.output_dir = output_dir
        os.makedirs(self.output_dir, exist_ok=True)

    def test_on_dataset(self, dataset_path, num_images=5):
        """Тестирование на изображениях из датасета"""
        test_images_dir = os.path.join(dataset_path, 'test', 'images')

        if not os.path.exists(test_images_dir):
            print(f"Директория тестовых изображений не найдена: {test_images_dir}")
            return

        image_files = glob.glob(os.path.join(test_images_dir, "*.jpg"))
        image_files = image_files[:num_images]
```

```
    print(f"Тестирование на {len(image_files)} изображениях из датасета...")

    for i, image_path in enumerate(image_files):
        try:
            # Очистка памяти перед каждой обработкой
            if torch.cuda.is_available():
                torch.cuda.empty_cache()

            results = self.model.predict(image_path, conf=0.25, imgsz=640)
            plotted_image = results[0].plot()
            output_path = os.path.join(self.output_dir, f'dataset_test_{i + 1:03d}.jpg')
            cv2.imwrite(output_path, plotted_image)

            num_detections = len(results[0].boxes) if results[0].boxes else 0
            print(f"Изображение {i + 1}/{len(image_files)}: обнаружено {num_detections} человек")

        except Exception as e:
            print(f"Ошибка обработки {image_path}: {e}")


def main():
    """Основная функция"""
    print(" ЗАПУСК ПРОГРАММЫ ОБНАРУЖЕНИЯ ЛЮДЕЙ")
    print("=" * 50)

    # Очистка памяти перед началом
    if torch.cuda.is_available():
        torch.cuda.empty_cache()

    # 1. Инициализация
    detector = PeopleDetectorTrainer(dataset_path='lab3')
    detector.setup_environment()

    # 2. Подготовка датасета
    if not detector.prepare_dataset():
        print(" Не удалось подготовить датасет. Проверьте структуру папок.")
        return

    detector.analyze_dataset()

    # 3. Инициализация модели
    if not detector.setup_model():
        return

    # 4. Обучение модели
    print("\n БЕЗОПАСНЫЕ ПАРАМЕТРЫ ОБУЧЕНИЯ:")
    print("  - Размер изображения: 640x640")
    print("  - Batch size: 12 (уменьшен для стабильности)")
```

```
    print("  - Количество эпох: 50 (уменьшено)")
    print("  - Workers: 4 (уменьшено)")
    print("  - Mosaic: отключен (экономия памяти)")
    print("  - Mixup: отключен (экономия памяти)")
    print("  - Лимит памяти GPU: 70%")

    if not detector.train_model(epochs=50, imgsz=640, batch=12):
        return

    # 5. Оценка модели
    print("\n ОЦЕНКА МОДЕЛИ...")
    metrics = detector.evaluate_model()

    # 6. Сохранение графиков
    print("\n СОХРАНЕНИЕ ГРАФИКОВ...")
    detector.save_training_plots()

    # 7. Тестирование детектора
    print("\n ТЕСТИРОВАНИЕ ДЕТЕКТОРА...")
    best_model_path = os.path.join(detector.output_dir, 'models', 'best.pt')

    if os.path.exists(best_model_path):
        visualizer = DetectionVisualizer(best_model_path,
                                         os.path.join(detector.output_dir, 'detections'))

        # Тестирование на датасете
        visualizer.test_on_dataset(detector.yolo_dataset_path, num_images=5)

        print(f" РЕЗУЛЬТАТЫ СОХРАНЕНЫ В: {detector.output_dir}")

        if metrics:
            print(f"\n ФИНАЛЬНЫЕ МЕТРИКИ МОДЕЛИ:")
            print(f"   mAP50: {metrics.box.map50:.3f}")
            print(f"   mAP50-95: {metrics.box.map:.3f}")
            print(f"   Precision: {metrics.box.precision:.3f}")
            print(f"   Recall: {metrics.box.recall:.3f}")
    else:
        print(" Лучшая модель не найдена")


if __name__ == "__main__":
    main()
```

Вывод программы:

```
Image sizes 640 train, 640 val
Using 4 dataloader workers
```

```
Logging results to /home/oppa/runs/detect/people_detection_20251111_214341
Starting training for 50 epochs...

