



Функциональные характеристики программы MislalabMedAI и информация по установке и эксплуатации программного обеспечения

Введение

Программа MislalabMedAI (далее Программа), разработанная ООО "МИСЛАБ", предназначена для автоматического распознавания графических медицинских документов. В результате распознавания пользователю возвращаются структурированные данные в машиночитаемом виде. Программа используется другим программным обеспечением для интеграции с медицинскими информационными системами. Задача программы - автоматизация процесса обработки и ввода медицинских документов.

Программа написана на языке Python и распространяется в виде образа для создания контейнера в среде виртуализации Docker.

Для работы Программы не требуется подключение к сети Интернет – клиенты могут использовать её в своём закрытом контуре локальной сети.

Требования к системе

Требования к аппаратному обеспечению - персональный компьютер (десктопная или серверная версии), Intel(R) Core(TM) i7-7700HQ CPU 2.80GHz (или совместимый), оперативная память 32Гб, жёсткий диск 500Гб.

Требования к программному обеспечению: ОС Ubuntu Linux 22.04 (или совместимой). Для запуска Программы требуется, чтобы на компьютере была установлена система виртуализации Docker 25.0.3 (или совместимая), архиватор, поддерживающий формат 7zip.

Установка программного обеспечения

- 1) Установить ОС Ubuntu Linux 22.04
- 2) Обновить список пакетов и их версий командой «sudo apt-get update»
- 3) Установить архиватор командой «sudo apt-get install p7zip-full»
- 4) Установить докер командой «sudo apt install docker.io»
- 5) Создать папку «mav» в домашнем каталоге пользователя при помощи команд «cd ~», «mkdir mav»
- 6) Скопировать архив «mislalab-med-ai.7z» с образом Программы в только что созданную папку «~/mav» (архив с Программы скачивается по предоставленной ссылке, например <https://www.tethys.ru/example-app/mislalab-med-ai.7z?token=3fc165f6-4601->



[4c14-8c2e-0d2a38c0c861](#) или может быть передан другим способом по договорённости с заказчиком)

- 7) Прейти в папку «» и распаковать образ Программы командой «7z a -t7z -mx=9 mislab-med-ai.7z mislab-med-ai.tar»
- 8) Импортировать образ Программы в Docker при помощи команды «sudo docker import mislab-med-ai.tar mislab-med-ai:latest»
- 9) Создать сеть при помощи команды «sudo docker network create mislab-med-ai-net»
- 10) Создать том при помощи команды «sudo docker volume create mislab-med-ai-vol»
- 11) Создать и запустить контейнер командой «sudo docker run -d --name mislab-med-ai-1 -w /src -u 0:0 --network mislab-med-ai-net -e 17013 -p 17013:17013 -v mislab-med-ai-vol:/src:rw mislab-med-ai:latest python3 /src/svc.py»

Начало работы

После создания и запуска контейнера Программы не требует дополнительных настроек. Для проверки успешного запуска Программы можно использовать команду «sudo docker logs mav_mav-svc-ocr_1».

При успешном запуске Программы, в последних строках лога Docker-а можно будет увидеть строки:

INFO: Started server process [1]

INFO: Waiting for application startup.

INFO: Application startup complete.

INFO: Uvicorn running on http://0.0.0.0:17013 (Press CTRL+C to quit)

Описание функциональных характеристик

Программа позволяет осуществлять распознавание направлений по форме № 057/у-04. Разбор документа на структурные элементы осуществляется последовательной обработкой документа при помощи ряда нейронных сетей. Этапами разбора документа являются:

1 Этап – распознавание типа документа. Задача этапа - отличать различные типы/или подтипы исследуемого документа для последующего детального анализа структуры документа. На этом этапе осуществляется обработка документа при помощи модели yolo-057-sel-box, построенной на базе нейронной сети YoloV8s и обученной на собственном датасете, а также захват основного поля документа из предоставленного изображения.



2 Этап – анализ компоновки документа для извлечения изображений необходимых полей. В результате этого этапа получается набор изображений полей документа интересующих нас типов (ОГРН, ОМС, код диагноза МКБ). На этом этапе используется модель yolo-057-prn-1-wb-1k, построенная на базе нейронной сети YoloV8l.

3 Этап – извлечение текста из изображений полей, полученных в рамках выполнения Этапа 2, с использованием моделей на базе нейронной сети EasyOCR.

4 Этап – постобработка полученного текста для устранения очевидных ошибок распознавания.

5 Этап – Формирование и отправка пользователю структурированной информации с результатами распознавания в формате JSON.

