Министерство науки и высшего образования Российской Федерации

ФГБОУ ВО «АЛТАЙСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Институт цифровых технологий, электроники и физики

Кафедра вычислительной техники и электроники (ВТиЭ)

**Отчёт по производственной эксплуатационной практике**

Выполнил студент 5.306М гр.:

\_\_\_\_\_\_\_\_\_\_\_\_\_ Лаптев А.В.

Проверил: проф. д.т.н. каф. ВТиЭ

\_\_\_\_\_\_\_\_\_ Белозерских В.В.

  Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

  «\_\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_2024г.

Барнаул 2024

**Содержание**

**[1. ВВЕДЕНИЕ 3](#_Toc2443)**

**[1.1.  Общие цели производственной эксплуатационной практики 3](#_Toc3404)**

**[1.2. Постановка задачи производственной эксплуатационной практики 3](#_Toc21884)**

**[2. ОБЩИЕ СВЕДЕНИЯ 4](#_Toc31813)**

**[2.1. Постановка задачи 4](#_Toc23761)**

**[2.2. Описание выполненных работ 4](#_Toc931)**

**[3. ЗАКЛЮЧЕНИЕ 8](#_Toc8246)**

**[ПРИЛОЖЕНИЕ 9](#_Toc6165)**

1. **ВВЕДЕНИЕ**
   1. **Общие цели производственной эксплуатационной практики**

Целью проведения практики является опознавание образов на изображениях (классификация) с помощью нейронной сети.

* 1. **Постановка задачи производственной эксплуатационной практики**

В ходе выполнения практики необходимо решить следующие задачи:

* сбор изображений для двух датасетов (по 100 изображений в каждом), каждый датасет состоит из изображений с утками, свиньями и овцами (баранами);
* классификация изображений из датасетов с использованием модели от Google;
* классификация изображений из датасетов с использованием модели YOLO;
* анализ результатов классификации, оценка ошибок первого и второго рода;
* составление отчета по выполненному заданию.

**Время прохождения практики:** 24.06.–5.07.2022.

1. **ОБЩИЕ СВЕДЕНИЯ**

Руководителям практики являлся Белозерских Василий Вениаминович.

* 1. **Постановка задачи**

Собрать два датасета (тренировочный и тестовый) с изображениями животых: утки, свиньи, овцы (бараны), по100 изображений в каждом.

Реализовать классификацию изображений при помощи нейронной сети от Google, проанализировать качество разделения классов и оценить ошибки первого и второго рода для модели.

Реализовать классификацию изображений при помощи YOLO, проанализировать качество разделения классов, оценить ошибки первого и второго рода.

После выполнения задачи классификации сравнить оба подхода между собой.

* 1. **Описание выполненных работ**

В первую очередь для решения поставленных задач нужно было создать два датасета с изображениями указанных животных: утки, свиньи, овцы (бараны). В обоих датасетах должно быть по 100 изображений с примерно одинаковым количеством примеров для каждого класса. Один из датасетов является тренировочным, а второй тестовым. В результате составления датасетов были получены следующие разбиения по классам (для обоих датасетов): по 33 изображения для уток и свиней и 34 изображения для овец (баранов).