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
       1/50      0.25G      1.517      1.963      1.323         65        640: 100% ━━━━━━━━━━━━ 1083/1083 5.8it/s 3:07
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 58/58 2.2it/s 26.2s
                   all       1370      10660      0.675      0.465      0.504      0.243

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
       2/50      2.27G      1.536      1.639       1.34         25        640: 100% ━━━━━━━━━━━━ 1083/1083 7.4it/s 2:27
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100━━━━━━━━━━━━ 58/58 7.3it/s 8.0s
                   all       1370      10660      0.624      0.449      0.489      0.232

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
       3/50      3.13G      1.624      1.656      1.419         22        640: 100% ━━━━━━━━━━━━ 1083/1083 8.0it/s 2:15
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 58/58 6.9it/s 8.4s
                   all       1370      10660      0.568      0.395      0.411      0.191

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
       4/50      3.14G      1.689      1.689      1.469         55        640: 100% ━━━━━━━━━━━━ 1083/1083 7.6it/s 2:23
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 58/58 6.5it/s 9.0s
                   all       1370      10660      0.642      0.414      0.463      0.223

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
       5/50      3.14G      1.611      1.594      1.435         25        640: 100% ━━━━━━━━━━━━ 1083/1083 7.8it/s 2:19
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 58/58 7.1it/s 8.2s
                   all       1370      10660      0.655      0.429      0.493      0.243

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
       6/50      3.14G      1.555      1.508      1.402         17        640: 100% ━━━━━━━━━━━━ 1083/1083 7.6it/s 2:22
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 58/58 6.8it/s 8.6s
                   all       1370      10660      0.612      0.449      0.496      0.252

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
       7/50      3.14G      1.514      1.454      1.373         30        640: 100% ━━━━━━━━━━━━1083/1083 7.6it/s 2:22
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 58/58 6.3it/s 9.3s
                   all       1370      10660      0.683      0.484      0.544       0.28

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
       8/50      3.14G      1.477      1.411      1.354         28        640: 100% ━━━━━━━━━━━━1083/1083 7.6it/s 2:23
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 58/58 7.1it/s 8.2s
                   all       1370      10660      0.691      0.485      0.557      0.295

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
       9/50      3.14G      1.451      1.373      1.334        120        640: 100% ━━━━━━━━━━━━ 1083/1083 7.6it/s 2:22
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%━━━━━━━━━━━━ 58/58 6.3it/s 9.2s
```

```
      all      1370     10660      0.681      0.481      0.543      0.292

Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
10/50      3.14G      1.423      1.338      1.318         19        640: 100% ━━━━━━━━━━━━ 1083/1083 7.2it/s 2:31
       Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 58/58 7.0it/s 8.3s
         all      1370      10660      0.732      0.476      0.542      0.299

Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
11/50      3.14G      1.405      1.302      1.304         27        640: 100% ━━━━━━━━━━━━ 1083/1083 7.8it/s 2:19
       Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 58/58 6.7it/s 8.6s
         all      1370      10660      0.703      0.512      0.581       0.32

Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
12/50      3.14G      1.384      1.286      1.293         20        640: 100% ━━━━━━━━━━━━ 1083/1083 7.5it/s 2:24
       Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 58/58 7.3it/s 8.0s
         all      1370      10660      0.712      0.515      0.591      0.327

Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
13/50      3.14G      1.361      1.267      1.282         17        640: 100% ━━━━━━━━━━━━ 1083/1083 7.4it/s 2:26
       Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 58/58 6.8it/s 8.6s
         all      1370      10660      0.713       0.53      0.595      0.329

Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
14/50      3.14G       1.35      1.243      1.272         26        640: 100% ━━━━━━━━━━━━ 1083/1083 7.8it/s 2:18
       Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 58/58 7.3it/s 8.0s
         all      1370      10660       0.75      0.516      0.602      0.338

Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
15/50      3.14G      1.335      1.219      1.261          9        640: 100% ━━━━━━━━━━━━ 1083/1083 7.9it/s 2:17
       Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 58/58 7.1it/s 8.1s
         all      1370      10660       0.72      0.532      0.599      0.336

Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
16/50      3.14G      1.326      1.199      1.251         36        640: 100% ━━━━━━━━━━━━ 1083/1083 7.7it/s 2:21
       Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 58/58 6.7it/s 8.6s
         all      1370      10660      0.752      0.536      0.619      0.348

Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
17/50      3.14G       1.31      1.184      1.242         25        640: 100% ━━━━━━━━━━━━ 1083/1083 7.4it/s 2:26
       Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 58/58 6.9it/s 8.4s
         all      1370      10660      0.744      0.547      0.627      0.353


Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
18/50      3.14G      1.298      1.168      1.238         23        640: 100% ━━━━━━━━━━━━ 1083/1083 7.4it/s 2:25
       Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 58/58 6.6it/s 8.8s
```

```
        all      1370     10660     0.748      0.54     0.623     0.353

Epoch   GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
19/50     3.14G      1.288       1.15      1.229         57        640: 100% ━━━━━━━━━━━━ 1083/1083 7.7it/s 2:22
       Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 58/58 6.9it/s 8.4s
        all      1370     10660     0.763     0.554     0.632     0.359

Epoch   GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
20/50     3.14G      1.275       1.145      1.222         25        640: 100% ━━━━━━━━━━━━ 1083/1083 7.6it/s 2:22
       Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 58/58 6.8it/s 8.5s
        all      1370     10660     0.742     0.552     0.635     0.368

Epoch   GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
21/50     3.14G      1.259       1.119       1.21         16        640: 100% ━━━━━━━━━━━━ 1083/1083 7.9it/s 2:18
       Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 58/58 6.9it/s 8.4s
        all      1370     10660      0.76     0.563     0.647     0.372

Epoch   GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
22/50     3.14G      1.252       1.101      1.205         34        640: 100% ━━━━━━━━━━━━ 1083/1083 7.6it/s 2:23
       Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 58/58 6.5it/s 8.9s
        all      1370     10660     0.768     0.562     0.642     0.373

Epoch   GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
23/50     3.14G      1.249       1.104      1.204         25        640: 100% ━━━━━━━━━━━━ 1083/1083 7.7it/s 2:22
       Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 58/58 6.5it/s 8.9s
        all      1370     10660     0.759     0.562     0.648     0.375

Epoch   GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
24/50     3.14G      1.235       1.084      1.199         33        640: 100% ━━━━━━━━━━━━ 1083/1083 7.7it/s 2:21
       Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 58/58 7.0it/s 8.3s
        all      1370     10660     0.775      0.56     0.647     0.375

Epoch   GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
25/50     3.14G       1.22       1.074      1.192         19        640: 100% ━━━━━━━━━━━━ 1083/1083 7.7it/s 2:20
       Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 58/58 6.4it/s 9.1s
        all      1370     10660     0.773     0.557      0.65     0.383

Epoch   GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
26/50     3.14G      1.211       1.058      1.186         29        640: 100% ━━━━━━━━━━━━ 1083/1083 7.6it/s 2:23
       Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 58/58 6.8it/s 8.5s
        all      1370     10660     0.761      0.58     0.665     0.392


Epoch   GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
27/50     3.14G      1.209       1.052      1.178         31        640: 100% ━━━━━━━━━━━━ 1083/1083 7.5it/s 2:24
       Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 58/58 6.7it/s 8.6s
```

```
        all      1370     10660      0.773      0.573      0.665      0.393

Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
28/50      3.14G      1.197       1.04      1.172         44        640: 100% ━━━━━━━━━━━━ 1083/1083 7.7it/s 2:20
       Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 58/58 7.2it/s 8.1s
         all      1370      10660      0.772      0.576      0.661       0.39

Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
29/50      3.14G      1.188      1.032      1.171         35        640: 100% ━━━━━━━━━━━━ 1083/1083 7.6it/s 2:22
       Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 58/58 7.1it/s 8.2s
         all      1370      10660      0.771      0.582      0.667      0.393

Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
30/50      3.14G      1.182      1.021      1.164         15        640: 100% ━━━━━━━━━━━━ 1083/1083 7.6it/s 2:23
       Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 58/58 7.0it/s 8.3s
         all      1370      10660      0.777      0.579       0.67      0.398

Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
31/50      3.14G      1.169       1.01      1.159         21        640: 100% ━━━━━━━━━━━━ 1083/1083 7.9it/s 2:17
       Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 58/58 7.0it/s 8.2s
         all      1370      10660      0.777      0.581      0.671      0.401

Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
32/50      3.14G      1.161      0.994      1.154         28        640: 100% ━━━━━━━━━━━━ 1083/1083 7.7it/s 2:22
       Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 58/58 6.9it/s 8.4s
         all      1370      10660      0.789       0.58      0.674      0.403

Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
33/50      3.14G      1.154     0.9868      1.147         25        640: 100% ━━━━━━━━━━━━ 1083/1083 7.6it/s 2:23
       Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 58/58 6.8it/s 8.5s
         all      1370      10660      0.775      0.593      0.677      0.404



Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
34/50      3.14G      1.145     0.9767      1.141         20        640: 100% ━━━━━━━━━━━━ 1083/1083 7.9it/s 2:18
       Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 58/58 7.1it/s 8.2s
         all      1370      10660      0.787      0.583      0.679      0.409
Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
35/50      3.14G      1.136      0.967      1.138         17        640: 100% ━━━━━━━━━━━━ 1083/1083 7.5it/s 2:24
       Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 58/58 7.0it/s 8.3s
         all      1370      10660      0.787      0.587       0.68       0.41

Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
36/50      3.14G      1.122     0.9551      1.133         20        640: 100% ━━━━━━━━━━━━ 1083/1083 7.5it/s 2:25
       Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 58/58 6.7it/s 8.6s
```

```
                all      1370     10660     0.783     0.589     0.681     0.412

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      37/50      3.14G      1.124     0.9468       1.13         46       640: 100% ━━━━━━━━━━━━ 1083/1083 7.9it/s 2:18
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 58/58 6.9it/s 8.4s
                   all      1370     10660      0.795      0.585      0.684      0.416

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      38/50      3.14G      1.109     0.9319       1.124         17       640: 100% ━━━━━━━━━━━━ 1083/1083 8.1it/s 2:14
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 58/58 6.5it/s 8.9s
                   all      1370     10660      0.795      0.582      0.681      0.417

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      39/50      3.14G      1.099      0.926       1.12         23       640: 100% ━━━━━━━━━━━━ 1083/1083 7.7it/s 2:20
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 58/58 6.9it/s 8.4s
                   all      1370     10660      0.784      0.593      0.687       0.42

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      40/50      3.14G      1.093     0.9161       1.111         66       640: 100% ━━━━━━━━━━━━ 1083/1083 7.5it/s 2:25
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 58/58 6.8it/s 8.6s
                   all      1370     10660      0.784      0.598      0.692      0.424
Closing dataloader mosaic

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      41/50      3.14G      1.081     0.9017       1.107         46       640: 100% ━━━━━━━━━━━━ 1083/1083 7.5it/s 2:24
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 58/58 6.5it/s 8.9s
                   all      1370     10660      0.787      0.602      0.692      0.424

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      42/50      3.14G      1.075     0.8935       1.104         38       640: 100% ━━━━━━━━━━━━1083/1083 7.4it/s 2:26
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 58/58 6.7it/s 8.7s
                   all      1370     10660      0.795      0.592       0.69      0.424

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      43/50      3.14G      1.065     0.8872       1.102         54       640: 100% ━━━━━━━━━━━━ 1083/1083 7.2it/s 2:31
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 58/58 6.7it/s 8.7s
                   all      1370     10660      0.799      0.589      0.689      0.424

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      44/50      3.14G      1.057     0.8754       1.096         55       640: 100% ━━━━━━━━━━━━ 1083/1083 7.2it/s 2:31
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 58/58 6.2it/s 9.4s
                   all      1370     10660      0.803      0.587       0.69      0.425

