

Пояснительная записка

к дипломной работе на тему:

# **"Детекция и классификация марок и моделей автомобиля на изображении"**

Автор: Александр Воробьев

Группа: DSU-31

# **ОГЛАВЛЕНИЕ**

[ПОСТАНОВКА ЗАДАЧИ ДЛЯ ГЛУБОГО ОБУЧЕНИЯ 3](#_Toc164112333)

[АНАЛИЗ ДАННЫХ 3](#_Toc164112334)

[МЕТОДИКА РЕАЛИЗАЦИИ 5](#_Toc164112335)

[Вариант №1 (основной): YOLOv8 5](#_Toc164112336)

[Вариант №2 (дополнительный): SSD300 @VGG16 8](#_Toc164112337)

[ИТОГИ ОБУЧЕНИЯ МОДЕЛИ 9](#_Toc164112338)

# **ПОСТАНОВКА ЗАДАЧИ ДЛЯ ГЛУБОГО ОБУЧЕНИЯ**

С целью повышения эффективности процесса контроля в данном проекте реализована автоматизация анализа изображения автомобиля на картинке с помощью нейронных сетей (далее - НС) по следующим направлениям:

* детекция (выявление наличия автомобиля и его местоположения на источнике данных – фотоснимке);
* классификация (отнесение выявленного автомобиля к одной из категорий).

Задача в целом относится к классу «Object Detection» и успешно реализуется средствами сверточных НС (CNN). Поэтому в качестве основной архитектуры НС была выбрана YOLOv8 («You Only Look Once» v8). Используемый фреймворк для реализации задачи – PyTorch.

Основной метрикой оценки качества обучения НС и сравнения результатов послужила mAP(mean Average Precision)[[1]](#footnote-1) с IoU=0.5 (далее - mAP@50), т.е. среднее значение Precision при различных Recall, с условием перекрытия (Intersection-over-Union) не менее 0.5[[2]](#footnote-2).

# **АНАЛИЗ ДАННЫХ**

В качестве исходных данных для решения задачи был использован размеченный датасет с фотографиями, общедоступный https://universe.roboflow.com/mxk/car-model-detection/dataset/1.

Набор данных содержит 8 132 цветных фотографий, в т.ч. 6 506 (80%) размеченных (Train) и 1 626 (20%) (Valid).

Фотографии сделаны в хороших погодных условиях, при отсутствии осадков (дождя, снега и иные).

Фотографии были размечены при помощи инструментов LabelImg и CVAT (Computer Vision Annotation Tool) в формате PASCAL VOC (к каждой фотографии из train/val-датасета есть файл разметки повреждений в формате xml). Ко многим фотографиям применена аугментация, т.е. фото могут повторяться с небольшими вариациями.