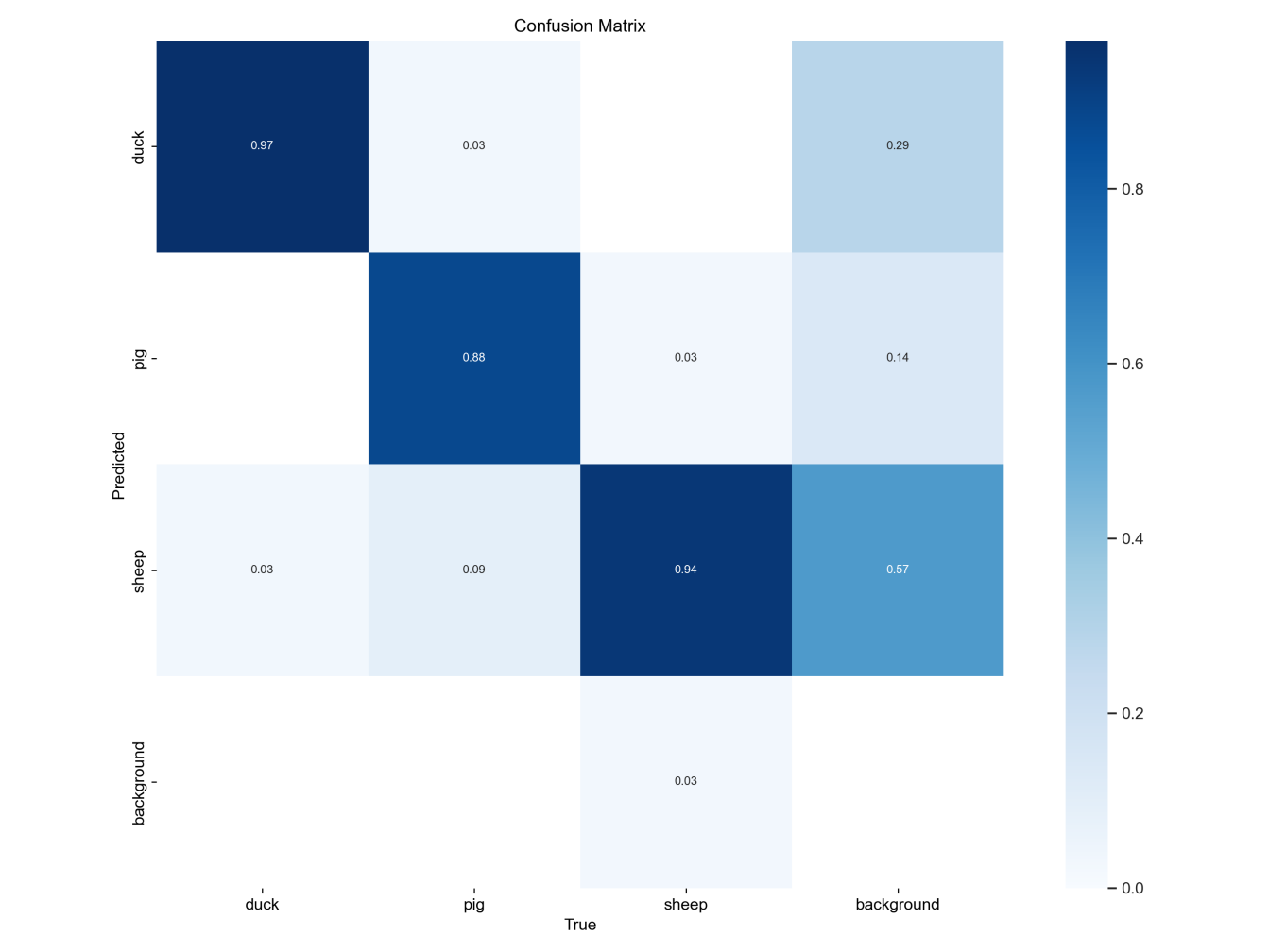
После составления датасетов необходимо было определиться с моделями нейронных сетей, на которых будет осуществляться решение задачи классификации.

В качестве нейронной сети от Google была выбрана TensorFlow в которой для обучения была выбрана модель MobileNetV2.

