МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное
образовательное учреждение высшего образования
«Самарский национальный исследовательский университет
имени академика С.П. Королева»
(Самарский университет)

Институт информатики и кибернетики
Факультет информатики
Кафедра технической кибернетики

Отчет по лабораторной работе №2

Дисциплина: «Инженерия данных»

Тема: **«Инференс и обучение НС»**

Выполнил: Мелешенко И.С.

Группа: 6233-010402D

Самара 2023

СОДЕРЖАНИЕ

Часть 1. Построение пайплайн для инференса данных.

Шаг 1. Разработка и реализация DAG-а

В рамках первого задания необходимо реализовать пайплайн, который реализует систему "Автоматического распознавания речи" для видеофайлов. Построенный пайплайн будет выполнять следующие действия поочередно:

- Производить мониторинг целевой папки на предмет появления новых видеофайлов.
- Извлекать аудиодорожку из исходного видеофайла.
- Преобразовывать аудиодорожку в текст с помощью нейросетевой модели.
- Формировать конспект на основе полученного текста.
- Формировать выходной .pdf файл с конспектом.

Для реализации описанных действий мы будем использовать DockerOperator, а также FileSensor для получения необходимого видеофайла.

Для работы task-а по ожиданию получения нового видео необходимо создать новое подключение к airflow. Для создания подключения переходим в Airflow по адресу http://localhost:8080/connection/list/ или мы можем в Airflow пройти по пути Admin>>Connections, как на рисунке ниже.

