

# **Инструкция**

## **по применению материалов для выполнения лабораторных (практических) работ**

### **1. ОБЩИЕ СВЕДЕНИЯ**

Разработанная программа распознавания подписей – «Система распознавания подписи» (SRS – Signature Recognition System) – предназначена для идентификации владельца рукописной подписи по сканированному образу.

Программа написана на языке Python с использованием фреймворка: FEDOT (платформа с открытым исходным кодом для решения задач автоматического моделирования и машинного обучения) и библиотеки OpenCV (открытый код с алгоритмами общего назначения, компьютерного зрения и обработки изображений).

Репозиторий с кодом программы можно изучить по адресу: <https://github.com/ElishaFlacon/signess> . Для его использования необходим компилятор Visual Studio Code или PyCharm, датасет со сканированными образами подписей и сводной таблицей данных, содержащей ФИО и индекс владельца подписи в лексикографическом порядке.

### **2. НАЗНАЧЕНИЕ И УСЛОВИЯ ПРИМЕНЕНИЯ**

#### **2.1 Назначение программного компонента**

Задача программы - распознать рукописную подпись на сканированном изображении и идентифицировать ее владельца.

Для успешного решения поставленной задачи в общем случае выполняются следующие операции:

1. Создание датасета.
2. Определение области печати с подписью.
3. Выделение признаков подписи человека на документе, скрытой информационным шумом (печатью);
4. Подготовка изображений:
  - a. обрезка изображения по контуру;
  - b. изменение размеров изображения;
  - c. переворачивание изображения;
5. Удаление цифрового шума:
  - a. удаление текста;
  - b. удаление печати;

6. Классификация подписи.
7. Идентификация подписи по имеющейся базе данных с другими подписями людей.

## **2.2 Область применения**

Данное решение позволяет вести работу с различными документами и деловыми бумагами. Оно применимо в сфере реализации государственных услуг, юриспруденции, в учреждениях медицинского и образовательного направления, в бизнесе и других сферах, в которых необходимо обрабатывать большие объёмы документации (для проверки принадлежности подписи на документе заявленному лицу).

Разработанная программа – это универсальная система для проверки принадлежности подписи на документе заявленному лицу, которую можно будет приспособлять под конкретные нужды заказчика. Потенциальными клиентами, которые непосредственно будут работать с программой, могут являться сотрудники из различных сфер профессиональной деятельности: частный бизнес – секретарь, делопроизводитель, финансист, специалист отдела кадров, офис-менеджер; государственное учреждение – бухгалтер, администратор; нотариальная контора – нотариусы и т.д.

## **2.3 Функциональные условия применения**

Для корректной работы SRS следует учитывать важные ограничения на применение:

- на изображении должна быть чётко видна только одна подпись, которую пользователь хочет идентифицировать;
- изображение должно быть в формате png, jpg, bmp или pdf.
- изображение не должно быть черно-белым.

## **2.4 Технические условия применения**

Вычислительная машина должна использовать ОС Windows или Linux со следующими характеристиками:

- процессор: Intel\AMD-совместимый, не менее 2\4 ядер, не менее 2.5\3 ГГц, соответственно;
- видеокарта: NVIDIA, не старше 7 поколения (GeForce GTX 750 Ti), для AMD не старше Rx 300 (R9 380x);
- оперативная память: не менее 8 Гб;
- дисковая подсистема: не менее 250 Гб;
- пропускная способность сетевых интерфейсов: не менее 1 Гбит/с.

Для сканирования анализируемого файла обязательно иметь сканер, подключенный к устройству посредством кабеля или беспроводной сети.

### **3. ОПИСАНИЕ ПРИКЛАДНЫХ ЗАДАЧ**

#### **3.1 Классы решаемых задач**

Решаемая задача системы – идентифицировать распознанную подпись на сканированном документе, скрытой круглой печатью.

#### **3.2 Примеры решения задач**

1. Задача – идентифицировать владельца подписи на документе.
2. Исходные данные – файл типа png, jpg, bmp или pdf с подписью, требуемой распознать.
3. Решение задачи – конвертация файла в изображение (если тип не является подходящим для работы программы), подготовка подписи к распознаванию (удаление текста, печати, разрывов на подписи, обрезка изображения по контурам подписи), получение подписи из документа, интерпретация выходных данных.
4. Результат – индекс владельца подписи из таблицы базы данных.