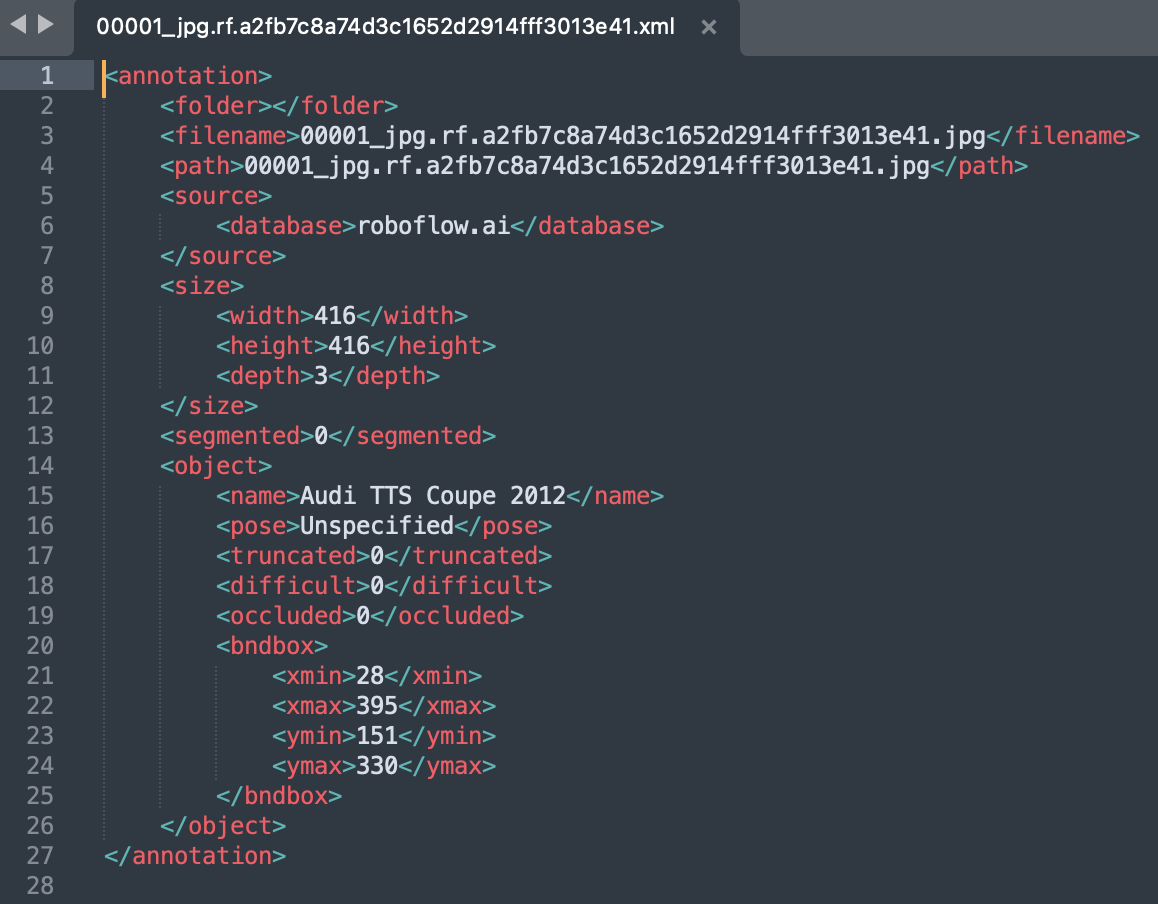


Рис.1 «Пример xml-файла разметки в формате PASCAL VOC»

