КАЗАХСКИЙ АГРОТЕХНИЧЕСКИЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИМЕНИ САКЕНА СЕЙФУЛЛИНА

Энергетический факультет

Кафедра эксплуатации электрооборудования
Специальность: D100 «Автоматизация и управление»

ОТЧЕТ

по зарубежной стажировке докторанта в Томском государственном университете систем управления и радиоэлектроники (ТУСУР)

Докторант: Амир Е. К.
Научный руководитель: Сарсикеев Е. Ж. (PhD)

АСТАНА 2024

СОДЕРЖАНИЕ

Введение

Тема: «Управление техническими системами»

Место стажировки: Федеральное государственное автономное образовательное учреждение высшего образования «Томский государственный университет систем управления и радиоэлектроники» г. Томск, Российская Федерация

Период прохождения стажировки: с 27 мая 20024 г. по 28 июня 2024 г.

Цели: Получение практических навыков по разработке и применению моделей машинного обучения в реальных проектах. Изучение статистических методов анализа данных и обоснования полученных результатов.

Результаты: Были проведены теоретические и лабораторные занятия в размере 120 часов. Проведено ознакомление с основными инструментами моделирования и разработки технологии машинного обучения, такие как «Keras», «TensorFlow», «Pandas», «Numpy», «Matplotlib» и т.д.. Также был проведен краткий курс введение в основые языка программирования «Python». Разработана и применена модель оценки ресурса пастбища на основе спутниковых мультиспектарльных изображении и показателей урожайности. Рассмотрены основные статистические методы анализа данных и их характерные особенности.

Компьютерное зрение - это область искусственного интеллекта, которая позволяет машинам интерпретировать визуальные данные из изображений, видео и других визуальных данных. Оно включает в себя получение, обработку, анализ и понимание цифровых изображений для извлечения значимой информации.

В основе компьютерного зрения лежит применение теорий и моделей для создания систем компьютерного зрения. Эти системы могут "видеть" и понимать реальный мир путем обработки визуальных данных.

Анализ изображений, с помощью которого компьютеры "видят", предполагает представление изображений в виде матриц 1 и 0. Каждый пиксель соответствует значению цвета, формируя изображение. Машинное обучение обеспечивает этот процесс, что приводит к тому, что мы называем компьютерным зрением[1].

Преимущества распознавания изображений с помощью МЛ:

1. Скорость и точность: Распознавание изображений с помощью ИИ отличается скоростью и точностью. Он может быстро и с высокой точностью идентифицировать объекты на изображениях.

2. Масштабируемость: Системы распознавания изображений на основе ML масштабируются без особых усилий. Анализируя тысячи или миллионы изображений, они сохраняют стабильную производительность.
3. Экономическая эффективность: По сравнению с ручным трудом использование ИИ для распознавания изображений экономически выгодно в долгосрочной перспективе.
4. Адаптивность: Системы ИИ адаптируются к различным сценариям, что делает их универсальными для различных приложений[2].

Глубокое обучение и извлечение абстрактных признаков:

Методы глубокого обучения играют важнейшую роль в распознавании изображений. Они автоматически обучаются на основе данных и извлекают абстрактные признаки, необходимые для идентификации объектов.

Эти признаки позволяют системам распознавать объекты даже в сложных ситуациях, способствуя высокой точности[3].

Современные системы распознавания изображений, основанные на искусственном интеллекте и машинном обучении, выявляют скрытые закономерности в коллекциях изображений. Эти системы принимают независимые интеллектуальные решения, основанные на знаниях, не очевидных для человеческого глаза. Благодаря распознаванию изображений отрасли получают выгоду от автоматизации процессов, повышения безопасности и улучшения качества обслуживания клиентов[5].

Применение технологии искусственного интеллекта для определения степени стравливания пастбища при помощи облачного сервиса “Teachable Machine”