### **4. ОБРАЩЕНИЕ К ПРОГРАММЕ**

#### **4.1 Точки входа в программу**

Точками входа в программу являются начала модулей, запускаемых в ходе выполнения программы:

- инициация класса Dataset;
- инициация класса Autograph;
- инициация класса Document;
- инициация класса FedotCNN.

#### **4.2 Базовые функции**

На рисунке 4.2.2 представлен основной модуль системы, отвечающий за сборку датасета и вызов методы классификации изображения. Для этого подключаем библиотеки `signess` и `inskrub`, более полные названия приведены ниже на рисунке 4.2.1. Делаем инициализацию переменных, хранящих в себе путь к папке с подписями и печатью одного человека, к датасету, к файлу csv с остальными подписями и к входному изображению в формате png. Сначала собирается датасет с

помощью класса Dataset, в который передаются нужные параметры. После этого создаётся не-обученная нейросеть при помощи метода FedotCNN. Остаётся только собрать пайплайн обучения модели и вызвать метод классификации изображения.

```
from signess.dataset import Dataset
from signess.network import FedotCNN

from inskrib.autograph import Autograph
from inskrib.documents import Document
```

Рисунок 4.2.1 – Подключаемые библиотеки

```
if __name__ == '__main__':
    set_random_seed(1)

    # config
    path_to_data = './example/docs'
    path_to_picture = './result/autographs/0-first_person-0.png'
    path_to_save_and_load = './model'

    path_to_dataset = create_dataset(path_to_data)
    create_network(path_to_dataset, path_to_picture, path_to_save_and_load)
    load_network(path_to_dataset, path_to_picture, path_to_save_and_load)
```

Рисунок 4.2.2 – Конфигурация и вызов главной функции программы в основном модуле

### 4.3 Пайплайны на основе функций компонента.

На рисунке 4.3.1 представлен пайплайн системы.

```
# create Network
network = FedotCNN()
# load dataset where path - path to .npz file
dataset = network.load_dataset(path=path_to_dataset)
# train model with loaded dataset by 3 epochs
network.train(dataset=dataset, num_epochs=3)

# predicts dataset
predicts = network.predict(dataset)
print(f'Predicts: {predicts}')

# classify picture
predict = network.classify(path_to_picture, path_to_dataset)
```

Рисунок 4.3.1 – Пайплайн обучения модели

## 5. УСТАНОВКИ И ПРОВЕРКА ПРОГРАММЫ

### 5.1 Установка и настройка программы

Программа представляет собой библиотеку для языка программирования Python. Ее можно установить при помощи пакетного менеджера pip командой `pip install signess`.

Для тестового запуска необходимо скопировать содержимое директории из репозитория библиотеки (<https://github.com/ElishaFlacon/signess>), предварительно установив саму библиотеку и запустить файл `example.py`. В нем находится пример кода для запуска программы.

### 5.2 Модульные и интеграционные тесты

В директории предусмотрены тестовые выборки различных подписей, на которых можно проверить работу библиотеки.

После скачивания библиотеки и установки необходимых файлов директории можно будет провести тестирование.

### 5.3 Контрольные примеры

В качестве тестового изображения используется документ, содержащий пример подписи. В результате выполнения программы был получен процент точности распознавания и результат идентификации подписи.

## 6. ВХОДНЫЕ И ВЫХОДНЫЕ ДАННЫЕ

### 6.1 Входные данные

В таблице 6.1.1 приведены состав и структура входных данных.

Таблица 6.1.1

Состав и структура входных данных

Наименование данных	Обозначение	Структура данных	Способ ввода данных	Ограничения
Исходный файл с подписью и печатью	self	Документ в классе AutographClass	Загрузка документов с подписью	Документ не чёрно-белый; тип файла pdf, jpg, png или bmp
Датасет с подписями	dataset	таблица csv с изображениями подписей в классе Data	Загрузка изображений с подписью	csv файл
Объект MatLike из OpenCV	picture	Изображение MatLike в классе AutographClass	Загрузка изображений с подписью	

Входные данные должны соответствовать следующим условиям:

- документ с подписью отсканирован;
- отсканированное изображение имеет тип png, jpg, bmp или pdf.
- на отсканированном изображении находится только одна единственная подпись.