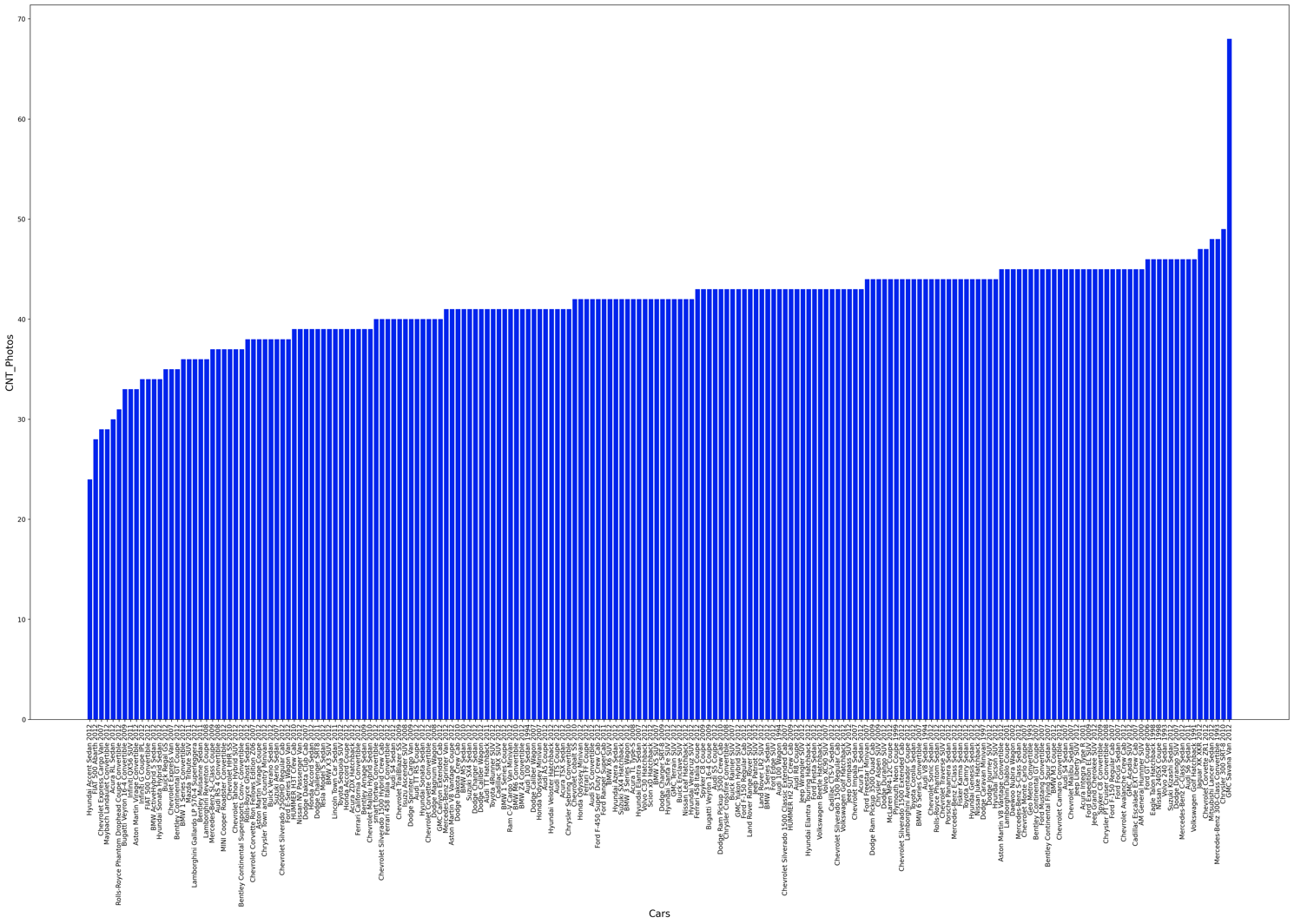


Рис.2 «Распределение автомобилей по классам в размеченном (train/val) датасете».

На основании анализа данных были сделаны следующие выводы:

Классы немного не сбалансированы между собой.

Весь датасет имеет полную разметку для всех изображений.

В рамках дипломного проекта для более скорейшего обучения моделей YOLOv8 и SSD300 VGG16 было принято решение ограничить объем классификации до 10 автомобилей (марок, моделей и года):

* Chevrolet Traverse SUV 2012
* Dodge Dakota Club Cab 2007
* Buick Verano Sedan 2012
* Acura TL Sedan 2012
* Audi TTS Coupe 2012
* Mitsubishi Lancer Sedan 2012
* Hyundai Sonata Hybrid Sedan 2012
* Ford F-450 Super Duty Crew Cab 2012
* Geo Metro Convertible 1993
* Dodge Charger Sedan 2012

После ограничений набор данных содержит 415 цветных фотографий, в т.ч. 336 (80%) размеченных (Train) и 79 (20%) (Valid). Автомобили были выбраны, чтобы постараться соблюсти небольшой дисбаланс классов.