Неправильная подборка изображении

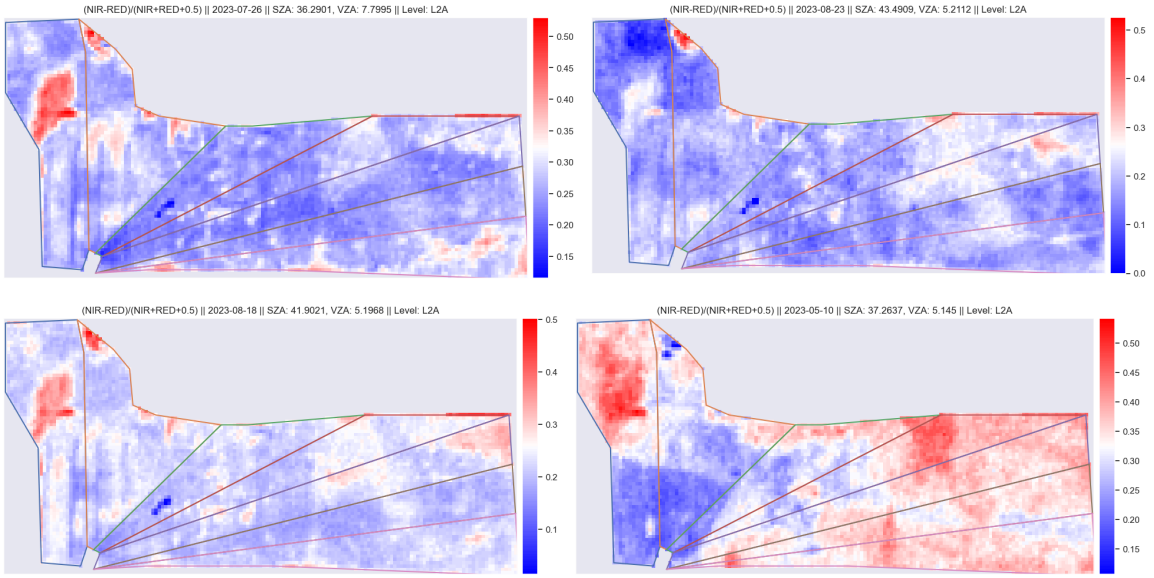


Рисунок 1 – Плохая выборка изображении более стравленного пастбища

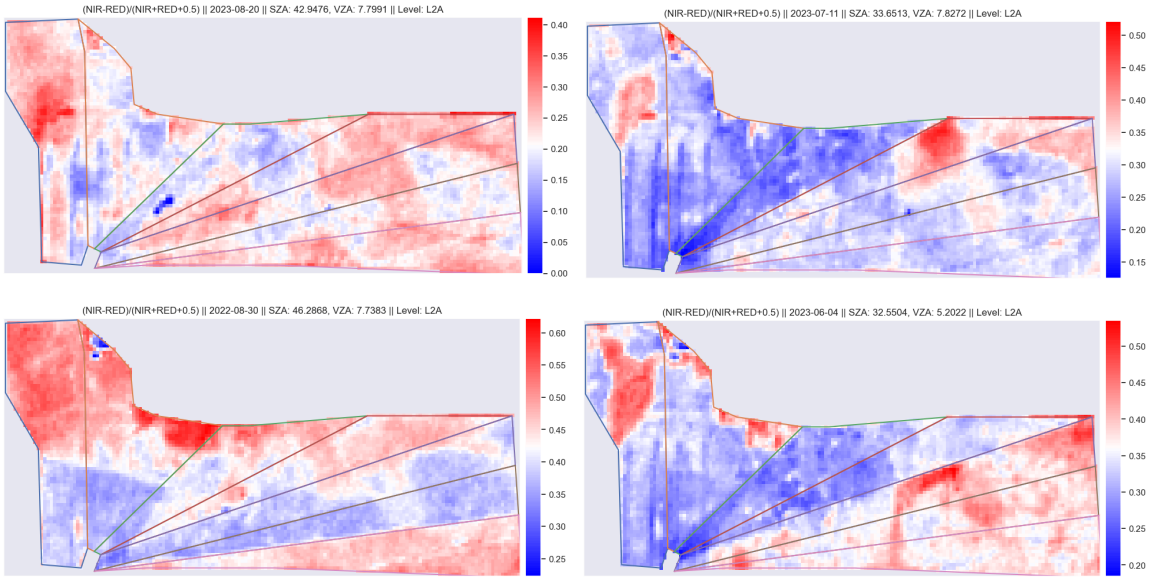


