



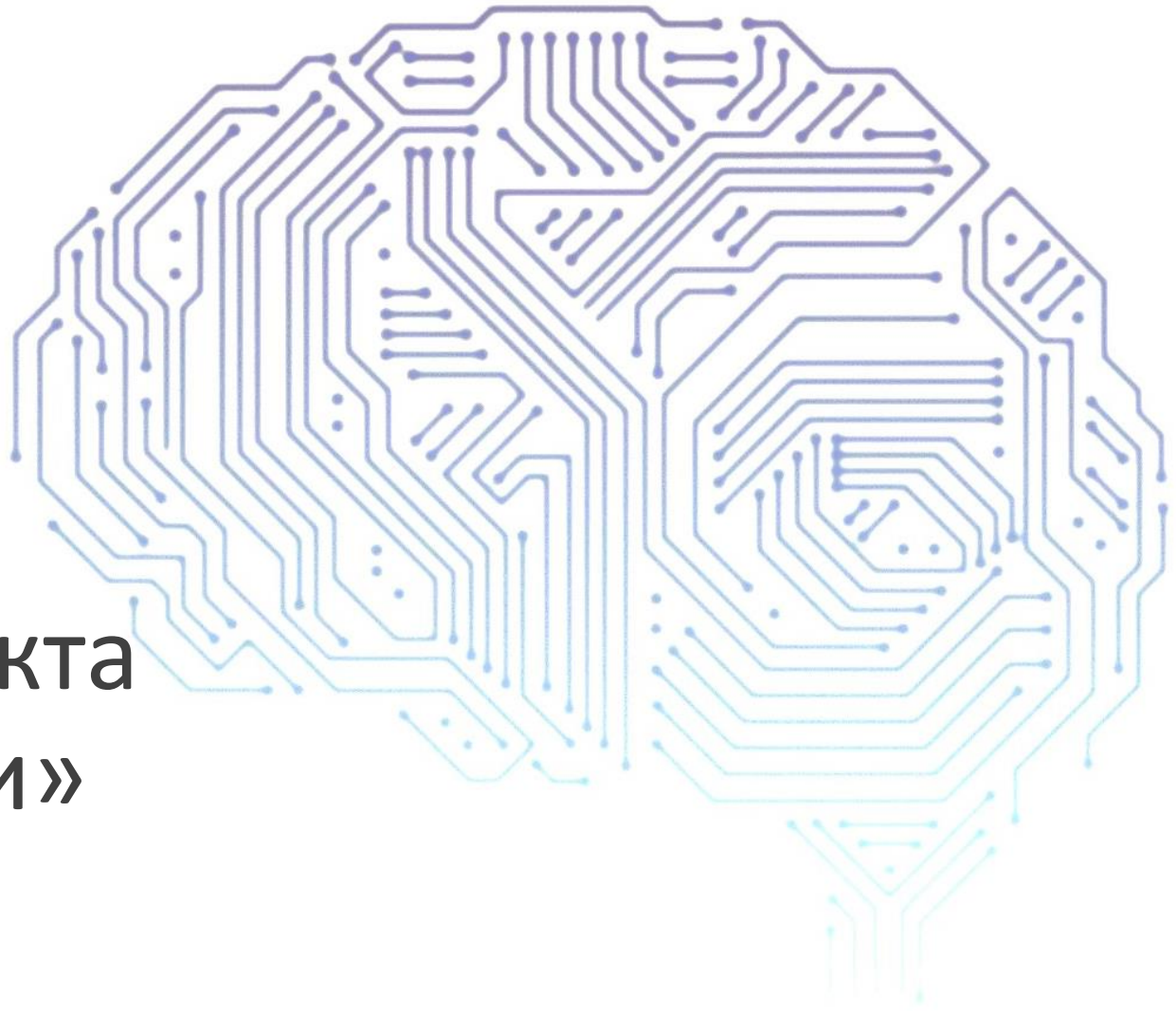
НМИЦ
ОНКОЛОГИИ
им. Н.Н.Блохина

«Использование искусственного интеллекта в диагностике и лечении»

Программа повышения квалификации (36 ак. часов)
2026 г.

