Тестовое задание ML-инженер

Время на выполнение: до 12ч

Разработайте систему поиска наиболее похожих растений на основе входного изображения среди изображений из заданной библиотеки.

Входные данные: одно цветное изображение растения.

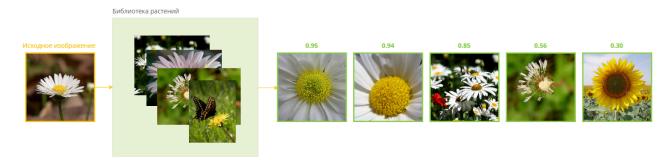
Выходные данные: топ-5 наиболее похожих изображений из библиотеки в виде словаря:

- ключ: image_path путь к изображению в библиотеке;
- значение: similarity_score показатель сходства (значение метрики);
- элементы словаря должны быть отсортированы по убыванию сходства (от наиболее к наименее похожему).

Для обучения и тестирования используйте датасет Flowers Recognition.

Оформление

- 1. Оформите этапы подготовки данных, обучения модели, валидации и тестирования в Jupyter Notebook.
- 2. Пример работы модели:
 - для 5 случайно выбранных изображений из тестовой выборки покажите результаты поиска;
 - оформите результат в виде сетки из 6 изображений:
 - входное изображение;
 - топ-5 найденных изображений из библиотеки.



3. Деплой модели:

- реализуйте модель в виде Docker-контейнера;
- создайте API-эндпоинт для загрузки изображения и получения результата поиска в формате JSON;
- поиск должен производиться по тестовой выборке датасета (имитирующего библиотеку).

См. чек-лист для самопроверки на следующей странице.

Чек-лист для самопроверки (требования)
□ Репозиторий на Github:
🗌 (опционально, будет плюсом) опишите ваш подход к решению
данной задачи, с какими проблемами столкнулись и как вы их решили;
□ регулярные коммиты с момента инициализации репозитория;
☐ README.md:
□ инструкция к запуску проекта;
□ пример использования (инференса) модели: загрузка
тестового изображения и отправка запроса на АРІ.
□ Воспроизводимость:
□ в репозитории размещены веса модели;
🗌 в репозитории размещен Dockerfile (и compose.yml, если
необходимо) для деплоя модели;
□ для библиотек задан random_seed.
□ Реализация проекта:
□ подготовка данных, обучение и тестирование оформлены в Jupyter Notebook;