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      45/50      3.14G      1.047     0.8658       1.092         32       640: 100% ━━━━━━━━━━━━ 1083/1083 7.1it/s 2:33
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 58/58 6.5it/s 8.9s
```

all 1370 10660 0.798 0.593 0.691 0.426

Epoch GPU_mem box_loss cls_loss dfl_loss Instances Size
46/50 3.14G 1.043 0.8585 1.088 11 640: 100% ━━━━━━━━━━━━ 1083/1083 7.7it/s 2:20
Class Images Instances Box(P R mAP50 mAP50-95): 100% ━━━━━━━━ 58/58 6.9it/s 8.4s
all 1370 10660 0.797 0.599 0.694 0.427

Epoch GPU_mem box_loss cls_loss dfl_loss Instances Size
47/50 3.14G 1.039 0.8528 1.085 29 640: 100% ━━━━━━━━━━━━ 1083/1083 7.6it/s 2:22
Class Images Instances Box(P R mAP50 mAP50-95): 100% ━━━━━━━━ 58/58 7.0it/s 8.3s
all 1370 10660 0.796 0.603 0.695 0.428

Epoch GPU_mem box_loss cls_loss dfl_loss Instances Size
48/50 3.14G 1.022 0.8392 1.079 63 640: 100% ━━━━━━━━━━━━ 1083/1083 7.4it/s 2:27
Class Images Instances Box(P R mAP50 mAP50-95): 100% ━━━━━━━━ 58/58 6.2it/s 9.3s
all 1370 10660 0.794 0.601 0.694 0.429

Epoch GPU_mem box_loss cls_loss dfl_loss Instances Size
49/50 3.14G 1.021 0.8362 1.078 24 640: 100% ━━━━━━━━━━━━ 1083/1083 7.4it/s 2:27
Class Images Instances Box(P R mAP50 mAP50-95): 100% ━━━━━━━━ 58/58 6.4it/s 9.0s
all 1370 10660 0.79 0.605 0.695 0.429