Шувалова Екатерина Викторовна

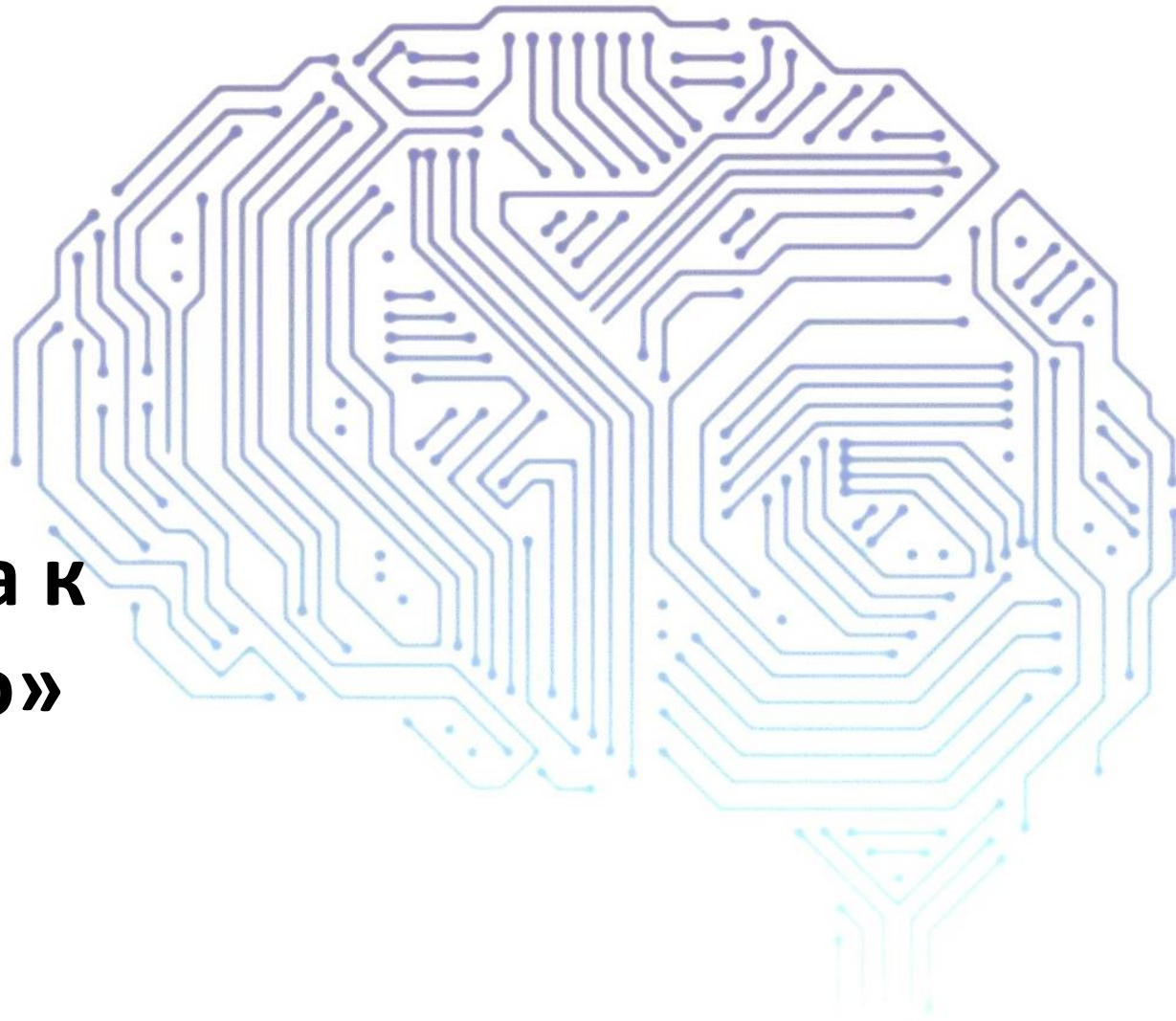
доцент, кандидат физико-математических наук





**НМИЦ
ОНКОЛОГИИ**
им.Н.Н.Блохина

Модуль 2. «Прикладные решения ИИ: от прототипа к клиническому внедрению»



