Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий
Кафедра систем искусственного интеллекта

УТВЕРЖДАЮ
Заведующий кафедрой
_____ Г. М. Цибульский
подпись

«____» _____ 2020 г.

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

09.04.01.10 «Интеллектуальные информационные системы»

Использование нейронных сетей для задачи распознавания объектов на городских территориях

Научный руководитель _____ доцент, канд. техн. наук А. В. Пятаева
подпись, дата

Выпускник _____ В. В. Ойнец
подпись, дата

Рецензент _____ доцент, канд. техн. наук В. В. Вдовенко
подпись, дата

Красноярск 2020

Продолжение титульного листа магистерской диссертации по теме «Использование нейронных сетей для задачи распознавания объектов на городских территориях»

Нормоконтролёр _____ доцент, канд. техн. наук А. В. Пятаева
подпись, дата

Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий
Кафедра систем искусственного интеллекта

УТВЕРЖДАЮ
Заведующий кафедрой
_____ Г. М. Цибульский
подпись

«____» _____ 2020 г.

ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ
в форме магистерской диссертации

Студенту Ойнецу Вячеславу Валерьевичу.

Группа КИ18-01-10М, направление 09.04.01 «Информатика и вычислительная техника», магистерская программа 09.04.01.10 «Интеллектуальные информационные системы».

Тема выпускной квалификационной работы «Использование нейронных сетей для задачи распознавания объектов на городских территориях».

Утверждена приказом по университету №_____от_____.

Руководитель ВКР А. В. Пятаева, доцент кафедры систем искусственного интеллекта ИКИТ СФУ, кандидат технических наук.

Исходные данные для ВКР: задание на магистерскую диссертацию.

Перечень разделов ВКР:

- введение,
- городские сцены как объект исследования,
- выбор архитектуры нейронной сети для решения задачи классификации,
- реализация классификации объектов на изображениях,
- заключение,
- список использованных источников.

Перечень графического материала: презентация «Использование нейронных сетей для задачи распознавания объектов на городских территориях».

Руководитель ВКР _____ А. В. Пятаева
подпись

Задание принял к исполнению _____ В. В. Ойнец
подпись

«____» _____ 2020 г.

Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий
Кафедра систем искусственного интеллекта

УТВЕРЖДАЮ
Заведующий кафедрой
_____ Г. М. Цибульский
подпись

«____» _____ 2020 г.

ГРАФИК
НАПИСАНИЯ И ОФОРМЛЕНИЯ
ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЫ
в форме магистерской диссертации

Студент: Ойнец Вячеслав Валерьевич.

Группа КИ18-01-10М, направление 09.04.01 «Информатика и вычислительная техника», магистерская программа 09.04.01.10 «Интеллектуальные информационные системы».

Тема выпускной квалификационной работы «Использование нейронных сетей для задачи распознавания объектов на городских территориях».

График выполнения выпускной квалификационной работы (ВКР) приведён в таблице 1.

Таблица 1 – График выполнения этапов ВКР

Наименование этапа	Срок выполнения	Примечания
Анализ предметной области, подбор литературы	До 14 февраля 2019	Выполнено
Составление плана работы над ВКР	До 14 марта 2019	Выполнено
Разработка и предоставление на проверку первой главы	До 30 мая 2019	Выполнено
Разработка и предоставление на проверку второй главы	До 30 сентября 2019	Выполнено
Разработка и предоставление на проверку третьей главы	До 29 декабря 2019	Выполнено
Доработка ВКР в соответствии с полученными замечаниями	До 15 апреля 2020	Выполнено
Разработка тезисов доклада и подготовка презентации для защиты	До 1 июня 2020	Выполнено
Согласование с руководителем тезисов доклада и презентации	До 15 июня 2020	Выполнено
Прохождение нормоконтроля	До 19 июня 2020	Выполнено
Ознакомление с отзывом и рецензией	До 1 июля 2020	Выполнено
Завершение ВКР к защите с учётом отзыва и рецензии	До 8 июля 2020	

Руководитель ВКР _____ А. В. Пятаева
подпись

Студент группы КИ18-01-10М _____ В. В. Ойнец
подпись

Реферат

Выпускная квалификационная работа по теме «Использование нейронных сетей для задачи распознавания объектов на городских территориях» содержит 46 страниц текстового документа, 29 иллюстраций, 1 таблицу, 30 использованных источников.

Проблема — распознавание объектов, имеющих отношение к дорожному движению, в городских условиях.

Объект исследования — городские сцены, содержащие объекты характерных классов.

Предмет исследования — алгоритмы и нейросетевые архитектуры, предназначенные для обнаружения и классификации объектов.

НЕЙРОННЫЕ СЕТИ, НАБОР ДАННЫХ, ГОРОДСКИЕ СЦЕНЫ, ОБУЧЕНИЕ, КОМПЬЮТЕРНОЕ ЗРЕНИЕ, РАСПОЗНАВАНИЕ ОБЪЕКТОВ, КЛАССИФИКАЦИЯ.

Актуальность работы обусловлена возрастающим применением систем, нуждающихся в восприятии окружающей городской обстановки и классификации различного рода объектов. В рамках работы будет проведено исследование алгоритма классификации изображений, полученных на городских территориях, с использованием нейронных сетей.

Исходя из цели были поставлены следующие задачи:

- выявление объектов интереса, присутствующих на городских сценах,
- выбор архитектуры нейронной сети для решения задачи классификации,
- проведение экспериментального исследования с применением выбранных решений.

СОДЕРЖАНИЕ

Введение

В настоящее время с развитием транспортных систем искусственные нейронные сети всё чаще применяются для восприятия окружающей городской обстановки через сегментацию и классификацию различного рода объектов. Это могут быть объекты, с которыми мы имеем дело на дорогах и вокруг них – такие как различные транспортные средства, пешеходы, дорожные знаки, дорожное полотно, тротуары, и прочее.

Причиной популярности нейронных сетей является то, что они способны самостоятельно выявить нужную закономерность на основе представленного набора данных. Обнаружение объекта – это процедура определения класса, к которому принадлежит объект, и оценки местоположения объекта путем вывода ограничивающего прямоугольника вокруг объекта. Обнаружение объекта является главным шагом в любой деятельности по визуальному распознаванию.

Для глубокого обучения возникает необходимость в большом массиве данных. Объёмные наборы данных также используются для изучения более сложных закономерностей, а для оптимизации используются параллельные вычисления. Поскольку нейронные сети нуждаются в большом количестве входных данных для обучения, поэтому были созданы различные наборы данных, которые обеспечивают обучение и проверку алгоритмов компьютерного зрения и, следовательно, играют решающую роль в проведении исследований.

Чтобы нейронная сеть имела возможность классифицировать искомый объект, предварительно проводится её обучение на наборах данных, содержащих примеры изображений, которые потребуется классифицировать. Наборы данных также активно используются для обучения нейронных сетей систем навигации мобильных устройств. Применяемые наборы данных могут быть очень разнообразными по характеру включаемых изображений.

В последние несколько лет были разработаны различные архитектуры нейронных сетей – например, YOLO, SSD, Faster R-CNN, RFCN, MobileNet. Все они нуждаются в инструментах для реализации, поэтому предлагаются также

различные фреймворки для работы с ними. Они поддерживают разные интерфейсы программирования, значительно ускоряют скорость работы архитектур. Например, такие фреймворки, как Keras, Tensorflow, PyTorch.

Актуальность задач распознавания объектов подтверждается наличием соответствующей специальности в перечне Всероссийской аттестационной комиссии Российской Федерации (ВАК РФ) – 05.13.17 Теоретические основы информатики, п. 7 (Разработка методов распознавания образов, фильтрации, распознавания и синтеза изображений, решающих правил. Моделирование формирования эмпирического знания).

В данной работе поставлена цель разработать алгоритм классификации изображений, полученных на городских территориях, с использованием нейронных сетей.

Для достижения данной цели решаются следующие задачи:

- выявление объектов интереса, присутствующих на городских сценах (выделение классов);

- выбор архитектуры нейронной сети и фреймворка для решения задачи классификации;

- проведение экспериментального исследования с применением выбранных решений.

1 Городские сцены как объект исследования

Семантическое понимание городской сцены перестало восприниматься как недостижимая цель, и на этом сосредоточены многие последние исследования в области компьютерного зрения. Поэтому разрабатываются соответствующие наборы данных, включающие широкий спектр сложных уличных сцен, в различных городах.

Выбирая подход к маркировке полученных городских изображений, определяют различное количество визуальных классов для аннотации, которые могут быть сгруппированы, например, в такие категории: плоская поверхность, конструкция, природный объект, транспортное средство, небо, человек, животное, пустота. [1] Классы выбираются на основе их частоты попадания на изображения, актуальности, удобства выполнения аннотаций, а также для обеспечения совместимости с другими разработанными наборами данных. Слишком редкие классы исключаются из рассмотрения.

В работе [2] принималось, что изображение принадлежало району некоторого города, если численность его населения составляла ≥ 50000, в противном случае – относилось к сельской местности.

Чтобы удовлетворять требованиям разнообразия, отбираются изображения, охватывающие различные времена года, различные погодные условия (солнечная, дождливая, облачная, туманная или снежная погода), а также различные условия освещения (на рассвете, в сумерках или ночью).

Изображения являются предпочтительными для рассмотрения и аннотирования в случае, если они содержат:

- несколько классов объектов,
- несколько объектов, отнесённых к какому-то из классов,
- объекты, принадлежащие к редким классам.

Когда изображения отобраны для аннотирования, их рассматривают с точки зрения выделения на них наиболее далёких от точки съёмки классов с по-

следующим приближением. При аннотировании рекомендуется начинать с классов объектов, расположенных позади. Так, например, аннотирование обычно начинается с неба (наиболее удаленный класс от центра камеры) и постепенно приближаются к переднему плану. Следовательно, небо может быть аннотировано одним многоугольником, несмотря на то, что на изображении могут присутствовать несколько областей. Это даёт преимущество в виде того, что области, перекрываемые объектами, расположенными ближе к камере, могут быть нарисованы быстрее, и их не обязательно корректировать в случае уточнения маркировки объектов переднего плана.

1.1 Наборы данных о типах транспортных средств

Распознавание типа транспортного средства признаётся ключевой технологией для построения систем наблюдения за дорожным движением, поэтому классификация транспортных средств по типам является важной частью интеллектуальных транспортных систем [3], [4]. Соответственно, применяются наборы данных, содержащие изображения автомобилей с разделением на популярные типы, такие как легковой автомобиль, микроавтобус, мотоцикл, грузовик и др. Примерами таких наборов данных являются Stanford Car Dataset [5], Caltech [6]. Также возможно, а иногда и предпочтительно, использование наборов данных собственной разработки [3], [7].

1.2 Наборы данных о дорожных знаках

Не менее важной функцией для управления в транспортных системах является способность распознавать дорожные знаки [8], [9]. Системы распознавания дорожных знаков предлагают дополнительную помощь водителю, предупреждая о наличии таких знаков на дороге. Так снижаются риски в ситуациях отвлечения внимания водителя, его усталости, недостаточного освещения или плохих погодных условий. Всё это способствует повышению безопасности движения.

Для оценки качества работы нейронных сетей в этом случае используются специальные наборы данных, содержащих изображения дорожных знаков– такие, как German Traffic Sign Recognition Benchmark [10], German Traffic Sign Detection Benchmark [11], Belgium Traffic Sign Classification [12].

1.3 Наборы данных с объектами в сложных погодных условиях

Разработаны специальные наборы данных для обучения нейронных сетей на видеоданных, полученных на открытых пространствах в сложных погодных условиях [13]. Под сложными погодными условиями понимают присутствие тумана или атмосферных осадков, недостаточную освещенность и пр. Набор данных FROSI Database (Foggy ROad Sign Images) [14] используется для работы нейронной сети, определяющей дальность обнаружения объектов в условиях тумана. Для решения проблемы распознавания объектов в условиях пониженной освещенности для нейронных сетей требуются большие объёмы обучающих данных, которые имеют различную интенсивность освещения.

1.4 Наборы данных с объектами в NIR-диапазоне

В свободном доступе наборов данных, подготовленных в условиях низкой интенсивности освещения, обнаружить не удалось, на основании чего можно сделать предположение, что задача по сбору и подготовке таких наборов данных представляет некоторую сложность. Возможным выходом из этой ситуации рассматривается использование изображений в видимом и ближнем инфракрасном (NIR-изображения) диапазоне спектра [15]. NIR-изображение менее чувствительно к изменению условий освещенности. По сравнению с NIR-изображениями, изображения в видимом свете предоставляют большую детализацию объектов. Поэтому имеет смысл использовать совместно как изображение в видимом свете, так и изображение в ближнем инфракрасном диапазоне, чтобы решить проблему изменения освещенности и в то же время сохранить преимущество

изображений в видимом свете. Для реализации этого подхода используются наборы данных как в видимом свете, так и сделанные в ближнем инфракрасном диапазоне (CASIA NIR [16], PolyU NIR [17], HIT LAB2 [18]).

1.5 Подход к аннотированию изображений в наборах данных

Важным этапом предварительной обработки данных при обучении нейронных сетей является аннотирование (или маркировка) изображений. Любая ошибка или неточность на этом этапе может отрицательно повлиять на качество набора данных. Кроме того, общая производительность модели может быть сильно занижена, существенно возрастает количество ошибок распознавания. Существует три варианта аннотирования изображений: ручное, полуавтоматическое и автоматическое аннотирование. Такие данные, как изображение автомобиля, голос и текст, обычно маркируются рабочими группами в ручном режиме. Для полуавтоматического или для аннотирования без участия человека, применяются специальные технологии, например, технология VDL (virtual detections line-based methods) [19]. При решении некоторых задач автоматическое аннотирование является неотъемлемой частью процесса для получения искомого результата. В статье [20] рассмотрен метод определения критических ситуаций, связанных с дорожным движением. В отличие от известных решений, для обнаружения и распознавания нормальных или критических ситуаций в нём используется классификация с последующим усилением на основе нескольких кадров видеопотока и алгоритмом автоматической аннотации.

1.6 Влияние структуры набора данных

При изучении вопроса об эффективности обучения нейронной сети рассматривается не только содержание набора данных, но и его структура. Например, в статье [21] показано, что качество аннотирования изображений набора

данных влияет на точность обнаружения и распознавания объектов нейронной сетью. Были выбраны три лучших и три худших класса по средней точности обнаружения. Учитывались также площади объектов и общая площадь изображения. Эксперимент показал, что при уменьшении средней величины площади распознанного объекта средняя точность обнаружения нейронной сетью снижалась, а количество нераспознанных объектов увеличивалось. Можно предположить, что неудовлетворительные результаты распознавания объектов бывают связаны с наличием в наборе данных большого количества аннотированных объектов небольшого размера.

2 Выбор архитектуры нейронной сети для решения задачи классификации

В данной главе приведены популярные архитектуры нейронных сетей, разработанных в последнее время, рассмотрены их особенности и сделан выбор для проведения экспериментальной части.

