Документация проекта Headliners

1. Архитектура и методы

- Модель: YOLO v8s
- Тип задачи: Object Detection (обнаружение объектов)
- Фреймворк: Ultralytics YOLO с CLI интерфейсом

Архитектура для решения задачи object-detection

В основе нашего решения лежит модель YOLO, которой присущи следующие характеристики:

Backbone (Базовая сеть)

- Цель: Извлечение признаков из входного изображения.
- Реализация: Сверточная сеть (CSPDarknet с использованием C2f-модуля).

Neck (Шея)

- Цель: Агрегация и комбинирование признаков с разных уровней для улучшения детекции объектов разного масштаба.
- Реализация: PAN-FPN (Path Aggregation Network - Feature Pyramid Network) (также с использованием C2f-модулей)

Head (Голова)

- Цель: Финальное предсказание.
- Реализация: Набор сверточных слоев, которые предсказывают:
    1. Bounding Boxes (ограничивающие рамки): Координаты (x, y, width, height) и уверенность (confidence score).
    2. Classes (Классы): Вероятность принадлежности объекта к каждому из классов.

Наше решение существенно расширило возможности YOLO для детекции инородных тел на снимках грудной клетки - за счёт обучения на нескольких вариантах аугментации изображений и аугментировании данных для детекции во время предобработки с последующим слиянием результатов, что даёт куда более точную картину и делает возможной обработку в том числе изображений плохого качества.

Структура библиотеки `headliners_CXRForeignViewer`

Библиотека `headliners_CXRForeignViewer` состоит из трех основных модулей:

1. config

Содержит переменные для настройки путей проекта. Обязательно настройте этот файл при запуске модели в режиме обучения. При запуске скрипта в режиме детекции скрипт сам запросит директорию, где хранятся DICOM файлы.

2. lib

   Подмодули:

   - augmentation - методы аугментации, использованные для подготовки train и test датасетов перед запуском модели в режиме обучения. Для обработки использовались фильтры `CLAHE`, `DCP`, а также `комбинации` этих фильтров с различными параметрами.
   - file_management - методы файлового менеджмента, использованные командой разработчиков для подготовки датасетов. Данный модуль делает возможной рекурсивную обработку директории, содержащей данные, целиком. Наличие вложенных папок не является препятствием для работы модели.
   - image_processing - методы, использованные для предобработки данных перед запуском модели в train или predict моде.
   - model_mapper - класс модели, позволяющий легко и удобно использовать её функционал.
   - postprocess - методы, использованные для постобработки результата модели.
   - metrics - методы расчёта метрик для оценки моделей.

3. model

   Модель, обученная командой. Директория содержит как параметры модели, так и описание её характеристик и эпох обучения, а также диаграммы.

4. scripts

   Демонстрационные скрипты. Содержит скрипт, показывающий работу метода `predict` модели `YoloViewer`.

5. interactive_detector

   Это самостоятельный модуль с графическим интерфейсом, специально разработанный командой для удобной разметки DICOM изображений. Он позволяет:

   - В интерактивном режиме размечать снимки с помощью боксов.
   - Накладывать на исходное изображение различные фильтры аугментации без непосредственного его изменения.
   - Автоматически получать координаты и параметры каждого боква.
   - Сохранять разметку в требуемом формате.

6. demo_files

   Директория с примерами файлов для демонстрации работы решения. `Важно!` Демонстрационные файлы DICOM не предполагают использования этой директории для

запуска модели в режиме обучения, несмотря на то, что помимо DICOM файлов директория содержит пример файла dataset.yaml.

7. docs

   Документация по проекту.

8. Демонстрационные скрипты и блокноты.

   Запуск решения возможен с помощью файлов `main.py`, `detect.ipynb`, `detect_autonomous.ipynb`. Подробнее см. `README.md`.

9. Результаты работы модели

   Результаты работы модели по дефолту помещаются в папку `results` и файл бинарной классификации `result.xlsx`. Эти пути можно настроить через `config`.

2. Данные

Структура данных

Программа принимает для работы путь к папке и работает только с изображениями в формате DICOM. Наличие внутри папки вложенных не является препятствием для работы модели.

- Структура обучающего датасета:
  - Исходные файлы;
  - Файлы с применённой к ним аугментацией;
  - Инвертированные копии файлов из категорий, указанных выше.

Разделение данных

Обучающий датасет был разбит на:

- Train детасет, из которого
  - 80% файлов для первичного обучения модели
  - 20% файлов - валидация
- Test датасет

Для более эффективного разделения применялось стратифицированное перемешивание с фиксированным random_seed=42.

3. Обоснование выбора методов

Модель

Модель YOLO v8s была выбрана по нескольким причинам.

- Во-первых, она использует сети агрегации путей (PAN) для лучшего слияния признаков и использует потерю фокуса для точного прогнозирования ограничивающей рамки, что обусловливает её высокую точность и скорость.
- Во-вторых, она является продуктом, который активно поддерживается и развивается командой разработчиков, что гарантирует долговременную работу нашей библиотеки и возможность её дальнейшего развития с улучшением качества моделей новых поколений.
- В-третьих, она является популярным свободно распространяемым продуктом, а значит, имеет хорошую интеграцию со всеми существующими системами, архитектурами и библиотеками.