Рисунок 2 – Плохая выборка изображении менее стравленного пастбища

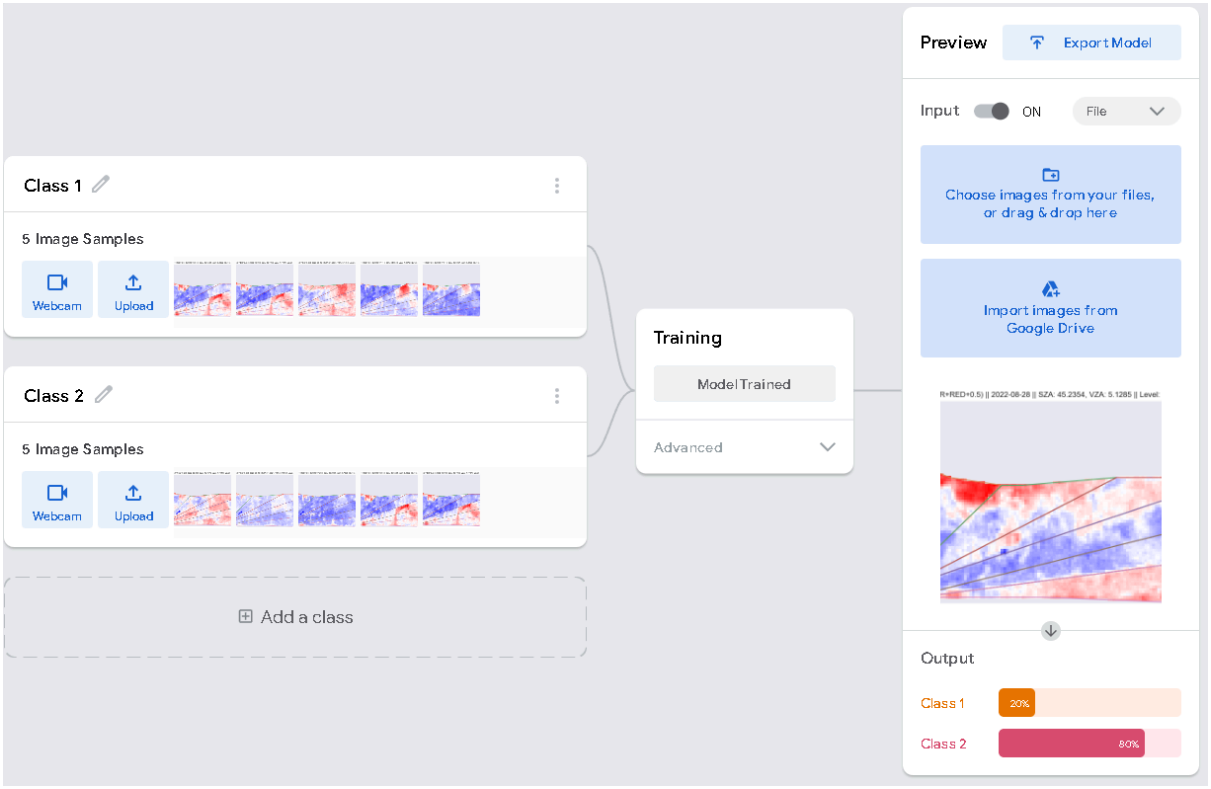


Рисунок 3 – Тестирование обученной модели. Неверное определение более стравленного пастбища