В практике использования моделей нейронных сетей наряду с самостоятельными архитектурами встречаются так называемые «мета-архитектуры», когда одни архитектуры, используемые для извлечения признаков, сочетаются с другими, используемыми для распознавания. При этом изучаются компромиссы между точностью и скоростью работы в разных мета-архитектурах, блоках извлечения признаков. Например, выбор блока извлечения признаков сильно изменяет результаты работы на различных мета-архитектурах.

2.1 Faster R-CNN ResNet101

Существует серия архитектур на основе R-CNN (Regions with Convolution Neural Networks features), которая включает R-CNN, Fast R-CNN, Faster R-CNN [22]. Для обнаружения объекта на изображении с помощью механизма Region Proposal Network (RPN) выделяются ограниченные регионы (bounding boxes). Первоначально вместо RPN применялся более медленный механизм Selective Search. Затем выделенные ограниченные регионы подаются на вход обычной нейросети для классификации. В архитектуре R-CNN есть явные циклы «for» перебора по ограниченным регионам, всего до 2000 прогонов через внутреннюю сеть AlexNet. Из-за явных циклов «for» замедляется скорость обработки изображений. Количество явных циклов, прогонов через внутреннюю нейросеть, уменьшается с каждой новой версией архитектуры, а также проводятся десятки других изменений для увеличения скорости.

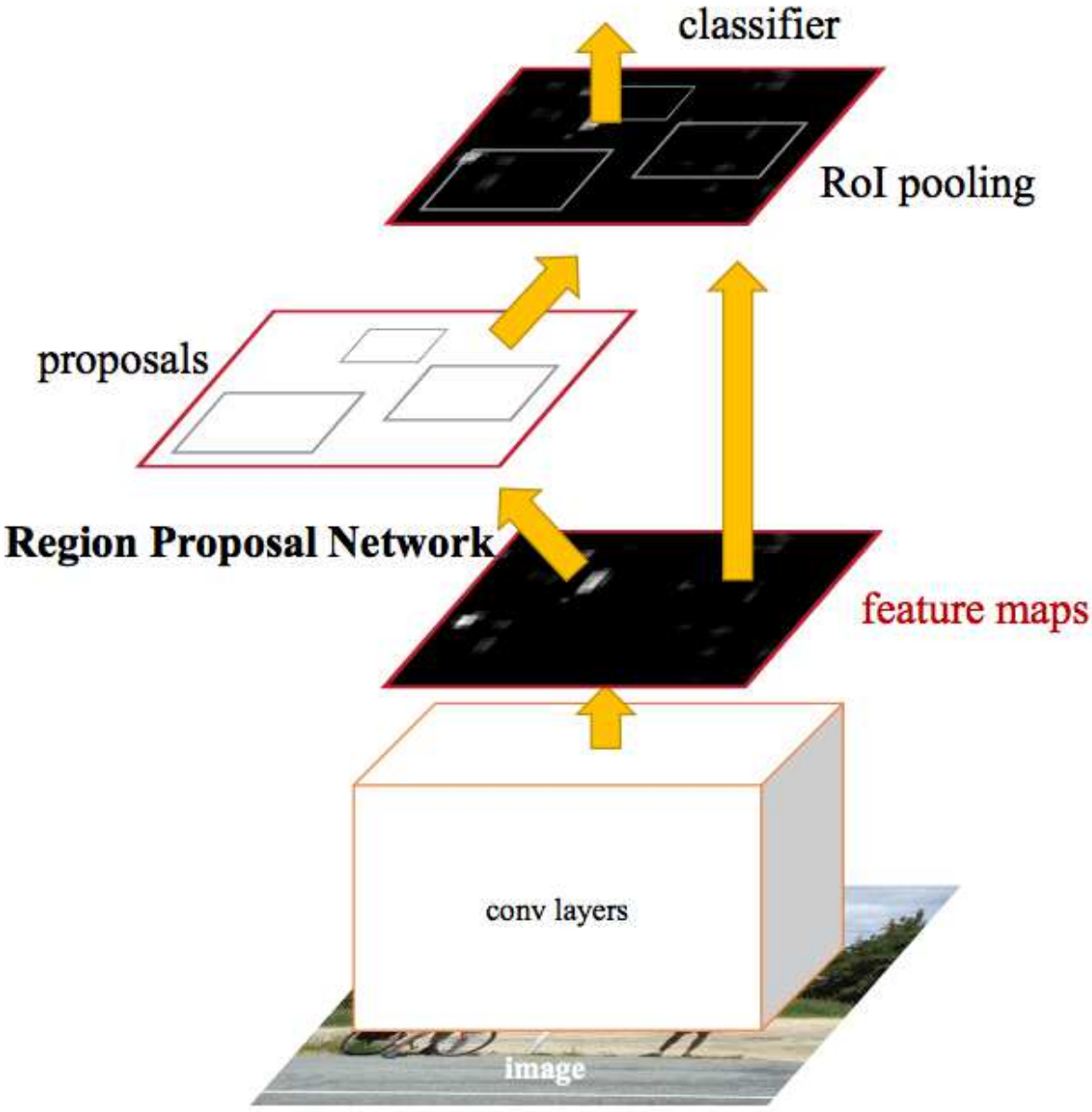

Рисунок 2.1 – Архитектура Faster R-CNN

ResNet — это сокращенное название для Residual Network (дословно – «остаточная сеть») [23]. Основной элемент ResNet – Residual-блок (остаточный блок) с shortcut-соединением, через которое данные проходят без изменений. Res-блок представляет собой несколько свёрточных слоёв с активациями, которые преобразуют входной сигнал x в F(x). Shortcut-соединение — это тождественное преобразование x → x.

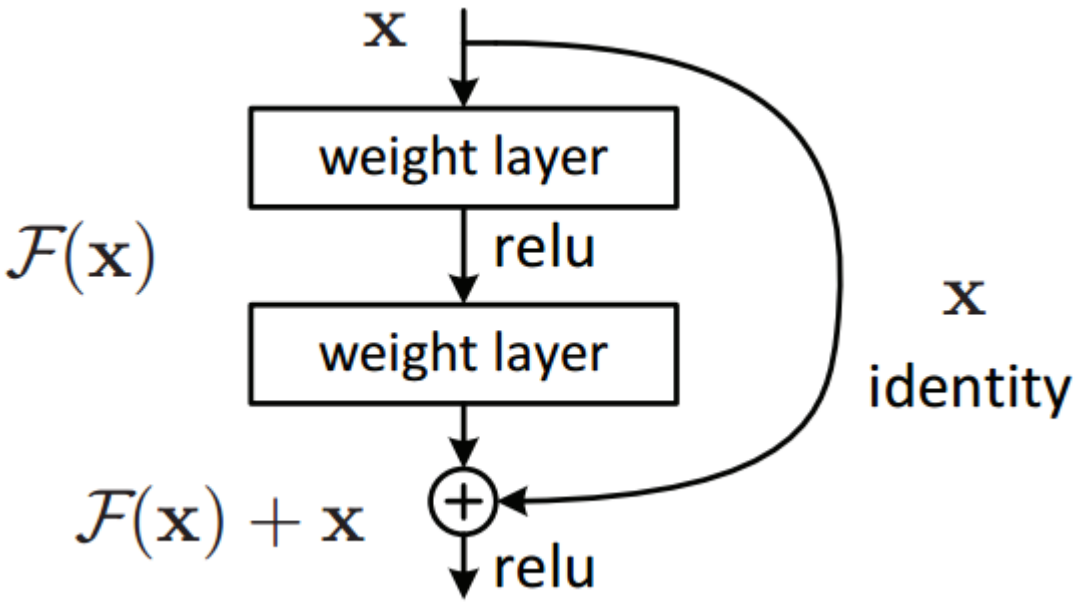

Рисунок 2.2 – Базовый элемент архитектуры ResNet

2.2 RFCN ResNet101

Название означает Region-based Fully Convolutional Network [24].

Это метод, в котором разработчики отказались от применения ресурсоёмких подсетей для обработки отдельных регионов изображения сотни раз на каждой картинке. Здесь детектор по регионам полностью свёрточный и производит совместные вычисления на всём изображении целиком.

Хотя модель Faster R-CNN относительно неплохо показывает себя в задачах обнаружения объектов, важно отметить, что эта архитектура значительно медленнее, чем более современные подходы, такие как RFCN.

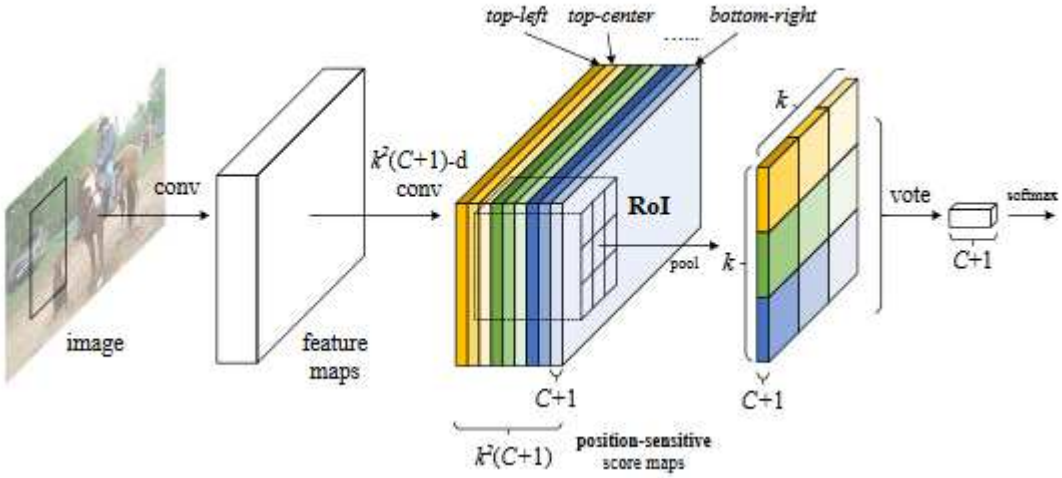

Рисунок 2.3 – Архитектура RFCN

2.3 UNet v2

Архитектура состоит из стягивающего пути для захвата контекста и симметричного расширяющегося пути, который позволяет осуществить точную локализацию [25].

Для UNet характерно достижение высоких результатов в различных реальных задачах, особенно для биомедицинских приложений, используя при этом небольшое количество данных.

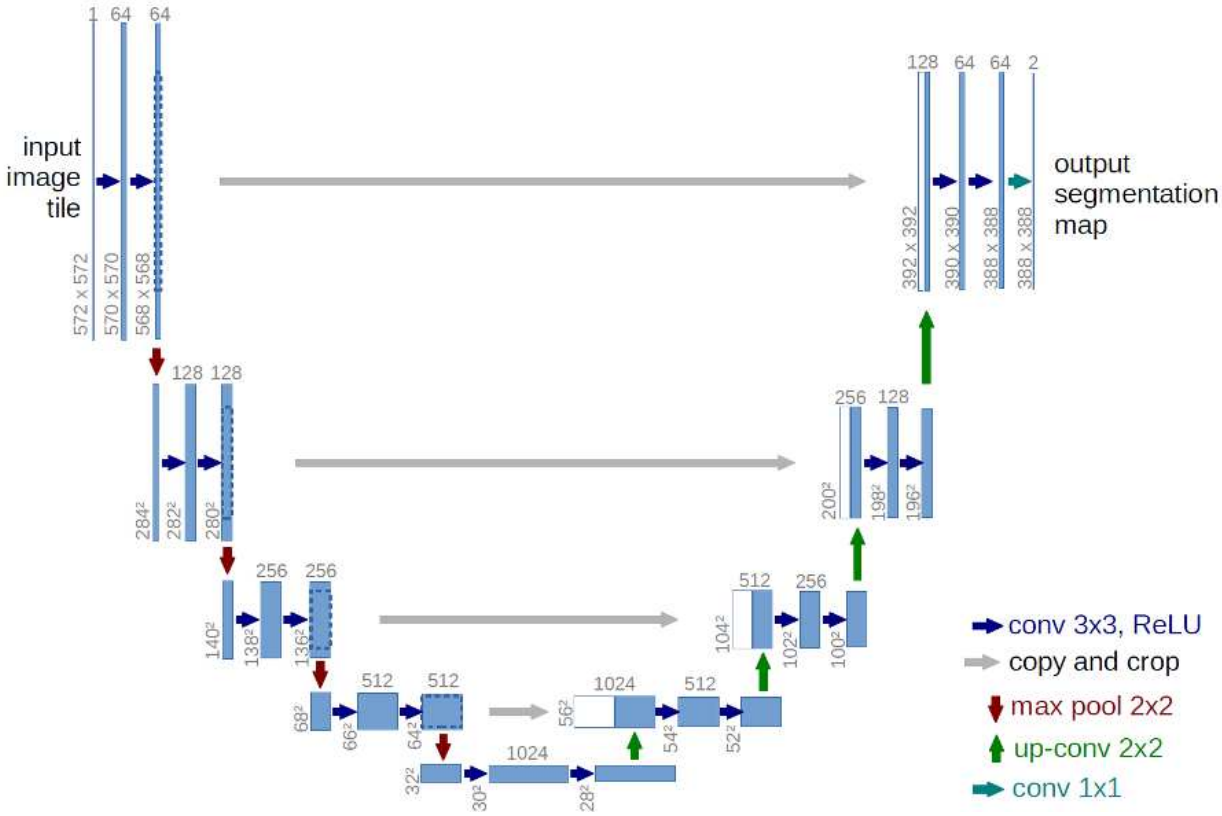

Рисунок 2.4 – Архитектура UNet

2.4 YOLO v3

YOLO – значит You Look Only Once (Вы смотрите только один раз). Главная особенность этой архитектуры по сравнению с другими состоит в том, что большинство систем применяют нейронную сеть несколько раз к разным регионам изображения, в YOLO нейронная сеть применяется один раз ко всему изображению сразу [26]. Сеть делит изображение на своеобразную сетку и предсказывает расположение ограничивающих прямоугольников и вероятности того, что там есть искомый объект для каждого участка.

Плюсы данного подхода состоит в том, что нейронная сеть смотрит на всё изображение сразу и учитывает контекст при детектировании и распознавании объекта. Так же YOLO в 1000 раз быстрее чем R-CNN и около 100 быстрее чем Fast R-CNN.

YOLO v3 – это усовершенствованная версия архитектуры YOLO. Она состоит из 106 свёрточных слоев и лучше детектирует небольшие объекты по сравнению с её предшественницей YOLO v2. Основная особенность YOLO v3 состоит в том, что на выходе есть три слоя, каждый из которых рассчитан на обнаружение объектов разного размера.