Исходный файл с подписью загружается из локальной папки.

## 6.2 Выходные данные

В таблице 6.2.1 приведены состав и структура выходных данных.

Таблица 6.2.1

Состав и структура выходных данных

Наименование данных	Обозначение	Структура данных	Способ вывода данных	Ограничения
Результат точности распознавания подписи нейронной сетью	predict	Число с плавающей точкой	Вызов метода predict в программе	От 0 до 1 включительно
Результат точности распознавания подписи нейронной сетью	accuracy	Число с плавающей точкой	Вызов метода accuracy в программе	От 0 до 1 включительно
Индекс владельца подписи в таблице базы данных	index	Целое число	Вызов метода classify_picture	Не меньше 0

Результатом работы программы является – **index** – индекс владельца подписи в таблице базы данных при её наличии (рисунок 6.2.1). Выходные данные выводятся на консоль компьютера.

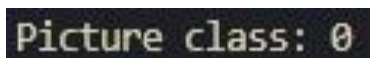


Рисунок 6.2.1 – Результат идентификации подписи

## 7. СООБЩЕНИЯ

При запуске программы для удобства пользователя будет выведен интерфейс (см рисунок

7.1

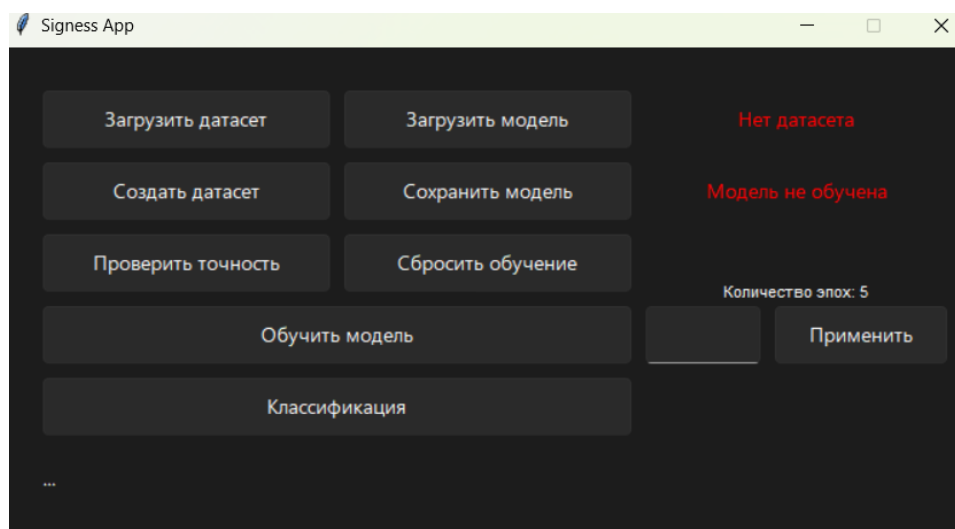


Рисунок 7.1 – Меню приложения

После выбора пользователем опции “Загрузить датасет”, а также выбора пути к датасету, ему выведется сообщение, показанное на рисунке 7.2.

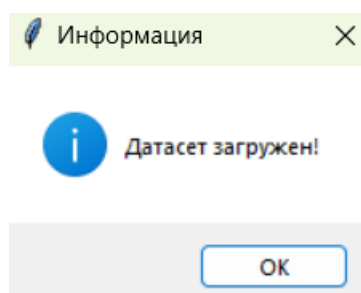


Рисунок 7.2 – Информация о загрузке датасета

Если на этапе выбора пути пользователь отменит операцию, то ему будет выведено сообщение, показанное на рисунке 7.3.

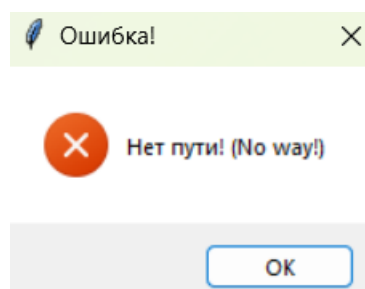


Рисунок 7.3 – Ошибка выбора пути

Если пользователь выберет опцию “Создать датасет”, и укажет путь к папке с изображениями, то программа оповестит его о начале работы (см. рисунок 7.4), а после о ее завершении (см. рисунок 7.5).