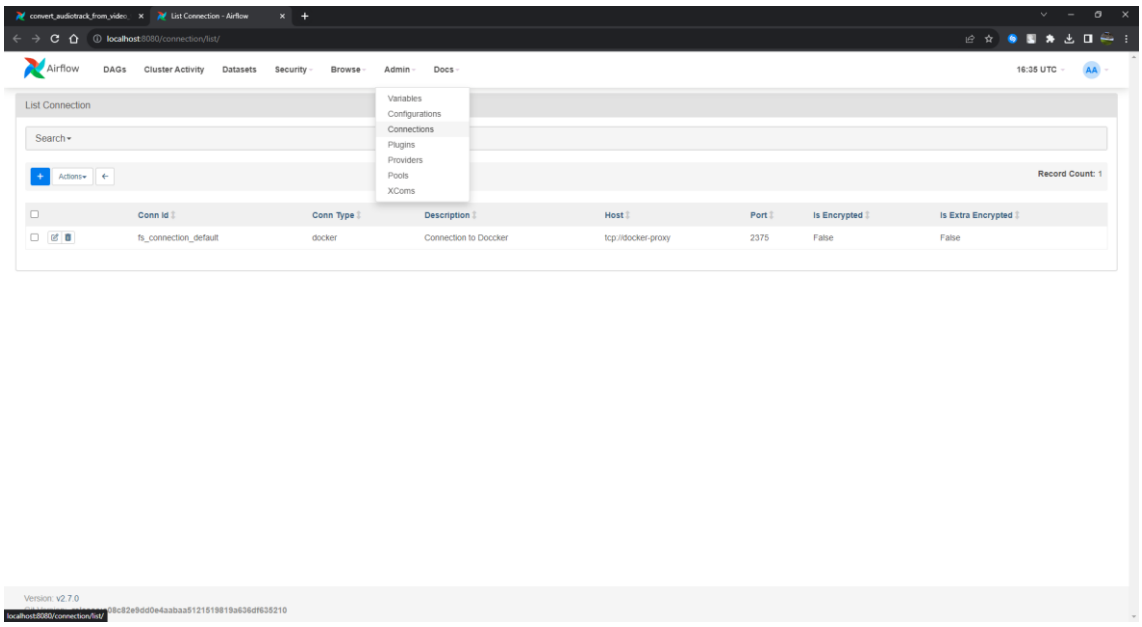

Рисунок 1 – Создание Connection

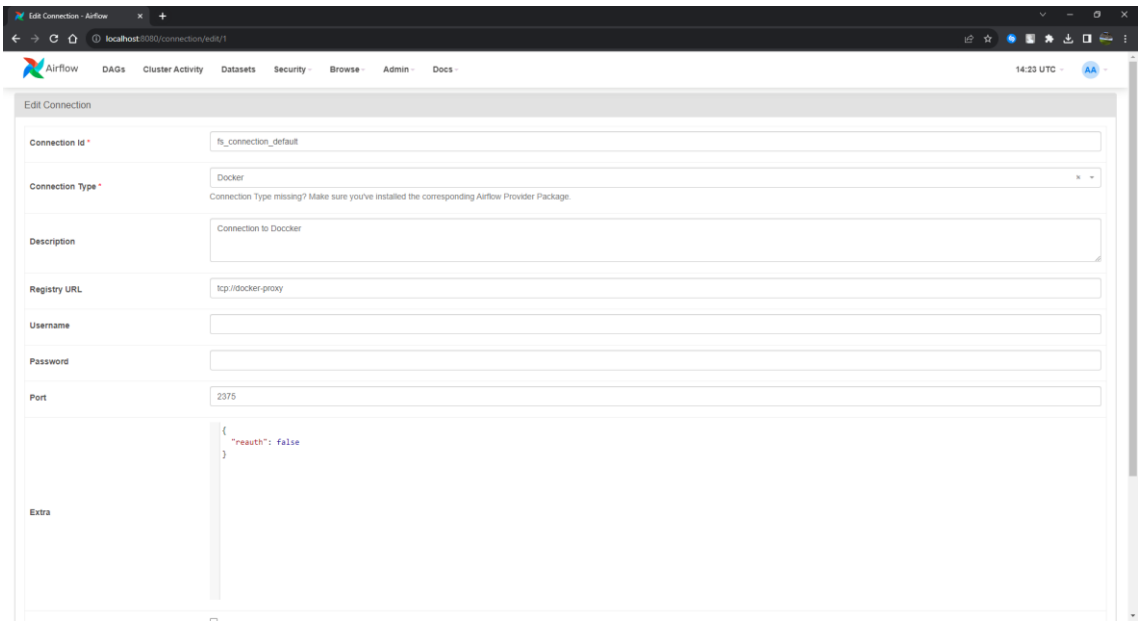

Рисунок 2 – Параметры Connection

Шаг 2. Регистрация на huggingface и получения токена API.

Далее для того, чтобы можно было преобразовать наш аудиофайл в текст, а после получить из него summary, необходимо зарегистрироваться на https://huggingface.co/ и получить токен API с правами записи для возможности посылки и получения запросов к сайту.