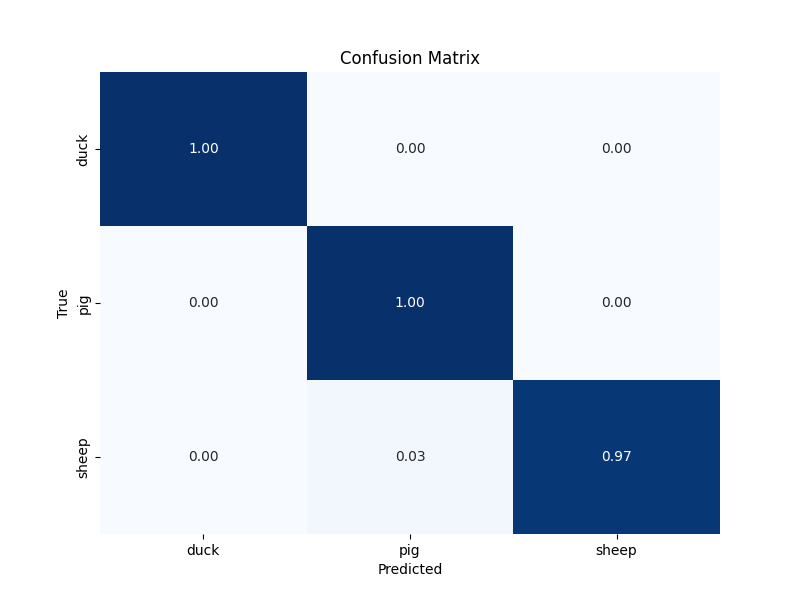
Для YOLO была выбрана модель YOLOv5, поскольку она оказалась наиболее проста в настройке.

Для того, чтобы классификация была в максимально равных условиях все параметры обучения были выбраны одинаковыми: размеры изображений на входе, количество эпох, batch size.

В ходе обучения обе модели показали высокие результаты в разделении классов. Ниже представлены визуализированные матрицы ошибок для каждой модели.



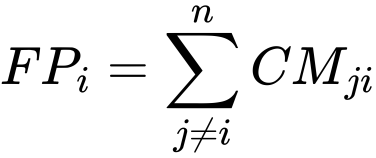
*Рис. 1 Матрица ошибок для YOLOv5.*



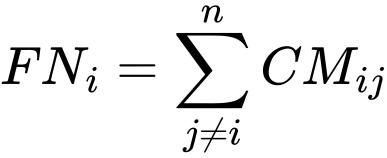
*Рис. 2 Матрица ошибок для TensorFlow.*

На основе матриц ошибок можно рассчитать ошибки первого и второго рода. Ошибки первого рода (False Positives) происходят, когда модель ошибочно классифицирует объект как принадлежащий к определенному классу, когда на самом деле он к этому классу не принадлежит. Ошибки второго рода (False Negatives) происходят, когда модель ошибочно не распознает объект как принадлежащий к определенному классу, хотя на самом деле он принадлежит к этому классу.

Расчет ошибок осуществляется по следующим формулам:

, где

*FPi －* количество ложноположительных ошибок для класса *i, CMji －* элемент матрицы ошибок, где *j －* истинный класс, *i* *－* предсказанный класс.

, где

*FNi* *－* количество ложноотрицательных ошибок для класса *i*, *CMij* *－* элемент матрицы ошибок, где *i* *－* истинный класс, *j* *－* предсказанный класс.

Для YOLOv5 были получены следующие значения ошибок на основе матрицы ошибок:

* ошибка первого рода (False Positives): утки － 0,03, свиньи － 0,03, овцы － 0,12;
* ошибка второго рода (False Negatives): утки － 0,03, свиньи － 0,12, овцы － 0,03.

Для TensorFlow были получены следующие значения ошибок на основе матрицы ошибок:

* ошибка первого рода (False Positives): утки － 0,00, свиньи － 0,03, овцы － 0,00;
* ошибка второго рода (False Negatives): утки － 0,00, свиньи － 0,00, овцы － 0,03.

В ходе обучения двух нейронных сетей в приблизительно одинаковых условиях видно, что задача классификации точнее решена для TensorFlow. Эта модель показывает более высокую точность для данных датасетов. В то же время у YOLO есть преимущество в скорости обучения. При достаточно высокой точности разделения на классы скорость обучения данной модели была примерно в 2 раза выше.

При этом, несмотря на то, что ошибки классификации для YOLO выше, они не являются критичными, а результаты являются удовлетворительными для того, чтобы задача классификации все равно решалась успешно. Точность классификации может быть повышена при использовании большего количества элементов разных классов в датасете, разнообразии элементов классов, повышении количества эпох для обучения и другими распространенными способами для повышения точности классификатора.

