МИНИСТЕРСТВО ТРАНСПОРТА РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «РОССИЙСКИЙ УНИВЕРСИТЕТ ТРАНСПОРТА» (РУТ(МИИТ)

Институт управления и цифровых технологий

Кафедра «Вычислительные системы, сети и информационная безопасность»

ОТЧЕТ ПО ПРОЕКТНОЙ ДЕЯТЕЛЬНОСТИ НА ТЕМУ: ИСКУССТВЕННАЯ НЕЙРОСЕТЬ(ИНС) ДЛЯ РАСПОЗНАВАНИЯ СВЕТОФОРА И ЕГО ЦВЕТА

Направление: 09.03.01 Информатика и вычислительная техника

Профиль: Вычислительные системы и сети

Выполнили:
студенты группы УВВ-211
Басова А. С.
Мартин Е. А.
Куприянов В. В.

Проверил:
Ст.преп. Абрамов А. В.
Доцент Малинский С. В.
(должность, ФИО)

МОСКВА 2023

АННОТАЦИЯ

Пояснительная записка 24 страниц, 16 рисунков, 3 таблицы, 6 источников

ИСКУССТВЕННЫЙ ИНТЕЛЛЕКТ, ИСКУССТВЕННАЯ НЕЙРОСЕТЬ, СВЕТОФОР

Объектом исследования является распознавание на изображении светофора и его цвета.

Предмет исследования – обученная нейросеть, распознающая светофор на изображении и его цвет.

Целью исследования является обучение нейросети, распознающей на изображении светофор и его цвет.

В ходе работы над проектом был проведен анализ предметной области, установлены задачи и области применения проекта. Так же был произведен анализ моделей Keras для обучения искусственной нейросети.

Задачей нашей группы было обучение искусственной нейросети для распознавания на изображении светофора и его цвета. Для выполнения данной задачи работа была разбита на несколько этапов.

Первым этапом нашей работы было формирования датасета изображений. Второй этап заключался в изучении моделей Keras и выбора 2-х моделей для обучения.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Нейронные сети — это математическая модель, которая имитирует работу человеческого мозга. Они состоят из нейронов, которые соединены между собой и обрабатывают информацию. Нейронные сети широко используются в области машинного обучения и искусственного интеллекта для решения различных задач, таких как распознавание образов, классификация данных, прогнозирование и многое другое.

Основные компоненты нейронной сети включают:

1. Нейроны: они являются основными строительными блоками нейронной сети. Каждый нейрон принимает входные данные, выполняет математические операции и передает результат в следующий слой.

2. Веса: каждая связь между нейронами имеет свой вес, который определяет важность этой связи для вычисления выходных данных.

3. Функция активации: функция активации определяет, какой будет выход нейрона на основе взвешенной суммы его входных данных. Она может добавить нелинейность в модель и позволить нейронной сети решать сложные задачи.

4. Слои: нейроны организованы в слои, которые могут быть сверточными, пулинговыми, полносвязными и другими. Каждый слой выполняет определенные операции с входными данными и передает их в следующий слой.

5. Функция потерь: она определяет, насколько хорошо модель выполняет свою задачу. Цель обучения нейронной сети - минимизировать функцию потерь, чтобы достичь оптимальной производительности.

6. Оптимизатор: он оптимизирует веса и параметры модели в процессе обучения. Оптимизаторы используются для настройки модели таким образом, чтобы минимизировать функцию потерь.

Обучение нейронной сети происходит путем подачи тренировочных данных на вход модели, вычисления выходных данных и сравнения их с ожидаемыми значениями. Затем используется алгоритм обратного распространения ошибки (backpropagation), чтобы подкорректировать веса и параметры модели, чтобы минимизировать ошибку. Процесс обучения повторяется до достижения достаточной производительности модели.

Существует множество различных архитектур нейронных сетей, таких как полносвязные нейронные сети, сверточные нейронные сети (CNN), рекуррентные нейронные сети (RNN), а также более сложные модели, такие как глубокие нейронные сети (DNN), рекуррентные сверточные нейронные сети (RCNN), генеративно-состязательные сети (GAN) и другие.

Нейронные сети имеют широкий спектр применений в различных областях, включая компьютерное зрение, обработку естественного языка, робототехнику, финансы и многие другие. Они продолжают развиваться и иметь важное значение для развития искусственного интеллекта.

1 ОБЗОР И ВЫБОР ИСКУССТВЕННОЙ НЕЙРОСЕТИ

Для обучения нейронных сетей были отобраны пять моделей Keras и рассмотрены их характеристики (Таблица 1 [1]).

Таблица 1 – Характеристики моделей нейронных сетей

Model	Size (MB)	Top-1 Accuracy[1]	Top-5 Accuracy[2]	Parameters[3]	Depth[4]	Times(ms) per inference step[5](CPU)	Time(ms) per inference step(GPU)
VGG16	528	71.3%	90.1%	138.4M	16	69.5	4.2
ResNet50	98	74.9%	92.1%	25.6M	50	58.2	4.6
InceptionV3	92	77.9%	93.7%	23.9M	48	42.2	6.9
InceptionResNetV2	215	80.3%	95.3%	55.9M	164	130.2	10.0
EfficientNetB0	29	77.1%	93.3%	5.3M	82	46.0	4.9

1.1 Модель Keras VGG16

Модель Keras VGG16 — это предварительно обученная сверточная нейронная сеть, которая была разработана в 2014 году в рамках проекта Visual Geometry Group (VGG) Кембриджского университета. Она является одной из наиболее популярных архитектур в области компьютерного зрения45.