Руководство пользователя

Программа предназначена для использования в автоматизированных системах обработки медицинской информации. Взаимодействие с Программой осуществляется посредством вызова Web-API методов:

- 1) «<http://127.0.0.1:17013/api/v1/ai/version>» - позволяет использовать GET запрос для получения версии ПО. Запрос выполняется без параметров, в заголовке передаётся «Authorization-Token». Далее – «запрос version». Этот метод используется в том случае, если в службу поддержки нужно передать информацию о том, какая именно версия Программы установлена у заказчика и какие настройки работы оно использует.
- 2) «<http://127.0.0.1:17013/api/v1/ai/doc-ocr>» - позволяет использовать POST запрос для распознавания изображения документа. В заголовке передаётся «Authorization-Token», а в качестве тела запроса – JSON вида «{ "img":{ "code":{ "v": "f57" }, "b64": "<Base64Image>" } }». Далее – «запрос doc-ocr». **Пример запроса и ответа на запрос приведён в приложении №1.**

После запуска Программы пользователю доступен Swagger по адресу «<http://127.0.0.1:17013/docs>», который можно использовать для проверки выполнения запросов. При формировании вызовов нужно использовать заголовок «Authorization-Token: 2c80b827-749e-42ec-99d1-cc08cb7ddb1f».

В качестве ответа на запрос version возвращается JSON вида:



```
{  
  
  "name": "mav-svc-ocr",  
  
  "branch": "master",  
  
  "date": "2024.03.07.14.40.26",  
  
  "commit": "64e52e1b4016bd45263947d79ceacc84a9802d6f",  
  
  "F57Parser": {  
  
    "F57SelectorBody": {  
  
      "model": "yolo-057-sel-box.8s100e16b-bg.pt"  
  
    },  
  
    "F57SelectorFields": {  
  
      "model": "yolo-057-prn-1-wb-1k.8l61e16.pt"  
  
    },  
  
    "F57OcrFieldOgrn": {  
  
      "model": "craft_standard",  
  
      "params": "en;clr"  
  
    },  
  
    "F57OcrFieldOms": {  
  
      "model": "craft_standard",  
  
      "params": "en;clr"  
  
    },  
  
    "F57OcrFieldMkb": {  
  
      "model": "craft_standard",  
  
      "params": "en;clr;rpl;add"  
  
    }  
  
  }  
  
}
```



```
}  
  
}  
  
}
```

Где поля имеют следующее значение:

Параметр	Значение
name	Внутреннее название проекта
branch	Название ветки CMS из которой был взят проект
date	Дата сборки
commit	ID коммита
F57Parser	Объект, содержащий параметры работы компонент анализа изображения документа

В качестве ответа на запрос doc-ocr возвращается JSON вида:

```
{  
  
  "img": {  
  
    "code": {  
  
      "v": "f57 t1",  
  
      "p": 0.9690848588943481  
  
    }  
  
  },  
  
  "features": [  
  
    {  
  
      "code": {  
  
        "v": "oms",  
  
        "p": 0.821002185344696  
  
      }  
  
    }  
  
  ]  
}
```



```
},  
  "value": {  
    "v": "0002220222000777",  
    "p": 0.9952449042697629  
  }  
},  
{  
  "code": {  
    "v": "mkb",  
    "p": 0.814726710319519  
  },  
  "value": {  
    "v": "C85.9",  
    "p": 0.5430103320873902  
  }  
},  
{  
  "code": {  
    "v": "ogrn",  
    "p": 0.8037108182907104  
  },  
  "value": {  
    "v": "0055556103555",
```



```
"p": 0.9953925198254994  
  
}  
  
],  
  
"errors": []  
  
}
```

В ответе в объекте «img» возвращается уточнённый тип документа (если это применимо), а в объекте «features» набор распознанных элементов документа. Для каждого элемента передаётся тип («code»), состоящий из строкового значения («v») и показателя уверенности модели в результате распознавания («p») и значение («value») так же состоящее из распознанного значения поля («v») и показателя уверенности модели в результате распознавания («p»).

Объект «errors» может содержать список ошибок, которые возникли в процессе распознавания документа. Для каждой ошибки указывается код («code») и расшифровка ошибки в человеко-читаемом виде («txt»). Например,

```
"errors": [  
  
  {  
  
    "code": 1,  
  
    "txt": "Указанный в запросе тип документа 'f5722' не поддерживается."  
  
  }  
  
]
```

Устранение неисправностей

Программа поставляется в виде готового образа, что сводит к минимуму количество потенциальных проблем, которые могут возникать у пользователя в процессе пуско-наладки. Описание основных возможных проблем:



Проявление	Возможная причина и меры по устранению
<p>Программа не отвечает на запросы, при анализе логов Docker видно, что контейнер был остановлен.</p>	<p>Наиболее частой причиной такого поведения системы является недостаток оперативной памяти.</p> <p>Необходимо отказаться от другого ПО, которое может потреблять доступную память или установить в систему больше оперативной памяти.</p>
<p>Программа не отвечает на запросы, из логов Docker видно, что контейнер запущен.</p>	<p>Наиболее частой причиной такого поведения является:</p> <ol style="list-style-type: none">1) Работа брандмауэра, который блокирует обращения к Программе по сети. Для решения этой проблемы необходимо отказаться от использования брандмауэра или настроить его соответствующим образом.2) Конфликт при использовании TCP портов. Если другое ПО использует порт 17013, то при создании контейнера можно изменить порт на тот, который в системе является свободным (поменять параметр «host_port» на нужный) <pre>«sudo docker run -d --name mislab-med-ai-1 -w /src -u 0:0 --network mislab-med-ai-net -e 17013 -p host_port:17013 -v mislab-med-ai-vol:/src:rw mislab-med-ai:latest python3 /src/svc.py»</pre>