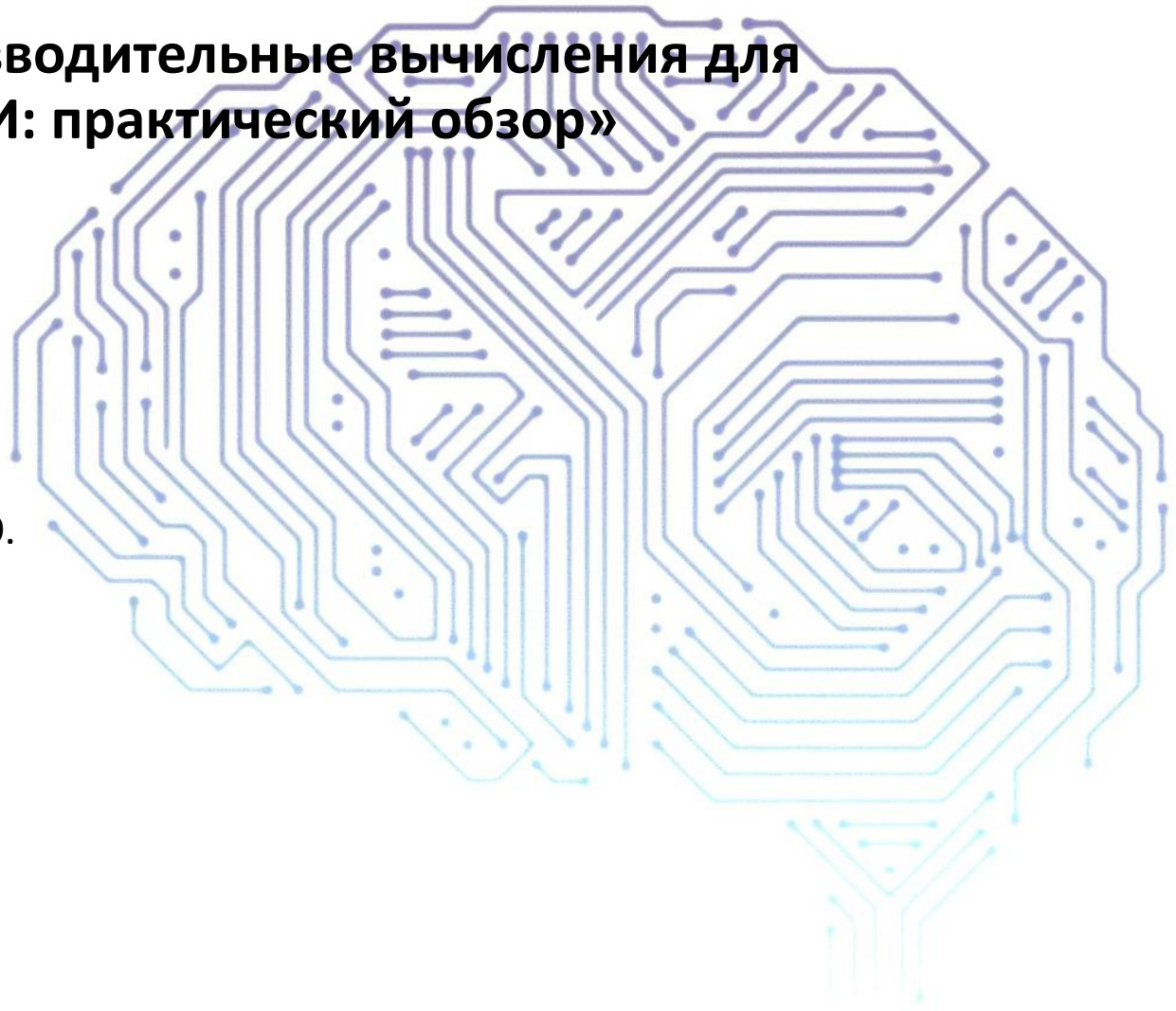
2.2 «YOLO. Высокопроизводительные вычисления для медицинского ИИ: практический обзор»



**НМИЦ
ОНКОЛОГИИ**
им.Н.Н.Блохина

2.2 «YOLO. Высокопроизводительные вычисления для медицинского ИИ: практический обзор»

- 1.Обнаружение объектов (YOLO) в медицине.
2. Практикум: подготовка данных для работы с YOLO.
3. Практикум: Выделение с помощью YOLO клеток
- 4.Роль GPU в медицинской диагностике
- 5.Гибридные платформы
6. Обзор RAPIDS (cuDF, cuML, cuGraph)
7. Библиотека Numba
- 8.No-code/Low-code (NVIDIA TAO Toolkit)
- 9.Сравнение CPU/GPU
- 9.Рекомендации по инфраструктуре в РФ