VGG16 состоит из 16 слоев, включая сверточные слои, слои объединения и полносвязные слои. Она имеет очень глубокую структуру, что

[1] Ответ модели должен быть в точности ожидаемым ответом
[2] Любой из пяти ответов с наибольшей вероятностью должен соответствовать ожидаемому ответу
[3] Количество параметров
[4] Количество слоев
[5] Время на шаг вывода

позволяет ей улавливать иерархические признаки изображений различных уровней абстракции.

Главная особенность VGG16 — это использование небольших сверточных фильтров размером 3x3 пикселя, применяемых последовательно. Такой подход помогает уменьшить количество параметров и улучшить обобщающую способность модели.

VGG16 предварительно обучена на большом наборе данных ImageNet, который содержит миллионы изображений из тысячи классов. Поэтому модель обладает способностью распознавать широкий спектр объектов и признаков.

В Keras VGG16 доступны две вариации:

1. VGG16 с тремя полносвязными слоями в конце, которые могут использоваться для классификации изображений.

2. VGG16 без последних полносвязных слоев, используемый для извлечения признаков из изображений, которые затем могут быть использованы в других моделях машинного обучения [2].

Keras предоставляет готовую реализацию VGG16, которую можно использовать для тренировки на своих данных или для извлечения признаков изображений в различных задачах компьютерного зрения. Архитектура работы модели представлена ниже (рисунок 1).

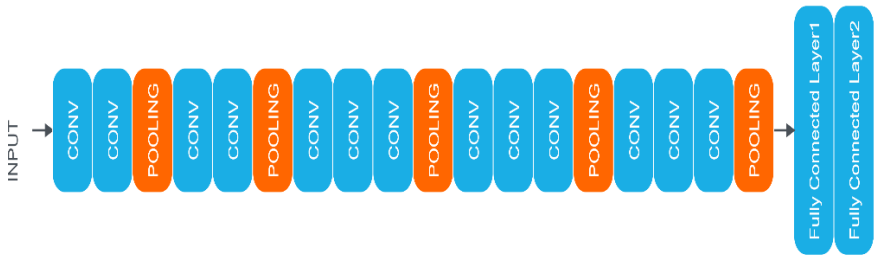

Рисунок 1 – Архитектура работы модели VGG16

1.2 Модель Keras ResNet50

Модель Keras ResNet50 — это еще одна предварительно обученная сверточная нейронная сеть, которая была разработана в 2015 году и является одной из самых популярных архитектур в области компьютерного зрения.

ResNet50 основана на идеи остаточных блоков (residual blocks), которые позволяют сети успешно обучаться при глубине больше 50 слоев. Основной проблемой глубоких нейронных сетей является затухание градиентов, что затрудняет процесс обучения. Однако, благодаря остаточным блокам, ResNet50 способна эффективно передавать информацию и градиенты через слои, что позволяет ей достичь очень глубоких структур и лучшей производительности.

ResNet50 состоит из 50 слоев, включая сверточные слои, слои объединения и блоки остатков. Она имеет очень глубокую структуру и способна извлекать высокоуровневые признаки изображений.

Модель ResNet50 также предварительно обучена на наборе данных ImageNet. Она обладает способностью распознавать широкий спектр объектов и признаков, как и VGG16 [3].

В Keras доступна готовая реализация ResNet50, которую можно использовать для классификации изображений или для извлечения признаков. Она также может быть дообучена на своих данных или использована в качестве базовой модели для решения других задач компьютерного зрения. Архитектура работы модели представлена ниже (рисунок 2).

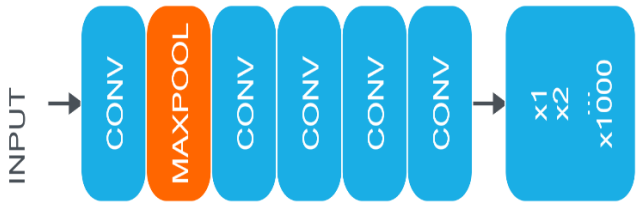

Рисунок 2 – Архитектура работы модели ResNet50

1.3 Модель Keras InceptionV3

Модель Keras InceptionV3 — это еще одна предварительно обученная сверточная нейронная сеть, которая была разработана командой Google в 2015 году. InceptionV3 является развитием предыдущей модели InceptionV1, которая была представлена в 2014 году.

Одной из главных особенностей InceptionV3 является использование модуля Inception, который позволяет сети параллельно изучать иерархические признаки различных размеров. Это позволяет модели эффективно улавливать как мелкие, так и крупные детали изображений.

InceptionV3 состоит из более чем 40 слоев, включая сверточные слои, слои объединения и блоки Inception. Она имеет глубокую и сложную структуру, которая позволяет ей извлекать высокоуровневые признаки изображений.

Модель InceptionV3 также предварительно обучена на наборе данных ImageNet. Она обладает способностью распознавать широкий спектр объектов и признаков. InceptionV3 также известна своей высокой точностью на задачах классификации изображений [4].