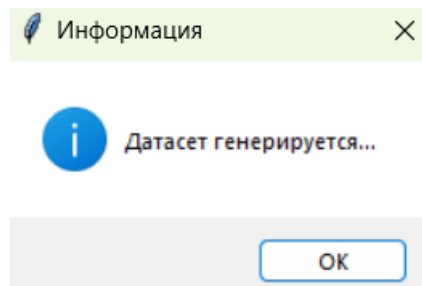


Рисунок 7.4 – Начало генерации датасета

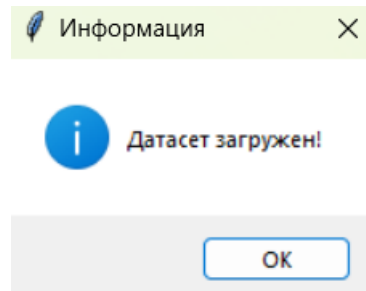


Рисунок 7.5 – Окончание генерации датасета

О том, что датасет загружен пользователь так же может узнать, посмотрев на главное меню (см. рисунок 7.6).

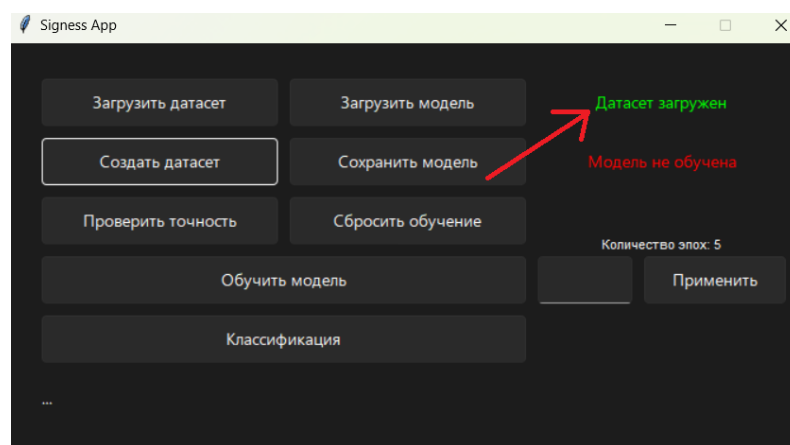


Рисунок 7.6 – Указатель готовности датасета

После выбора опции “Обучить модель”, приложение оповестит о том, что модель обучается (см. рисунок 7.7).



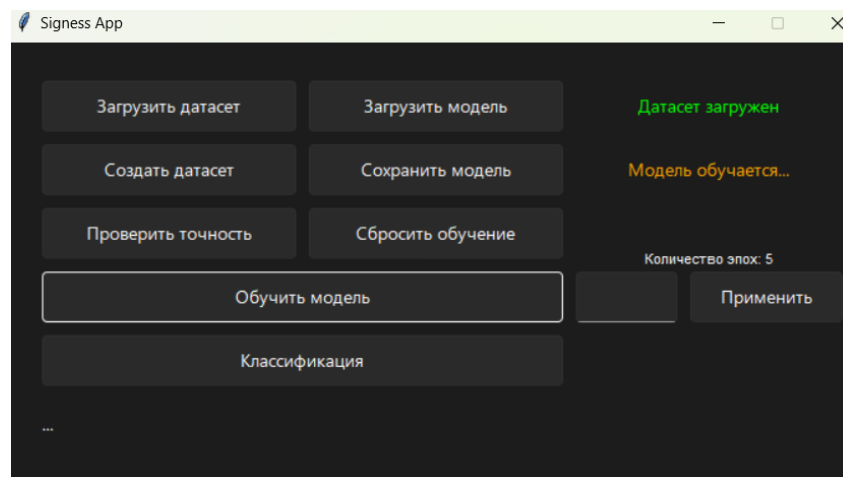


Рисунок 7.7 – Процесс обучения модели

И после обучения модели программа сообщит пользователю об этом (см. рисунок 7.8).

