Описание классов и методов

1. ОПИСАНИЕ ЛОГИЧЕСКОЙ СТРУКТУРЫ

1.1 Используемые методы

Используемые алгоритмы решения поставленных задач являются в большей мере базовыми алгоритмами библиотек OpenCV и FEDOT.

Преобразование изображения, включая подготовку, устранение цифрового шума, а также конвертацию, полностью обеспечивают базовые функции OpenCV.

Основной моделью нейронной сети является глубокая сверточная нейронная сеть FedotCNN, позволяющая производить более глубокий анализ изображения. Реализация нейронной сети обеспечена базовым функционалом библиотеки FEDOT.

1.2 Алгоритм(ы) программы

На рисунке 1.2.1 представлена блок-схема алгоритма работы основного модуля.

Рисунок 1.2.1 – Алгоритм работы основного модуля

На рисунке 1.2.2 представлена блок-схема алгоритма работы модуля извлечения подписи с изображения.

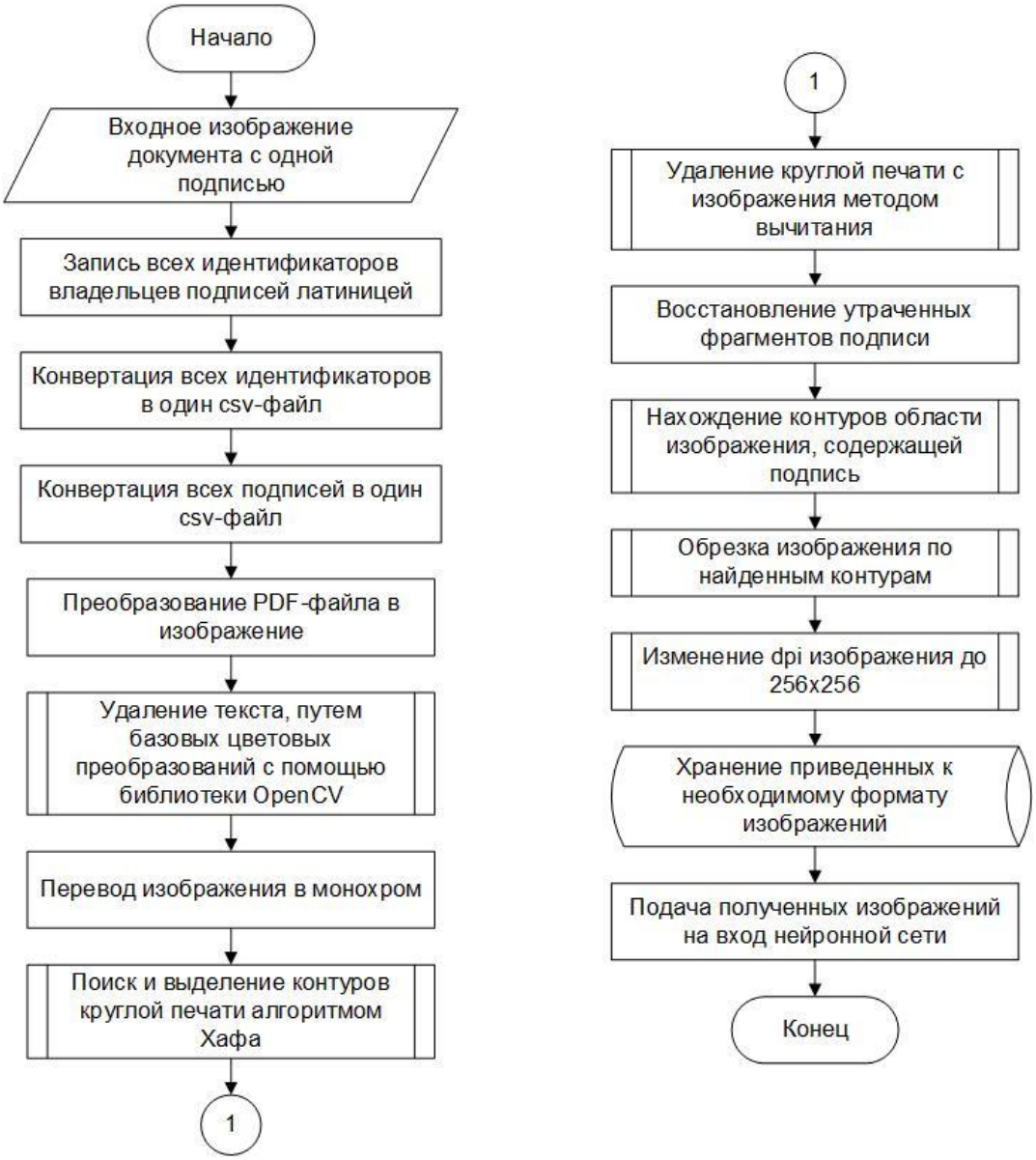

Рисунок 1.2.2 – Алгоритм работы модуля извлечения подписи с изображения

На рисунке 1.2.3 представлена блок-схема алгоритма работы модуля идентификации владельца подписи.

