**Федеральное агентство связи**

**Ордена Трудового Красного Знамени**

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

**«Московский технический университет связи и информатики»**

Кафедра «Информатики»

**Отчет по практической работе**

на тему:

«Обработка изображений и распознавание образов»

Выполнил: студент группы

БВТ1901

Кускова А. Е.

Руководитель:

Мосева М. С.

Москва 2020

**Оглавление**

[1. Постановка задачи и анализ предметной области 3](#_Toc531125128)

[2. Выбор инструментов разработки 4](#_Toc531125130)

[3. Сбор и разметка датасета 10](#_Toc531125131)

[4. Обучение и оценка модели 13](#_Toc531125132)

[Заключение 14](#_Toc531125133)

1. Постановка задачи и анализ предметной области

Требуется создать нейросеть для определения качественного состава транспортных потоков (тип ТС, марка/модель).

Система распознавания типов транспортных средств и их моделей пользуется большой популярностью в наши дни. Самое очевидное ее применение – обеспечение дисциплины на дорогах. Ведь автоматический контроль соблюдения правил дорожного движения, нарушение которых влечет за собой неотвратимость наказания, является важным аспектом обеспечения безопасности дорожного движения. Преимущества автоматической регистрации понятны – система регистрирует каждый проезжающий автомобиль, что не всегда может делать работник патрульно-постовой службы. Было решено реализовать алгоритм с применением сверточной нейронной сети, одним из преимуществ которых является слабая чувствительность к искажениям входного сигнала.

Сверточная нейронная сеть — это Deep Learning-алгоритм, который может принимать входное изображение, присваивать важность различным областям/объектам в изображении и может отличать одно от другого. Сеть имеет входные и выходные данные и сверточные слои. В нашем случае на вход поступает изображение, включающее в себя транспортные средства, а на выходе мы получаем новое изображение с распознаванием этих транспортных средств и их моделей.

2. Выбор инструментов разработки

Для реализации поставленной задачи были использованы такие инструменты разработки, как:

* Язык программирования Python 3.6.8
* PyCharm — интегрированная среда разработки для языка программирования Python
* Архитектура современной нейронной сети для сегментации объектов на изображениях Mask R-CNN
* Пакет необходимых библиотек: numpy, scipy, pillow, cython, matplotlib, scikit-image, tensorflow 1.3.0, keras 2.0.8, opencv-python, h5py, imgaug, IPython[all]
* LabelImg – инструмент для графических аннотаций изображений.

3. Обзор архитектур нейронных сетей

В процессе работы были рассмотрены основные архитектуры нейронных сетей: Fully Connected Feed-Forward Neural Networks, FNN; Recurrent Neural Networks, RNN; Convolutional Neural Networks, CNN.

**Fully Connected Feed-Forward Neural Networks, FNN**

Fully Connected Feed-Forward Neural Networks, FNN – полносвязные нейросети прямого распространения. Эти нейросети отлично подходят для классификации, однако имеется ряд трудностей, с которыми можно столкнуться при их использовании – большое количество параметров и затухающие градиенты. Для обучения сети с большим количеством параметров, необходимо использовать много обучающих примеров, которые есть не всегда.

**Recurrent Neural Networks, RNN**

Recurrent Neural Networks, RNN – рекуррентные нейросети. Главное отличие их от обычных FNN-сетей в том, что появляется циклическая связь. То есть скрытый слой свои же значения отправляет сам на себя на следующем шаге. По сути, рекуррентные нейросети — это обычный компьютер. Задача — его правильно обучить. Потенциально он может считать любой алгоритм. Однако научить его бывает сложно. Рекуррентные сети обладают дополнительной памятью за счет того, что содержимое слоя передается нейросети обратно.

**Convolutional Neural Networks, CNN**

Convolutional Neural Networks, CNN – свёрточные сети. Они решают 3 основных задачи: классификация, детекция и сегментация. Сверточная нейросеть — это обычная Feed-Forward сеть, обладающая дополнительными свойствами. Она имеет сверточные слои, которые обычно рисуются в виде набора плоскостей или объемов. Каждая плоскость на таком рисунке или каждый срез в этом объеме — это, по сути, один нейрон, который реализует операцию свертки. Subsampling, или Pooling-слои, которые уменьшают размер изображения и полносвязные слои, использующиеся на выходе модели для классификации.

Для выполнения поставленной задачи было решено использовать Convolutional Neural Networks, а именно Mask R-CNN.

3. Сбор и разметка датасета

Данная нейросеть будет включать в себя несколько классов. По типу транспортного средства (‘car’, ‘bus’, ‘truck’), по его маркам (‘bmw’, ‘toyota’, ‘peugeot’; ‘kia’, ‘zonda’, ‘toyota’; ‘man’, ‘hyundai’, ‘scania’).

Для каждого класса был собран датасет, включающий в себя порядка 150 фотографий. Те, в свою очередь были разбиты на фото для обучения (80%) и для тестирования (20%).

В дополнение к каждой фотографии потребовались файлы аннотаций, которые были созданы благодаря инструменту LabelImg. Файл аннотации содержит элемент «размер», описывающий форму фотографии, и один или несколько элементов «объект», которые описывают ограничивающие рамки для объектов классов на фотографии.

4. Обучение и оценка модели

Код по созданию и обучению нейросети предоставлен на github. Оценка модели:

*Определение модели с помощью конфигурации и установление для аргумента mode значение inference вместо training:*

cfg = PredictionConfig()

model = MaskRCNN(mode='inference', model\_dir='./', config=cfg)

*Затем мы можем загрузить веса из сохраненной модели.*

model.load\_weights('mask\_rcnn\_ 0005.h5', by\_name=True)

*Далее мы можем оценить модель. Во-первых, изображение и маска истинности могут быть загружены из набора данных для заданного image\_id. Этого можно добиться с помощью вспомогательной функции load\_image\_gt ().*

image, image\_meta, gt\_class\_id, gt\_bbox, gt\_mask = load\_image\_gt(dataset, cfg, image\_id, use\_mini\_mask=False)

*Затем значения пикселей загруженного изображения должны быть масштабированы таким же образом, как это было выполнено для обучающих данных, например центрировано. Этого можно добиться с помощью вспомогательной функции mold\_image ().*

scaled\_image = mold\_image(image, cfg)

*Затем размеры изображения необходимо расширить на один образец в наборе данных и использовать в качестве входных данных для создания прогноза с помощью модели.*

sample = expand\_dims(scaled\_image, 0)

yhat = model.detect(sample, verbose=0)

r = yhat[0]

*Затем прогноз можно сравнить с фактами и метриками, рассчитанными с помощью функции compute\_ap ().*

AP, \_, \_, \_ = compute\_ap(gt\_bbox, gt\_class\_id, gt\_mask, r["rois"], r["class\_ids"], r["scores"], r['masks'])

APs.append(AP)

mAP = mean(APs)

return mAP

*Теперь мы можем рассчитать MAP для модели на поезде и тестовых наборах данных.*

train\_mAP = evaluate\_model(train\_set, model, cfg)

print("Train mAP: %.3f" % train\_mAP)

test\_mAP = evaluate\_model(test\_set, model, cfg)

print("Test mAP: %.3f" % test\_mAP)

МАР выше 90% или 95% — это хороший результат.

Заключение

Таким образом, используя основы языка Python, была создана нейросеть для определения качественного состава транспортных потоков.