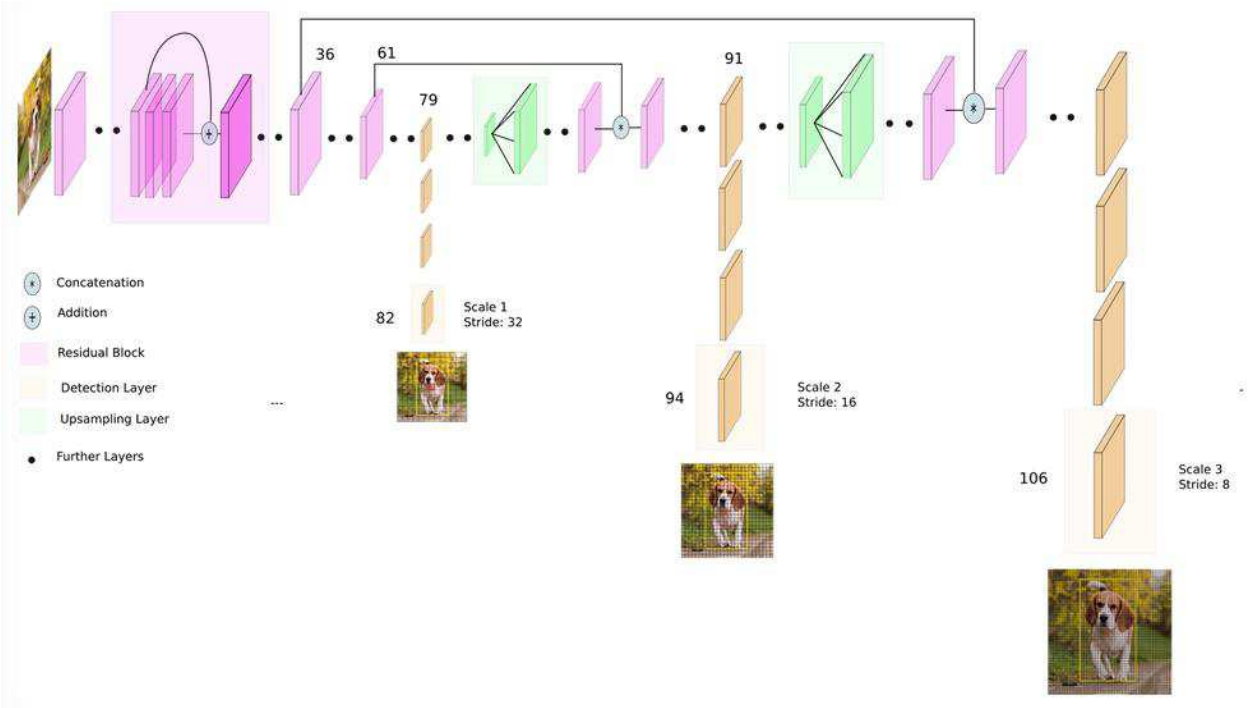

Рисунок 2.5 – Архитектура YOLO v3

2.5 DeepLab v3 Plus

DeepLab – это современная модель семантической сегментации, разработанная Google и находящаяся в открытом доступе. Архитектура DeepLab v3 содержит энкодер-декодер для получения четких границ объекта путем постепенного восстановления пространственной информации [27]. Такой вид архитектуры успешно применялись ко многим задачам компьютерного зрения, включая обнаружение объектов, позы человека, а также семантическую сегментацию.

DeepLab v3 Plus ещё более улучшает Deep Lab v3, добавляя простой, но эффективный модуль декодера для ещё большего повышения точности сегментации, особенно вдоль границ объекта.

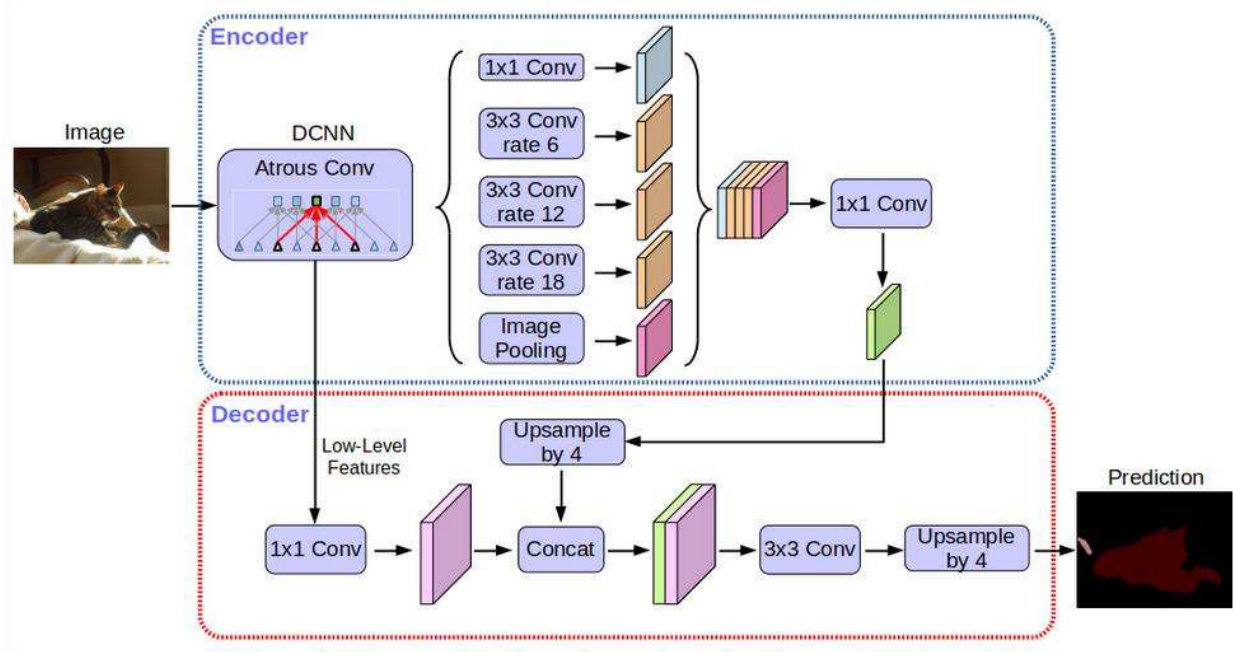

Рисунок 2.6 – Архитектура DeepLab v3 Plus

2.6 SSD MobileNet v2

SSD (Single Shot MultiBox Detector) [28] – в этой архитектуре используются наиболее удачные особенности архитектуры YOLO и добавляются новые, чтобы повысить скорость и точность нейронной сети. Отличительная особенность: различение объектов за один прогон с помощью заданной сетки окон (default box) на пирамиде изображений. Пирамида изображений закодирована в сверточных тензорах при последовательных операциях свертки и пулинга (при операции max-pooling пространственная размерность убывает). Таким образом, определяются как большие, так и малые объекты за один прогон сети.

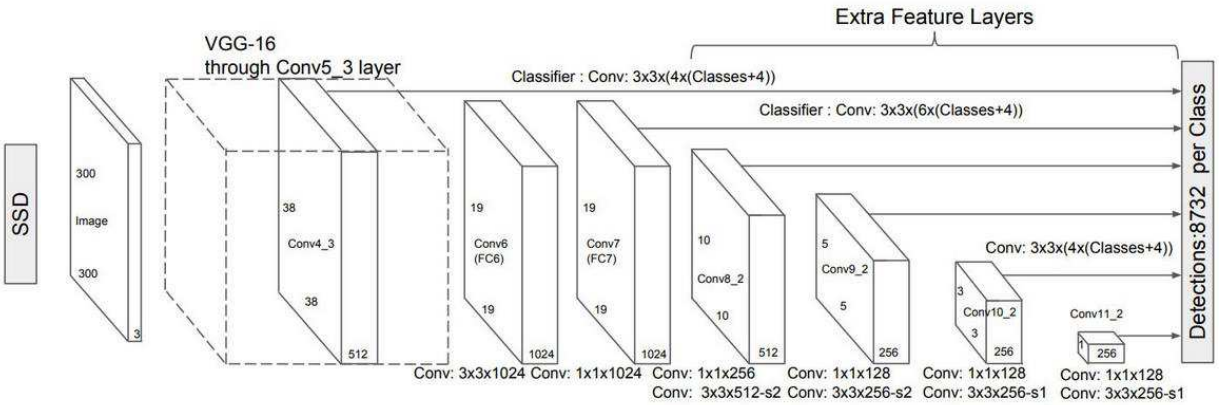

Рисунок 2.7 – Архитектура SSD

В структуре MobileNet свёрточная часть состоит из одного обычного свёрточного слоя с 3x3 свёрткой в начале и тринадцати блоков с постепенно увеличивающимся числом фильтров и понижающейся пространственной размерностью тензора.

Особенностью данной архитектуры является отсутствие max pooling-слоёв. Вместо них для снижения пространственной размерности используется свёртка с параметром stride, равным 2.

В MobileNet v2 имеется ключевая особенность: блок, называемый авторами расширяющим свёрточным блоком (в оригинале expansion convolution block или bottleneck convolution block with expansion layer), содержит слой 1x1-свертки с линейной функцией активации, понижающей число каналов [29]. Авторы статьи выдвигают гипотезу, что «целевое многообразие» высокой размерности, полученное после предыдущих шагов, можно «уложить» в подпространство меньшей размерности без потери полезной информации, что, собственно и делается на этом шаге.

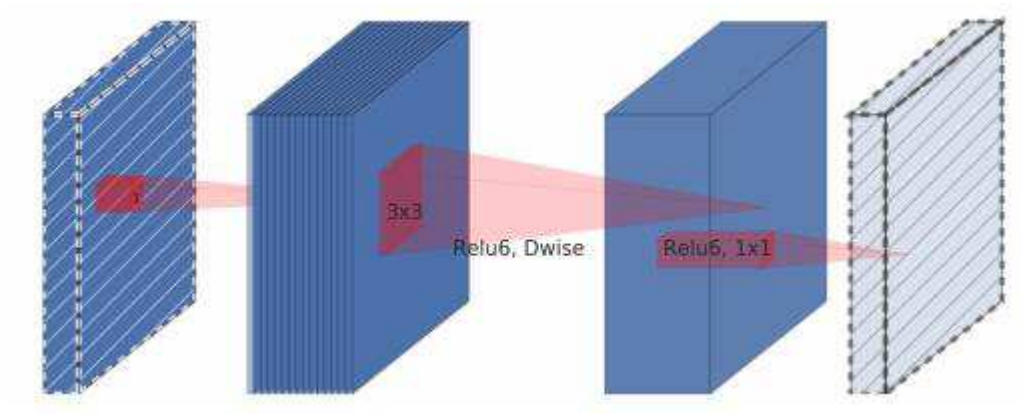

Рисунок 2.8 – Архитектура MobileNet v2

SSD + MobileNet v2 – комбинация из двух архитектур нейронных сетей. Первая сеть MobileNet v2 работает быстро и увеличивает точность распознавания. MobileNet v2 применяется вместо VGG-16, которая использовалась ранее. Вторая сеть SSD определяет местоположение объектов на изображении.

2.7 Выбор архитектуры

Учитывая особенности и области применения рассмотренных архитектур нейронных сетей, для проведения дальнейших экспериментов отберём те, которые построены с учётом недостатков более ранних разработок и специализированы на классификации объектов на городских сценах:

- RFCN ResNet101,
- YOLO v3,
- SSD MobileNet v2.

3 Реализация классификации объектов на изображениях

3.1 Общая информация

Supervisely – это веб-платформа для управления, аннотирования, проверки и экспериментов с наборами данных и нейронными сетями для компьютерного зрения [30]. Supervisely помогает как пользователям с опытом в области машинного обучения, так и без опыта, создавать современные приложения для компьютерного зрения. Платформа оптимизирована для того, чтобы организовать весь рабочий процесс – от необработанных данных до развертывания нейронных сетей – обходясь без программирования.

Supervisely позволяет управлять наборами данных, моделями нейросетей, пользовательскими расширениями и многими другими объектами.

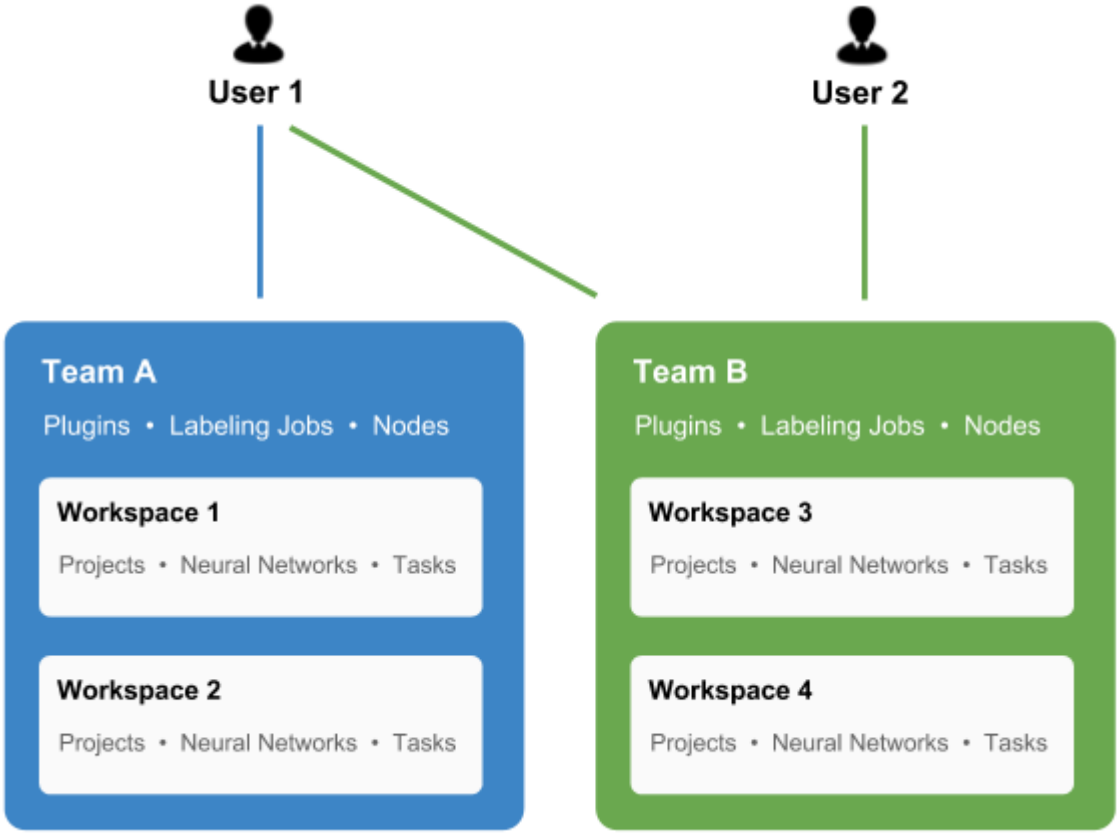

Рисунок 3.1 – Команды

Команда – это группа пользователей и ресурсов. Пользователи в одной команде используют одни и те же ресурсы, например, проекты и модели. В любой момент пользователь работает в определенной команде, и все новые элементы, такие как проекты, которые он создаёт, будут создаваться в этой команде. Нельзя создавать проекты вне команды или иметь один и тот же проект в нескольких командах. Команду можно поменять в любой момент или создать новую в меню.

На данный момент доступны следующие роли участников команд:

- Администратор: имеет полный доступ в команде.