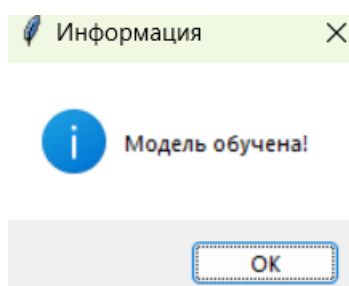


Рисунок 7.8 – Информация об обучении модели

Так же о том, что модель обучена можно узнать в главном меню (см. рисунок 7.9).

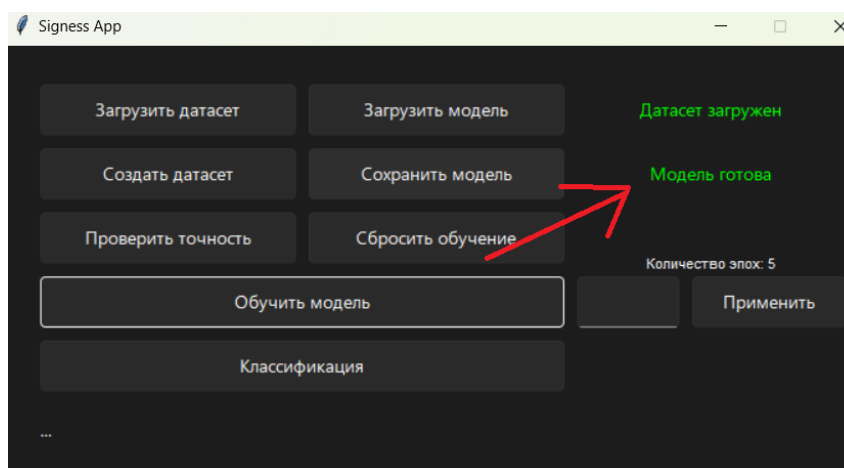


Рисунок 7.9 – Указатель готовности модели

При выборе опции “Проверить точность” необходимо будет указать путь к датасету, после чего программа проверит на нем точность обученной модели, и сообщит об итогах пользователю (см. рисунок 7.10).

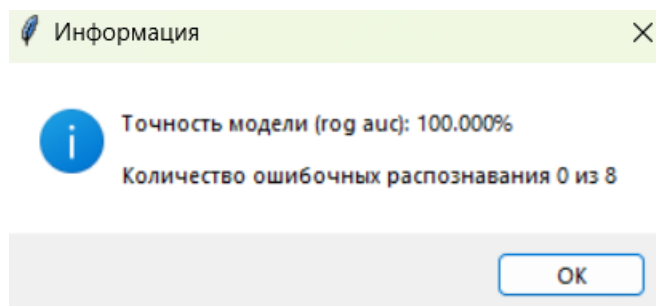


Рисунок 7.10 – Информация о точности

Опция “Сбросить обучение” удалит обучение модели. При завершении пользователь получит сообщение, показанное на рисунке 7.11.

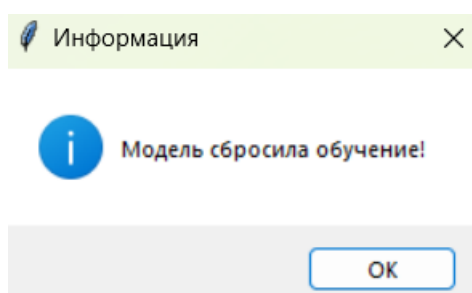


Рисунок 7.11 – Информация о сбросе обучения

Опция “Сохранить модель” создаст, в выбранной пользователем папке, папку с моделью, которая сейчас готова к использованию.

После чего подобные папке можно использовать в опции “Загрузить модель”, которая позволит использовать выбранную модель в приложении. Программа сообщит пользователю необходимые требования к модели, загружаемой таким образом (см. рисунок 7.12).

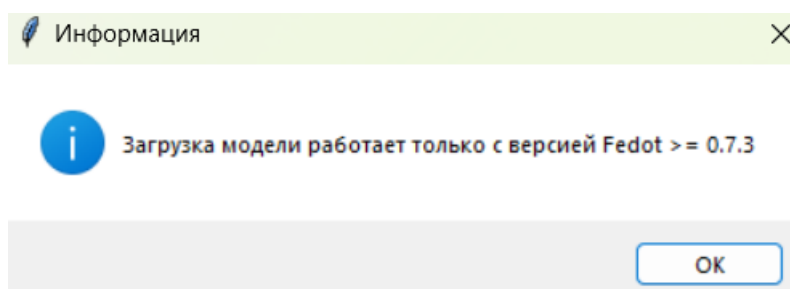


Рисунок 7.12 – Информация о необходимой версии FEDOT

Опция “Классификация” покажет принадлежность к классам, заданным в обученной модели, выбранного пользователем файла (см. рисунок 7.13).