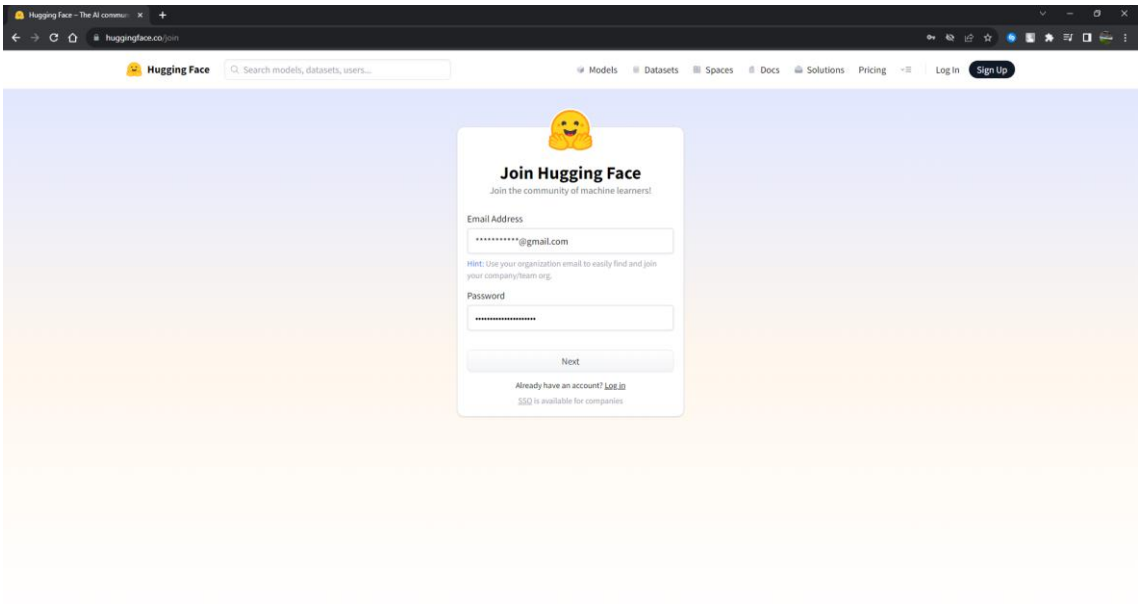

Рисунок 3 – Регистрация на huggingface

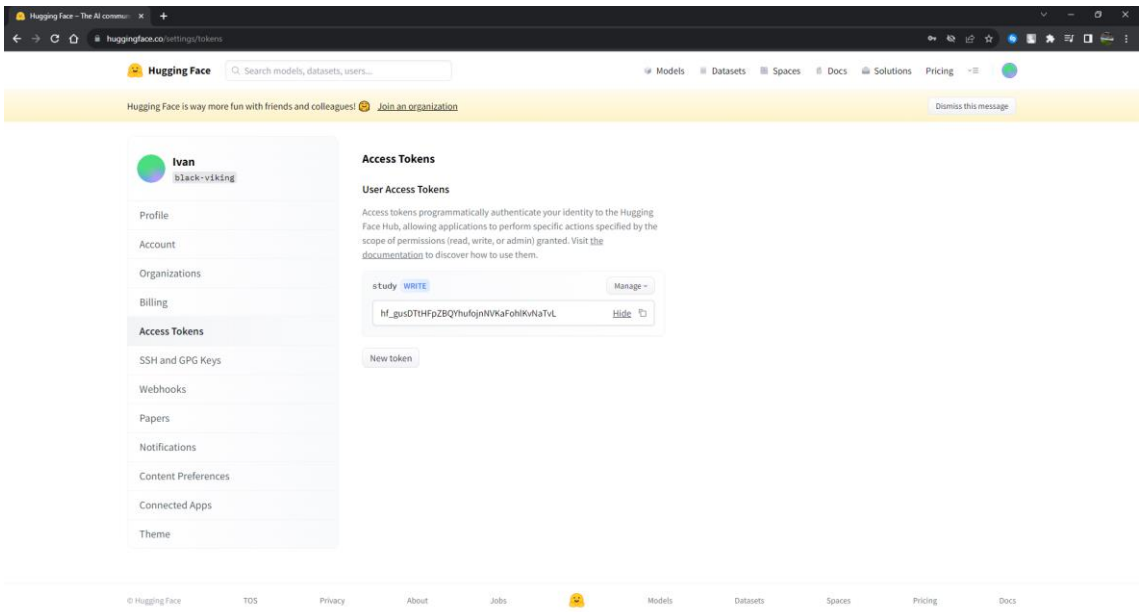

Рисунок 4 – Получение токена API

Шаг 3. Создание Docker образа с необходимыми библиотеками.

Для сохранения конспекта в PDF, необходимо было использовать библиотеку fpdf. Создадим необходимый для этого образ в Docker, который будет содержать в себе необходимы библиотеки для выполнения всей лабораторной работы. Процесс представлен ниже.

Вначале создадим Dokerfile который будет содержать в себе инструкции по сборке и развертыванию контейнера. Контейнер мы создаем на основе tensorflow, который нам пригодится при выполнении второй части работы. Так же добавим следующие библиотеки, которые нам понадобятся в будущем:

- Scikit-learn
- Numpy
- Pandas
- FPDF

Пример Dokerfile, который я использовала в лабораторной работе приведен на рисунке ниже, а также в репозитории с решением лабораторной работы. После того как Dokerfile создан переходим в консоль и выполним процесс сборки.

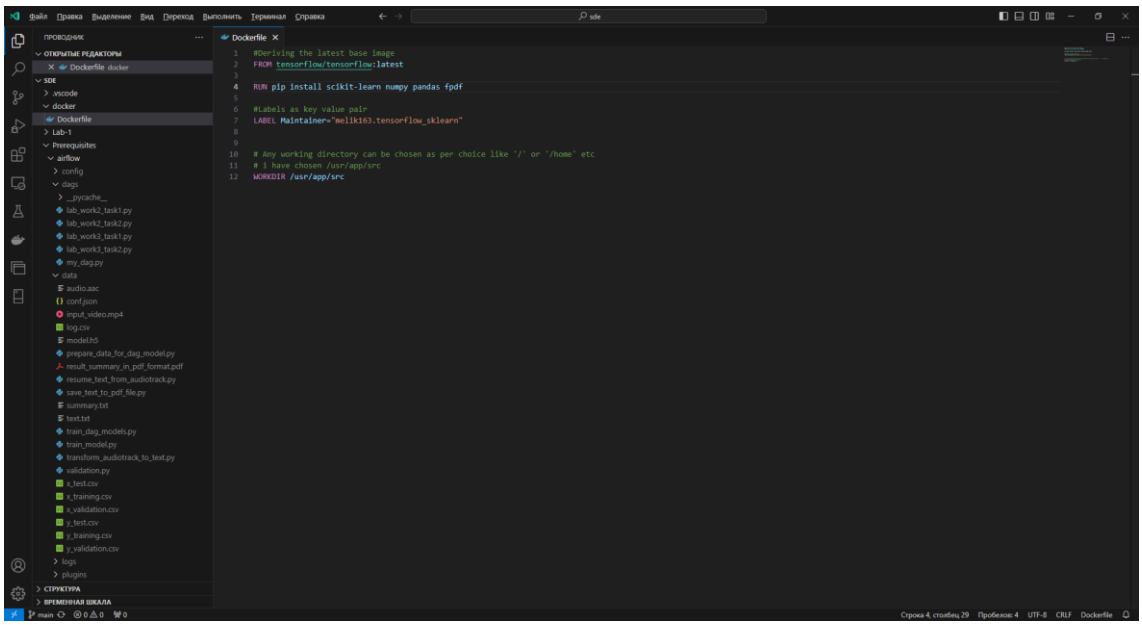

Рисунок 7 – Создание Dokerfile

Первым делом произведем сборку образа, при помощи команды:

$ docker build . –t our_tensorflow_container

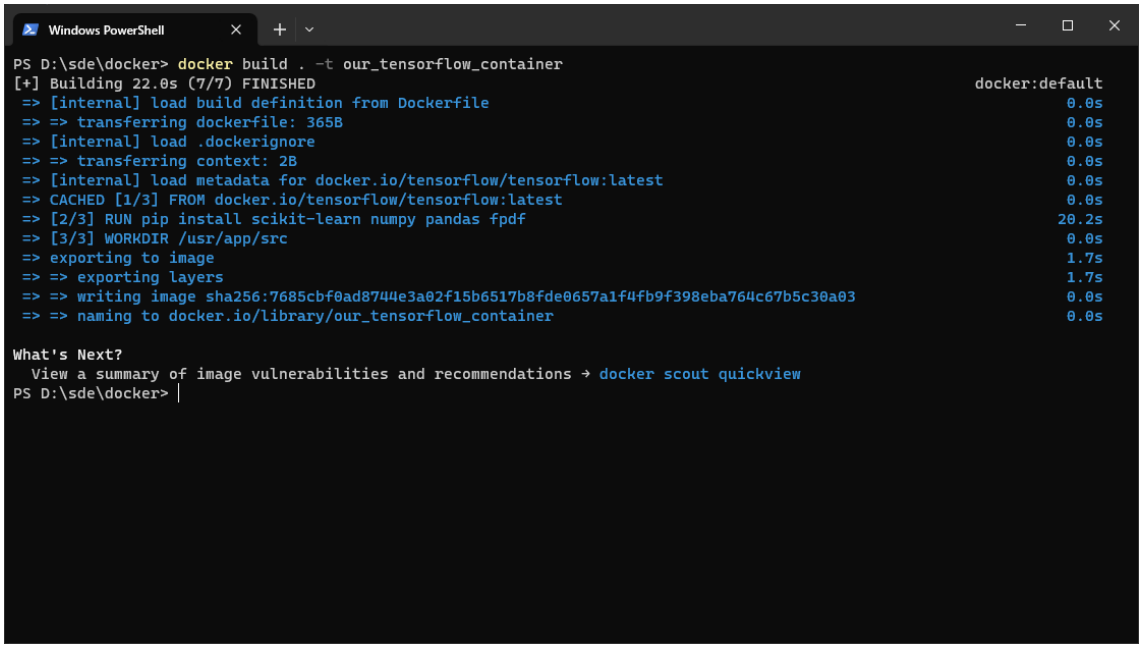

Рисунок 8 – Сборка образа

После успешной сборки, необходимо произвести проверку того, что наш Docker образ был создан. Для этого используем команду

$ docker images

После проверки произведем присвоение tag нашему образу, для того чтобы можно было произвести отправку нашего контейнера в DockerHub. Для этого воспользуемся командой:

$ docker tag –t our_tensorflow_container melik163/ our_tensorflow_container:1.0

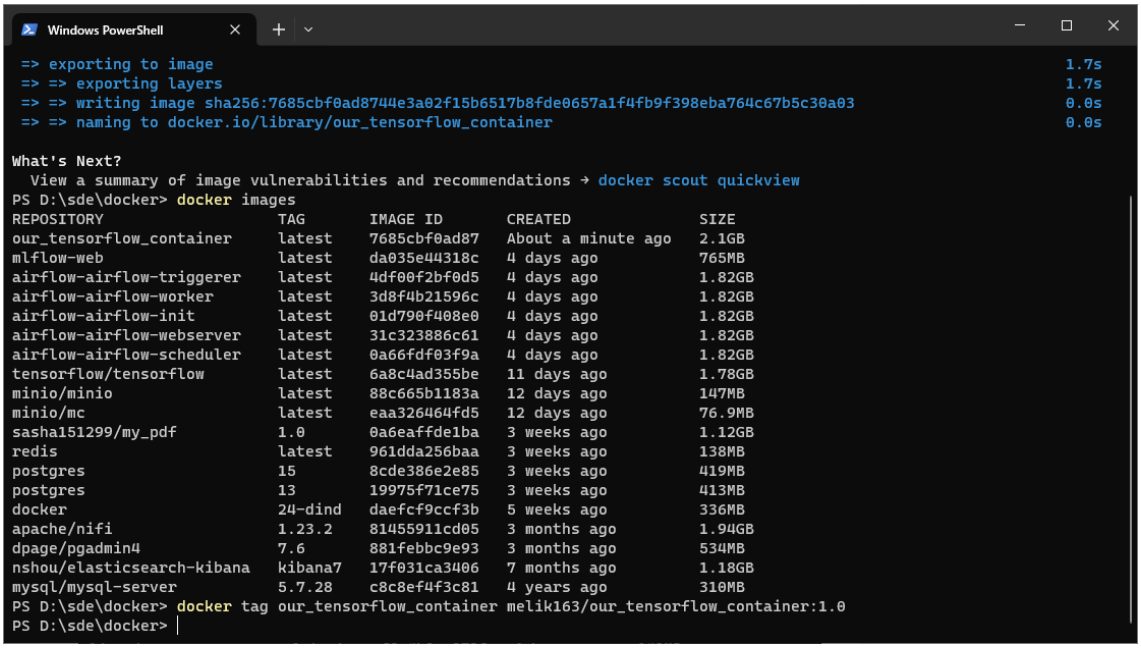

Рисунок 9 – Присвоение тега образу

В итоге после всех приготовлений, произведем отправку нашего образа в DockerHub, при помощи команды:

$ docker push melik163/ our_tensorflow_container:1.0

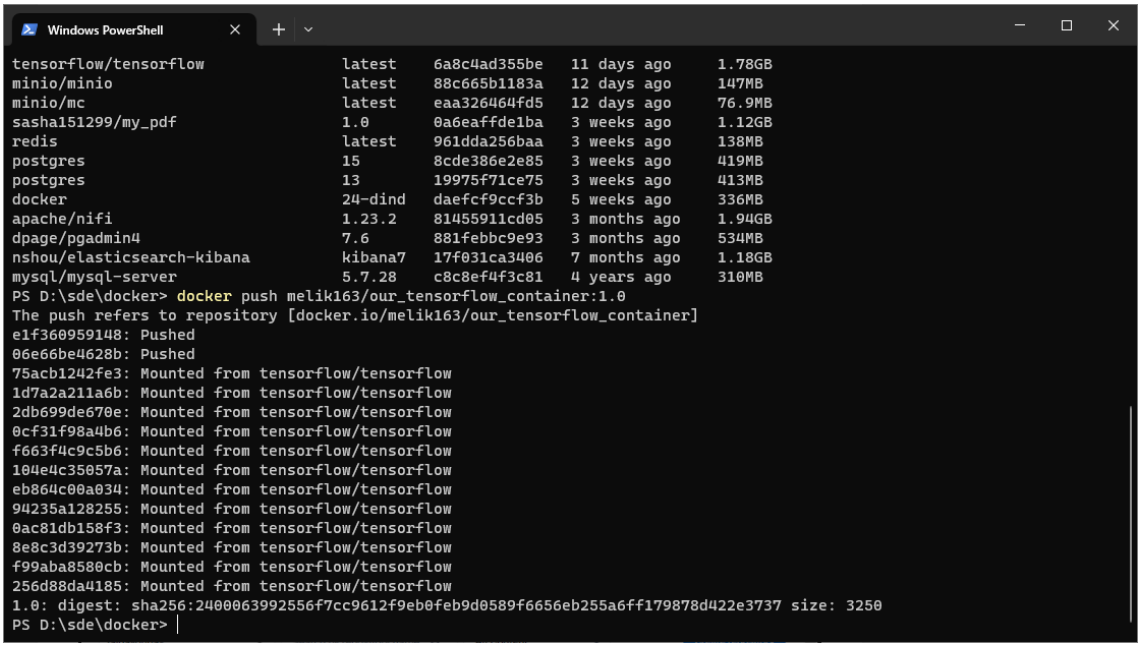

Рисунок 10 – Отправка образа в DockerHub

Шаг 4. Подготовка DAG-а.

В результате выполнения данной части работы был разработан DAG, состоящий из 5 task-ов:

wait_get_new_videofile – осуществляет «прослушивание» указанной директории, на предмет появления в ней видеофайла, который будет принят далее в работу.

extract_audiotrack_from_video – осуществляет извлечение аудиодорожки из исходного видеофайла для дальнейшей работы. Для извлечения аудиодорожки из видео была использована библиотека ffmpeg, которая была получена из Docker-образа jrottenberg/ffmpeg.

transform_audiotrack_to_text – осуществляет обработку, распознавание и трансформацию аудиофайла в текстовый файл. Данная операция осуществлялась при помощи запросов в сервис huggingface.

resume_text_from_audiotrack – осуществляет суммаризацию текстового файла, который получен на предыдущих этапах.

save_get_text_from_txt_to_pdf – осуществляет сохранение полученного результата в файл формата pdf.

Шаг 5. Запуск DAG-а.

Теперь после всех необходимых настроек и приготовлений, мы можем запустить наш DAG. Для этого переходим в airflow: http://localhost:8080/home и находим наш только что созданный DAG:

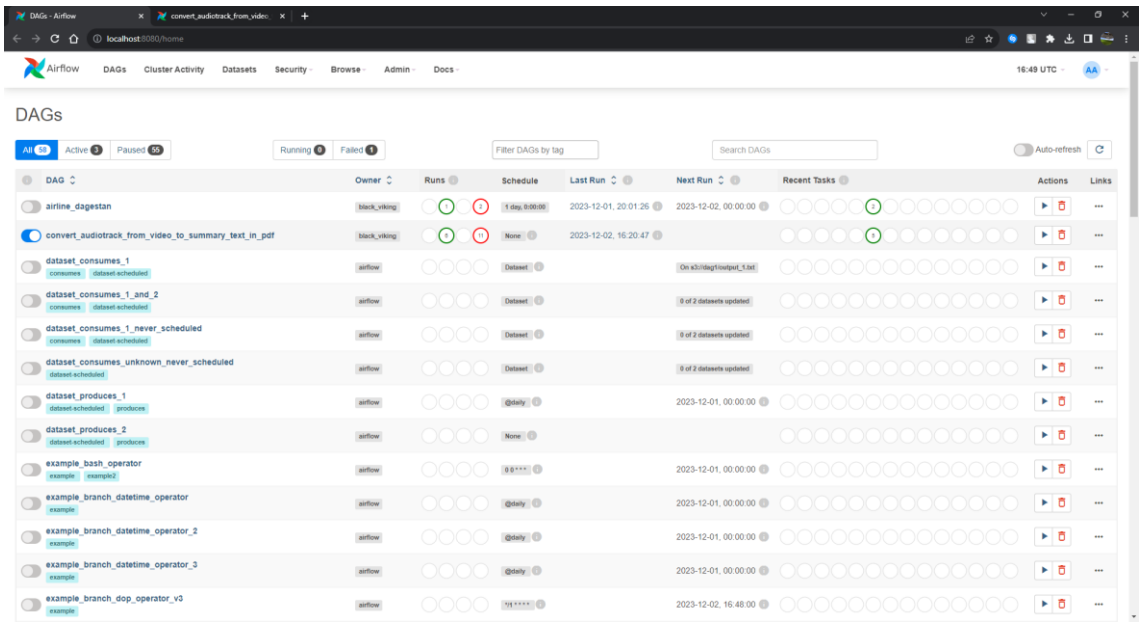

Рисунок 5 – Поиск DAG-а.

Далее запускаем наш DAG и наслаждаемся процессом.

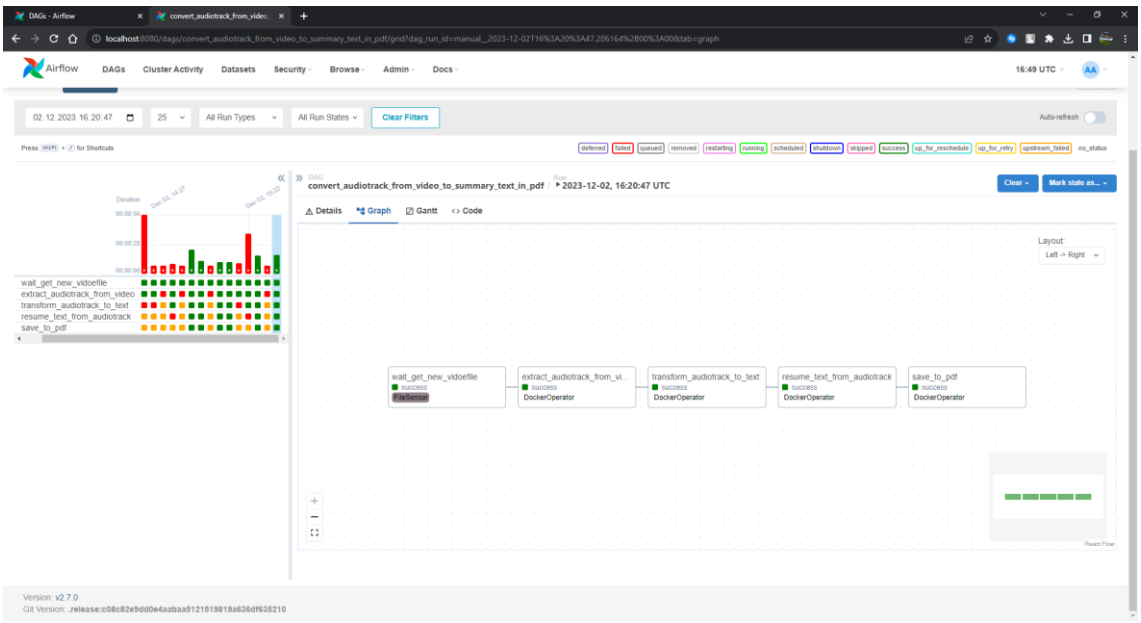

Рисунок 6 – Запуск DAG-а.

В качестве исходного видео использовался фрагмент из кинофильма «Крестный отец» длительностью 3 минуты 11 секунд. После чего мы получали аудиодорожку, которая использовалась в качестве основы для получения текстового файла.

Рисунок 11 – Результат работы huggingface по преобразованию аудио в текст

Далее полученный результат мы еще раз передавали huggingface для получения уже конспекта по отправленному нами файлу. Полученный результат записывали pdf-файл.

