Документация

Ограничения

При разработке решения наша команда определила следующие ограничения, основываясь на техническом задании:

1. пользователь может загружать документы следующих форматов: файлы офисных пакетов MS Office и LibreOffice, PDF файлы с текстовым содержимым, изображениями документов и распознанными изображениями документов, графические файлы форматов PNG и Jpeg;
2. пользователь за раз может загрузить разумное количество файлов (до 1000 файлов);
3. наше решение должно находить из персональных данных (далее — ПДн) только ФИО на русском языке в различных форматах;
4. пользователями решения будут только конкретные сотрудники департаментов Москвы, поэтому решение не будет содержать механизмов разграничения прав доступа.

Гипотеза

В нашем решении выбран механизм поиска ПДн на уровне текста (natural language processing). Такой выбор обусловлен тем, что будет очень легко дополнять механизм поиска ПДн новыми категориями (телефоны, адреса и т. д.), также это позволяет подходить к поиску ПДн с учётом структуры текста.

1. Сперва происходит основной структурный поиск ПДн по тексту всего документа форматов: Фамилия И.О., И.О. Фамилия, Фамилия Имя Отчество, Имя Отчество Фамилия.
2. Так как в определенных форматах документов фамилии, имена и отчества могут находиться не рядом, то дополнительно проводится словарный поиск. Словарь содержит большой массив популярных имен и фамилий во всех падежах (склонение производилось собственным алгоритмом, присутствующем в решении).
3. Добавлен экспериментальный поиск по структуре email-ов, и телефонов. Email-ы определяются по наличию символа @ и разрешенных символов как в адресе пользователя, так и в доменном имени mail-сервера. Телефоны определяются числами, скобками и тире и по их длине.
4. Поиск адреса происходит по структуре и наличию слов из словаря адресных объектов, который содержит названия населенных пунктов и улиц, загруженных из ФИАС.

Так как скрытие персональных данных должно происходить в графических файлах, то после обнаружения ПДн, вычисляются координаты текстовых блоков, содержащих ПДн, с помощью утилит poppler utils. По этим координатам происходит скрытие ПДн уже в графических файлах страниц документов.

Так как для нашего алгоритма требуется наличие текстовых и графических представлений страниц документов, то они организуются следующими способами:

1. преобразование в PDF:
    1. офисные файлы MS Office и LibreOffice преобразуются в PDF с помощью конвертера LibreOffice;
    2. графические файлы распознаются с помощью открытого пакета Tesseract OCR, использующего нейронные сети LSTM, и сохраняются в PDF;
    3. файлы PDF, содержащие текст остаются без изменений;

    4. файлы PDF с изображениями также распознаются с помощью Tesseract OCR;
2. затем получившиеся PDF разбиваются на отдельные страницы с помощью утилиты PDFTK;
3. при страниц в виде картинок происходит преобразование страниц из PDF в jpg с помощью poppler utils;
4. затем получившиеся пары файлов страницы подаются на вход алгоритму поиска и обезличивания.

Дополнительно так как при распознавании могут быть перепутаны похожие кириллические и латинские символы, то перед алгоритмом обезличивания идет поиск таких ошибок и их корректировка.

Используемые сторонние средства

В решении используются следующие opensource продукты:

1. Для алгоритма распознования:
    1. Tesseract OCR версии 4;
    2. poppler utils;
    3. pdftk;
    4. libreOffice;
2. для организации web приложения:
    1. php фреймворк yii2;
    2. СУБД PostgreSQL;
    3. веб-сервер apache2;
    4. фронтенд фреймворк React.

Описания web интерфейса

В web-системе представлен интерфейс пакетной загрузки документов и интерфейс просмотра результатов преобразования.
Каждый загруженный пакет документов пофайлово передается в очередь на преобразование, когда преобразование завершается, то результат преобразования и итоговые файлы передаются обратно в интерфейс для просмотра пользователя. Результаты всех преобразованных документов складываются и отображаются как единый результат преобразования пакета документов.

Требования

- Ubuntu 20.04
- docker
- docker-compose v2
- make
- git

Конфигурация

Для конфигурации контейнеров:

- бекенда `backend/docker-compose.yml`

- фронтенда `frontend/docker-compose.yml`

Установка

```
git clone https://github.com/it-animals/moscow-depersonalization-data.git

cd moscow-depersonalization-data/frontend

cp .env.example .env

cd ..

make init
```

После успешной установки проекта, проект будет доступен по адресу:

http://localhost:8001

Команды

Первая инициализация проекта

```
make init
```

Рестарт контейнеров

```
make restart
```

Запуск контейнеров

```
make up
```

Выключение контейнеров

```
make down
```

Выключение контейнеров и удаление данных

```
make down-clear
```

Статус контейнеров

```
make status
```

Запуск слушателя новых заданий на конвертацию

```
make queue
```