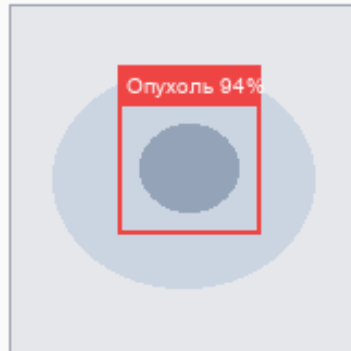
НМИЦ
ОНКОЛОГИИ
им.Н.Н.Блохина

2.2 «YOLO. Высокопроизводительные вычисления для медицинского ИИ: практический обзор»

Обнаружение объектов (YOLO) в медицинской диагностике

Применение моделей семейства YOLO для детекции патологий в реальном времени

1. Принцип Single-Shot детекции



You Only Look Once:
Поиск координат и класса
за один проход сети

Сферы применения в диагностике:

- Рак желудка по эндоскопии (YOLOv8)
- Переломы костей на рентгене (G-YOLOv11)
- Анализ и подсчет клеток в мазках крови
- Классификация и локализация опухолей MPT

Современные архитектуры:

ScaleMamba-YOLO, Elastic YOLO, Light-YOLOv11

Оптимизация под вычислительные среды:

- Глубокие модели критичны к ресурсам процессора
- Необходима адаптация под слабое/ограниченное железо
- Квантование (INT8/FP16) для ускорения инференса
- Развертывание легких версий прямо внутри контура МИС

Цель: Стабильный FPS на встроенных процессорах клиник

Преимущества для медицины:

- ☐ Сверхвысокая скорость обработки (работа 'в потоке')
- ☐ Одновременное нахождение десятков разнородных очагов
- ☐ Сквозная детекция без предварительной жесткой разметки ткани

Позволяет онкологу находить патологии прямо во время процедуры

Технологические вызовы и ограничения:

- ☐ Трудности с поиском мелких объектов (микрокальцинаты)
- ☐ Риск ложных срабатываний на размытых кадрах эндоскопии
- ☐ Сильная зависимость точности от калибровки и контраста камер

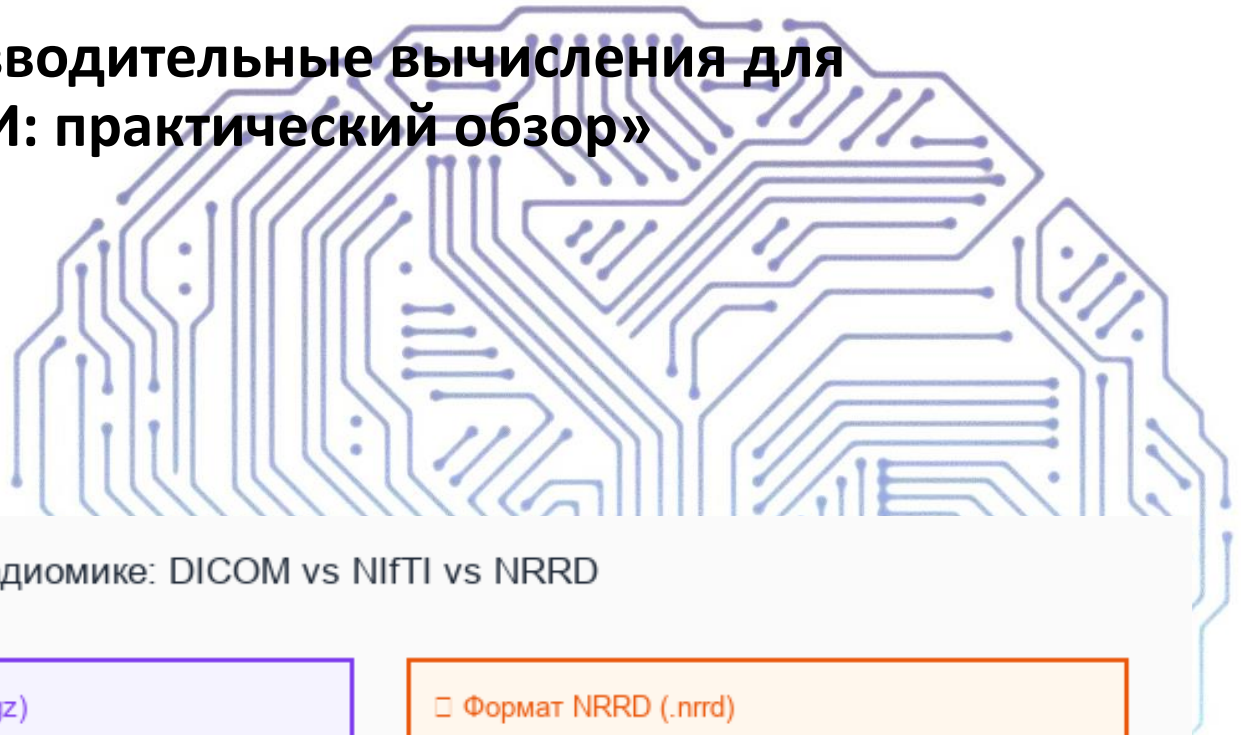
Решается предобработкой снимков и каскадным обучением



