

## Лабораторная работа №9

### Yolo

1. Установка и настройка среды:

Установить необходимую библиотеку Yolo.

Настроить конфигурационные файлы Yolo для работы с набором данных.

2. Обучение модели Yolo:

Загрузить подготовленный набор данных в Yolo.

Обучить модель с использованием выбранных гиперпараметров.

Отслеживать процесс обучения и метрики (например, loss, accuracy).

3. Оптимизация модели:

Провести эксперименты с разными конфигурациями Yolo и гиперпараметрами.

Оптимизировать модель для достижения максимальной точности и скорости.

4. Реализация детекции объектов:

Разработать систему, которая будет использовать обученную модель Yolo для детекции объектов на видеопотоке.

```
git clone https://github.com/ultralytics/yolov5 # clone
cd yolov5
pip install -r requirements.txt # install
```

### 1. Подготовка набора данных для YOLOv5

Рекомендуется применить следующие шаги предварительной обработки: **Изменить размер (Stretch)** - до квадратного входного размера модели (640x640 - это YOLOv5 по умолчанию).





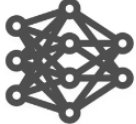
### 2.

#### Создайте `dataset.yaml`

[COCO128](#) Это пример небольшого обучающего набора данных, состоящего из первых 128 изображений в [COCO](#) train2017.

[data/coco128.yaml](#)

#### Выбор модели

				
Nano	Small	Medium	Large	XLarge
YOLOv5n	YOLOv5s	YOLOv5m	YOLOv5l	YOLOv5x
4 MB <sub>FP16</sub> 6.3 ms <sub>V100</sub> 28.4 mAP <sub>COCO</sub>	14 MB <sub>FP16</sub> 6.4 ms <sub>V100</sub> 37.2 mAP <sub>COCO</sub>	41 MB <sub>FP16</sub> 8.2 ms <sub>V100</sub> 45.2 mAP <sub>COCO</sub>	89 MB <sub>FP16</sub> 10.1 ms <sub>V100</sub> 48.8 mAP <sub>COCO</sub>	166 MB <sub>FP16</sub> 12.1 ms <sub>V100</sub> 50.7 mAP <sub>COCO</sub>

#### Обучение

Обучить модель YOLOv5s на COCO128, указав набор данных, размер батча, размер изображения и либо предварительно обученную модель `--weights yolov5s.pt` (рекомендуется), или случайно инициализированный `--weights '' --cfg yolov5s.yaml` (не рекомендуется).

```
train.py --img 640 --epochs 3 --data coco128.yaml --weights yolov5s.pt
```

Все результаты обучения сохраняются в `runs/train/` с увеличивающимися директориями запуска, то есть `runs/train/exp2`, `runs/train/exp3` И так далее.

### 3. Анализ

Файл с результатами `results.csv` обновляется после каждой эпохи as `results.png`. Можно построить график любого `results.csv` файл вручную:

```
from utils.plots import plot_results
```

```
plot_results("path/to/results.csv") # plot 'results.csv' as  
'results.png'
```