1. **ЗАКЛЮЧЕНИЕ**

В ходе производственной эксплуатационной практики была решена задача классификации для двух моделей нейронных сетей (YOLO, TensorFlow), проанализированы результаты, оценена ошибка первого и второго рода для обоих моделей.

Для достижения поставленной цели надо было решить следующие задачи:

* сбор изображений для двух датасетов (по 100 изображений в каждом), каждый датасет состоит из изображений с утками, свиньями и овцами (баранами);
* классификация изображений из датасетов с использованием модели от Google;
* классификация изображений из датасетов с использованием модели YOLO;
* анализ результатов классификации, оценка ошибок первого и второго рода.

В ходе практики, поставленные передо мной задачи были выполнены в срок и в полном объеме.

**ПРИЛОЖЕНИЕ**

Программный код для обучения модели TensorFlow:

import numpy as np

from sklearn.metrics import confusion\_matrix, classification\_report

from keras.\_tf\_keras.keras.preprocessing.image import ImageDataGenerator

from keras.\_tf\_keras.keras.applications import MobileNetV2

from keras.\_tf\_keras.keras.layers import Dense, GlobalAveragePooling2D

from keras.\_tf\_keras.keras.models import Model

def evaluate\_model(model, test\_generator):

    '''Функция для классификации моделей'''

    predictions = model.predict(test\_generator)

    predicted\_classes = np.argmax(predictions, axis=1)

    true\_classes = test\_generator.classes

    class\_labels = list(test\_generator.class\_indices.keys())

    return predicted\_classes, true\_classes, class\_labels

if \_\_name\_\_ == '\_\_main\_\_':

    # Загрузка предобученной модели

    base\_model = MobileNetV2(weights='imagenet', include\_top=False, input\_shape=(224, 224, 3))

    x = base\_model.output

    x = GlobalAveragePooling2D()(x)

    x = Dense(1024, activation='relu')(x)

    predictions = Dense(3, activation='softmax')(x)

    model = Model(inputs=base\_model.input, outputs=predictions)

    for layer in base\_model.layers:

        layer.trainable = False

    model.compile(optimizer='adam', loss='categorical\_crossentropy', metrics=['accuracy'])

    # Подготовка данных

    train\_datagen = ImageDataGenerator(rescale=1./255, horizontal\_flip=True, zoom\_range=0.2)

    test\_datagen = ImageDataGenerator(rescale=1./255)

    train\_generator = train\_datagen.flow\_from\_directory(

        'C:/Users/Lapte/Documents/university\_projects/1\_course\_master/Practice/datasets/train\_dataset',

        target\_size=(224, 224),

        batch\_size=32,

        class\_mode='categorical'

    )

    test\_generator = test\_datagen.flow\_from\_directory(

        'C:/Users/Lapte/Documents/university\_projects/1\_course\_master/Practice/datasets/test\_dataset',

        target\_size=(224, 224),

        batch\_size=32,

        class\_mode='categorical',

        shuffle=False

    )

    # Обучение модели

    model.fit(train\_generator, epochs=100, validation\_data=test\_generator)

    # Проведение классификации

    predicted\_classes, true\_classes, class\_labels = evaluate\_model(model, test\_generator)

    cm = confusion\_matrix(true\_classes, predicted\_classes)

    report = classification\_report(true\_classes, predicted\_classes, target\_names=class\_labels)

    # Вычисление ошибок первого и второго рода

    fp = cm.sum(axis=0) - np.diag(cm)  # False Positives

    fn = cm.sum(axis=1) - np.diag(cm)  # False Negatives

    print(f"Confusion Matrix:\n{cm}\nClassification Report:\n{report}\n")

    print(f"False Positives (Type I errors) per class: {fp}\nFalse Negatives (Type II errors) per class: {fn}")

Конфигурационный файл для YOLOv5:

train: C:/Users/Lapte/Documents/university\_projects/1\_course\_master/Practice/datasets/images/train

val: C:/Users/Lapte/Documents/university\_projects/1\_course\_master/Practice/datasets/images/val

nc: 3  # количество классов

names: ['duck', 'pig', 'sheep']  # имена классов