Epoch GPU_mem box_loss cls_loss dfl_loss Instances Size
50/50 3.14G 1.014 0.8278 1.075 30 640: 100% ━━━━━━━━━━━━ 1083/1083 7.3it/s 2:28
Class Images Instances Box(P R mAP50 mAP50-95): 100% ━━━━━━━━ 58/58 7.4it/s 7.9s
all 1370 10660 0.794 0.606 0.696 0.43

50 epochs completed in 2.124 hours.
Optimizer stripped from /home/oppa/runs/detect/people_detection_20251111_214341/weights/last.pt, 5.4MB
Optimizer stripped from /home/oppa/runs/detect/people_detection_20251111_214341/weights/best.pt, 5.4MB

ОЦЕНКА МОДЕЛИ...
Оценка модели на тестовой выборке...
Ultralytics 8.3.227 🚀 Python-3.12.3 torch-2.8.0+cu129 CUDA:0 (NVIDIA GeForce RTX 5060, 8150MiB)
YOLO11n summary (fused): 100 layers, 2,582,347 parameters, 0 gradients, 6.3 GFLOPs
val: Fast image access ✅ (ping: 0.4±0.3 ms, read: 47.3±23.6 MB/s, size: 35.6 KB)
val: Scanning /home/oppa/mamka/yolo_dataset/test/labels... 60 images, 0 backgrounds, 0 corrupt: 8%
- ━━━━━━━━━━━━━━━━━━ 60/738 177.9it/s 0.1s<3.8val: Scanning /home/oppa/mamka/yolo_dataset/test/labels... 129 images,
0 backgrounds, 0 corrupt: 17% ━━━─━━━━━━━━━━━━━ 129/738 324.4it/s 0.2s<val: Scanning
/home/oppa/mamka/yolo_dataset/test/labels... 197 images, 0 backgrounds, 0 corrupt: 27% ━━━━─━━━━━━━━━━━━
197/738 426.6it/s 0.3s<val: Scanning /home/oppa/mamka/yolo_dataset/test/labels... 265 images, 0 backgrounds, 0