- Разработчик: аналогичен администратору, но может удалять только созданные им объекты и не может приглашать новых членов в команду.

- Менеджер: не имеет доступа к таким вещам, как Python Notebooks или нейронные сети, но может просматривать и изменять проекты и задания по аннотированию.

- Аннотатор: имеет доступ только к одной задаче – аннотированию (маркировке) объектов (Labeling Job).

- Наблюдатель: может только просматривать объекты в команде.

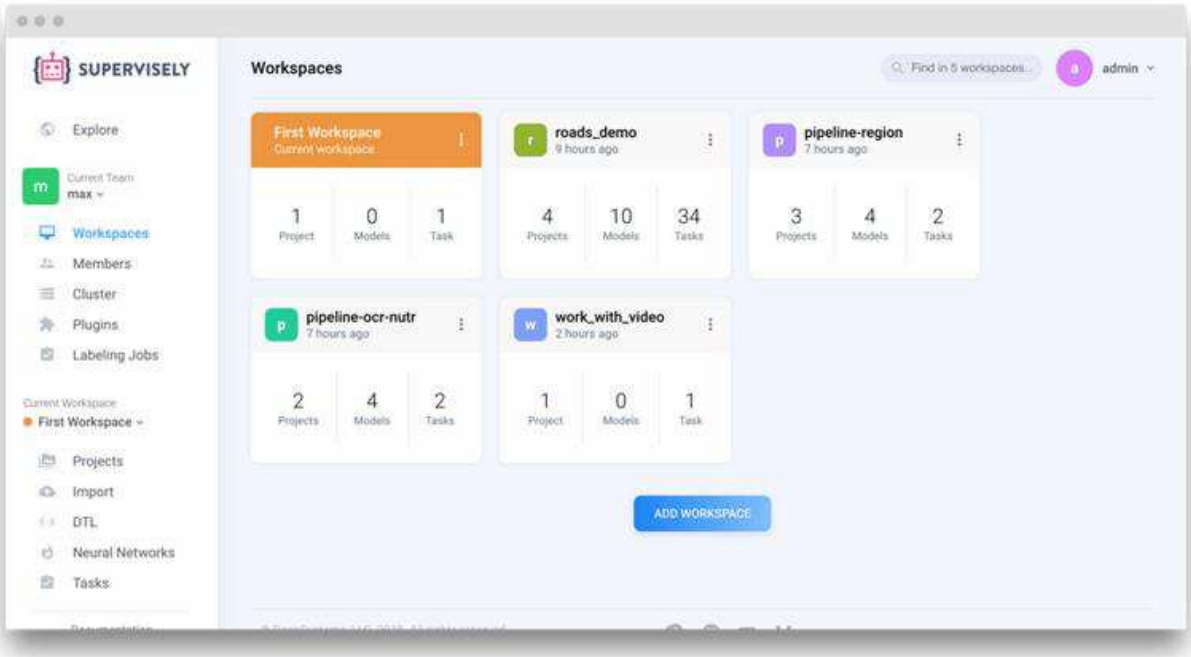

Рисунок 3.2 – Рабочие области

В любой момент участник работает в определенной рабочей области, и все созданные им проекты, нейронные сети и задачи становятся частью текущей рабочей области и текущей команды.

3.2 Импорт данных

Чтобы загрузить данные в Supervisely или добавить готовые наборы данных, имеется модуль Import.

Импорт данных позволяет загружать данные из внешних источников и автоматически преобразовывать их в формат json. Поддерживаемые форматы изображений: bmp, jpeg/jpg, png.

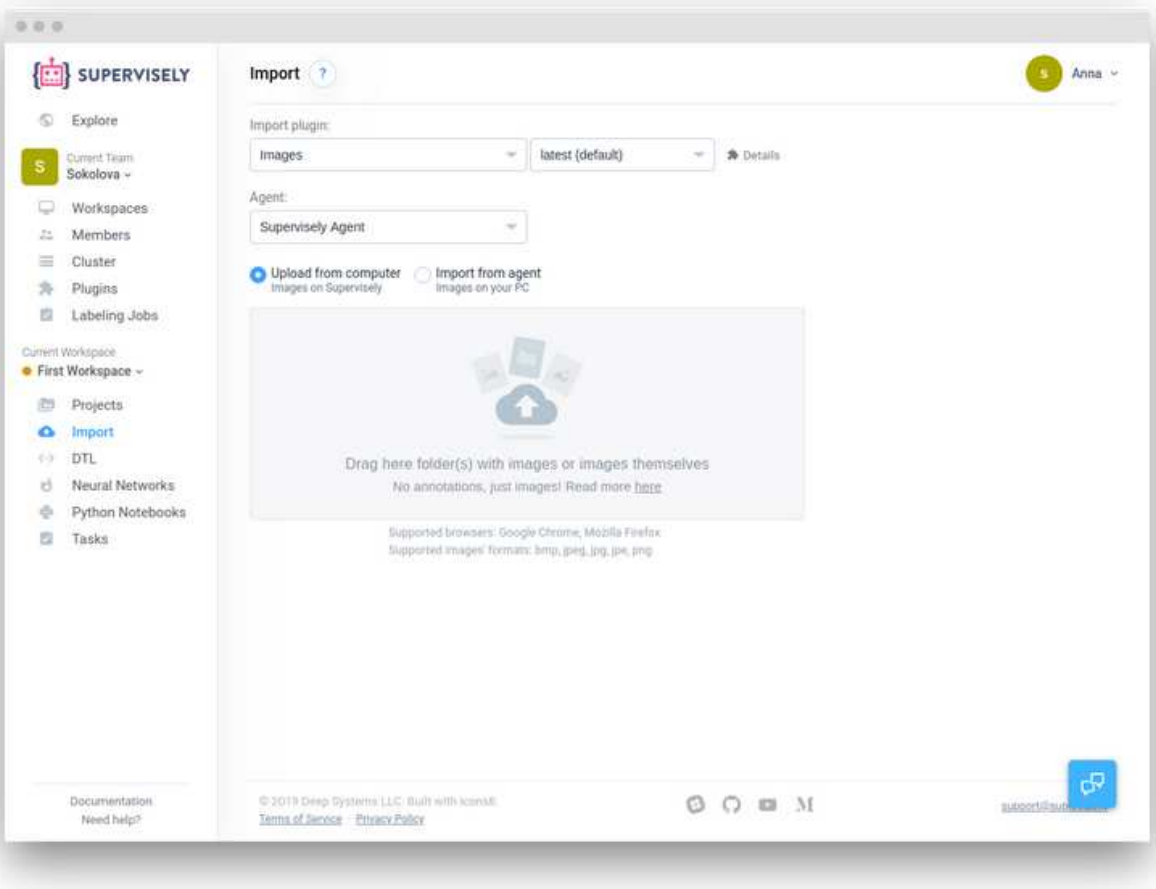

Рисунок 3.3 – Импорт данных

Для предотвращения ошибок настоятельно рекомендуется избегать использования нелатинских символов в именах файлов.

Существует два способа импорта данных: создать локальные (по отношению к сервису) файлы, то есть загруженные на сервер, или указать удалённые (по отношению к сервису) файлы, то есть хранящиеся на компьютере пользователя.

Чтобы создать локальные файлы, достаточно «перетащить» файлы и папки прямо в соответствующее поле браузера. Для этого нужно открыть раздел Import. Загрузка сразу целыми папками облегчает работу, если нужно загружать большое количество файлов.

Существует несколько поддерживаемых типов форматов:

- Изображения (по умолчанию): используется для загрузки папок с изображениями,

- Формат Supervisely: используется для загрузки папок с изображениями и их аннотациями в формате Supervisely,

- Aberystwyth,

- Kitti,

- CityScapes: используется для загрузки папки с набором данных Cityscapes,

- Mapillary: используется для загрузки папки с набором данных Mapillary,

- PascalVoc: используется для загрузки папки с набором данных PascalVoc.

Каждый из перечисленных форматов имеет свои требования к структуре файлов. Подробности о требуемой структуре можно узнать в документации.

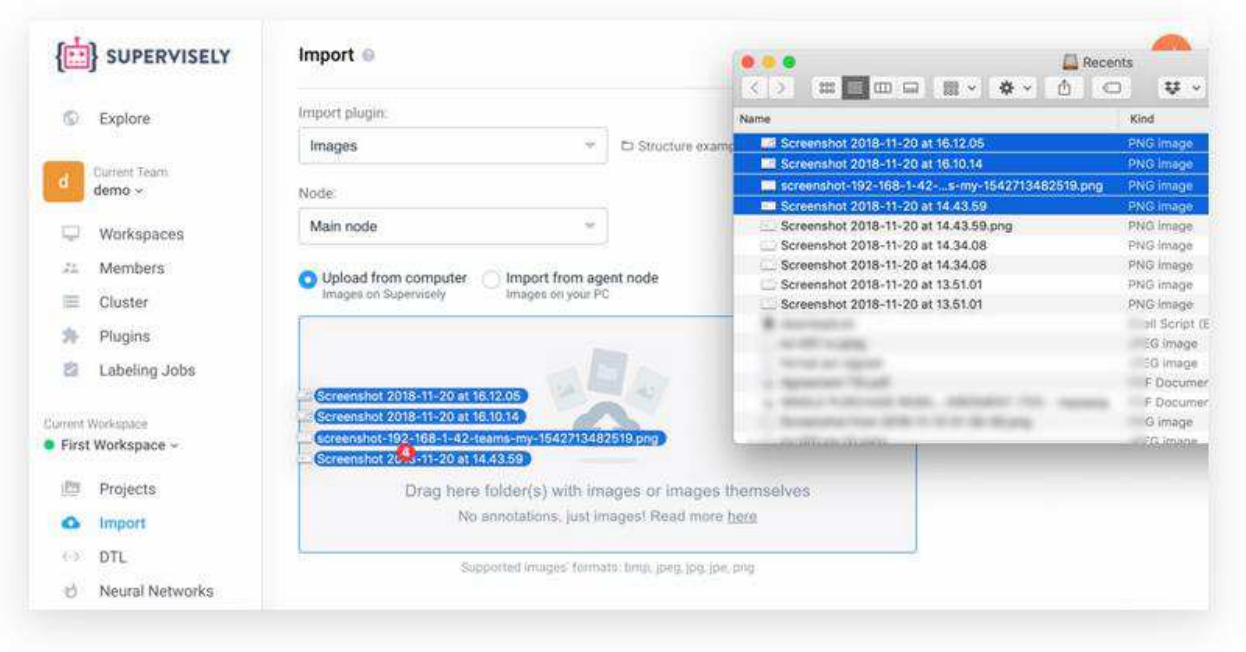

Рисунок 3.4 – Структура файлов

После перетаскивания файлов или папок произойдёт перенаправление на страницу «Статус задачи» (Task Status), где модуль Import будет загружать все файлы и папки, и отображается индикатор выполнения. Во время процесса загрузки не следует закрывать браузер. После завершения процесса импорта будет добавлен новый проект.

Также можно выбрать способ, при котором файлы изображений хранятся на частном компьютере пользователя, и полностью отказаться от загрузки. Для этого необходимо добавить скрипт «агента» в вычислительный кластер. У этого решения есть некоторые плюсы и минусы по сравнению с загрузкой.

Плюсы:

- Сохраняется приватность данных. Никто, включая Supervisely, не имеет доступа к вашим файлам.

- Эксперименты проводятся быстрее. При запуске процедур Train, Inference, DTL не нужно скачивать и загружать файлы с серверов – агент может сразу начать обработку.

Минусы:

- Сложнее организовать общий доступ. Разные пользователи могут иметь доступ к одним и тем же файлам, только если они работают на одном и том же узле вычислительной сети.

- Нет резервного копирования данных. Если пользователь удаляет папку, доступ к наборам данных будет навсегда потерян.

- Аннотация изображений занимает больше времени. Каждый раз, когда открывается изображение с местного узла, браузер должен загрузить его с этого компьютера через Интернет.

Итого: данное решение подходит для одиночной учетной записи, но не оптимально для групповой работы.

Чтобы задать удалённые данные, нужно поместить файлы в папку импорта, которая находится в папке агента на компьютере. По умолчанию это папка ~ / .supervisely-agent / <agent-token> / import.

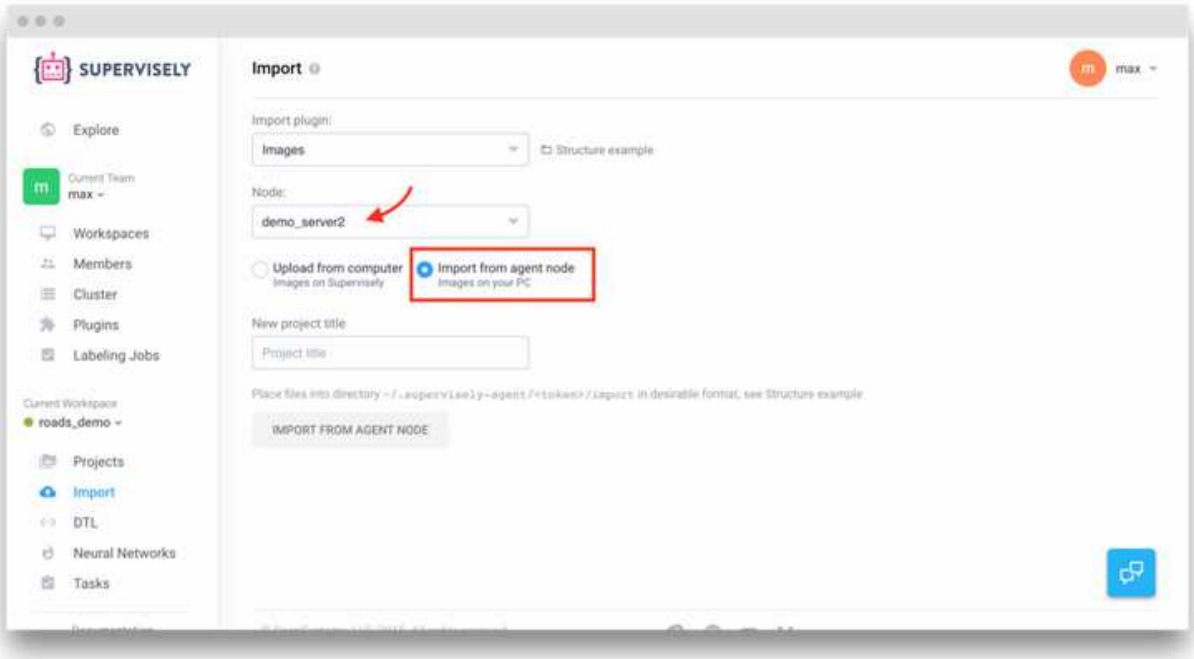

Рисунок 3.5 – Импорт через агента

Затем нужно перейти в раздел Import и выбрать опцию «Import from agent node», выбрать имя для проекта и узел, в котором хранятся нужные файлы. После этого нажать кнопку «Import from agent node», чтобы начать импорт. Так же, как

и в предыдущем варианте, произойдёт переход на страницу «Статус задачи» (Task Status), где будет показан индикатор выполнения.