Правильная подборка изображении

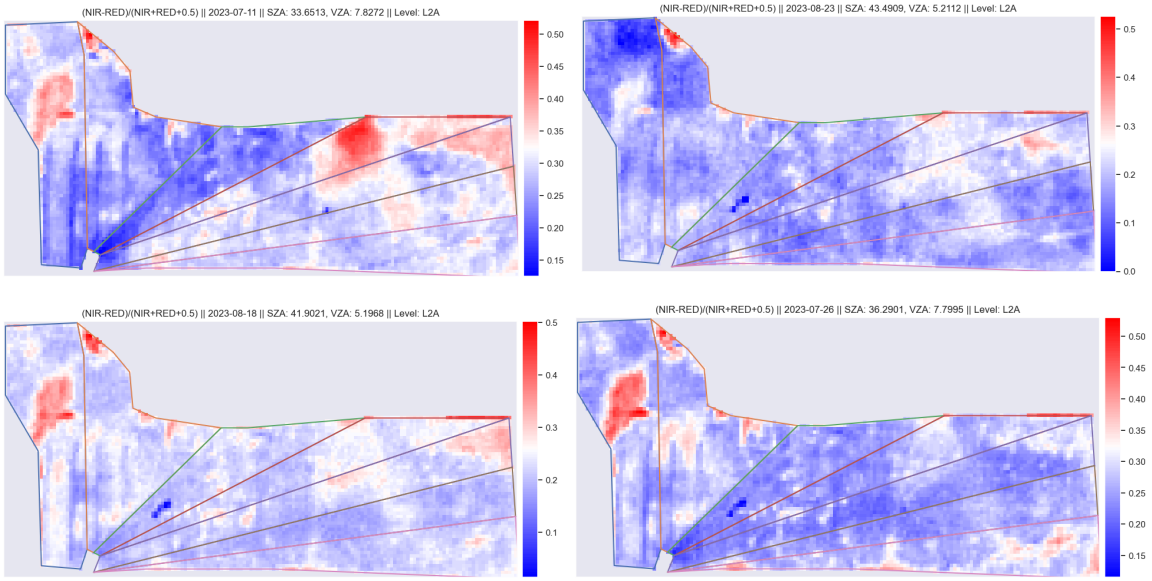


Рисунок 4 – Правильная выборка изображении более стравленного пастбища

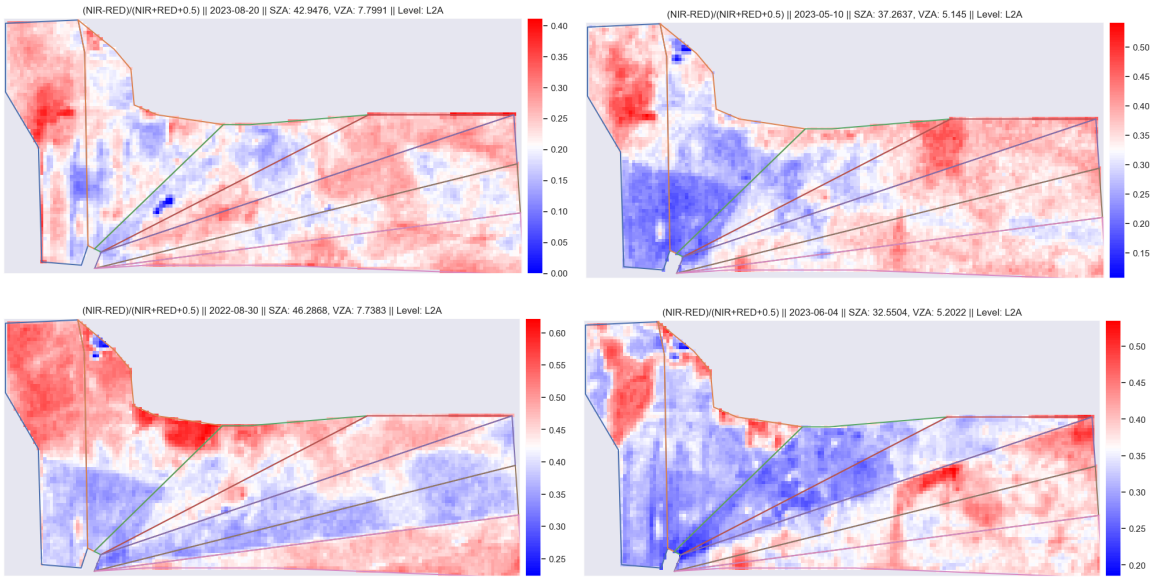


Рисунок 5 – Правильная выборка изображении менее стравленного пастбища

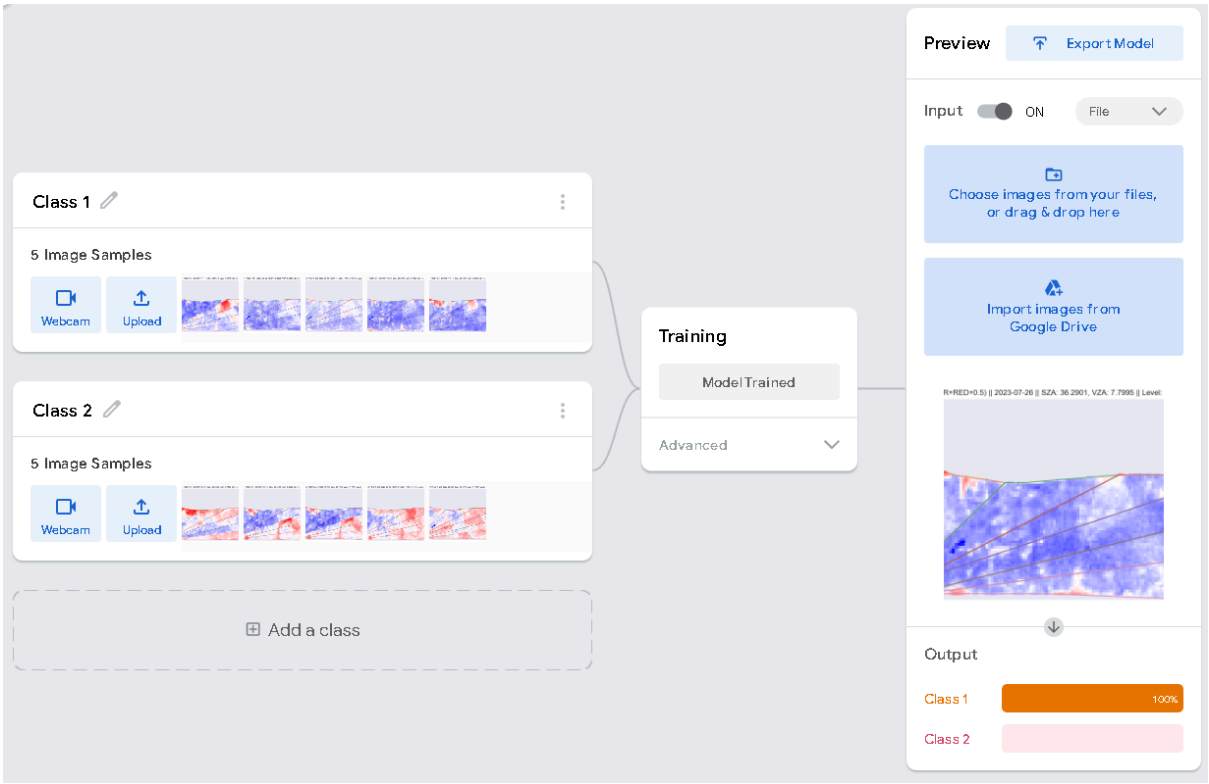


Рисунок 6 – Тестирование обученной модели. Правильное определение более стравленного пастбища