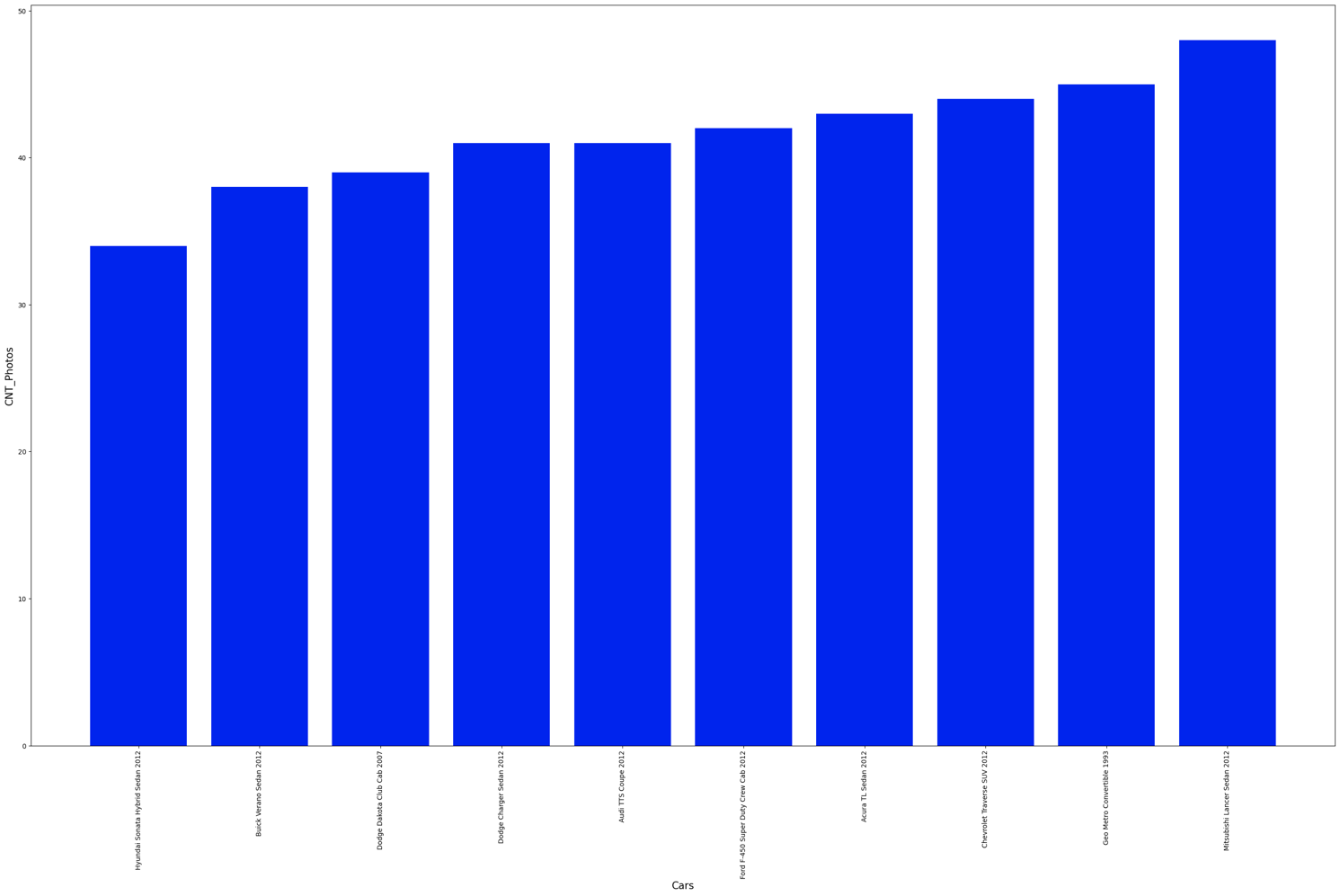


Рис.3 «Распределение автомобилей по классам в размеченном (train/val) датасете после ограничения».

# **МЕТОДИКА РЕАЛИЗАЦИИ**

# **Вариант №1 (основной): YOLOv8**

YOLOv8 является одной из State-of-the-Art архитектур, используемых в детекции объектов, и относится к архитектуре One-Stage detector: подход, который предсказывает координаты определённого количества *bounding box*'ов с результатами классификации и вероятности нахождения объекта, и в дальнейшем корректируя их местоположение. В целом такую архитектуру можно представить в следующем виде[[3]](#footnote-3):