НМИЦ
ОНКОЛОГИИ
им.Н.Н.Блохина

2.2 «YOLO. Высокопроизводительные вычисления для медицинского ИИ: практический обзор»

Практикум: работа с YOLO.



Форматы медицинских данных в радиомике: DICOM vs NIfTI vs NRRD

❑ Формат DICOM (.dcm)

Digital Imaging & Communications in Medicine

- ❑ Главный стандарт госпитальных МИС/PACS.
- ❑ Каждый file (срез) хранит 2D-слой КТ/МРТ.
- ❑ Хранит ФИО пациента, ID, параметры сканера.
- ❑ Исследование разбито на сотни мелких файлов.
- ❑ Применение: Хранение на томографах.
- ❑ Идеален для ведения истории болезни
- ❑ Тяжело загружать пачкой в нейросети

❑ Формат NIfTI (.nii / .nii.gz)

Neuroimaging Informatics Technology Initiative

- ❑ Родной формат для Deep Learning (nnU-Net).
- ❑ Склеивает все 2D-срезы в один 3D/4D файл.
- ❑ Хранит только геометрию и аффинные матрицы.
- ❑ Анонимен: Персональные данные стерты.
- ❑ Применение: Обучение 3D-нейросетей.
- ❑ Один пациент = один файл, быстрая память
- ❑ Теряет клинический контекст и ID пациента

❑ Формат NRRD (.nrrd)