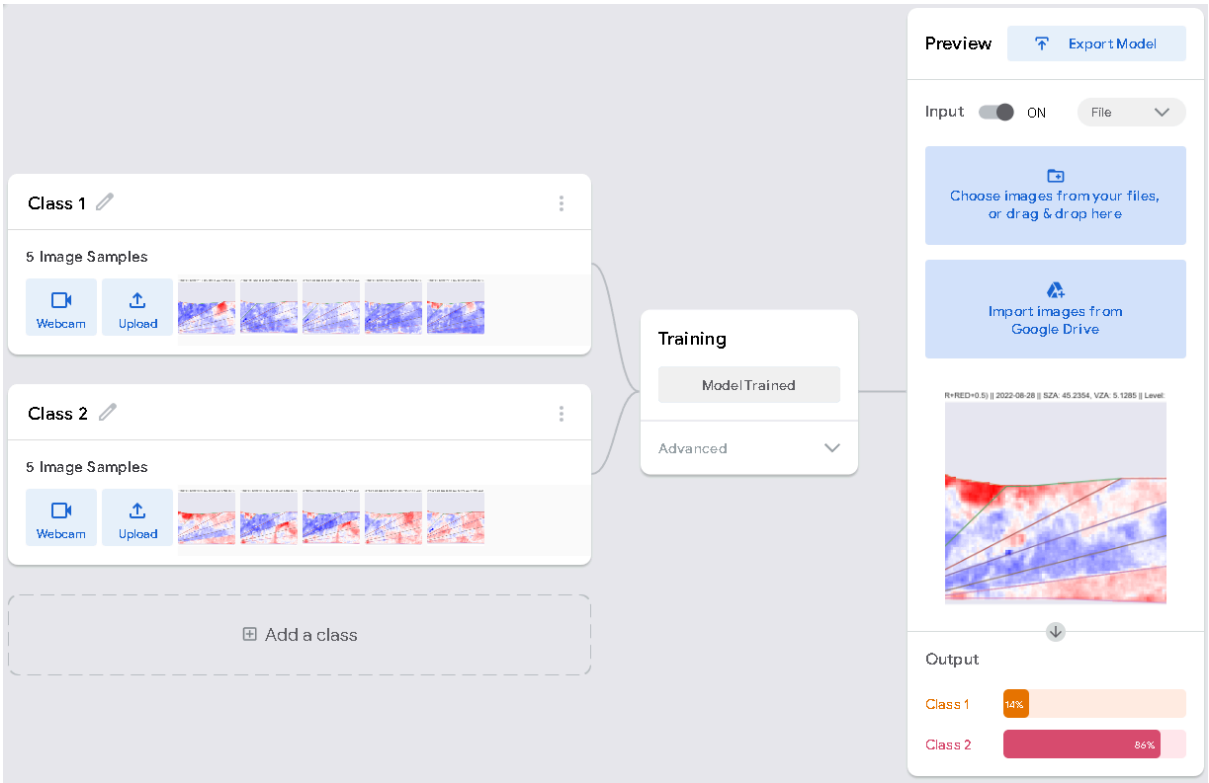


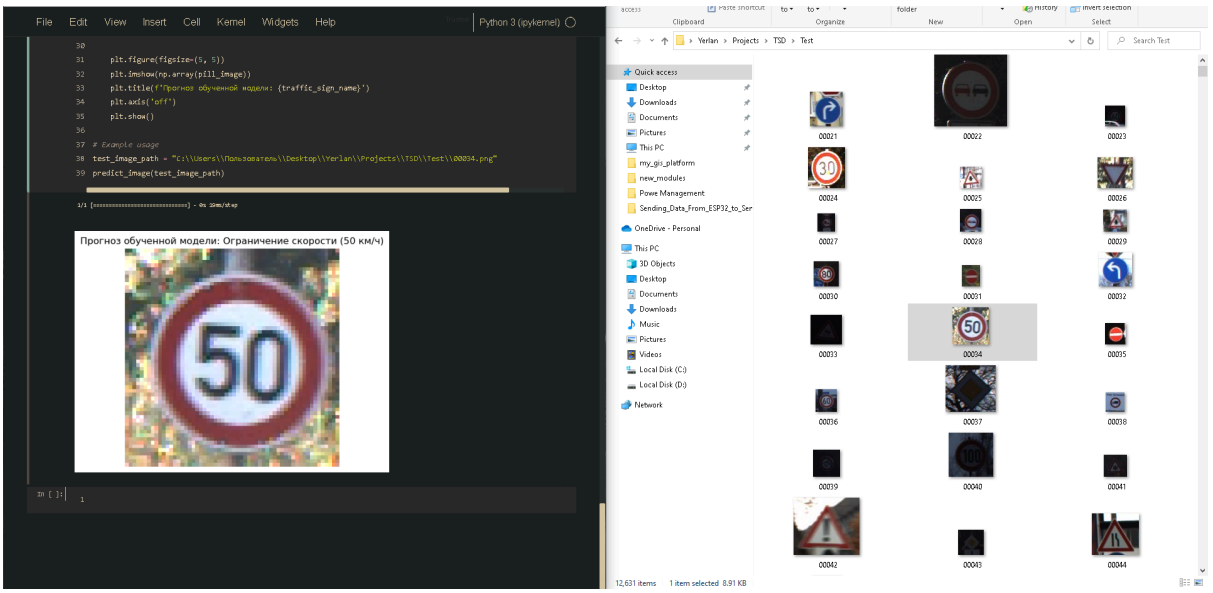
Рисунок 7 – Тестирование обученной модели. Правильное определение менее стравленного пастбища