Keras предоставляет готовую реализацию InceptionV3, которую можно использовать для классификации изображений или для извлечения признаков. Модель может быть дообучена на своих данных или использована в качестве базовой модели для других задач компьютерного зрения. Архитектура работы модели представлена ниже (рисунок 3).

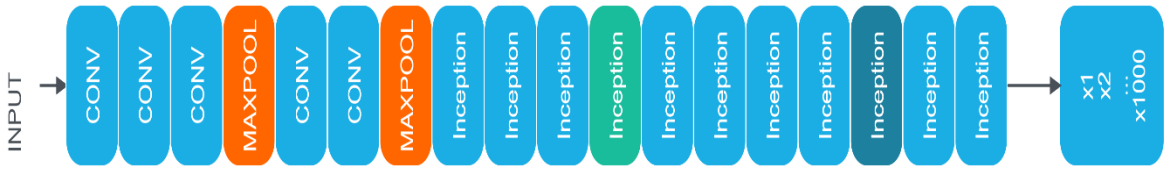

Рисунок 3 – Архитектура работы модели InceptionV3

1.4 Модель Keras InceptionResNetV2

Модель Keras InceptionResNetV2 — это предварительно обученная сверточная нейронная сеть, которая представляет собой комбинацию двух популярных архитектур - Inception и ResNet.

InceptionResNetV2 объединяет преимущества модуля Inception, позволяющего эффективно изучать иерархические признаки различных размеров, и идеи остаточных блоков (residual blocks) из архитектуры ResNet, обеспечивающих более глубокое обучение сети.

Модель InceptionResNetV2 состоит из более чем 570 слоев, включая сверточные слои, слои объединения, блоки Inception и блоки остатков. Она имеет очень глубокую и сложную структуру, которая позволяет ей извлекать высокоуровневые признаки изображений.

InceptionResNetV2 также предварительно обучена на наборе данных ImageNet. Эта модель обладает способностью распознавать широкий спектр объектов и признаков. Она известна своей высокой точностью на задачах классификации изображений [5].

Keras предоставляет готовую реализацию InceptionResNetV2, которую можно использовать для классификации изображений или для извлечения признаков. Модель может быть дообучена на своих данных или использована в качестве базовой модели для других задач компьютерного зрения. Архитектура работы модели представлена ниже (рисунок 4).

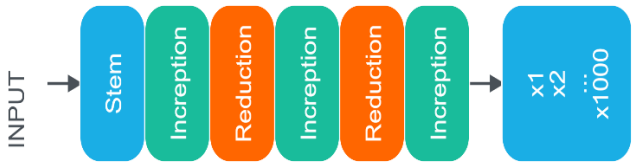

Рисунок 4 – Архитектура работы модели InceptionResNetV2

1.5 Модель Keras EfficientNetB0

Модель Keras EfficientNetB0 — это предварительно обученная сверточная нейронная сеть, которая отличается от других моделей своей эффективностью и меньшим количеством параметров при сохранении высокой точности.

EfficientNetB0 основана на идее автоматического масштабирования глубины, ширины и разрешения сети. Архитектура модели оптимизирована с помощью методов масштабирования, таких как Compound Scaling и Neural Architecture Search (NAS).

Модель EfficientNetB0 состоит из сверточных блоков, включая сверточные слои, слои объединения и точечные свертки. Она имеет глубокую структуру, которая позволяет ей извлекать высокоуровневые признаки изображений.

EfficientNetB0 также предварительно обучена на большом наборе данных ImageNet. Она обладает способностью распознавать широкий спектр объектов и признаков. Модель EfficientNetB0 известна своей хорошей производительностью и эффективностью в задачах классификации изображений [6].

Keras предоставляет готовую реализацию EfficientNetB0, которую можно использовать для классификации изображений или для извлечения признаков. Модель может быть дообучена на своих данных или использована в качестве базовой модели для решения других задач компьютерного зрения. Архитектура работы модели представлена ниже (рисунок 5).

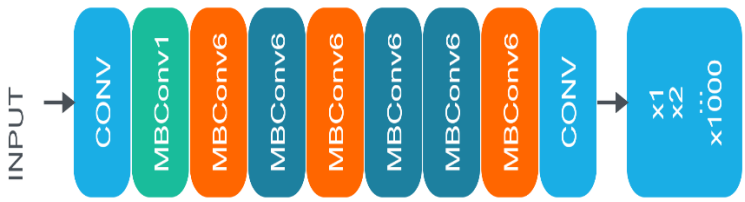

Рисунок 5 – Архитектура работы модели EfficientNetB0

1.6 Вывод

В итоге мы выбрали две модели: VGG16 и ResNet50. Выбранные модели являются достаточно простыми и основополагающими для многих других моделей.

2 ОБУЧАЮЩАЯ И ТЕСТИРУЮЩАЯ ВЫБОРКИ

Для формирования DataSet мы сделали снимки светофоров. Была использована камера iPhone 13 mini со следующими характеристиками: 12 Мп, f/1.6, 1.7 мкм, OIS сдвигом матрицы. Были сделаны 300 снимков, на 150 из них есть светофоры, на остальных – нет. Пятьдесят светофоров горят зеленым цветом, другие пятьдесят – желтым, остальные горят красным. Получившиеся изображения программа преобразует под размер 224р*224р. Примеры изображений представлены ниже (рисунки 6–9).