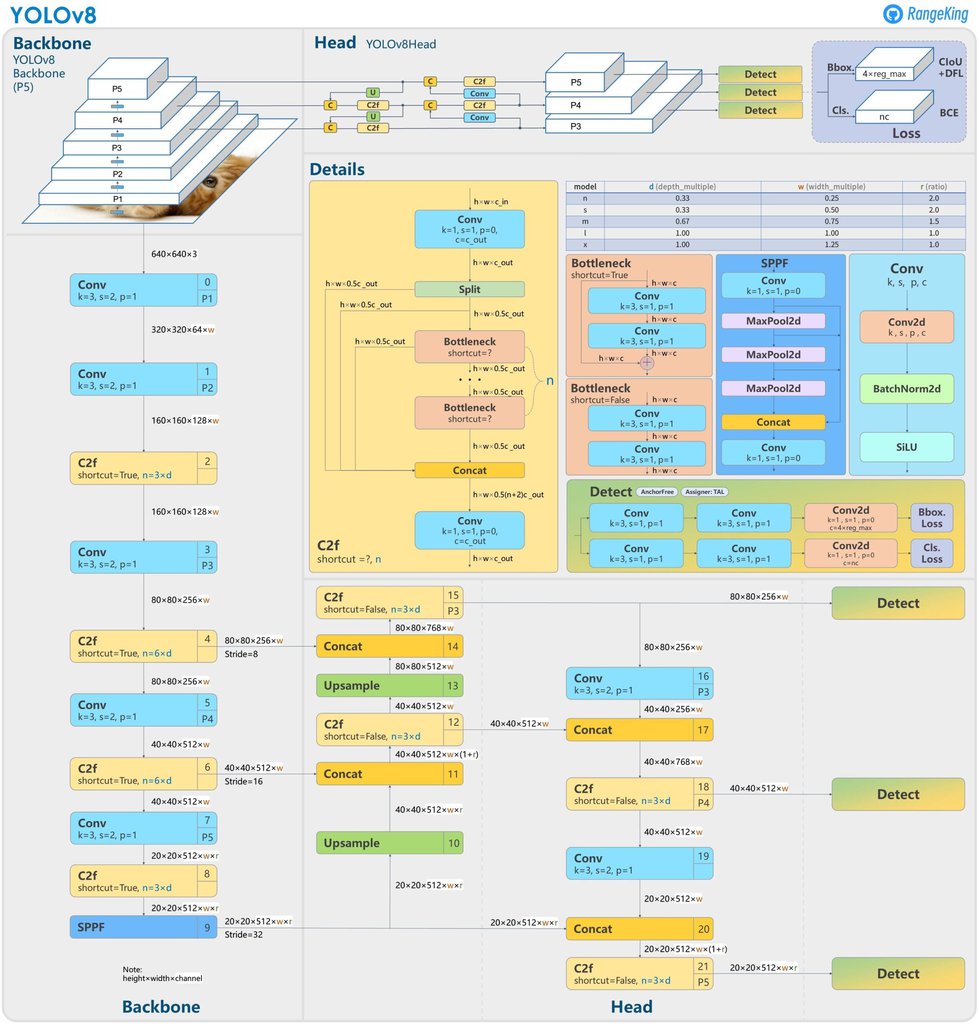


Рис.4 «Визуализация модели YOLOv8».

Модель YOLOv8 была реализована на PyTorch, предобучена разработчиками на датасете MS COCO (80 классов), и в полноразмерных вариантах показывала [mAP@0.5](mailto:mAP@0.5) на уровне выше 50%, а также показывала достаточно высокую сходимость за приемлемое время, что и предопределило выбор данной модели в качестве основной для решения задачи.

В ходе подготовки данных и обучения модели на YOLOv8 были проделаны следующие шаги:

1. Размеченный (train) датасет вручную разбит на трейн-(80%) и тест-(20%) выборки с сохранением %-ного соотношения между автомобилями;
2. Разметка файлов сконвертирована из формата PASCAL VOC (XML) в формат YOLO (TXT, координаты bounding boxes отнормированы по центру изображения, все классы объектов сохранены);
3. Склонирован с github репозиторий Ultralytics, содержащий pretrained-модели YOLOv8 разной степени сложности (от Small до Large), с целью дальнейшего обучения на нашем ограниченном датасете;
4. Сформирован настроечный YAML-файл (пути к данным + перечисление классов);
5. Проведено предварительное обучение на малом датасете (данные только по Чехии) с целью выбора подходящей модели и гиперпараметров для всего датасета. Для контроля процесса обучения использовался Tensorboard, а также файлы с метриками, формируемые непосредственно моделью YOLOv8 в ходе обучения;
6. После подбора подходящей модели и гиперпараметров проведено обучение на всем датасете (по всем странам);
7. Проведен референс модели на тестовом датасете (фото без разметки).