Разработка модели машинного обучения для распознавания дорожных знаков с помощью CNN и Keras на Python в среде разработки Jupyter Notebook

В этой лабораторной работе мы построим модель для классификации дорожных знаков, доступных на изображении, по многим категориям, используя сверточную нейронную сеть (CNN) и библиотеку Keras.

Набор данных изображений состоит из более чем 50 000 изображений различных дорожных знаков (ограничение скорости, перекрестки, сигналы светофора и т.д.) В наборе данных для классификации изображений представлено 43 различных классов (видов дорожных знаков).

Рисунок – 8. Пример использованных изображении дорожных знаков для обучения модели



Слева – Применение модели в среде разработки Jupyter Notebook; Справа – папка с тестовыми изображениями

Рисунок – 9. Пример прогноза обученой модели на новых тестовых данных.

Рисунок – 10. Примеры прогнозов обученой модели на новых тестовых данных

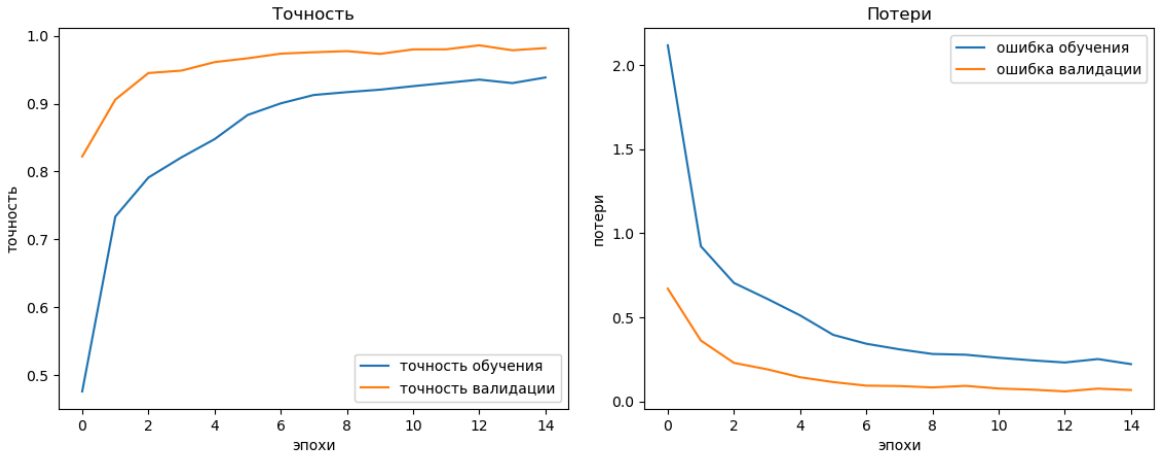


Рисунок – 11. Динамика изменении параметров точности и потери в процессе обучения модели

```
Epoch 1/15
981/981 [==============================] - 70s 70ms/step - loss: 2.1156 - accuracy: 0.4761 - val_loss: 0.6699 - val_accuracy: 0.8221
Epoch 2/15
981/981 [==============================] - 69s 71ms/step - loss: 0.9217 - accuracy: 0.7335 - val_loss: 0.3615 - val_accuracy: 0.9059
Epoch 3/15
981/981 [==============================] - 64s 66ms/step - loss: 0.7038 - accuracy: 0.7911 - val_loss: 0.2288 - val_accuracy: 0.9450
Epoch 4/15
981/981 [==============================] - 67s 69ms/step - loss: 0.6100 - accuracy: 0.8209 - val_loss: 0.1909 - val_accuracy: 0.9486
Epoch 5/15
981/981 [==============================] - 66s 68ms/step - loss: 0.5110 - accuracy: 0.8476 - val_loss: 0.1435 - val_accuracy: 0.9611
Epoch 6/15
981/981 [==============================] - 71s 73ms/step - loss: 0.3954 - accuracy: 0.8833 - val_loss: 0.1149 - val_accuracy: 0.9667
Epoch 7/15
981/981 [==============================] - 71s 73ms/step - loss: 0.3427 - accuracy: 0.9004 - val_loss: 0.0936 - val_accuracy: 0.9736
Epoch 8/15
981/981 [==============================] - 73s 74ms/step - loss: 0.3094 - accuracy: 0.9127 - val_loss: 0.0911 - val_accuracy: 0.9756
Epoch 9/15
981/981 [==============================] - 66s 67ms/step - loss: 0.2818 - accuracy: 0.9169 - val_loss: 0.0831 - val_accuracy: 0.9773
Epoch 10/15
981/981 [==============================] - 65s 66ms/step - loss: 0.2774 - accuracy: 0.9205 - val_loss: 0.0923 - val_accuracy: 0.9732
Epoch 11/15
981/981 [==============================] - 73s 74ms/step - loss: 0.2592 - accuracy: 0.9257 - val_loss: 0.0766 - val_accuracy: 0.9799
Epoch 12/15
981/981 [==============================] - 70s 71ms/step - loss: 0.2440 - accuracy: 0.9305 - val_loss: 0.0700 - val_accuracy: 0.9800
Epoch 13/15
981/981 [==============================] - 70s 72ms/step - loss: 0.2312 - accuracy: 0.9353 - val_loss: 0.0596 - val_accuracy: 0.9858
Epoch 14/15
981/981 [==============================] - 66s 67ms/step - loss: 0.2516 - accuracy: 0.9302 - val_loss: 0.0758 - val_accuracy: 0.9786
Epoch 15/15
981/981 [==============================] - 69s 70ms/step - loss: 0.2215 - accuracy: 0.9385 - val_loss: 0.0678 - val_accuracy: 0.9818
```

Рисунок – 12. Динамика изменении параметров точности и потери в процессе обучения модели с длительностью в 15 эпох