Рисунок 6 – Светофор с зеленым цветом

Рисунок 7 – Светофор с красным цветом

Рисунок 8 – Светофор с желтым цветом

Рисунок 9 – Столб

3 ОБУЧЕНИЕ И ТЕСТИРОВАНИЕ ИСКУССТВЕННОЙ НЕЙРОСЕТИ

В процессе обучения моделей нейронных сетей распознавать по изображению наличие светофора и каким цветом он горит, мы определили некоторые значения следующих характеристик: время обучения нейронной сети, количество эпох, качество обучения на обучающей и тестирующей выборках. Эти данные мы представили в виде таблиц (таблицы 2–3). Также мы отобразили сам процесс обучения нейросетей (рисунки 10–11). На этих графиках видно, что у модели VGG16 после 5-й эпохи началось переобучение - до нее шло обучение, а у модели ResNet50 переобучение началось после 11-й эпохи. На последнем графике показано сравнение обучения двух моделей (рисунок 12).

Таблица 2 – Результаты обучения и тестирования нейронных сетей

	Время обучения	Кол-во эпох	Качество обучающей выборки		Качество тестирующей выборки	
VGG16	38:18	5	0.96	96%	0.86	86%
ResNet50	16:24	11	0.75	75%	0.67	67%

Таблица 3 – Результаты тестирования нейронных сетей по классам

	Ошибка распознавания класса «Светофора нет»	Ошибка распознавания класса «Светофор красный»	Ошибка распознавания класса «Светофор желтый»	Ошибка распознавания класса «Светофор зеленый»	Ошибка распознавания класса «Светофор есть»
VGG16	5%	25%	35%	30%	16%
ResNet50	10%	5%	65%	80%	27%