После завершения процесса импорта будет добавлен новый проект. При этом загружаются не сами изображения, а только метаинформация.

3.3 Проекты

В разделе «Проекты» отображаются все имеющиеся проекты и наборы данных. Наборы данных содержат изображения и связанные с ними аннотации. Одно и то же изображение может принадлежать разным проектам или наборам данных. Однако, изображения не дублируются – можно загружать изображения несколько раз, но в системе будет храниться только одна версия.

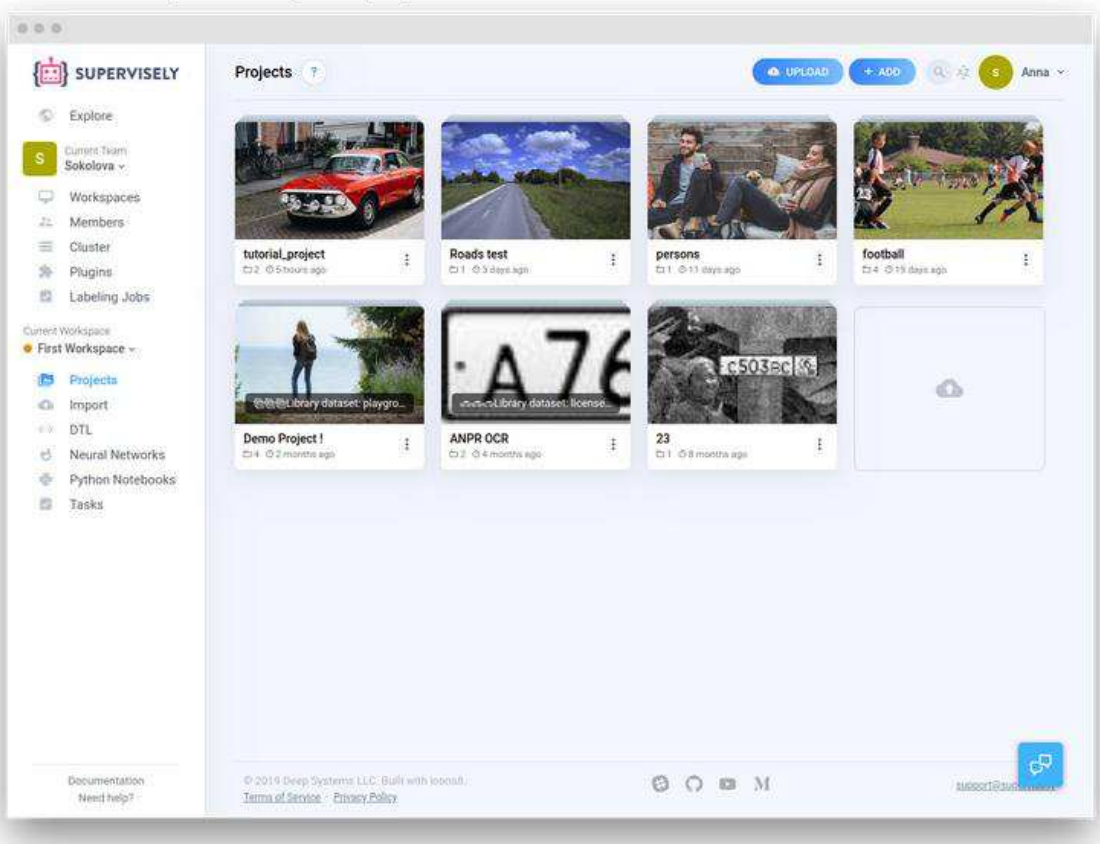

Рисунок 3.6 – Раздел проектов

Классы присваиваются объектам внутри проекта и доступны для любого набора данных внутри этого проекта.

Каждый проект содержит один или несколько наборов данных. Набор данных содержит аннотированные (размеченные) изображения. При нажатии на проект переходим к просмотру его наборов данных.

На вкладке «Классы» определены классы проекта. Столбцы таблицы «Классы»: «title» (название класса), «shape» (тип фигуры: один из ["многоугольник", "прямоугольник", "линия", "точка", "растровое изображение"]), «color», «hotkey». Можно добавить новый класс или удалить существующий.

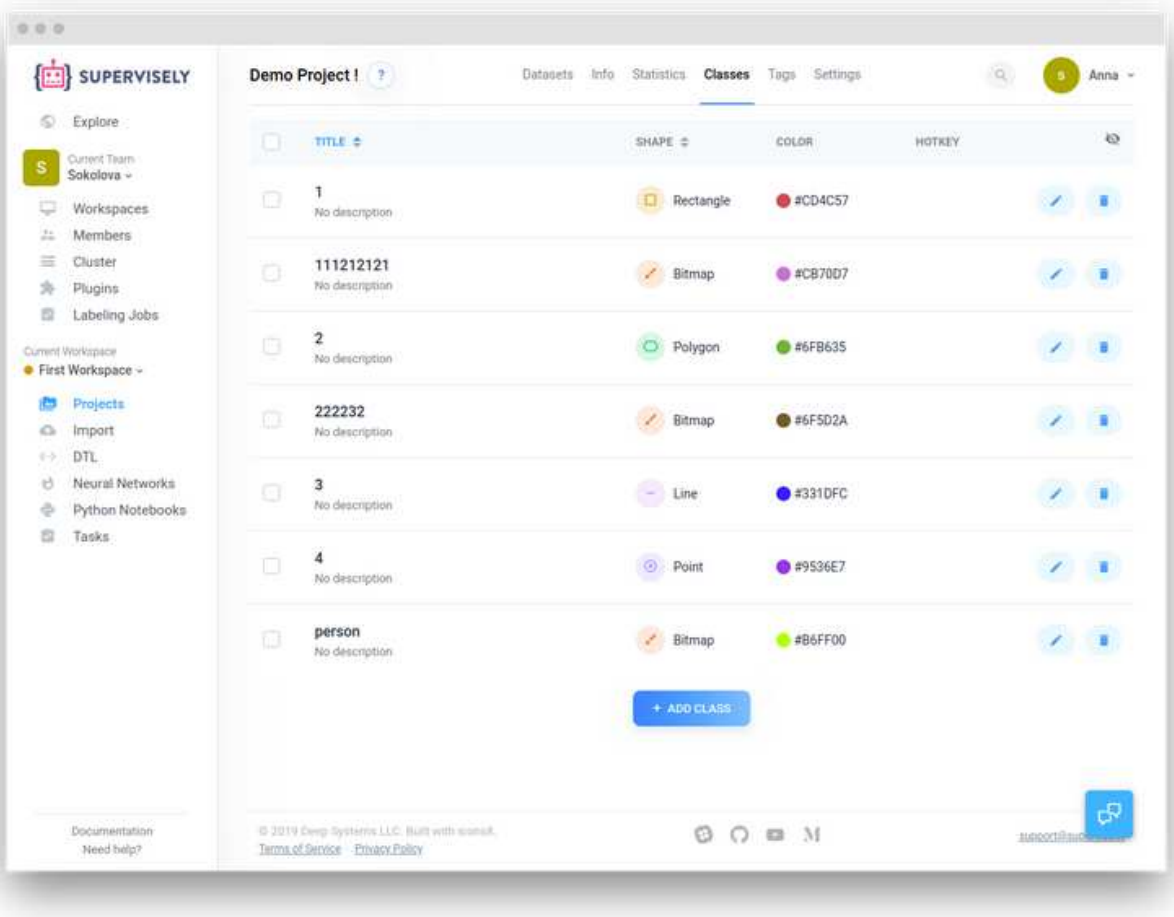

Рисунок 3.7 – Классы

Модуль аннотации – это веб-приложение, которое запускается в вашем браузере с помощью javascript и доступное из любого места. Для его работы есть некоторые системные рекомендации. Во-первых, рекомендуемый браузер: хотя

и поддерживаются все распространенные веб-браузеры, настоятельно рекомендуется использовать Google Chrome или Mozilla Firefox, которые поддерживают новейшие технологии для визуализации аннотаций. Во-вторых, для работы с большими изображениями и большим количеством аннотаций рекомендуется использовать компьютер с аппаратным ускорением.

Модуль аннотации поддерживает работу только с одним набором данных одновременно. Чтобы начать расставлять аннотации, необходимо открыть проект и щелкнуть соответствующий набор данных.

3.4 Кластер

Многие задачи в Supervisely (например, обучение нейронных сетей или запуск DTL) требуют длинных и сложных вычислений. Более того, часто вычислительные мощности распределяются между несколькими машинами.

Supervisely позволяет создать вычислительный кластер. Пользователи могут подключать свои собственные компьютеры или серверы и использовать их для распределения задач. В разделе создания кластера создаётся новый Агент Supervisely – менеджер задач с открытым исходным кодом, доступный в виде образа Docker. Агент будет подключаться к Supervisely API, и теперь мы можем запускать задачи на подключенном компьютере-хосте. Вычислительный кластер Supervisely можно масштабировать по горизонтали без каких-либо ограничений (все ограничения определены лицензионным соглашением). Пользователи могут добавлять десятки узлов и выполнять сотни задач параллельно. Эта технология позволяет организовывать и администрировать частный дата-центр внутри организации без каких-либо специальных знаний. Специалисты могут легко обмениваться вычислительными ресурсами и выполнять несколько экспериментов одновременно (обучение, работа нейросетей, и т. д.).

Задачи, которые могут выполняться Агентами:

- Импорт (преобразование наборов данных в формат Supervisely),
- Язык преобразования данных (Data Transformation Language, DTL),

- Обучение нейронных сетей,

- Вывод нейронных сетей.

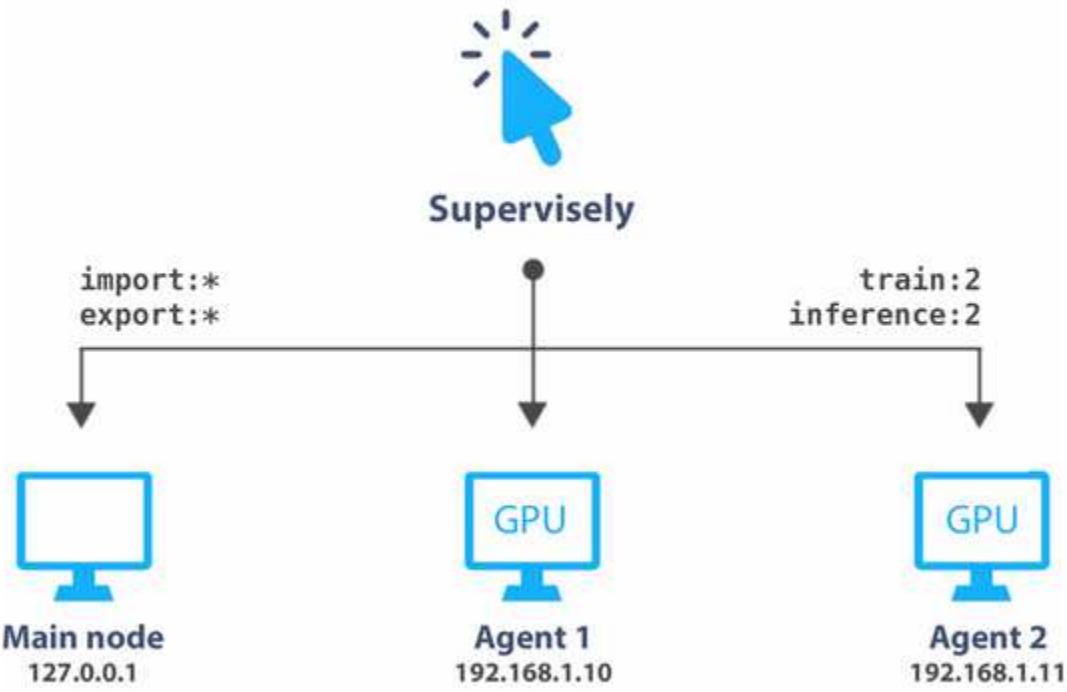

Рисунок 3.8 – Кластер

Жизненный цикл задачи следующий:

1. Пользователь выбирает узел, который будет обрабатывать задачу.

2. Пользователь запускает задачу (например, обучение нейросети).

3. Агент, запущенный на выбранном узле, обрабатывает задачу.

4. Агент загружает данные (проект и / или нейронную сеть), необходимые для задачи, и помещает их в папку задачи.

5. Агент создаёт Docker-контейнер, связанный с этой задачей.

6. Во время выполнения задачи Агент считывает логи stdout и stderr и передает их на сервер.

7. Когда задача завершена, Агент при необходимости загружает результаты на сервер Supervisely.

Кроме того, Агент отправляет необходимую системную информацию на сервер Supervisely в фоновом режиме. Эта информация используется для проверки работоспособности и мониторинга сервера.

Когда мы добавляем новый узел, генерируется уникальный секретный токен, связанный с нашим Агентом. Токен используется для авторизации, и необходимо держать его в секрете. Токен выглядит примерно так: mZVdyTnTGSPQz2iM4kkiNhH2IlqiOmnt.

По умолчанию Агент хранит модели нейросетей, изображения, логи и другую информацию в ~/.supervisely-agent/<token>.

Компьютер, который мы хотим использовать в Supervisely, должен соответствовать нескольким системным требованиям для запуска Агента:

- ОС Linux (ядро 3.10),
- Docker (версия 18.0),
- GPU с поддержкой CUDA 9.0,
- установленный пакет Nvidia-Docker.

Открываем раздел «Кластер» в Supervisely. Если ещё нет установленных Агентов, то мы увидим кнопку «Подключить свой первый узел», после чего откроется диалоговое окно установки агента.

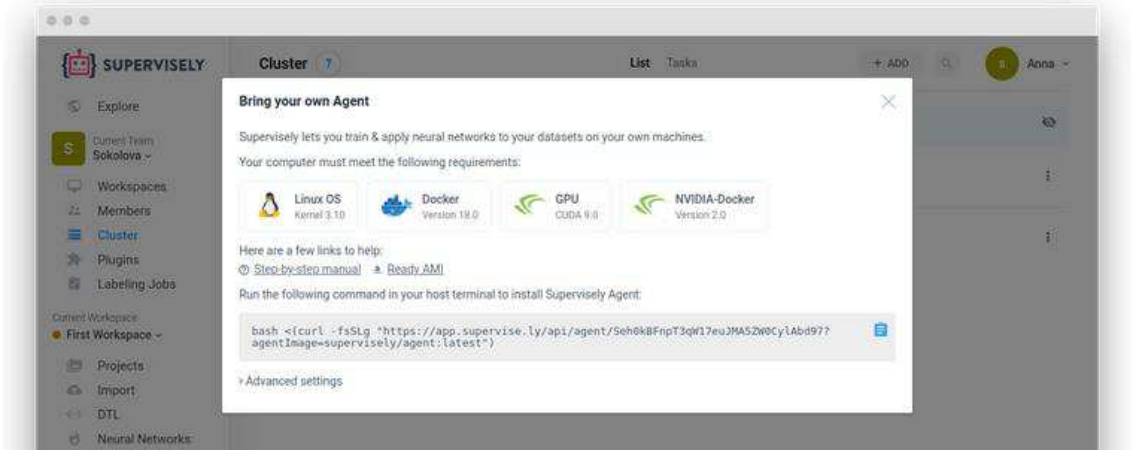

Рисунок 3.9 – Добавление Агента

Копируем команду, которую мы видим на экране, вставляем её в окне терминала в ОС Linux на компьютере (хосте), который мы используем с Supervisely, и запускаем её. Она запускает Агент Supervisely как образ Docker с именем «supervisely-agent/».