Для оценки результатов обучения были использованы следующие метрики[[4]](#footnote-4):

1. Intersection-over-Union (если IoU > 0.5, то предсказание можно считать «хорошим»)

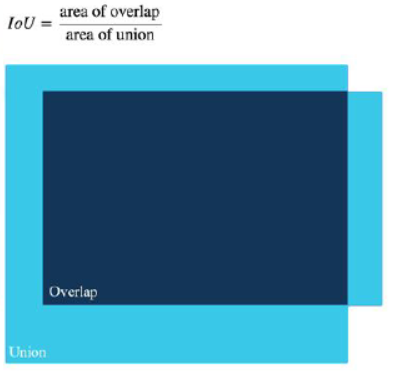
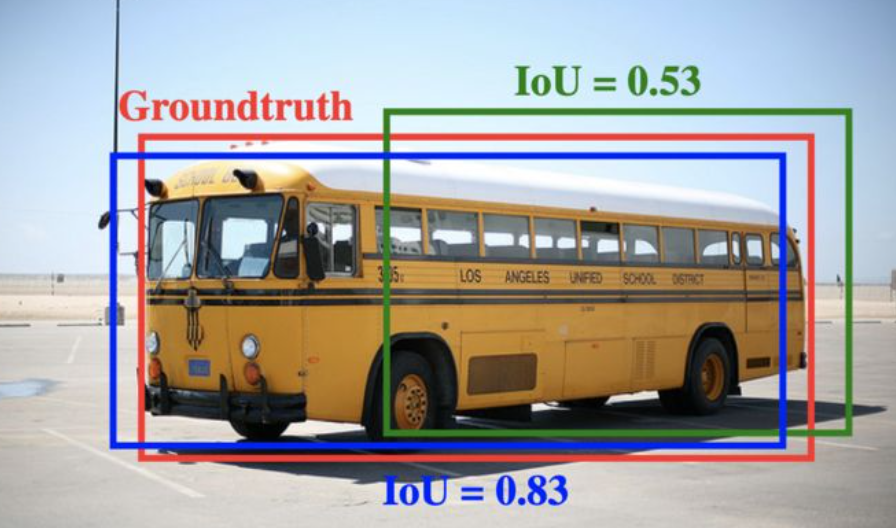


Рис.5 «Визуализация Intersection-over-Union».

1. Precision (насколько точно предсказанный объект попал в истинный Groundtruth), Recall (насколько полно мы нашли все совпадения предсказаний с истинным Groundtruth):

Precision = True positive / (True positive + False positive)

Recall = True positive / (True positive + False negative)

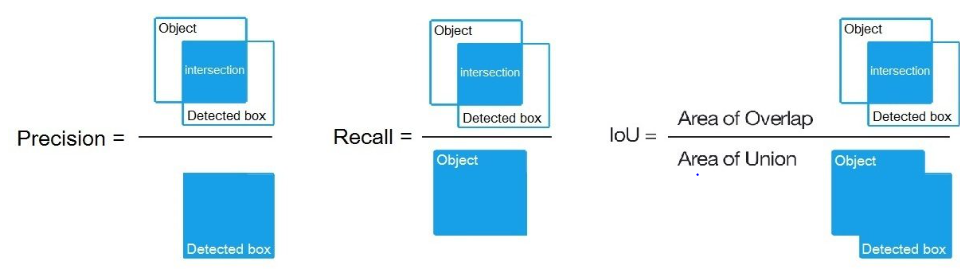
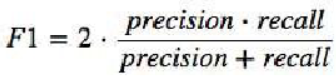


Рис.6 «Визуализация Precision».

1. F1-score (гармоническое среднее между Precision и Recall) и mean Average Precision (среднее значение Precision при различных Recall, усредненное по всем классам). mAP@0.5 означает, что mAP рассчитан для порога IoU>0.5 (средняя доля правильно предсказанных объектов составляет выше 50%):



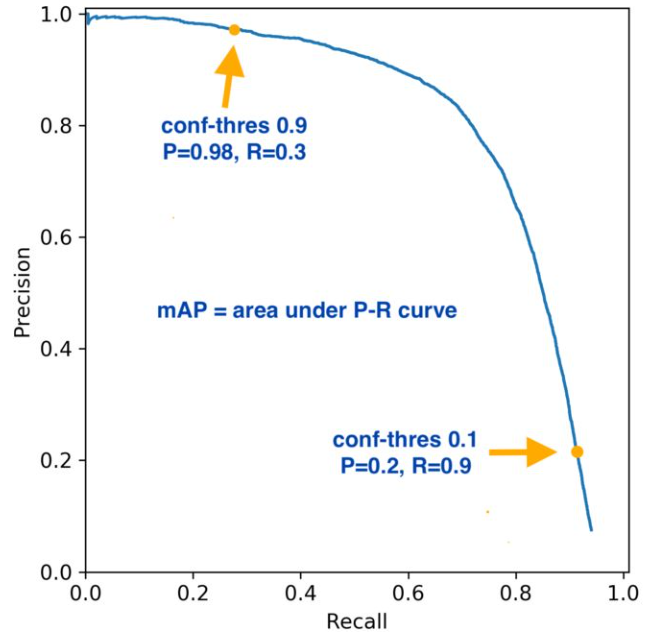


Рис.7 «Визуализация F1-Score».