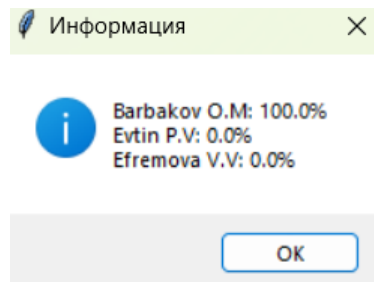


Рисунок 7.13 – Классификация файла

После этого в главном меню можно будет увидеть класс файла (см. рисунок 7.14).

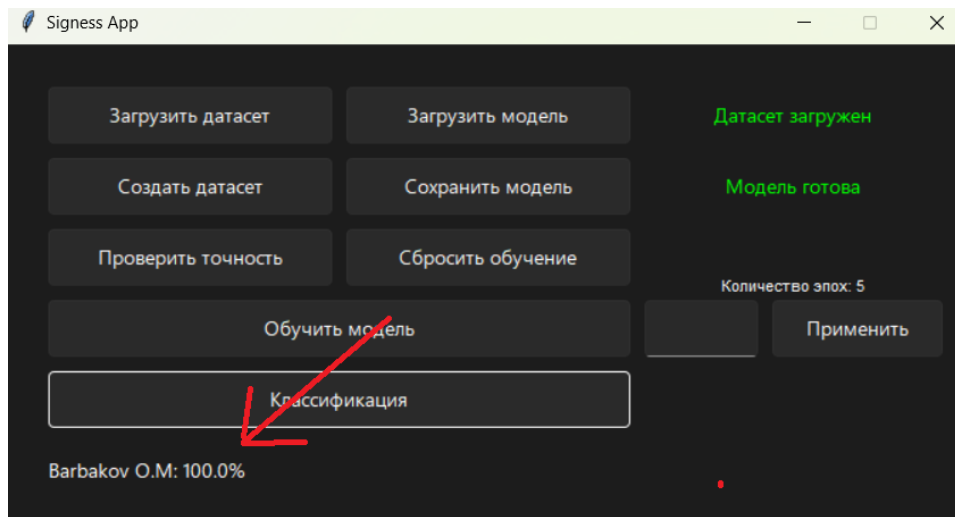


Рисунок 7.14 – Класс файла в главном меню

Если файл будет выбран некорректно, программа оповестит об этом пользователя. (см. рисунок 7.15).

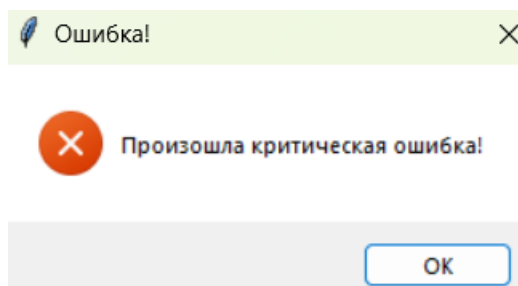


Рисунок 7.15 – Ошибка выбора файла

## **Перечень лабораторных/практических работ**

### **Лабораторная/практическая работа №1**

#### **Тема «Подготовка данных»**

##### **Задание:**

- 1) Собрать датасет, включающий не менее 100 изображений документов с подписью и печатью. Использовать только открытые источники данных.
- 2) Создать сводную таблицу описания датасета: ФИО владельца подписи и индекс. Упорядочить в лексикографическом порядке.

##### **Ограничения:**

- на изображении должна быть чётко видна только одна подпись, которую пользователь хочет идентифицировать;
- изображение должно быть в формате png, jpg, bmp или pdf.
- изображение должно быть в цветовой матрице RGB, т.е. не должно быть черно-белым.