Ниже приведен листинг обученной модели.

```
# Импортирование необходимых библиотек
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import cv2
import tensorflow as tf
```

```
from PIL import Image
import os
from sklearn.model_selection import train_test_split
from keras.utils import to_categorical
from keras.models import Sequential, load_model
from keras.layers import Conv2D, MaxPool2D, Dense, Flatten, Dropout
from sklearn.metrics import accuracy_score

# Объявление переменных
data = []
labels = []
classes = 43
cur_path = os.getcwd()

# Получение изображений и их меток
for i in range(classes):
    path = os.path.join(cur_path, 'Train', str(i))
    images = os.listdir(path)
    for a in images:
        try:
            image = Image.open(os.path.join(path, a))
            image = image.resize((30, 30))
            image = np.array(image)
            data.append(image)
            labels.append(i)
        except Exception as e:
            print(f"Ошибка загрузки изображения {a}: {e}")

# Преобразование списков в массивы numpy
data = np.array(data)
labels = np.array(labels)
print(data.shape, labels.shape)

# Разделение данных на обучающий и тестовый наборы
X_train, X_test, y_train, y_test = train_test_split(data, labels, test_size=0.2,
random_state=42)
print(X_train.shape, X_test.shape, y_train.shape, y_test.shape)

# Преобразование меток в кодировку one-hot
y_train = to_categorical(y_train, classes)
y_test = to_categorical(y_test, classes)

# Построение модели
```

```
model = Sequential()
model.add(Conv2D(filters=32, kernel_size=(5, 5), activation='relu', input_shape=X_train.shape[1:]))
model.add(Conv2D(filters=32, kernel_size=(5, 5), activation='relu'))
model.add(MaxPool2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.25))
model.add(Conv2D(filters=64, kernel_size=(3, 3), activation='relu'))
model.add(Conv2D(filters=64, kernel_size=(3, 3), activation='relu'))
model.add(MaxPool2D(pool_size=(2, 2)))
model.add(Dropout(rate=0.25))
model.add(Flatten())
model.add(Dense(256, activation='relu'))
model.add(Dropout(rate=0.5))
model.add(Dense(classes, activation='softmax'))

# Компиляция модели
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])

# Обучение модели
epochs = 15
history = model.fit(X_train, y_train, batch_size=32, epochs=epochs, validation_data=(X_test, y_test))

# Построение графиков для точности
plt.figure(0)
plt.plot(history.history['accuracy'], label='точность обучения')
plt.plot(history.history['val_accuracy'], label='точность валидации')
plt.title('Точность')
plt.xlabel('эпохи')
plt.ylabel('точность')
plt.legend()
plt.show()

plt.figure(1)
plt.plot(history.history['loss'], label='ошибка обучения')
plt.plot(history.history['val_loss'], label='ошибка валидации')
plt.title('Потери')
plt.xlabel('эпохи')
plt.ylabel('потери')
plt.legend()
plt.show()
```

```
# Тестирование точности на тестовом наборе данных
y_test_df = pd.read_csv('Test.csv')
labels = y_test_df["ClassId"].values
imgs = y_test_df["Path"].values
data = []

for img in imgs:
    image = Image.open(img)
    image = image.resize((30, 30))
    data.append(np.array(image))

X_test = np.array(data)
pred = np.argmax(model.predict(X_test), axis=-1)

# Точность на тестовых данных
print(f"Точность: {accuracy_score(labels, pred)}")

# Сохранение модели
model.save("traffic_classifier.h5")

# Загрузка модели
# model = load_model('traffic_classifier.h5')

# Словарь для пометки всех классов дорожных знаков
rus_classes = {
    1: 'Ограничение скорости (20 км/ч)', 2: 'Ограничение скорости (30 км/ч)', 3: 'Ограничение скорости (50 км/ч)',
    4: 'Ограничение скорости (60 км/ч)', 5: 'Ограничение скорости (70 км/ч)', 6: 'Ограничение скорости (80 км/ч)',
     7: 'Конец ограничения скорости (80 км/ч)', 8: 'Ограничение скорости (100 км/ч)', 9: 'Ограничение скорости (120 км/ч)',