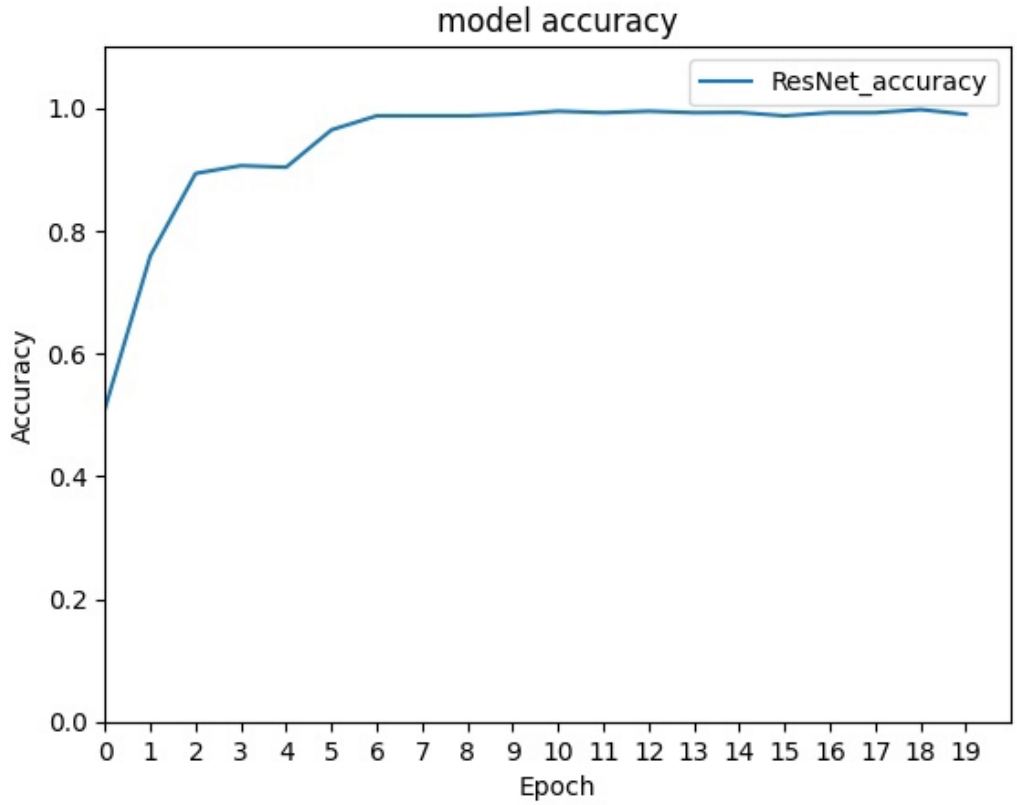

Рисунок 10 – Отображение обучения нейронной сети модели ResNet50

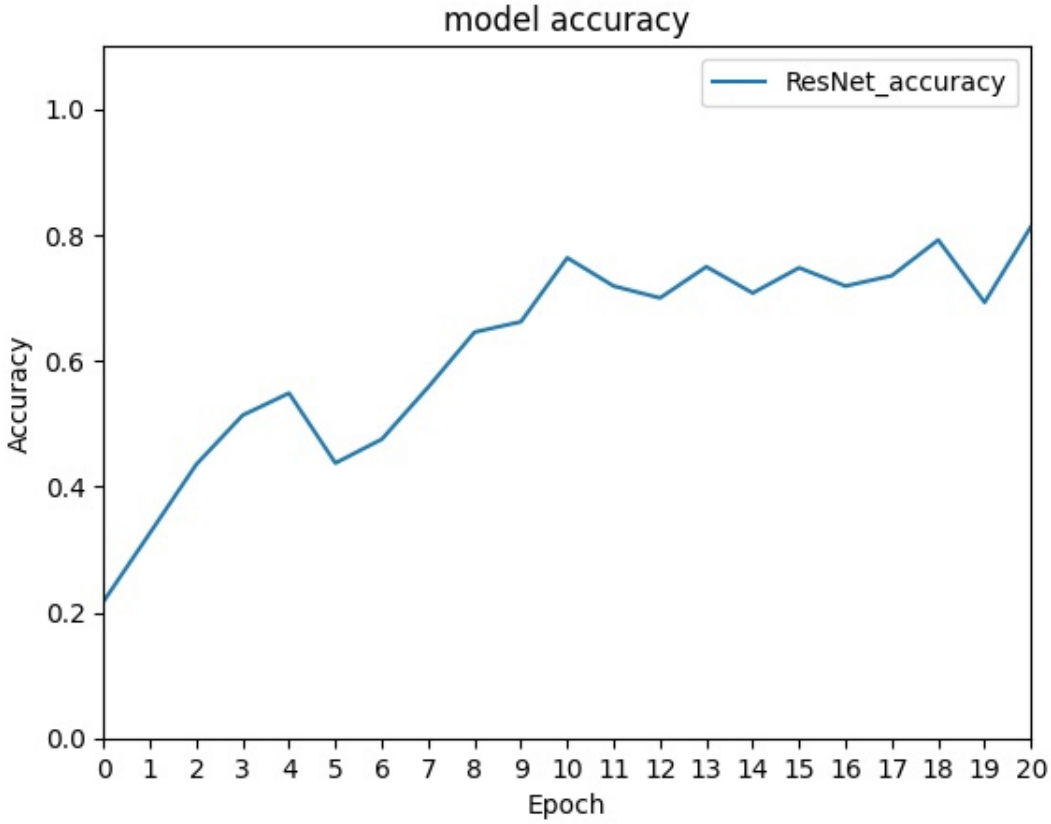

Рисунок 11 - Отображение обучения нейронной сети модели VGG16

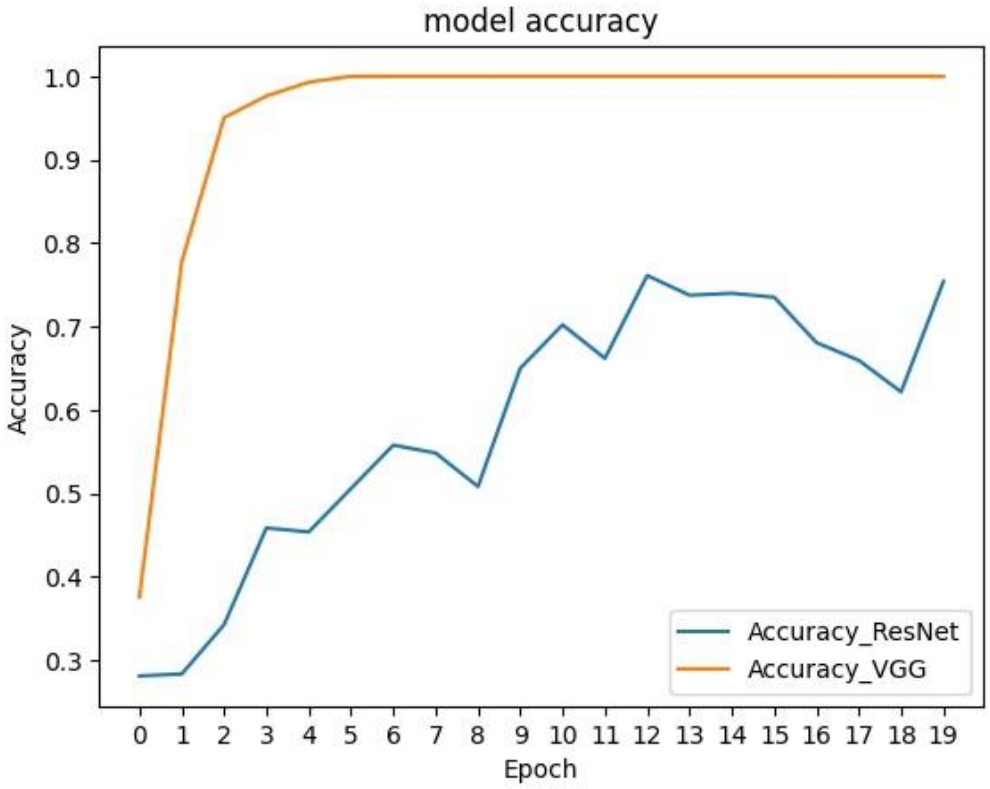

Рисунок 12 – Отображение сравнения обучения двух моделей

В результате мы выбрали модель VGG16, так как ее архитектура проще, чем ResNet50, и качество обучения на тестирующей выборке у этой модели оказалось лучше. Также VGG16 стала лучшим выбором для небольших наборов данных, так как ее более простая архитектура имеет меньше параметров и может иметь меньшие требования к вычислительным ресурсам.