### **Лабораторная/практическая работа №2**

#### **Тема «Обучение модели»**

##### **Задание:**

- 1) Запустить программу Signess (<https://github.com/ElishaFlacon/signess>).
- 2) Подготовить тестовую и обучающую выборки из собранного ранее датасета.
- 3) Обучить модель на созданной ранее обучающей выборке.
- 4) Идентифицировать владельцев подписи на тестовой выборке.
- 5) Проанализировать полученные результаты

##### **Ограничения:**

- Модель не должна обучаться дольше 6 часов.
- Точность идентификации должна быть не ниже 80%.
- Изображение должно быть в цветовой матрице RGB, т.е. не должно быть черно-белым.

### **Лабораторная/практическая работа №3**

#### **Тема «Точность идентификации»**

##### **Задание:**

- 1) Увеличить датасет за счёт расширения классов идентификации.
- 2) Подготовить тестовую и обучающую выборки из собранного ранее датасета.
- 3) Запустить программу Signess (<https://github.com/ElishaFlacon/signess>).

- 4) Обучить модель на созданной ранее обучающей выборке.
- 5) Идентифицировать владельцев подписи на тестовой выборке.
- 6) Проанализировать полученные результаты

**Ограничения:**

- Модель не должна обучаться дольше 6 часов.
- Точность идентификации должна быть не ниже 80%.
- Изображение должно быть в цветовой матрице RGB, т.е. не должно быть черно-белым.

## **Лабораторная/практическая работа №4**

### **Тема «Чувствительность модели»**

**Задание:** проверить чувствительность модели на двух датасетах, содержащих документы хорошего качества и эти же документы в ухудшенном качестве.

- 1) Подготовить два датасета: в первом датасете качество изображений хорошее, во втором датасете качество этих же изображений ухудшенное (внести размытость, разрывы печати, удаление областей и т.п.)
- 2) Запустить программу Signess (<https://github.com/ElishaFlacon/signess>).
- 3) Идентифицировать владельцев подписи на тестовых выборках.
- 4) Проанализировать полученные результаты.
- 5) Последовательно проводить эксперимент, ухудшая качество входных изображений. Сделать выводы о чувствительности модели. Выразить в процентах (%) зависимость чувствительности модели от качества входного изображения.

**Ограничения:**

- Модель не должна обучаться дольше 6 часов.
- Точность идентификации должна быть не ниже 80%.
- Изображение должно быть в цветовой матрице RGB, т.е. не должно быть черно-белым.

*Пример фрагмента выполнения работы:*

**Задача:** Проверить чувствительность модели на двух датасетах, содержащих документы хорошего качества и эти же документы в ухудшенном качестве.

**Ожидаемый результат:** существенное уменьшение точности распознавания.

**Состав датасетов:**

- а) датасет 1 - 8 файлов формата png, хорошего качества, количество классов - 3;
- б) датасет 2 - 8 файлов формата png, ухудшенного качества, количество классов - 3.

**Результат:**

а) точность датасета1 - 100% (см. рисунок 1)

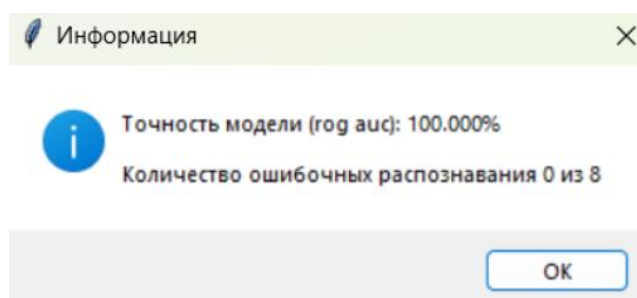


Рисунок 1 - Точность исходного датасета

б) точность датасета2 - 74% (см. рисунок 2)

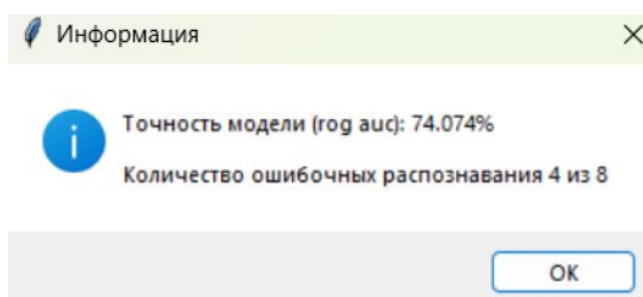


Рисунок 2 - Точность ухудшенного датасета

На рисунке 3 показаны «выпады», которые выделила модель:



Рисунок 3 - «Выпады» при обучении модели

## Лабораторная/практическая работа №5

### Тема «Время обучения модели»

**Задание:** определить зависимости времени обучения модели от количества классов и эпох.

- 1) Подготовить тестовую и обучающую выборки.
- 2) Запустить программу Signess (<https://github.com/ElishaFlacon/signess>).