Рисунок 1.2.3 – Алгоритм работы модуля идентификации владельца подписи

1.3 Структура программы

Структура программы представляет собой диаграмму классов разрабатываемой программы (см. рисунок 1.3.1). Диаграмма состоит из 5 классов: PictureClass, StampClass, DateClass, AutographClass, ClassificationClass.

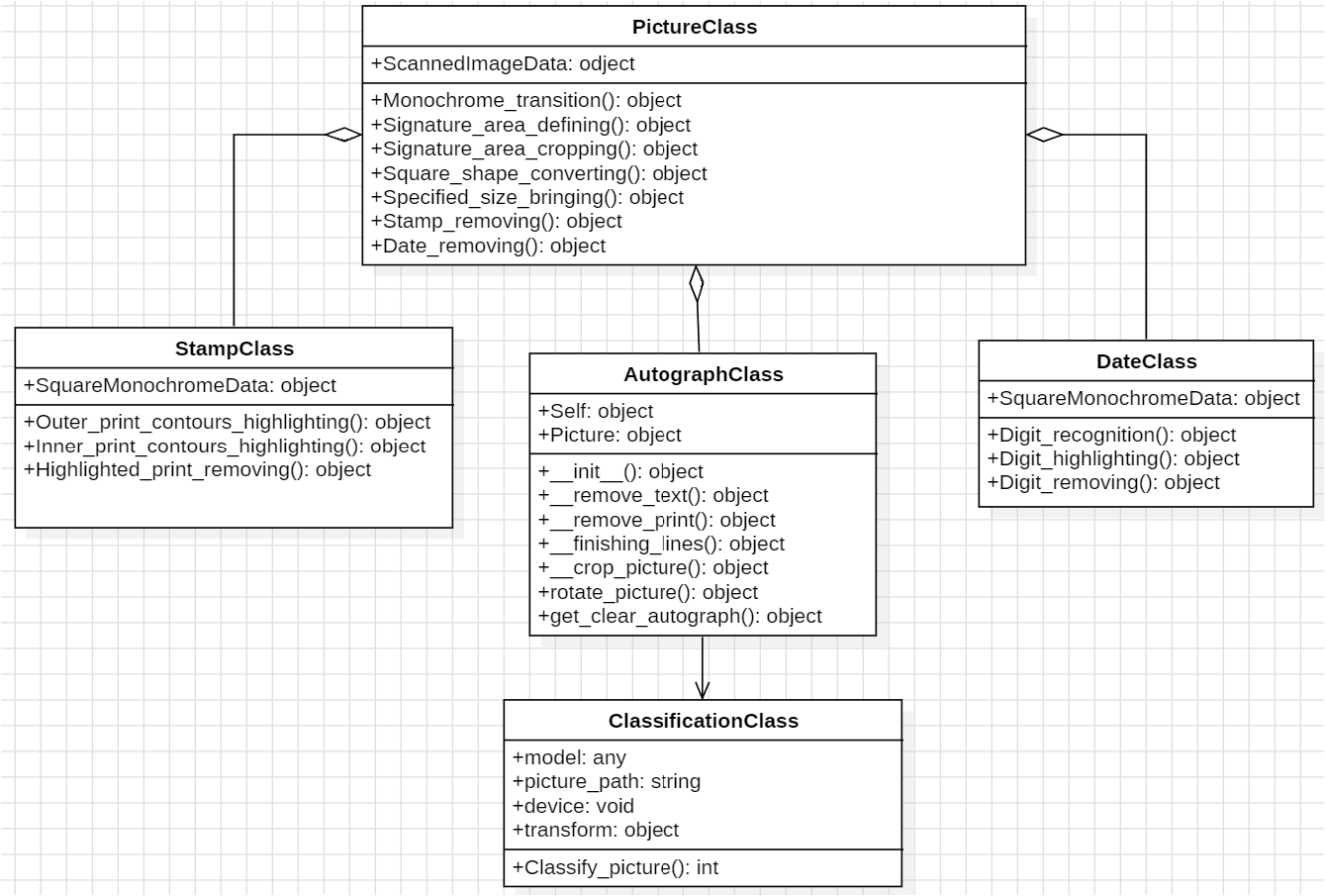

Рисунок 1.3.1 – Диаграмма классов SRS

Класс «PictureClass» имеет один атрибут – скан-образ документа с подписью и печатью (ScannedImageData) – а также несколько методов для работы с ним: переход в монохромный режим (Monochrom_transition), определение области с подписью(Signature_area_defining), обрезание области с подписью по контуру (Signature_area_cropping), преобразование обрезанного скан-образа в квадратную форму (Square_shape_converting), приведение к нужному разрешению (Specified_size_bringing), удаление печати с области с подписью (Stamp_removing) и удаление даты с области с подписью (Date_removing). Данный класс обрабатывает и подготавливает входное изображение для дальнейшей работы с ним и идентификации подписи. Данный класс состоит из трёх подклассов – классов «StampClass», «DateClass» и «AutographClass».