    10: 'Проезд запрещен', 11: 'Запрещен проезд автомобилей массой более 3,5 тонн', 12: 'Право прохода на перекрестке',
    13: 'Приоритетная дорога', 14: 'Уступи дорогу', 15: 'Остановка', 16: 'Нет транспортных средств',
    17: 'Запрещено движение транспортных средств массой более 3,5 тонн', 18: 'Въезд запрещен', 19: 'Общая осторожность',
     20: 'Опасный поворот налево', 21: 'Опасный поворот направо', 22: 'Двойной поворот', 23: 'Ухабистая дорога',
     24: 'Скользкая дорога', 25: 'Дорога сужается справа', 26: 'Дорожные работы', 27: 'Дорожные сигналы',
    28: 'Пешеходы', 29: 'Дети переходят дорогу', 30: 'Велосипедный переход', 31: 'Осторожно, гололед/снег',
```

```
    32: 'Переход диких животных',
    33: 'Конечная скорость + ограничения на проезд',
    34: 'Поворот направо',
    35: 'Поворот налево',
    36: 'Движение только вперед',
    37: 'Ехать прямо или направо',
    38: 'Ехать прямо или налево',
    39: 'Держитесь правее',
    40: 'Держитесь левее',
    41: 'Круговой перекресток обязателен',
    42: 'Конец запрета проезда',
    43: 'Запрещается проезд транспортных средств массой более 3,5 тонн'
}

# Функция для предсказания по одному изображению
def predict_image(image_path):
    # Загрузка и обработка изображения
    pill_image = Image.open(image_path)
    image = pill_image.resize((30,30))
    image = np.expand_dims(image, axis=0)
    image = np.array(image)
    pred = model.predict([image])
    predicted_class = np.argmax(pred, axis=-1)

    # Получение названия дорожного знака
    traffic_sign_name = rus_classes[predicted_class[0]+1]

    # Визуализация изображения и его класса
    plt.figure(figsize=(5, 5))
    plt.imshow(np.array(pill_image))
    plt.title(f'Прогноз обученной модели: {traffic_sign_name}')
    plt.axis('off')
    plt.show()

# Пример использования
test_image_path                                                                    =
"C:\\Users\\Пользователь\\Desktop\\Yerlan\\Projects\\TSD\\Test\\00034.png"
predict_image(test_image_path)
```

Листинг кода модели машинного обучения

Выводы

Машинное обучение, направление искусственного интеллекта, наделяет компьютерные системы способностью обучаться и совершенствоваться на основе опыта без явного программирования. Оно включает в себя как контролируемые, так и неконтролируемые методы, позволяющие решать задачи быстрее и эффективнее, чем это может сделать обычный человеческий мозг. Кроме того, успешное машинное обучение основывается на понимании данных, выборе подходящих алгоритмов и документировании архитектуры сети и гиперпараметров для воспроизводимости.

Полученные модели машинного обучения, а прежде всего, **приобретенные навыки** по проектированию данных моделей в ходе пройденной стажировки в будущем **могут быть использованы в диссертационной работе** для определения урожайности рассматриваемого пастбища, при условии наличия достаточного количества и качества тренировочных данных.

Ссылки на источники

1. Seeing Like a Machine: A Beginner's Guide to Image Analysis in Machine https://www.datacamp.com/tutorial/seeing-like-a-machine-a-beginners-guide-to-image-analysis-in-machine-learning.
2. How Machine Learning is Revolutionizing Image Recognition. https://techbullion.com/how-machine-learning-is-revolutionizing-image-recognition/.
3. How it Works and Benefits of Using Image Recognition. https://blog.transtrack.co/en/technology/image-recognition/.
4. AI Image Recognition Applications and Business Benefits MindTitan. https://mindtitan.com/resources/blog/ai-image-recognition-applications-and-benefits/.
5. The Role of Machine Learning in Image Recognition and Natural Language https://blog.emb.global/machine-learning-in-image-recognition-and-nlp/.