- 3) Обучить модель с фиксацией времени обучения и количества эпох: увеличиваем эпохи, не меняя количество классов (3 эксперимента); увеличиваем количество классов не меняя количество эпох (3 эксперимента); увеличиваем и количество классов и количество эпох (3 эксперимента).
- 4) Идентифицировать владельцев подписи на тестовой выборке
- 5) Свести полученные результаты в таблицу:

Мощность тестовой выборки	Количество классов	Количество эпох	Время обучения
Увеличиваем эпохи, не меняя количество классов			
...			
...			
...			
Увеличиваем количество классов не меняя количество эпох			
...			
...			
...			
Увеличиваем и количество классов и количество эпох			
...			
...			
...			
			Среднее значение

- 6) Описать зависимости и сделать выводы

**Ограничения:**

- Модель не должна обучаться дольше 6 часов.
- Точность идентификации должна быть не ниже 80%.
- Изображение должно быть в цветовой матрице RGB, т.е. не должно быть черно-белым.

*Пример фрагмента выполнения работы:*

Задача тестирования: Определить среднее время обучения модели.

Ожидаемый результат: обучение модели на датасете из 1946 изображений и из 195 изображений будет длиться не более 6 часов.

а) Состав датасета: 1946 файлов разного формата, неодинакового качества, количество классов – 3.

Количество эпох обучения модели: 5.

На рисунке 4 представлена точность модели, обученной на датасете из 1946 изображений

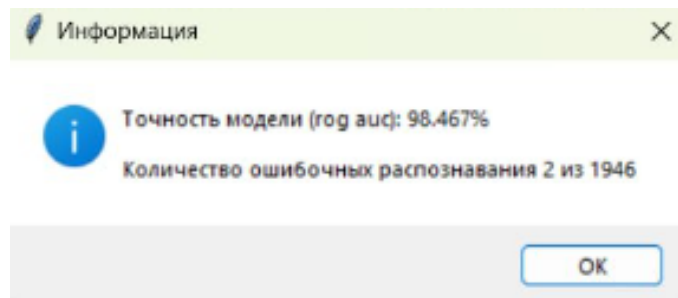


Рисунок 4 - Точность модели обученной на датасете из 1946 изображениях

Результат:

Время обучения модели: 19 минут.

б) Состав датасета: 195 файлов разного формата, неодинакового качества, количество классов – 3.

Количество эпох обучения модели: 5.

Точность модели в этом датасете можно увидеть на рисунке 5

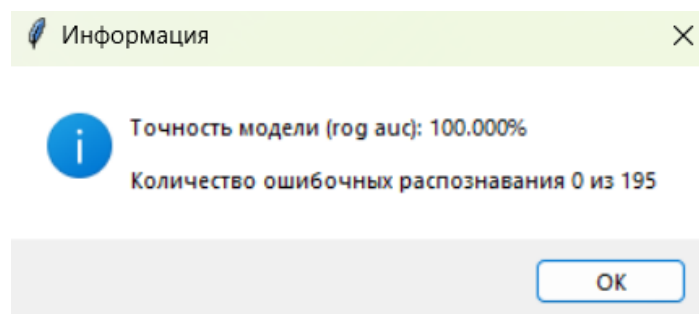


Рисунок 5 - Точность модели обученной на датасете из 195 изображениях

Результат:

Время обучения модели: 2 минуты.

Вывод:

Обучение модели на датасете из 1946 изображений заняло 19 минут при 5 эпохах.

При том же количестве эпох модель обучилась на датасете из 195 изображений за 2 минуты. Из этого можно сделать вывод, что модель обучается минуту за каждые 100 изображений в датасете.

В ходе выполнения теста было выявлено, что ограничение при загрузке датасета/обучении нейросети: для 600 файлов; необходимо выделять 1Gb оперативной памяти (при текущей настройке в config.py);