4 РАЗРАБОТКА ПО

После обучения ИНС мы приступили к разработке программного обеспечения, так как ПО позволяет внедрять и применять искусственный интеллект в различных областях.

4.1 Интеграция ИНС

ИНС загружается из H5 файла, который мы получаем в результате обучения нейронной сети. При запуске программы мы загружаем нейросеть.

Формат H5 — это открытый формат, предназначенный для компактного хранения данных моделей нейронных сетей. Он обеспечивает высокий уровень сжатия данных, позволяя сохранить в себе полную информацию о модели, включая ее архитектуру, веса, оптимизаторы и метрики.

4.2 Интерфейс

Интерфейс реализован в виде одного класса главного окна с помощью библиотеки PyQt. Структура интерфейса была создана в программе qt designer и загружена в программу в самом начале. Реализованы 3 строки и одна кнопка. Функционал кнопки выделен в функцию, которая запрашивает изображение с диска, а позже обрабатывает её с помощью обученной нейронной сети

4.3 Протокол тестирования

На рисунках 13–16 представлен протокол тестирования ПО для всех классов.

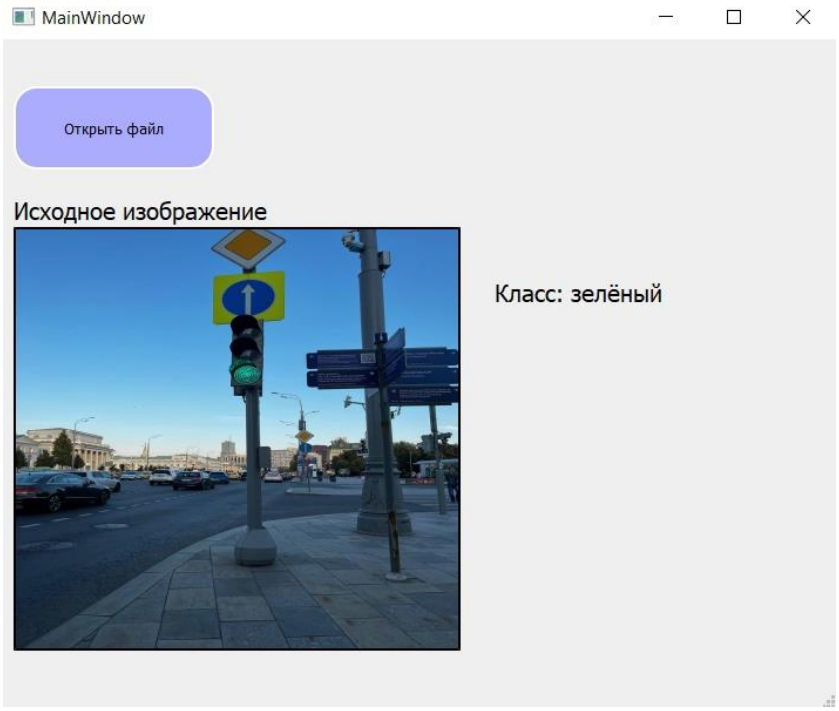

Рисунок 13 – Протокол тестирования ПО для класса «зеленый»

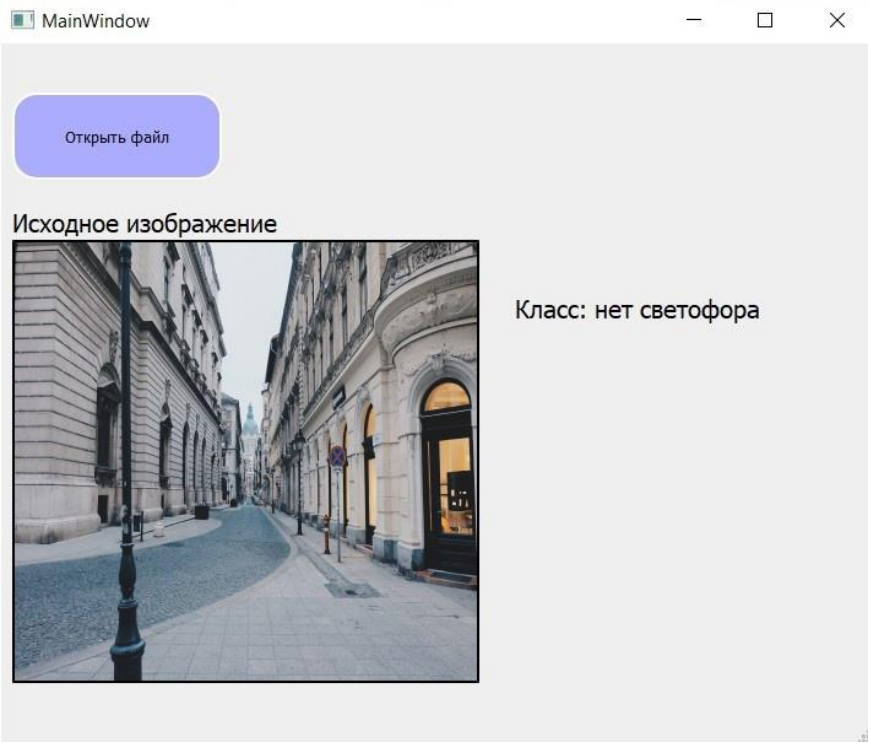

Рисунок 14 – Протокол тестирования ПО для класса «нет светофора»