corrupt: 36% ━━━━━─━━━━━━━━━━━ 265/738 501.5it/s 0.4s<val: Scanning
/home/oppa/mamka/yolo_dataset/test/labels... 330 images, 0 backgrounds, 0 corrupt: 45% ━━━━━━─━━━━━━━━━━
330/738 543.2it/s 0.5s<val: Scanning /home/oppa/mamka/yolo_dataset/test/labels... 399 images, 0 backgrounds, 0

corrupt: 54% ━━━━━━━━━━━━ 399/738 583.5it/s 0.6s<val: Scanning /home/oppa/mamka/yolo_dataset/test/labels... 466 images, 0 backgrounds, 0 corrupt: 63% ━━━━━━━━━━━━━━- ━━━━ 466/738 606.8it/s 0.7s<val: Scanning /home/oppa/mamka/yolo_dataset/test/labels... 533 images, 0 backgrounds, 0 corrupt: 72% ━━━━━━━━━━━━━━- ━━━ 533/738 622.6it/s 0.8s<val: Scanning /home/oppa/mamka/yolo_dataset/test/labels... 600 images, 0 backgrounds, 0 corrupt: 81% ━━━━━━━━━━━━━━- ━━ 600/738 636.5it/s 0.9s<val: Scanning /home/oppa/mamka/yolo_dataset/test/labels... 665 images, 0 backgrounds, 0 corrupt: 90% ━━━━━━━━━━━━━━━━- ─ 665/738 638.9it/s 1.0s<val: Scanning /home/oppa/mamka/yolo_dataset/test/labels... 737 images, 0 backgrounds, 0 corrupt: 100% ━━━━━━━━━━━━━━━━- 737/738 661.1it/s 1.1sval: Scanning /home/oppa/mamka/yolo_dataset/test/labels... 738 images, 0 backgrounds, 0 corrupt: 100% ━━━━━━━━━━━━━━━━ 738/738 661.3it/s 1.1s

val: New cache created: /home/oppa/mamka/yolo_dataset/test/labels.cache

Class	Images	Instances	Box(P	R	mAP50	mAP50-95): 100% ━━━━ 93/93 5.3it/s 17.6s