Если все сделано правильно, мы получим следующий вывод в терминале:

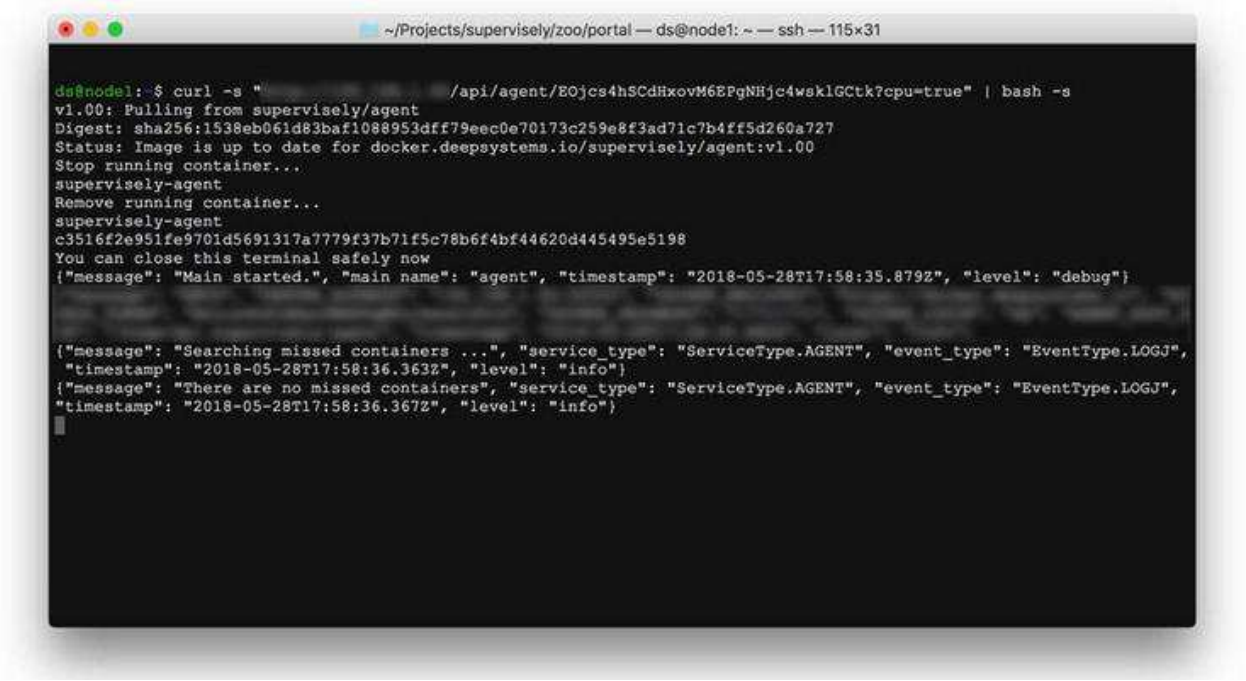

Рисунок 3.10 – Успешный запуск Агента

Теперь мы должны увидеть свой узел на странице кластера со статусом «Запущено (Running)». Это означает, что мы можем запускать задачи на этом Агенте.

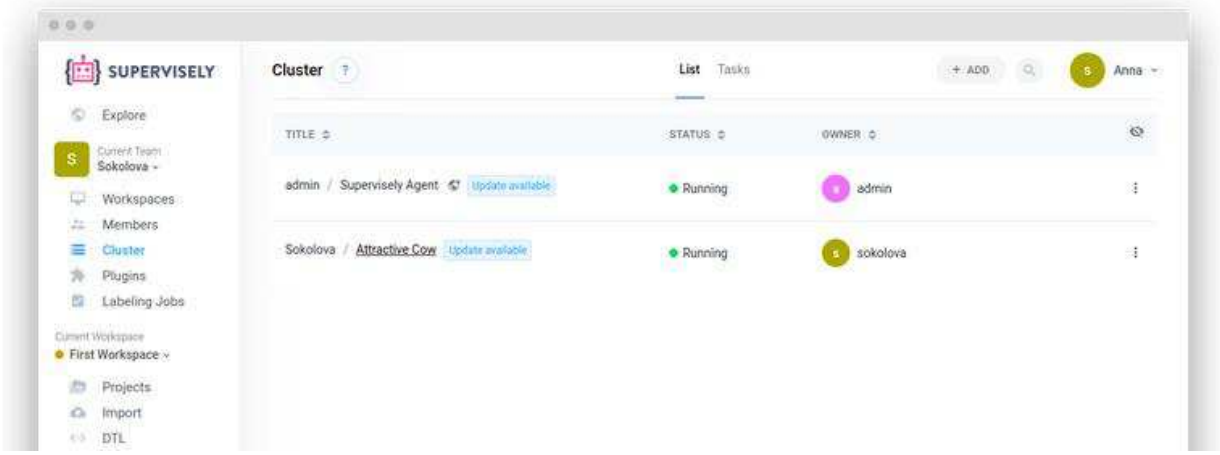

Рисунок 3.11 – Агент запущен

Нажатие на имя узла (сгенерированное автоматически) показывает информацию об оборудовании и другую системную информацию для контроля состояния узла.

3.5 Нейронные сети

В настоящее время нейронные сети используются для решения большинства задач компьютерного зрения. С Supervisely мы можем обучать нейронные сети на наших собственных изображениях, контролировать и визуализировать процесс обучения. После того, как модель нейросети готова, мы можем применять её для распознавания изображений или загрузить модель и использовать ее в своих продуктах.

Supervisely поддерживает большинство современных моделей для решения общих задач компьютерного зрения. В разделе Explore есть вкладка Models, где указаны все поддерживаемые нейронные сети.

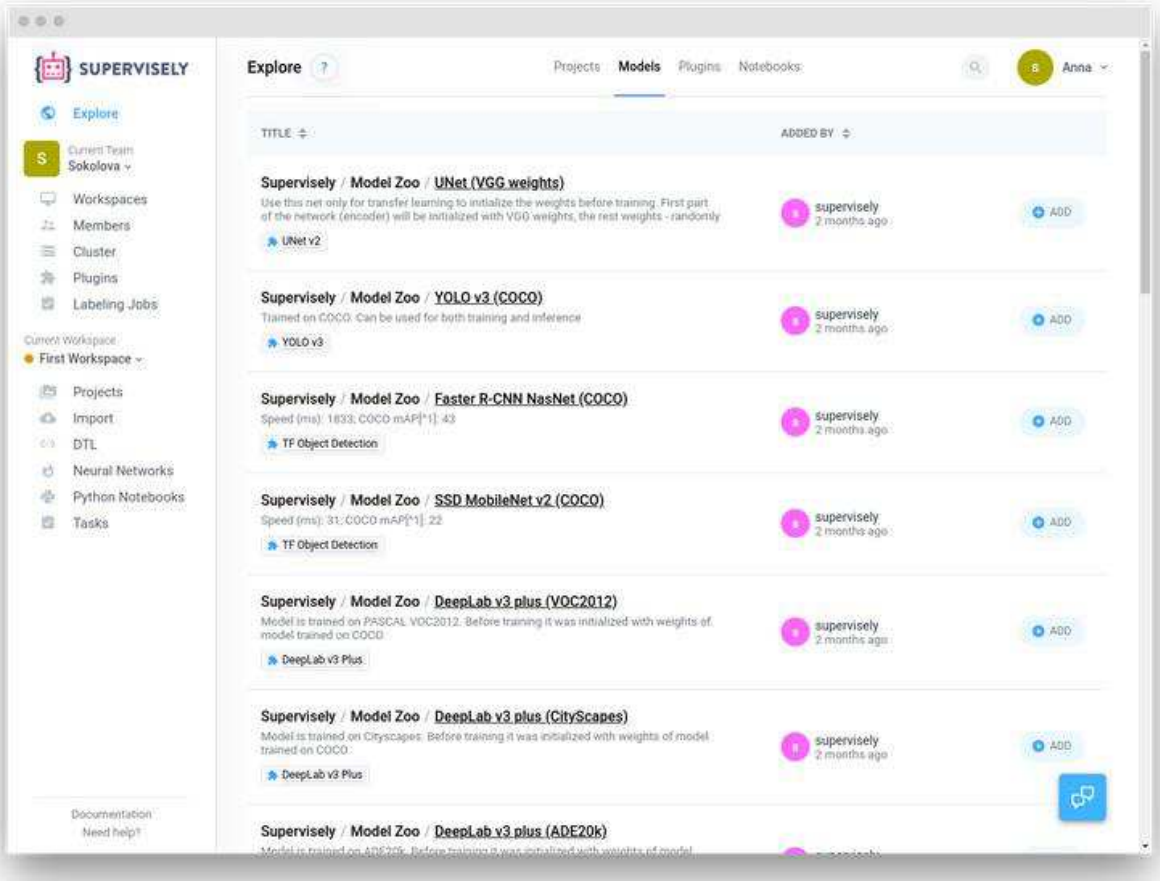

Рисунок 3.12 – Доступные нейросети

Нейронные сети лежат в основе Supervisely. Некоторые ключевые особенности:

- Доступ к современным моделям. В коллекции моделей Supervisely имеется большое количество современных моделей, которые можно обучать и использовать в своих продуктах. Некоторые из них: UNet, Mask R-CNN, YOLO v3, DeepLab v3, MobileNet SSD, PSPNet, ICNet, Faster-RCNN.

- Автоматизация исследований. Построение нейронной сети, которая обеспечивает желаемый уровень точности, может потребовать большого количества процедур обучения. В Supervisely имеется возможность сохранять, просматривать и воспроизводить результаты экспериментов, делиться результатами с другими членами команды.

- Использование предварительно обученных моделей. Часто система компьютерного зрения создается для работы с «хорошо известными» объектами, такими как люди или автомобили. Для таких случаев в коллекции есть несколько предварительно обученных моделей. Эти модели можно использовать в качестве готового к использованию компонента через API или для ускорения процесса аннотирования.

- Умный инструмент (Smart Tool) для сегментации определенных объектов на изображениях. Smart Tool позволяет делать аннотации с минимальным количеством кликов. Чтобы адаптировать Smart Tool для конкретных объектов, его можно обучить в Supervisely.

Выбираем необходимую нам модель и нажимаем кнопку «Добавить», чтобы клонировать модель в текущую рабочую область. Она появится в разделе «Нейронные сети». В этом разделе мы можем просматривать и управлять своими моделями.

Модели добавляются в этот список несколькими способами:

- Из списка моделей в разделе «Обзор»,

- Обучение автоматически сохраняет последнюю контрольную точку,

- Ручная загрузка.

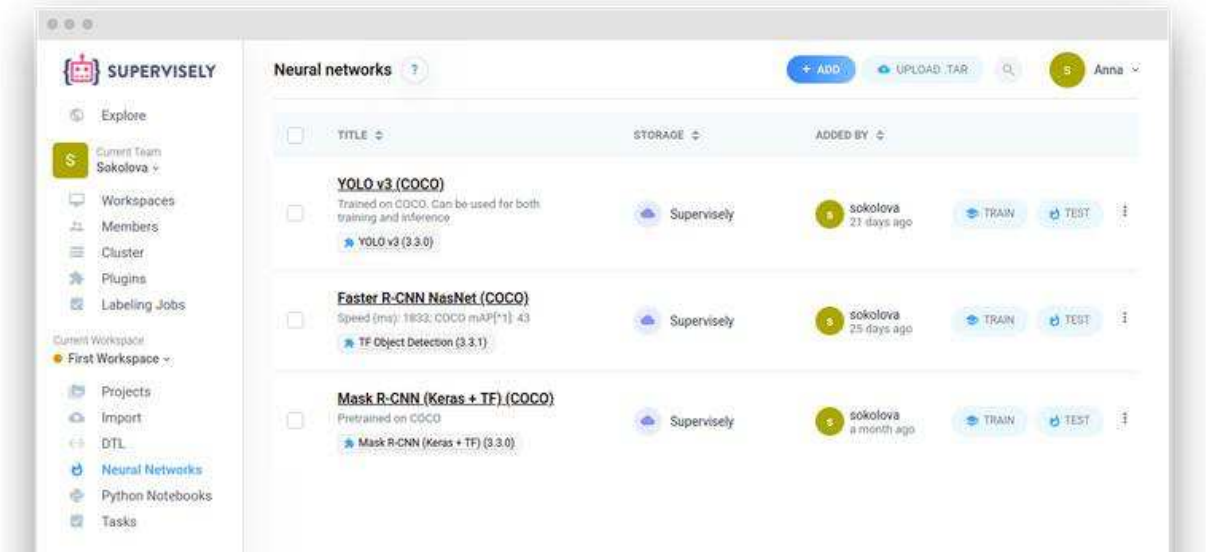

Рисунок 3.13 – Добавленные нейросети

Во время обучения несколько снимков модели сохраняются и могут быть восстановлены после завершения процесса обучения. Это может быть полезно для понимания динамики обучения и выбора наилучшей доступной модели для дальнейшего использования.

В контекстном меню, связанном с задачей, есть пункт «Контрольные точки (Checkpoints)», который дает доступ ко всем контрольным точкам, созданным в процессе обучения.

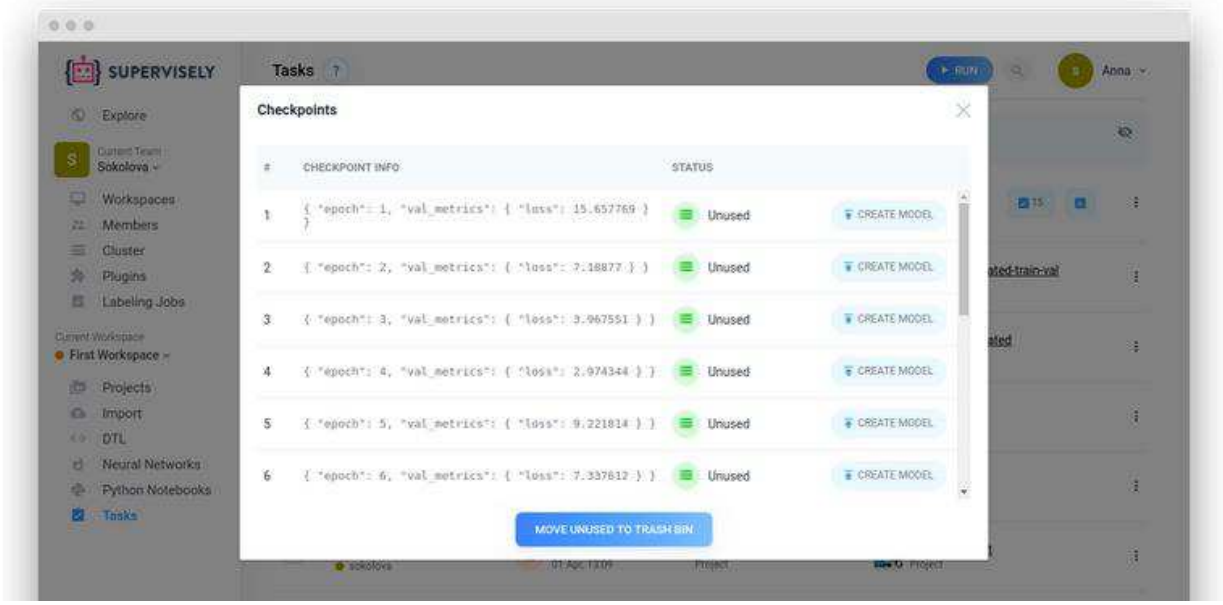

Рисунок 3.14 – Контрольные точки

Если нужно сохранить промежуточный результат, то выбираем контрольную точку, которую хотим сохранить, и указываем новое имя. После этого нажимаем кнопку «Создать». Теперь модель под новым именем доступна в разделе «Нейронные сети», и с ней возможны все те же операции, как и с другими моделями.