Препроцессинг

Одной из основных задач, которая стояла перед командой, было повысить точность детекции на изображениях плохого качества и контрастности, что в случае анализа медицинских данных может быть критичным. Для этого изображения подвергались дополнительной обработке.

Лучшие параметры для фильтров подбирались с целью, с одной стороны, максимально возможно повысить качество изображения, с другой стороны - не допустить появления на нём избыточных артефактов, которые модель может интерпретировать как значимые отклонения.

Постпроцессинг

В ходе детекции изображение получает несколько вариантов с разной аугментацией, после чего все эти варианты обрабатываются моделью, а результаты анализируются и суммируются, что обеспечивает намного лучшую точность по сравнению с анализом одного изображения.

4. Процесс обучения

Параметры обучения

Модель обучалась со следующими параметрами:

task=detect mode=train epochs: 100 imgsz: 640 batch: 8 conf=0.25

Обработка данных

Для удобства и ускорения работы нами был разработан независимый модуль интерактивной разметки, который позволил, с одной стороны, быстро произвести разметку обучающих данных, а с другой - подобрать оптимальные параметры для предобработки изображений.

При аугментации данных мы использовали следующие фильтры:

- CLAHE
- DCP
- Комбинация фильтров выше

Это позволило существенно расширить первоначальный датасет и обучить модель обрабатывать данные разного качества. Также это сделало результаты модели легко интерпретируемыми.

С помощью тестов мы ваявили оптимальные параметры для фильтров, которые увеличивали точность модели:

- Параметр clickLimit для фильтра CLAHE: 5.0
- Параметр tileGridSize для фильтра CLAHE: (8,8)
- Параметр размерности (patch) для фильтра DCP: 15
- Все изображения обрабатывались в двух модах: Gray и RGB

5. Методы оценки и результаты тестирования

Основные метрики

Для расчёта метрик модели использовался инструмент ClearML. Мы оценивали следующие метрики:

- Precision: Точность обнаружения. Получен результат: 0.977
- Recall: Полнота обнаружения. Получен результат: 0.967
- F1-score: Гармоническое среднее. Получен результат: 0.972
- mAP (0.5-0.95): Mean Average Precision. Получен результат: 0.838
- ROC-AUC: Площадь под ROC-кривой. Получен результат: 0.846

Функции расчета метрик

def calculate_iou(box1, box2): # Расчет Intersection over Union

def calculate_precision_recall(pred_boxes, true_boxes, iou_threshold=0.5): # Расчет precision и recall

def calculate_map(predictions, targets, iou_threshold=0.5): # Расчет mean Average Precision

def calculate_detection_roc_auc(predictions, targets, iou_threshold=0.5): # Расчет ROC-AUC для детекции

Результаты тестирования

В ходе тестирования мы сравнили несколько моделей и выявили наиболее подходящую для нашей задачи.

Здесь мы приводим результаты лишь некоторых тестов, которых было проведено несколько десятков.

Параметры модели	**Метрики**
Модель: YOLO v8s Эпох: 100	Precision: 0.977 Recall: 0.967 F1: 0.972 mAP: 0.846

Данная модель дала оптимальный показатель соотношения правильных обнаружений и случайных неверных детекций.

Параметры модели	Метрики
Модель: YOLO v8s	Precision: 1.0
Эпох: 200	Recall: 0.953
	F1: 0.972
	mAP: 0.9

Данная модель дала хороший показатель точности, но продемонстрировала слишком высокое количество неверных детекций.

Параметры модели	Метрики
Модель: YOLO 11l	Precision: 0.836
Эпох: 100	Recall: 0.845
	F1: 0.840
	mAP: 0.647

Это модель нового поколения, которая даёт хороший показатель точности, но только при долгом обучении на большом количестве данных, что в медицинской сфере не всегда возможно.

Параметры модели	Метрики
Модель: YOLO 11l	Precision: 0.951
Эпох: 200	Recall: 0.955
	F1: 0.953
	mAP: 0.837

При обучении в течении большего количества эпох показатели модели несколько повышаются, но затем точность выходит на плато.

Параметры модели	Метрики
Модель: YOLO 11x	Precision: 0.962
Эпох: 200	Recall: 0.85
	F1: 0.903
	mAP: 0.733

Модель ещё более высокого поколения показала хорошую обучаемость, но требовала очень большого времени на обучение и существенного расширения обучающей выборки, при том что достичь максимальной точности так и не удалось.

6. Заключение

Разработанная система демонстрирует высокую эффективность в обнаружении инородных тел на рентгенограммах ОГК. Усовершенствование архитектуры YOLO позволило достичь баланса между точностью и скоростью работы, что важно для клинического применения.

7. Дорожная карта

Куда же мы двигаемся дальше?

Для повышения точности модели в дальнейшем возможно:

- Расширение датасета за счёт увеличения методов аугментации. Это требует большего времени на тестирование результатов.
- Расширение исходной обучающей базы. Осложнено политикой конфиденциальности в отношении медицинских данных, которая влияет на доступность датасетов.
- Переход на модели старшего поколения с увеличением времени обучения. Требует значительных временных затрат и мощностей GPU.
- Увеличение количества вариантов разметки. Это требует привлечения большего числа специалистов в области медицины и времени на тестирование.
- Добавление функции мультиклассовой детекции для распознавания типа инородного объекта. Требует более широкой выборки данных и более длительного обучения.