Класс «StampClass» хранит в себе атрибут – квадратный монохром (SquareMonochromeData), преобразованный от изначального входного изображения путём перевода в чёрно-белый градиент, обрезания области с печатью и подписью и перевода в квадратный формат размером NxN пикселей. Этот класс также осуществляет методы к атрибуту: выделение внешних контуров (outer_print_contours_highlighting), выделение внутренних контуров печати (inner_print_contours_highlighting), удаление выделенной печати (highlighted_print_removing). Все

методы нужны для выделения и удаления с подписи информационного шума в виде круглой печати.

Классу «DateClass» передаётся один атрибут – квадратный монохром (SquareMonochromeData), почти тот же, что и в классе «StampClass», но без печати. Используемые классом методы: распознавание цифр (Digit_recognition), выделение цифр (Digit_highlighting), удаление цифр (Digit_removing). Все методы применяются для выделения и удаления информационного шума в виде прописанной даты.

Класс «AutographClass» содержит два атрибута – исходный загруженный файл с подписью и печатью (Self) и объект MatLike из OpenCV (picture). Методы для работы с имеющимися данными состоят из: подготовки входного изображения – приведение к нужному размеру, выделение круглой печати – (__init__), удаления текста с изображения (__remove_text), удаления круглой печати (__remove_text), восстановления утраченных фрагментов подписи (__finishing_lines), обрезания изображения по контуру подписи(), задания правильной ориентации скан-образу (__crop_picture), получения обработанной подписи (get_clear_autograph).

Класс «ClassificationClass» имеет несколько атрибутов: обученная модель из FEDOT, путь к классифицируемому изображению (picture_path). Classify_picture() – метод, позволяющий получить идентификатор объекта, которому принадлежит подпись на сканированном образе. Идентификация происходит уже после подготовки, выделения подписи и её распознавания.

На рисунке 1.3.2 представлена диаграмма компонентов системы.

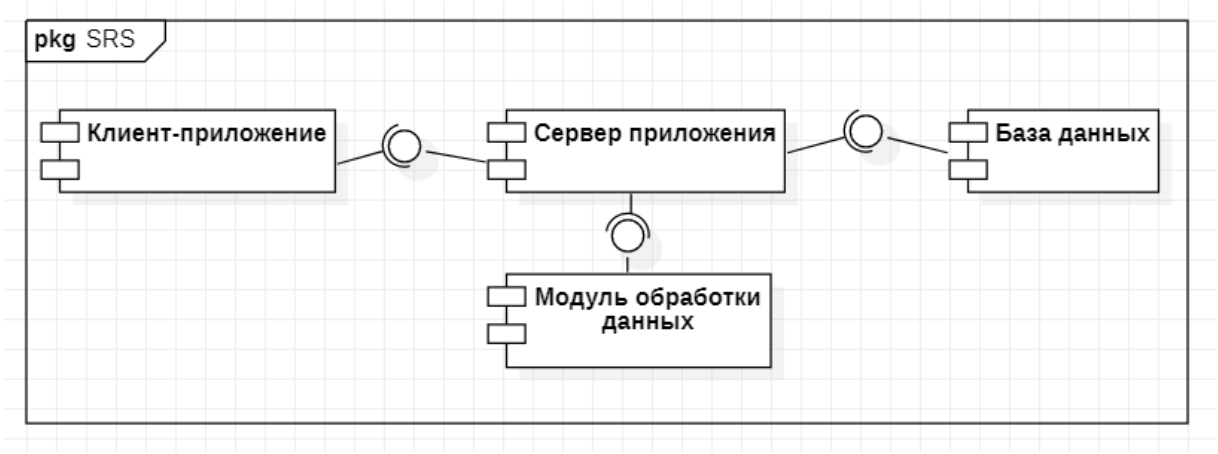

Рисунок 1.3.2 – Диаграмма компонентов

Система состоит из четырех основных компонентов:

- Клиент-приложение – блок, содержащий в себе интерфейс пользователя;
- Сервер приложения – блок, управляет доступом к данным и транзакциями;
- База данных – блок хранения преобразованных изображений;

Модуль обработки данных – основной блок системы, содержащий в себе набор функций и нейронных сетей, предназначенных для выполнения основной функции программы: распознавание скрытой подписи.

2. ПРИМЕРЫ ИСПОЛЬЗОВАНИЯ

2.1 Базовые функции программы

На рисунке 2.1.2 представлен основной модуль системы, отвечающий за сборку датасета и вызов методы классификации изображения. Для этого подключаем библиотеки signess и inskrib, более полные названия приведены ниже на рисунке 2.1.1. 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 необученная 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 2.1.1 – 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 2.1.2 – Конфигурация и вызов главной функции программы в основном модуле

2.2 Пайплайны на основе функций компонента.

На рисунке 2.2.1 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 2.2.1 – Пайплайн обучения модели