Если модель хранится на сервере Supervisely, то её можно использовать на всех вычислительных узлах. Перед выполнением задачи (обучение или вывод) Агент загрузит модель в локальное хранилище, если её там ещё не существует.

Если модель хранится на определенном узле в кластере, то её можно использовать (обучение или вывод) только на этом узле. Перед тем, как сохранить

эту модель или использовать ее на других узлах кластера, необходимо загрузить её на сервер Supervisely.

Если модель была добавлена в учетную запись из Model Zoo, то по умолчанию её основное хранилище находится на сервере Supervisely. Это означает, что эту модель можно использовать на всех вычислительных узлах кластера.

3.6 Обучение и вывод

Чтобы начать обучение, нажимаем кнопку «Обучение (Train)» в списке моделей в разделе «Нейронные сети». Откроется страница плагина запуска, некоторые поля будут заполнены автоматически.

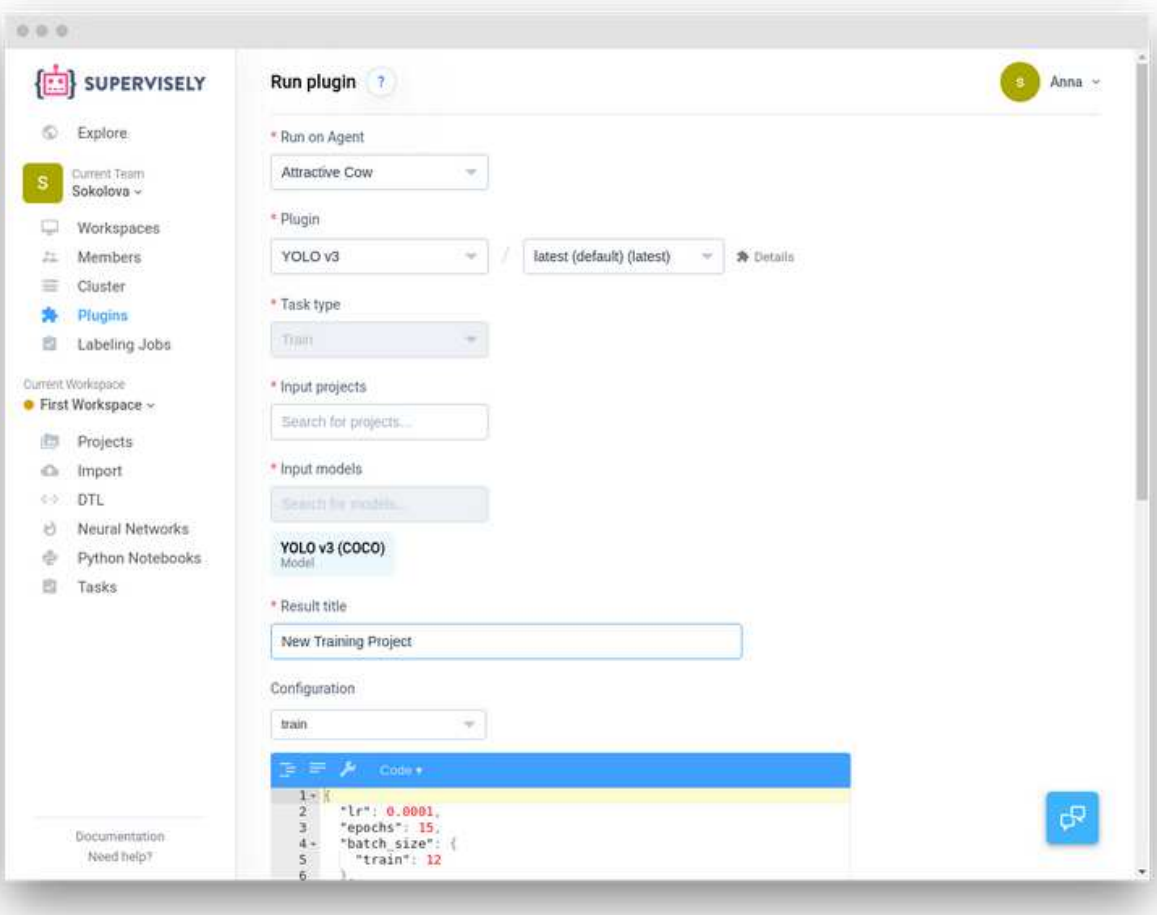

Рисунок 3.15 – Запуск обучения

Нужно сконфигурировать следующие поля:

1. Агент (выбрать Агента, на котором будет обучаться модель).

2. Проект (выбрать проект из текущей рабочей области).

3. Заголовок (ввести имя для будущей модели, может быть изменено позже).

4. Конфигурация (выбрать плагин для обучения).

Нажимаем «Запустить (Run)», чтобы начать обучение. Будет создано новое задание в разделе «Задачи». В контекстном меню модели (значок «три точки») можно просматривать логи, чтобы контролировать ход выполнения задачи или остановить обучение.

После окончания обучения на странице «Нейронные сети» последняя контрольная точка будет сохранена как новая модель с выбранным именем.

Выполнение процедуры логического вывода нейронной сети в целом происходит аналогично процедуре обучения. Чтобы начать вывод, нажимаем кнопку «Вывод (Test)» в списке моделей в разделе «Нейронные сети». Откроется страница плагина запуска, некоторые поля будут заполнены автоматически.

Нужно сконфигурировать следующие поля:

1. Агент (выбрать Агента, с которым будет работать модель).

2. Проект (выбрать проект из текущей рабочей области, к которому нужно применить модель).

3. Заголовок (ввести имя для будущего проекта, может быть изменено позже).

4. Конфигурация (выбрать плагин для обучения).

Нажимаем «Запустить (Run)», чтобы начать вывод. Будет создано новое задание в разделе «Задачи». В контекстном меню модели (значок «три точки») можно просматривать логи, чтобы контролировать ход выполнения задачи или остановить вывод.

После окончания вывода на странице «Проекты» в списке появится новый проект с выбранным именем. Если проект с таким именем уже существует, автоматически будет добавлен случайный суффикс.

3.7 Анализ результатов

Произведён вывод отобранных моделей нейронных сетей. В качестве изображений для проверки результатов использовались материалы, полученные камерой автомобильного регистратора. Рассматриваемая сцена представляет собой кадры последствий дорожно-транспортного происшествия, произошедшего на ул. Профсоюзов г. Красноярска весной 2020 года, и является типичной городской сценой, где представлены объекты характерных классов.

Для примера, на котором можно видеть особенности классификации, представлен один и тот же кадр с различными результатами классификации разных моделей – объекты детектируются неодинаково.

Рисунок 3.16 – Результат вывода RFCN ResNet101

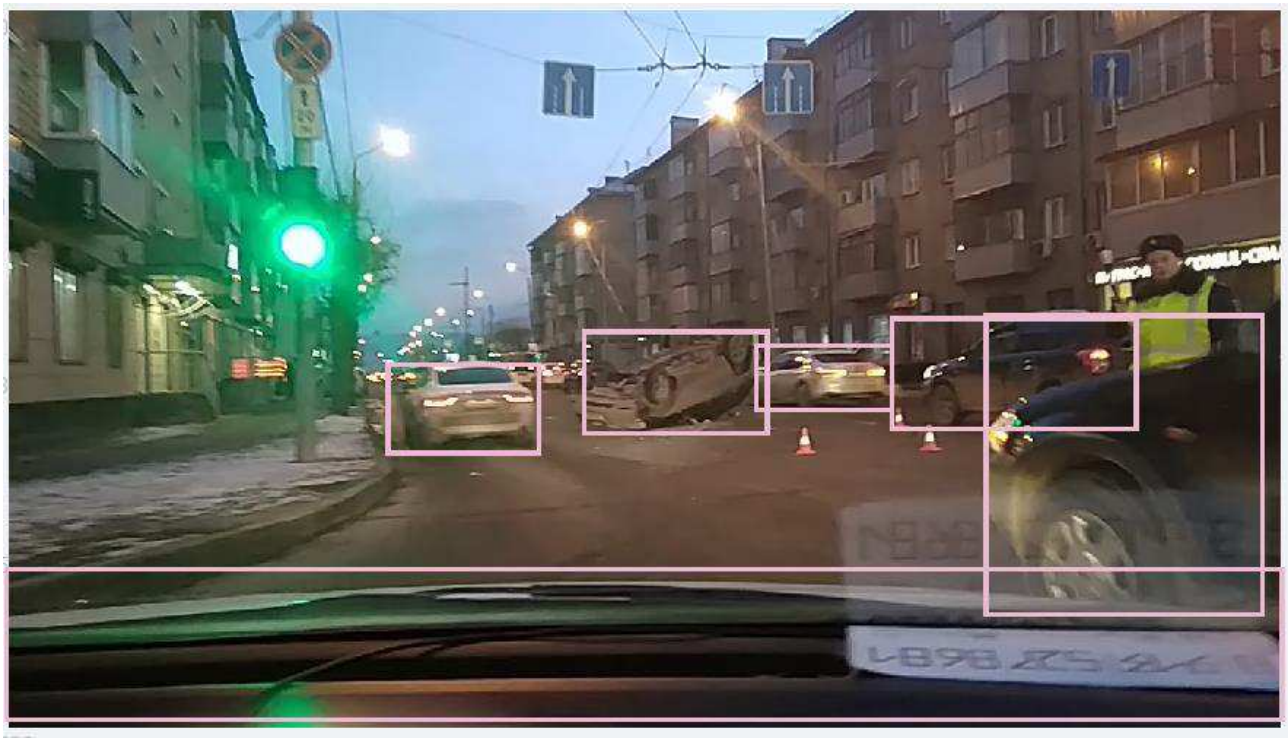

Рисунок 3.17 – Результат вывода YOLO v3

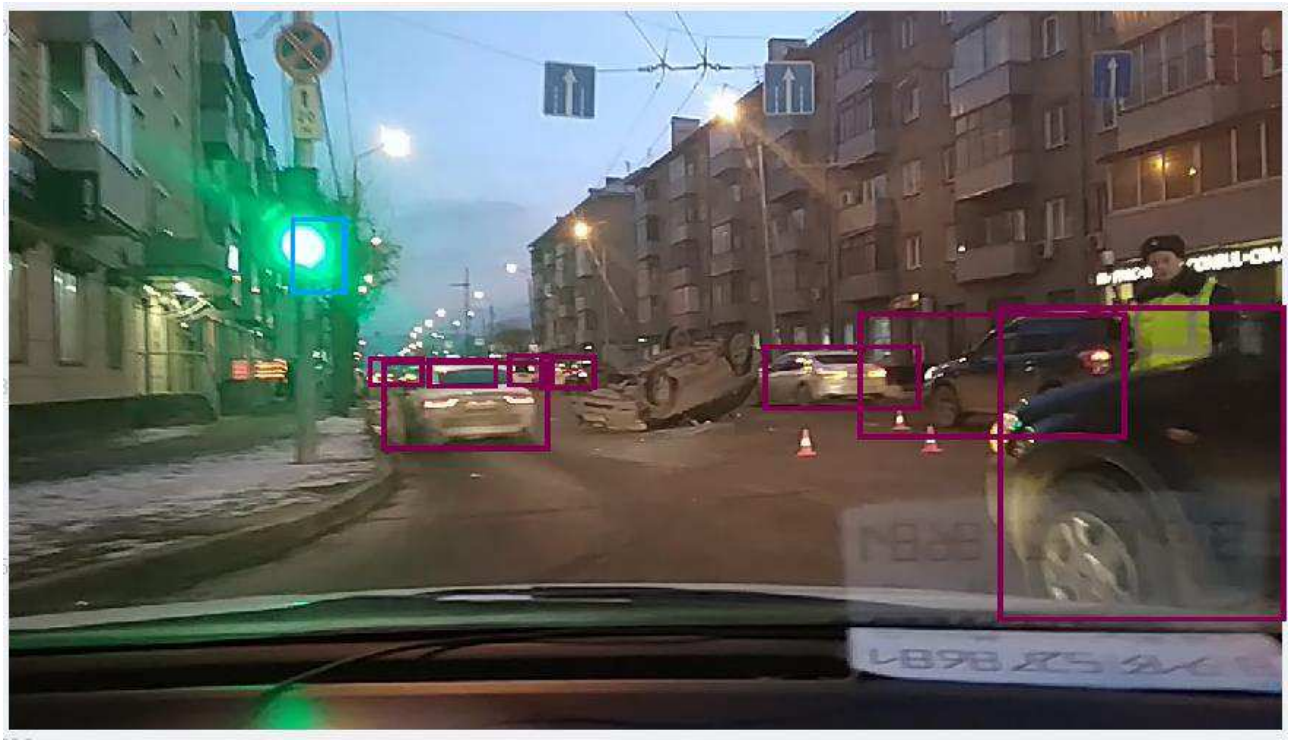

Рисунок 3.18 – Результат вывода SSD MobileNet v2

Было проведена оценка точности классификации использованных моделей с помощью распространённых метрик. Для этого используем следующие понятия или четыре группы результатов:

- Истинно положительные (ИП),

- Истинно отрицательные (ИО),

- Ложно положительные (ЛП),

- Ложно отрицательные (ЛО).

Тогда можно вычислить значения по следующим метрикам:

- Полнота (recall): представляет собой процент правильно классифицированных объектов от общего фактического количества существующих объектов.

$$\text{Полнота} = \frac{\text{ИП}}{\text{ИП} + \text{ЛО}}$$

- Точность (precision): представляет собой процент правильно классифицированных объектов от общего количества объектов, отнесённых к соответствующему классу.

$$\text{Точность} = \frac{\text{ИП}}{\text{ИП} + \text{ЛП}}$$

Поскольку эти две метрики не учитывают всех видов результатов, то они могут дать нам смещенную оценку, особенно в случае несбалансированных классов (где количество наблюдений, относящихся к каким-то классам, значительно отличается других).