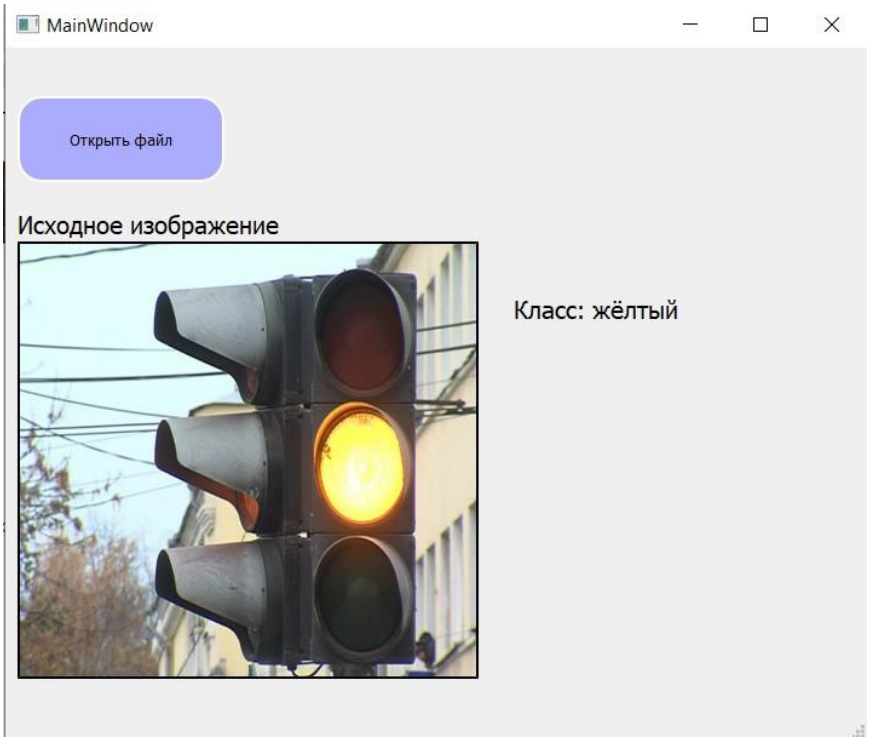

Рисунок 15 – Протокол тестирования ПО для класса «желтый»

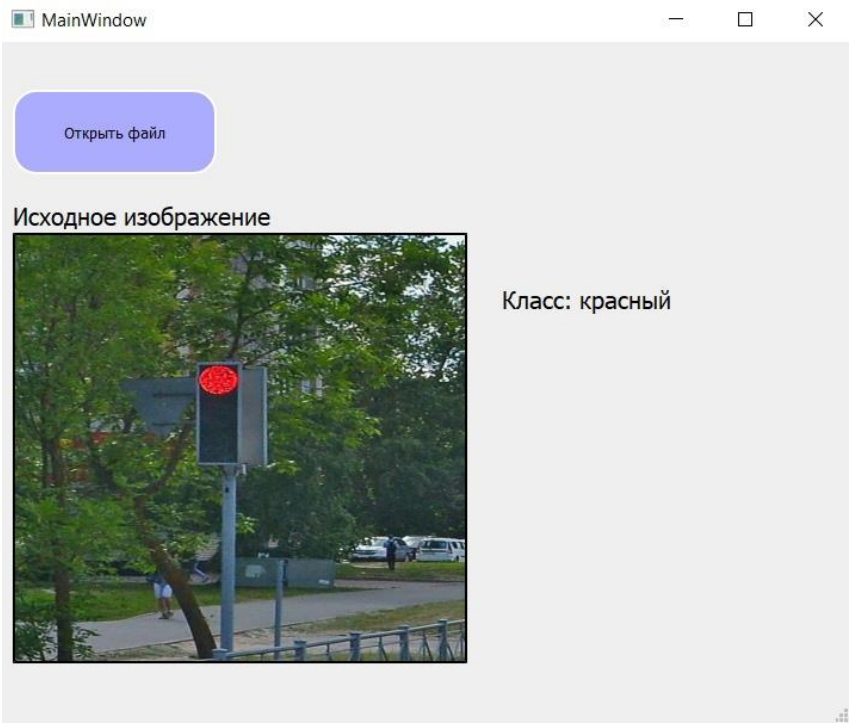

Рисунок 16 – Протокол тестирования ПО для класса «красный»

4.4 Вывод

По результатам тестирования можно сделать вывод, что разработанное ПО работает корректно.

ЗАКЛЮЧЕНИЕ

В ходе данного проекта была разработана и успешно протестирована искусственная нейронная сеть (ИНС) для распознавания светофора и определения его цвета. Использование нейронных сетей для распознавания цвета светофора представляет собой важную область исследований в области компьютерного зрения и автоматизированных систем управления транспортом.

В процессе проведения исследования были реализованы оптимальные методы обучения нейронных сетей для эффективного распознавания цветов светофора в различных ракурсах съемки. Разработанная ИНС показала достаточно высокую точность и надежность в распознавании цветовых сигналов.