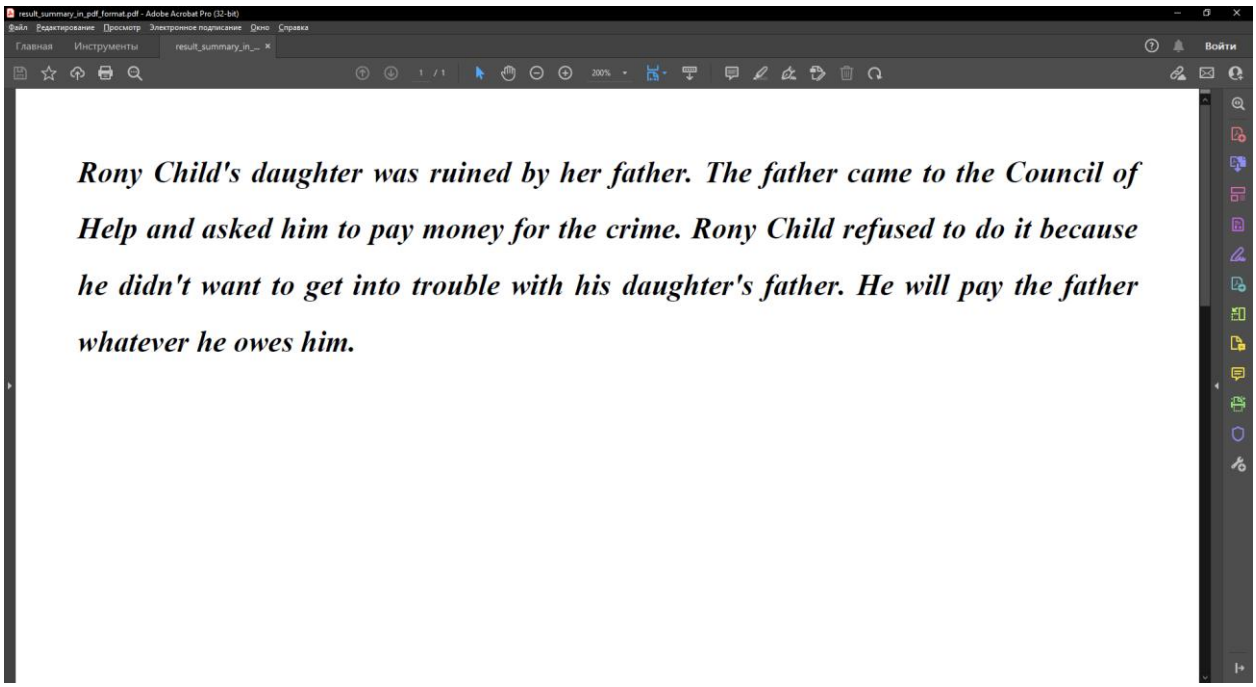

Рисунок 12 – Конспект текстового файла.

Получилось неплохо. Перейдем ко второй части.

Часть 2. Пайплайн, который реализует систему автоматического обучения/дообучения нейросетевой модели

В рамках второй части лабораторной работы нам необходимо было разработать пайплайн, который реализует систему автоматического обучения/дообучения нейросетевой модели.

Шаг 1. Разработка DAG

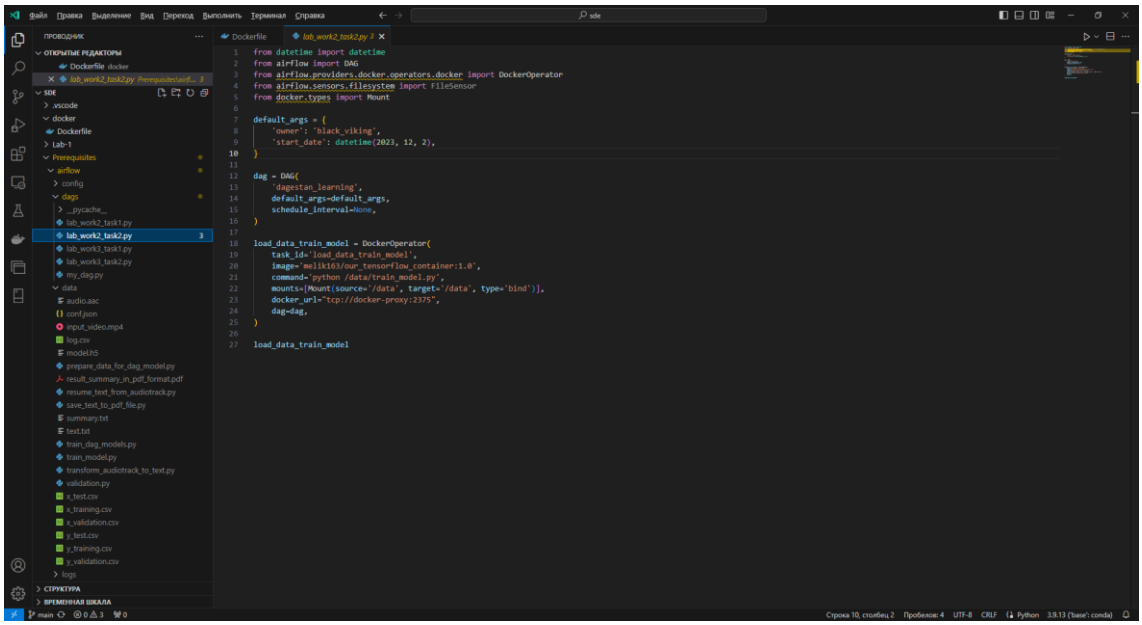

Рисунок 13 - Пайплайн

DAG запускал код, который получал датасет вин load_wine из sklearn.datasets, после чего мы проводили разбиение данных. Которые передаются в нейросеть, после чего модель проходит обучение. Процесс обучения логируется.