all	738	2783	0.868	0.832	0.893	0.656

Speed: 0.8ms preprocess, 17.4ms inference, 0.0ms loss, 1.2ms postprocess per image

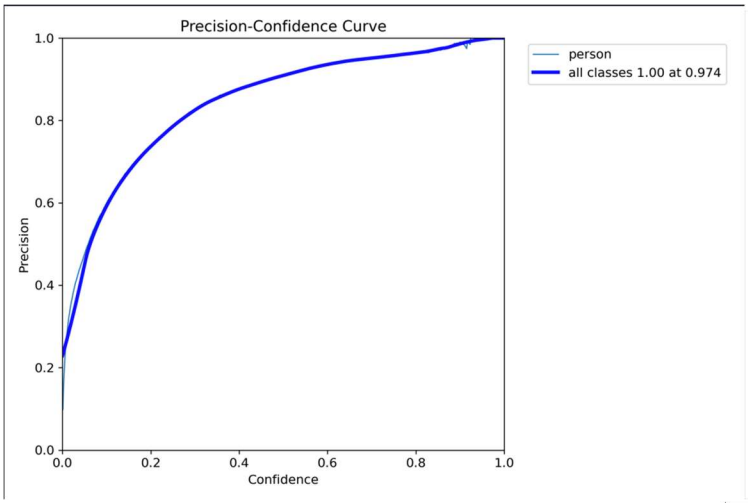
Results saved to /home/oppa/runs/detect/val

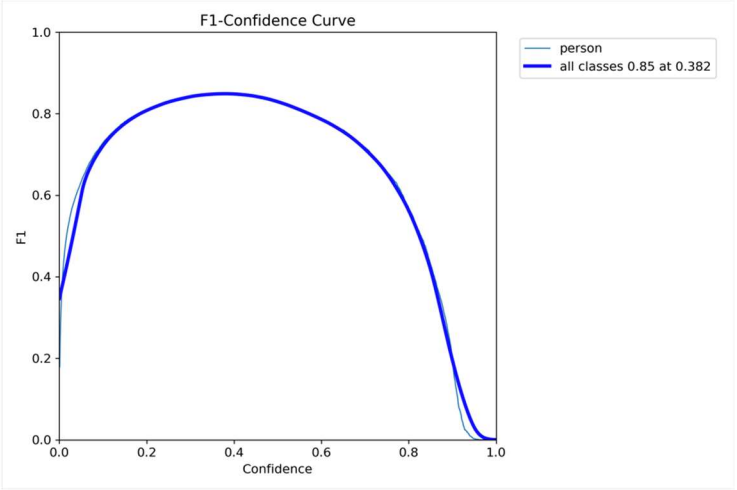
Результаты оценки:

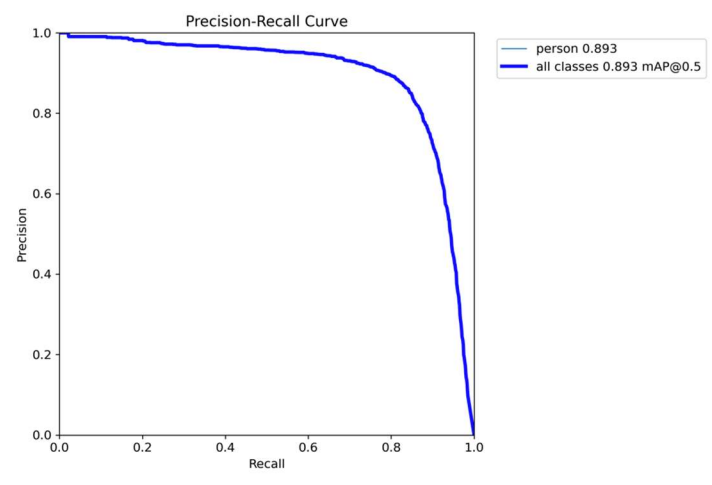
mAP50-95: 0.6556

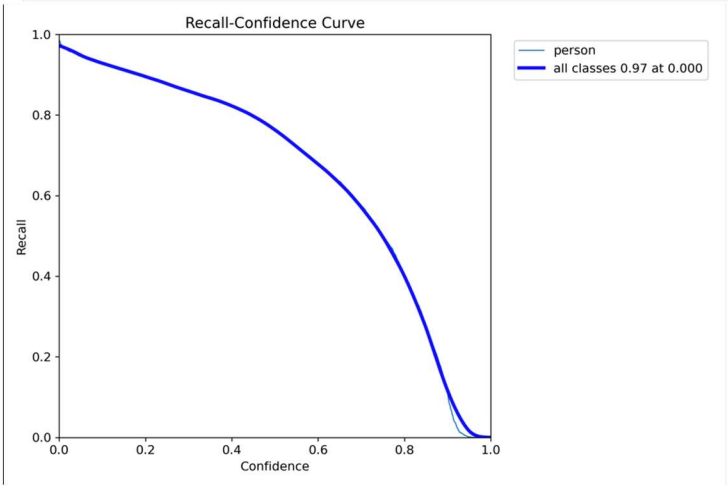
mAP50: 0.8933

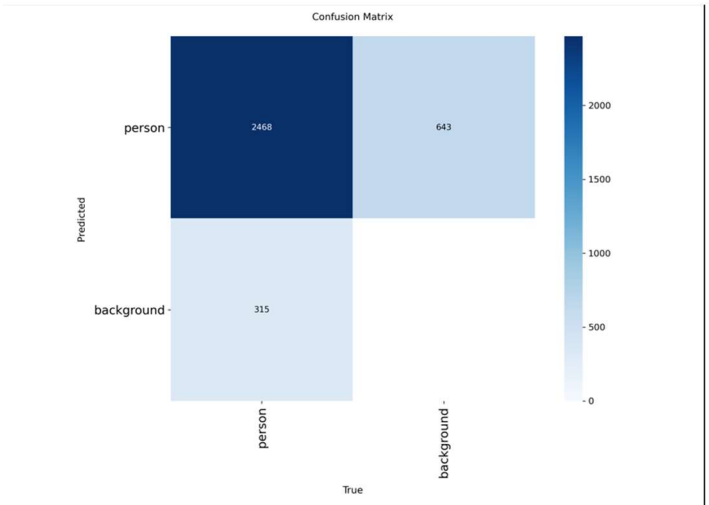
mAP75: 0.7308

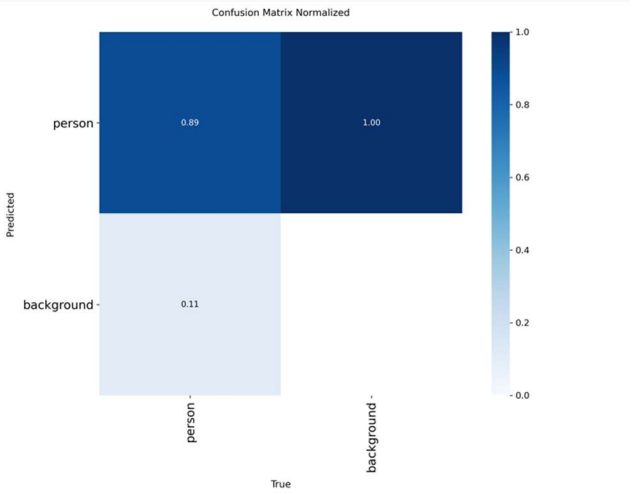


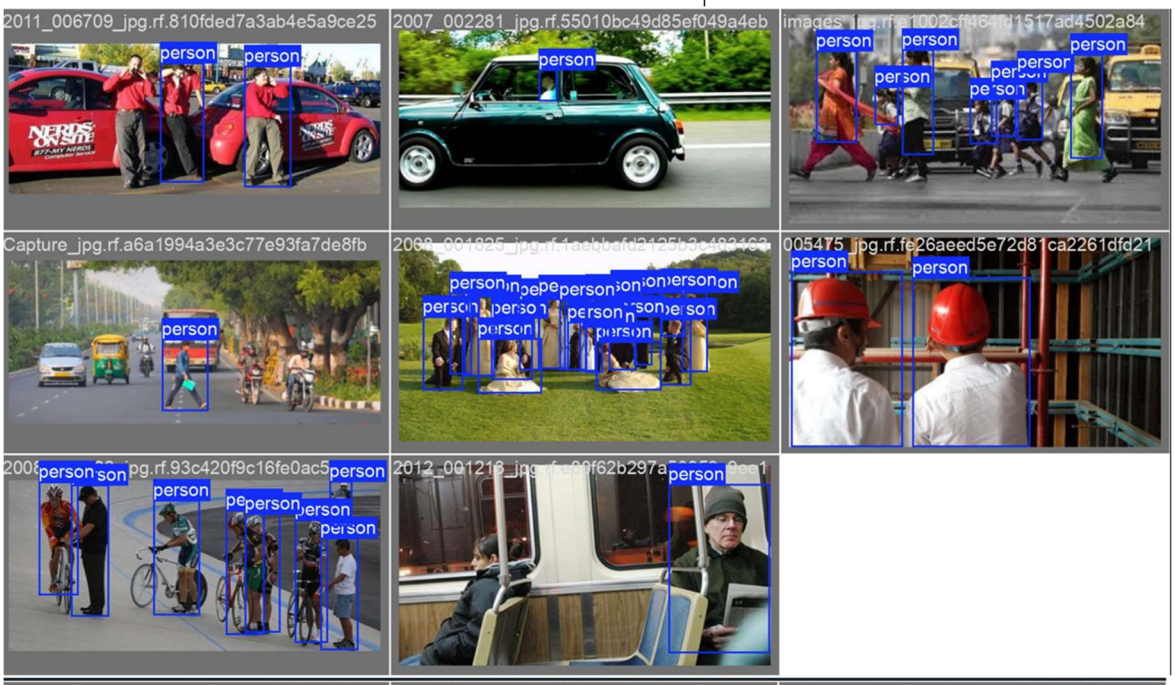







Confusion Matrix
person
background
Predicted
2468
643
315
2000
1500
1000
500
0
person
background
True


Confusion Matrix Normalized
person
background
Predicted
0.89
1.00
0.11
1.0
0.8
0.6
0.4
0.2
0.0
person
background
True


2011_006709_jpg.rf.810fded7a3ab4e5a9ce25
2007_002281_jpg.rf.55010bc49d85ef049a4eb
Capture_jpg.rf.a6a1994a3e3c77e93fa7de8fb
005475_jpg.rf.fe26aeed5e72d81ca2261dfd21
2012_001216_jpg.rf.80f62b297a
person


2011_006709_jpg.rf.810fded7a3ab4e5a9ce25
2007_002281_jpg.rf.55010bc49d85ef049a4eb
Capture_jpg.rf.a6a1994a3e3c77e93fa7de8fb
005475_jpg.rf.fe26aeed5e72d81ca2261dfd21
2012_001216_jpg.rf.80f62b297a
person 0.3
person 0.7
person 0.9
person 0.8

Вывод: осуществил обучение нейросетевого детектора для решения задачи обнаружения заданных объектов.