Результаты данного проекта имеют важное практическое применение в области управления транспортным потоком и обеспечения безопасности на дорогах. Использование искусственной нейронной сети для определения цветов светофора может значительно повысить эффективность и точность систем управления светофорами, а также способствовать созданию более интеллектуальных и автоматизированных систем управления транспортным движением.

В целом, результаты данного проекта свидетельствуют о потенциале искусственных нейронных сетей в области распознавания светофоров и автоматизации систем управления транспортным движением. Дальнейшие исследования в этой области могут привести к созданию более эффективных и надежных систем распознавания цветов светофоров с использованием искусственных нейронных сетей.

ПРИЛОЖЕНИЕ А

```
/*
© Copywrite by Kupriyanov Vladislav Vitalievich student of УВВ-211, RUT(MIIT)
All rights are protected.
Non-licence code distribution is strongly forbidden
*/


from PIL import Image
import tensorflow as tf
from tensorflow import keras
import numpy as np
from keras.models import Sequential, load_model
import sys
from PyQt5.QtWidgets import QMainWindow, QApplication, QPushButton, QFileDialog, QLabel
from PyQt5 import uic
from PyQt5.QtGui import QPixmap

model = keras.models.load_model('C:/Users/й/OneDrive/Desktop/Project/h5/VGG16.h5')

class MainWindow(QMainWindow):

    def __init__(self):
        super(MainWindow, self).__init__()
        #load the ui file
        uic.loadUi("dialog.ui", self)
```

```
        self.path = ""
        #define widgets
        self.button = self.findChild(QPushButton, "pushButton")
        self.label1 = self.findChild(QLabel, "label")
        self.label2 = self.findChild(QLabel, "label_2")
        self.label3 = self.findChild(QLabel, "label_3")
        #Click the DropDown box
        self.button.clicked.connect(self.clicker)

        #Show the app
        self.show()

    def clicker(self):
        self.path = QFileDialog.getOpenFileName(self, "Open File", "", "jpg(*.jpg)")
        #Открываем изображение
        self.pixmap = QPixmap(self.path[0])
        #Добавляем изображение в строку
        self.label3.setPixmap(self.pixmap)

        class_labels = ['жёлтый','зелёный', 'нет светофора', 'красный']
        img = Image.open(self.path[0])
        img = img.resize((224,224))
        img = np.array(img)
        img = np.expand_dims(img, axis = 0)
        img = img / 255.

        preds = model.predict(img)
        print(preds)
        class_index = np.argmax(preds)
        predicted_class = class_labels[class_index]
```

```
        self.label2.setText("Класс: " + str(predicted_class))

"""
def predict():
    class_labels = ['жёлтый','зелёный', 'котики', 'красный']
    model                                                                    =
keras.models.load_model('C:/Users/Vlad/Desktop/Project/h5/ResNet50.h5')

    cat_folder = '/content/drive/MyDrive/Project/Test/PєPsC,PёPєPё'
    img = Image.open("F:\conda\Project\Validation\\Nonlight\\cat.4350.jpg")
    img = img.resize((224,224))
    img = np.array(img)
    img = np.expand_dims(img, axis = 0)
    img = img / 255. # PќPsCЂPjP°P»PёP·P°C†PёCЏ

    preds = model.predict(img)

    class_index = np.argmax(preds)
    predicted_class = class_labels[class_index]
    confidence = preds[0][class_index]

    print(f"Predicted class: {predicted_class}")
    print(f"Confidence: {confidence}")

"""

app = QApplication(sys.argv)
window = MainWindow()
print(window.path)
app.exec()
```

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Keras Documentation: Keras Applications. Электронный ресурс. keras.io. URL: https://keras.io/api/applications/ [Дата обращения: 01.10.23]
2. VGG16 — нейросеть для выделения признаков изображений. Электронный ресурс. neurohive.io. URL: https://neurohive.io/ru/vidy-nejrosetej/vgg16-model/ [Дата обращения: 11.10.23]
3. ResNet (34, 50, 101): «остаточные» CNN для классификации изображений. Электронный ресурс. neurohive.io. URL: https://neurohive.io/ru/vidy-nejrosetej/resnet-34-50-101/ [Дата обращения: 11.10.23]
4. 10 предварительно обученных моделей для встраивания изображений, которые должен знать каждый специалист по данным. Электронный ресурс. vc.ru. URL: https://vc.ru/u/1389654-machine-learning/678485-10-predvaritelno-obuchennyh-modeley-dlya-vstraivaniya-izobrazheniy-kotorye-dolzhen-znat-kazhdyy-specialist-po-dannym [Дата обращения: 11.10.23]
5. 10 предварительно обученных моделей для встраивания изображений, которые должен знать каждый специалист по данным. Электронный ресурс. vc.ru. URL: https://vc.ru/u/1389654-machine-learning/678485-10-predvaritelno-obuchennyh-modeley-dlya-vstraivaniya-izobrazheniy-kotorye-dolzhen-znat-kazhdyy-specialist-po-dannym [Дата обращения: 11.10.23]
6. Обзор EfficientNet: повышение точности и надежности CNN. Электронный ресурс. evogeek.ru. URL: https://evogeek.ru/articles/277119/ [Дата обращения: 11.10.23]