F-мера (F-score) представляет собой совместную оценку точности и полноты, поэтому позволяет получить более сбалансированную характеристику модели.

$$\text{F-мера} = 2 \cdot \frac{\text{Точность} \cdot \text{Полнота}}{\text{Точность} + \text{Полнота}}$$

В таблице 3.1 представлены сравнительные результаты оценки точности моделей по различным метрикам. Как видно из таблицы, более высокую точность показала модель RFCN ResNet101, за ней с небольшим отставанием идёт YOLO v3.

Таблица 3.1 – Результаты оценки точности

Архитектура	Полнота	Точность	F-мера
RFCN ResNet101	0,7551	0,7236	0,7355
YOLO v3	0,7262	0,7292	0,7205
SSD MobileNet v2	0,5496	0,6223	0,5777

Ниже представлены сравнительные графики распределения значений метрик по отдельным кадрам изображений, из которых видно, что две лучших модели в среднем получают более высокие значения.

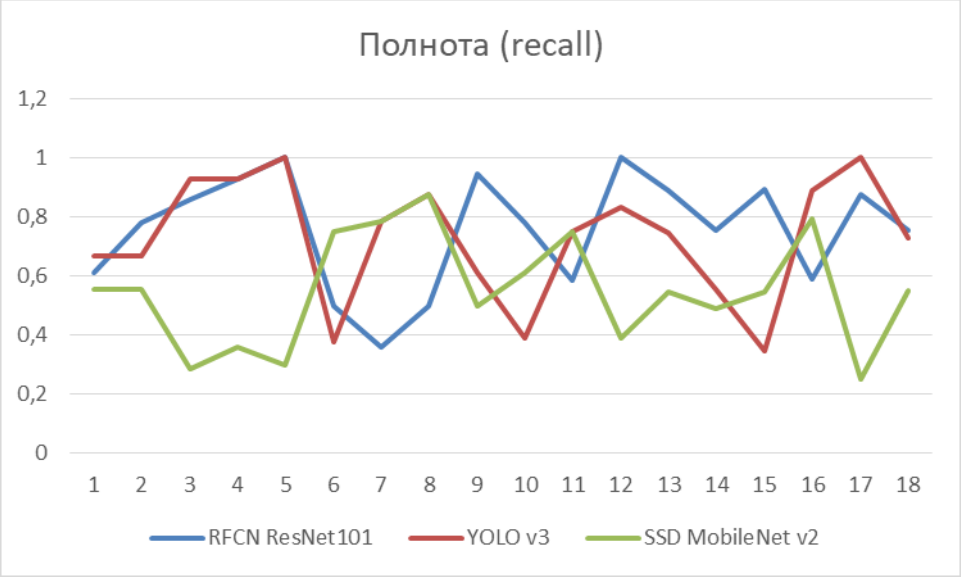

Рисунок 3.19 – Значения полноты

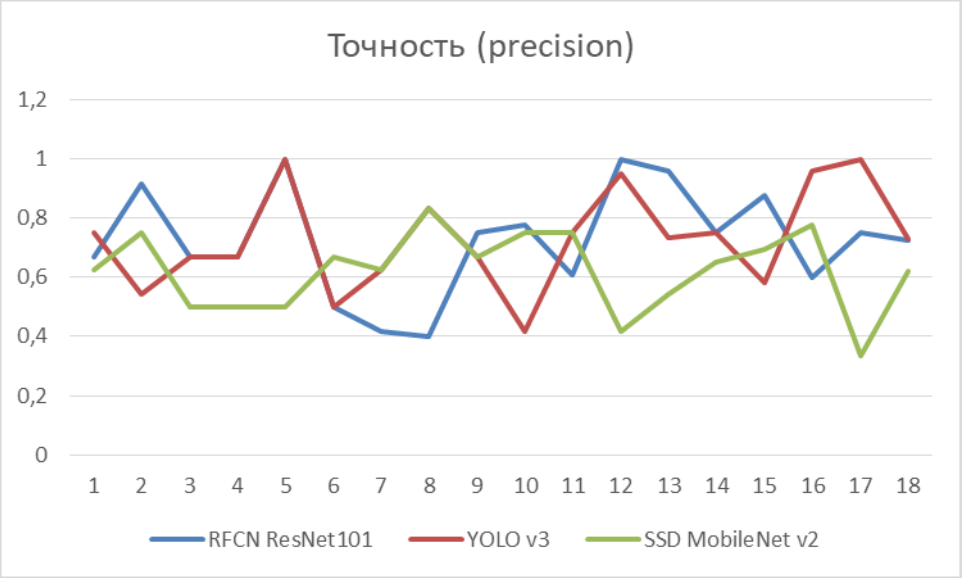

Рисунок 3.20 – Значения точности

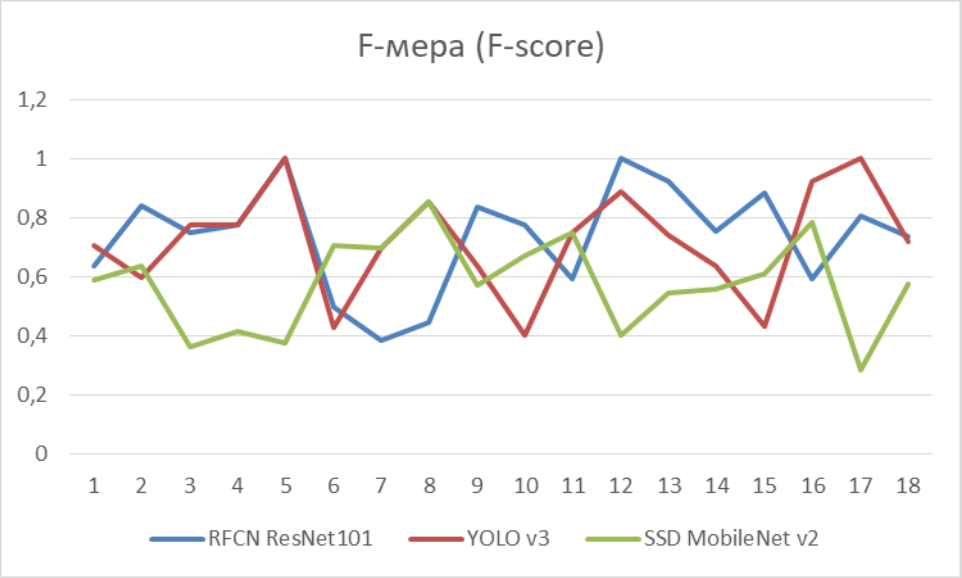

Рисунок 3.21 – Значения F-меры

В ходе подготовки данной работы и по её материалам были опубликованы две статьи в научно-технической конференции с международным участием «Робототехника и искусственный интеллект» в 2018 и 2019 годах.

Заключение

Выполнена научно-исследовательская работа по теме «Использование нейронных сетей для задачи распознавания объектов на городских территориях».

На определенных этапах исследования были решены следующие задачи:

- выявлены объекты интереса, присутствующие на городских сценах,
- рассмотрены различные архитектуры нейронных сетей для решения задач классификации,
- отобраны изображения для проверки работы алгоритмов классификации,
- проведено экспериментальное исследование с применением выбранных решений,
- выполнена оценка точности классификации по рассмотренным метрикам.

Решение этих задач позволило достичь цели – разработать алгоритм классификации изображений, полученных на городских территориях, и опробовать технические средства для его реализации.

Список использованных источников

1. Cordts M., Omran M., Ramos S., Rehfeld T., Enzweiler M., Benenson R., Franke U., Roth S., Schiele B. The Cityscapes Dataset for Semantic Urban Scene Understanding // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016. – pp. 3213-3223.

2. Neuhold G., Ollmann T., Rota Bulò S., Kontschieder P. The Mapillary Vistas Dataset for Semantic Understanding of Street Scenes // Proceedings of International Conference on Computer Vision (ICCV), 2017. – pp. 4990-4999.

3. Muchtar K., Afdhal N., Nugraha I. Attention-based Approach for Efficient Moving Vehicle Classification // Proceedings of 4th International Conference on Computer Science and Computational Intelligence (ICCSCI), 2019. – pp. 683–690.

4. Li S., Lin J., Li G., Bai T., Wang H., Pang Y. Vehicle type detection based on deep learning in traffic scene // Proceedings of 8th International Congress of Information and Communication Technology (ICICT-2018), 2018. – pp. 564–572.

5. Krause J., Stark M., Deng J., Fei-Fei L. 3D Object Representations for Fine-Grained Categorization. 4th IEEE Workshop on 3D Representation and Recognition, at ICCV 2013 (3dRR-13), Sydney, Australia, 12, 2013. [Электронный ресурс]: режим доступа – http://vision.stanford.edu/pdf/3drr13.pdf.

6. Caltech Computational Vision Group Datasets. [Электронный ресурс]: режим доступа – http://www.vision.caltech.edu/archive.html.

7. Madokoro H., Kainuma A., Sato K. Non-Rectangular RoI Extraction and Machine Learning Based Multiple Object Recognition Used for Time-Series Areal Images Obtained Using MAV // Proceedings of 22nd International Conference on Knowledge-Based and Intelligent Information & Engineering Systems, 2018. – pp. 462–471.

8. Shustanov A., Yakimov P. CNN Design for Real-Time Traffic Sign Recognition // Proceedings of 3rd International Conference “Information Technology and Nanotechnology” (ITNT-2017), Samara, Russia, 4, 2017. – pp. 718–725.

9. Saouli A., El Aroussi M., Fakhri Y. Traffic Sign Recognition Based On Multi-feature Fusion and ELM Classifier // Proceedings of The First Inter-national Conference On Intelligent Computing in Data Sciences, 2018. – pp. 146–153.

10. Stallkamp J., Schlipsing M., Salmen J., Igel C. The German Traffic Sign Recognition Benchmark: A multi-class classification competition // Proceedings of the IEEE International Joint Conference on Neural Networks, 2011. – pp. 1453–1460.

11. Houben S., Stallkamp J., Salmen J., Schlipsing M., Igel C. Detection of Traffic Signs in Real-World Images: The German Traffic Sign Detection Benchmark // Proceedings of the IEEE International Joint Conference on Neural Networks, 2013. – p. 1288.

12. Timofte R., Mathias M., Benenson R., Van Gool L. Traffic Sign Recognition - How far are we from the solution? // Proceedings of International Joint Conference on Neural Networks (IJCNN 2013), Dallas, USA, 8, 2013. [Электронный ресурс]: режим доступа – http://www.vision.ee.ethz.ch/~timofter/publications/ Mathias-IJCNN-2013.pdf.

13. Chaabani H., Kamoun F., Bargaoui H., Outay F., Yasar A. A Neural network approach to visibility range estimation under foggy weather conditions // Proceedings of International Workshop on Connected & Intelligent Mobility (CIM 2017), 2017. – pp. 466–471.

14. The FROSI (Foggy ROad Sign Images) database. [Электронный ресурс]: режим доступа – https://www.livic.ifsttar.fr/linstitut/cosys/laboratoires/livic-ifsttar/logiciels/bases-de-donnees/frosi/.

15. Kai G., Shuai W., Yong X. Face recognition using both visible light image and near-infrared image and a deep network // Proceedings of CAAI Transactions on Intelligence Technology, 2017. – pp. 39–47.

16. Li S., Yi D., Lei Z., Liao S. The CASIA NIR-VIS 2.0 face database // Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops. (CVPRW), 2013. – pp. 348-353.

17. Baochang Z., Lei Z., Zhang D., Linlin S. Directional Binary Code with Application to PolyU Near-Infrared Face Database // Proceedings of Pattern Recognition Letters, 10, 2010. – pp. 2337–2344.

18. Yong X., Aini Z., Jian Y., Zhang D. Bimodal biometrics based on a representation and recognition approach // Proceedings of SCI, 2011. – p. 50.

19. Namatevs I., Sudars K., Polaka I. Automatic data labeling by neural networks for the counting of objects in videos // Proceedings of ICTE in Transportation and Logistics (ICTE 2018), 2018. – pp. 151–158.

20. Pashchenko F. F., Amosov O. S., Amosova S. G., Ivanov Y. S., Zhiganov S. V. Deep Neural Network Method of Recognizing the Critical Situations for Transport Systems by Video Images // Proceedings of The 2nd International Conference on Emerging Data and Industry 4.0 (EDI40), Leuven, Belgium, 4, 2019. – pp. 675–682.

21. Verbitsky N. S., Chepin E. V., Gridnev A. A. Experimental studies of a convolutional neural network for application in the navigation system of a mobile robot // Proceedings of the 9th Annual International Conference on Bio-logically Inspired Cognitive Architectures (BICA), 2018. – pp. 611–616.

22. Ren S., He K., Girshick R., Sun J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks // Proceedings of IEEE Transactions on Pattern Analysis and Machine Intelligence, 2016. – pp. 1137-1149.

23. He K., Zhang X., Ren S., Sun J. Deep Residual Learning for Image Recognition // Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, 2016. – pp. 770-778.

24. Dai J., Li Y., He K., Sun J. R-FCN: object detection via region-based fully convolutional networks // Proceedings of the 30th International Conference on Neural Information Processing Systems (NIPS), 2016. – pp. 379–387.

25. Long J., Shelhamer E., Darrell T. Fully convolutional networks for semantic segmentation // Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015. – pp. 3431-3440.

26. Redmon J., Farhadi A. YOLOv3: An Incremental Improvement (2018). [Электронный ресурс]: режим доступа – https://arxiv.org/pdf/1804.02767.pdf

27. Chen L.-C., Zhu Y., Papandreou G., Schroff F., Adam H. Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation // Proceedings of the European Conference on Computer Vision (ECCV), 2018. – pp. 833-851.

28. Liu W., Anguelov D., Erhan D., Szegedy C., Reed S., Fu C.-Y., Berg A. C. SSD: Single Shot MultiBox Detector // Proceedings of European Conference on Computer Vision (ECCV), 2016. – pp. 21-37.

29. Sandler M., Howard A., Zhu M., Zhmoginov A., Chen L. MobileNetV2: Inverted Residuals and Linear Bottlenecks. // Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, 2018. – pp. 4510-4520.

30. Supervisely Docs – A web platform to manage, annotate, validate and experiment with datasets & neural networks for computer vision. [Электронный ресурс]: режим доступа – https://docs.supervise.ly/

Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий
Кафедра систем искусственного интеллекта

УТВЕРЖДАЮ
Заведующий кафедрой
_____ Г. М. Цибульский
подпись

«____» _____ 2020 г.

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

09.04.01.10 «Интеллектуальные информационные системы»

Использование нейронных сетей для задачи распознавания объектов
на городских территориях

Научный руководитель _____ 050720 доцент, канд. техн. наук А. В. Пятаева
подпись, дата

Выпускник _____ 06.07.2020 В. В. Ойнец
подпись, дата

Рецензент _____ 06.07.2020 доцент, канд. техн. наук В. В. Вдовенко
подпись, дата

Красноярск 2020