# **Вариант №2 (дополнительный): SSD300 @VGG16**

Данный вариант был выбран для сравнения с YOLOv8 и обучения модели НС «с нуля» (были использованы как default pretrained-веса VGG16\_Weights.IMAGENET1K\_V1, так и не использованы, - разница нивелировалась в течение нескольких эпох обучения). Обобщенное описание модели SSD приведено ниже[[5]](#footnote-5):

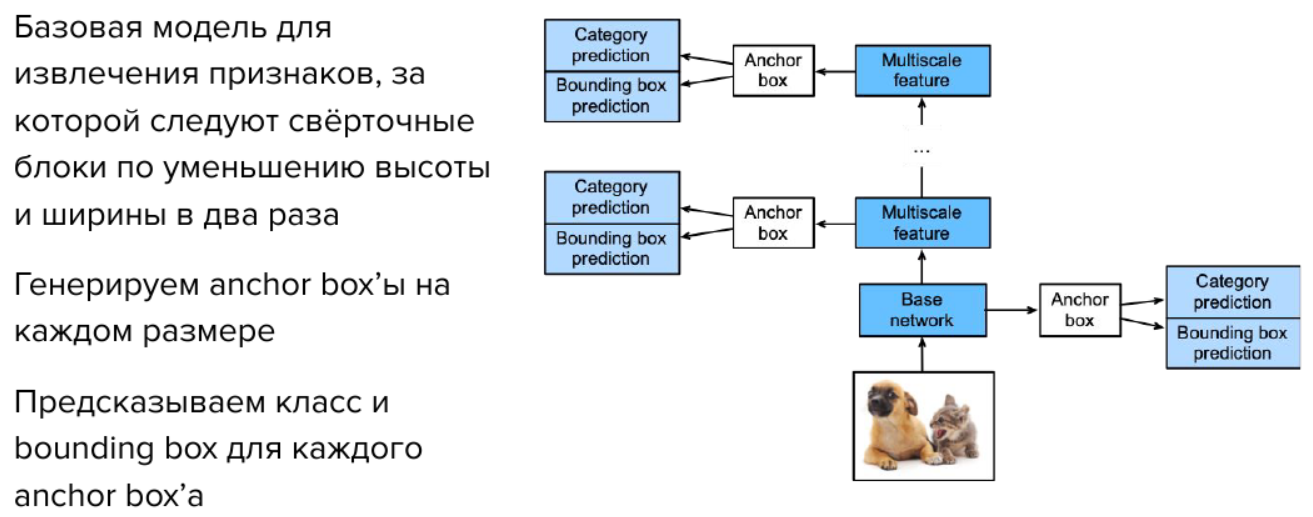


Рис.8 «Визуализация обобщенного устройства модели SSD».

В ходе подготовки данных и обучения модели на SSD300 @VGG16 были проделаны шаги:

1. На основании размеченного train/val-датасета, использованного ранее для YOLOv8, были сформированы файлы разметки в формате JSON (список классов, список train-файлов и train-объектов в них, список валидационных файлов и val-объектов в них);
2. Проведена аугментация загружаемых данных - random crop, переворот, изменение размера и фотометрических параметров (яркости, насыщенности, контраста);
3. Описана структура SSD300 на архитектуре VGG16;
4. На датасете были подобраны гиперпараметры обучения;
5. Проведено обучение на всём датасете;
6. Рассчитаны значения метрики mAP@50 для сравнения c YOLOv8;
7. Проведен референс модели на выборочных снимках из тестового датасета.

# **ИТОГИ ОБУЧЕНИЯ МОДЕЛИ**

Обучение сетей проводилось с использованием GPU на домашнем ПК в конфигурации Intel i9 и GPU nVidia Geforce 4070 Ti (Python-3.11.7, torch-2.2.2+cu118).