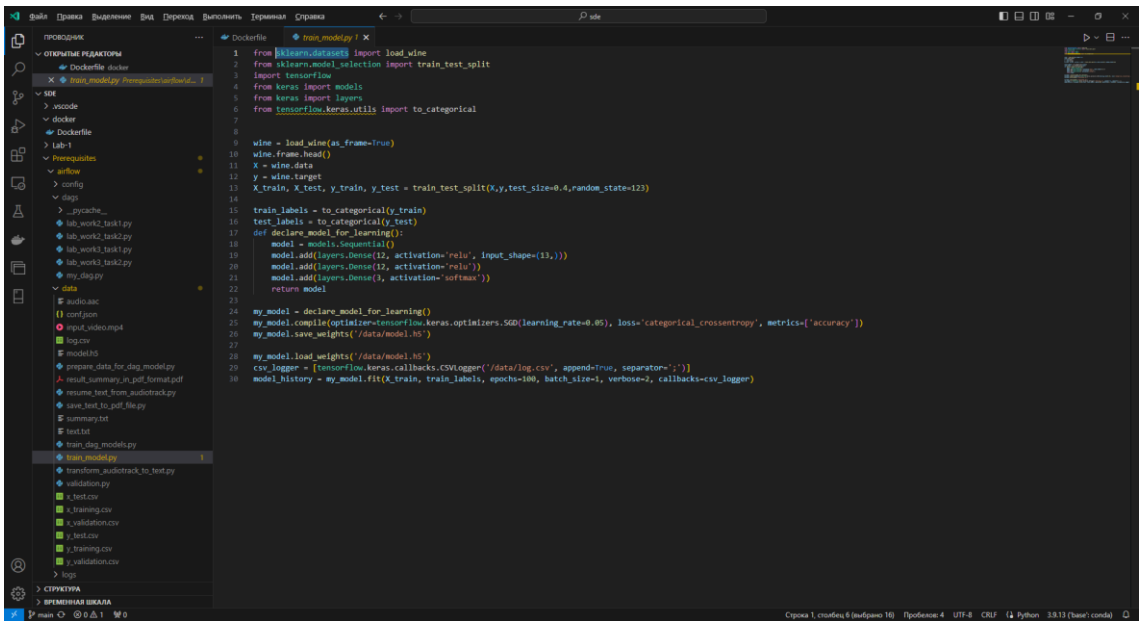

Рисунок 14 – Код обучения модели.

Шаг 2. Запуска DAG-а.

В процессе запуска DAG-а модель была обучена и показала какие-то результаты, которые мы записали в файл. В итоге получили вот такой лог обучения:

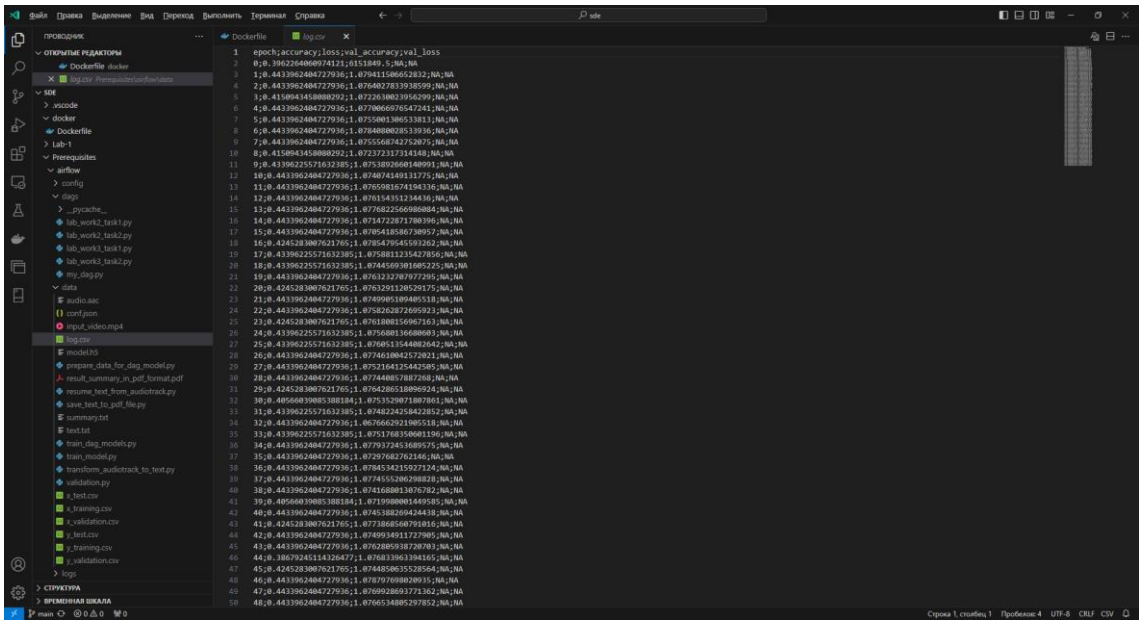

Рисунок 15 – Лог процесса обучения нейросети.

Заключение

В заключении хотелось бы отметить полезные навыки, полученные в результате выполнения лабораторной работы:

1. Работа с DAG в Airflow
2. Работс сетями на huggingface