Облачные ресурсы (Colab, Kaggle) было решено не использовать в силу ограниченности

бесплатных лимитов.

В серии экспериментов были достигнуты следующие результаты:

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Показатель** | **YOLOv8** | **SSD300@VGG16** |
| 1 | mAP@50 по всем классам (%%), в т.ч.: | 0.984 | 1.0 |
|  | Acura TL Sedan 2012 | 0.987 | 1.0 |
|  | Audi TTS Coupe 2012 | 0.97 | 1.0 |
|  | Buick Verano Sedan 2012 | 0.987 | 1.0 |
|  | Chevrolet Traverse SUV 2012 | 0.995 | 1.0 |
|  | Dodge Charger Sedan 2012 | 0.988 | 1.0 |
|  | Dodge Dakota Club Cab 2007 | 0.975 | 1.0 |
|  | Ford F-450 Super Duty Crew Cab 2012 | 0.989 | 1.0 |
|  | Geo Metro Convertible 1993 | 0.987 | 1.0 |
|  | Hyundai Sonata Hybrid Sedan 2012 | 0.969 | 1.0 |
|  | Mitsubishi Lancer Sedan 2012 | 0.995 | 1.0 |
| 2 | Оптимальная pretrained- модель | YOLOv8s (291 layers, 20915769 parameters, 20915769 gradients, 48.4 GFLOPs) | N/A |
| 3 | Кол-во epoch обучения | 100 | 170 |
| 4 | Batch Size | 16 | 16 |
| 5 | Image Size | All (rescale to 416) | All (rescale to 300x300) |
| 6 | Learning Rate | 0.01 | 0.0001 |
| 7 | Метод оптимизации | SGD | AdamW |
| 8 | Общее время обучения для достижения mAP (п.1) | ~25 минут | ~78 минут |
| 9 | Итоговый (суммарный) Loss | N/A | 1.98 |

Для визуального сравнения результатов были сделаны несколько предсказаний после обучения на датасете:

1. На тестовом (неразмеченном) датасете, который не участвовал в обучении моделей:

a) результаты предсказаний: Audi TT Coupe 2012 (слева – YOLO, справа - SSD):



b) результаты предсказаний: Dodge Charger Sedan 2012 (слева – YOLO, справа - SSD):



**ВЫВОДЫ И ЗАКЛЮЧЕНИЯ**

Задача детекции и классификации автомобилей (марка, модель и год) с применением методов глубокого обучения была успешно реализована. Наилучшие результаты показала архитектура YOLOv8. Вместе с высокой скоростью и качеством на обучении, устойчивостью к выбросам, эта архитектура также показывает малое время обучения, что позволяет использовать её для решения задач реал-тайм детекции и классификации.

Тем не менее, реализация на базе SSD300@VGG16 также показала положительные результаты: несмотря на более длительное время обучения при более высоких метриках, визуально качество детекции находится на высоком уровне.

В ходе работы над проектом были выявлены направления для дальнейшего повышения качества решения задачи:

* Использование вычислений в формате FP16 (вместо FP32) для обучения SSD300@VGG16, что позволит существенно сократить время на обучение модели с сохранением высокой точности предсказаний. При этом YOLOv8 уже по умолчанию обучается в режиме AMP (Automatic Mixed Precision);
* После определения автомобиля (марка, модель и год) определять цвет автомобиля путем добавления дополнительного шага дедукции цвета – потребует дополнительное обучение модели, возможно с разделением на две: детекция автомобиля и детекция цвета;
* До обучить модель на детекцию салона автомобиля исходя из его марки, модели и года.

1. Формулы метрик приведены ниже по тексту [↑](#footnote-ref-1)
2. Весь ноутбук (анализ/обработка данных, обучение моделей, референсы на фото и роликах) выложен на Github [↑](#footnote-ref-2)
3. https://jonathan-hui.medium.com/yolov4-c9901eaa8e61 [↑](#footnote-ref-3)
4. https://www.kdnuggets.com/2020/08/metrics-evaluate-deep-learning-object-detectors.html [↑](#footnote-ref-4)
5. Лекция Нетологии по курсу Deep Learning «Компьютерное зрение» [↑](#footnote-ref-5)