Nearly Raw Raster Data

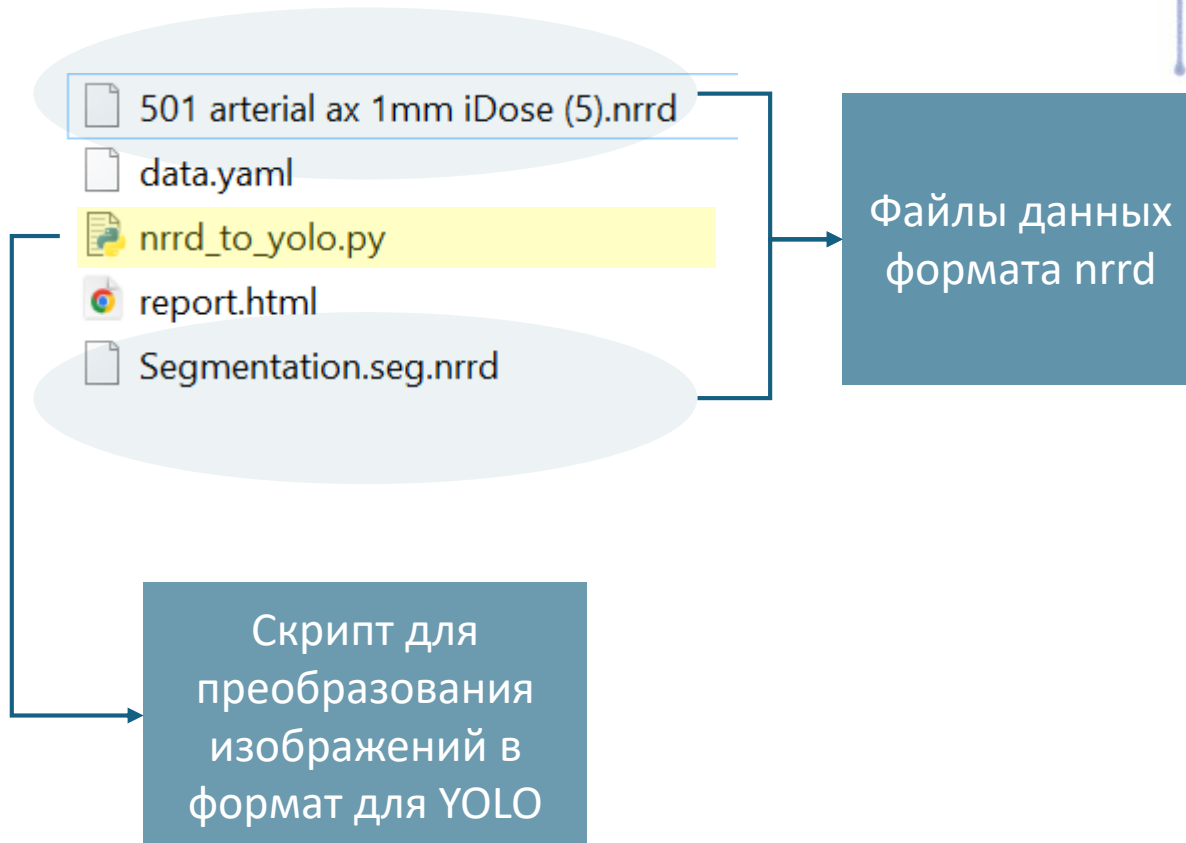
- ❑ Золотой стандарт для хранения масок разметки.
- ❑ Хранит N-мерные растровые массивы объектов.
- ❑ Имеет гибкий текстовый заголовок сверху файла.
- ❑ Отделяет метаданные от сырой пиксельной маски.
- ❑ Применение: Сохранение ROI в 3D Slicer / ITK.
- ❑ Идеален для хранения сложных масок органов
- ❑ Плохо поддерживается старыми выверами



НМИЦ
ОНКОЛОГИИ
им.Н.Н.Блохина

2.2 «YOLO. Высокопроизводительные вычисления для медицинского ИИ: практический обзор»

Практикум: подготовка данных для работы с YOLO.



```
nrrd_to_yolo.py x dia.py
4 import os
5 import re
6 import numpy as np
7 import nrrd
8 import pydicom
9 from pydicom.uid import ExplicitVRLittleEndian, generate_uid
10 from pathlib import Path
11 import matplotlib
12 matplotlib.use('Agg')
13 import matplotlib.pyplot as plt
14 from matplotlib.patches import Rectangle
15 from matplotlib.colors import ListedColormap
16 from scipy import ndimage
17 import cv2
```



НМИЦ
ОНКОЛОГИИ
им.Н.Н.Блохина

2.2 «YOLO. Высокопроизводительные вычисления для медицинского ИИ: практический обзор»

Практикум: работа с YOLO.

Архитектура пайплайна: Основной цикл по срезам (ось Z) и генерация YOLO-датасета

Шаг 1. Экстракция и подготовка среза

- Извлечение 2D-среза: По оси Z (от 0 до D-1)
- Транспонирование (.T): Приведение к (H, W)
- Сохранение dcm/: Датасет DICOM с метаданными

Контрастирование и Нормализация:

- Расчет 0.5 и 99.5 перцентилей + клиппинг
- Линейное масштабирование в $\rightarrow$ uint8

- Сохранение images/: PNG-срезы без разметки для детектора

Шаг 2. Обработка разметки и масок

- Поиск меток: Выделение уникальных label на слое
- Изоляция: Создание бинарной маски для каждого класса

Анализ связности (scipy.ndimage.label):

- Использование 8-связности пикселей
- Разделение класса на изолированные объекты obj_idx

- Вычисление Bounding Box: Мин/Макс по X и Y
- Алгоритм Дугласа-Пекера (cv2.findContours):
Аппроксимирует контур многоугольником (полигоном)

Шаг 3. Запись YOLO-файлов и верификация

- slice_XXXX.txt в images_labels/:
Формат: <class_id> <x_center> <y_center> <w> <h>
- slice_XXXX.txt в labels/polygons/:
Формат сегментации: <class_id> <x1> <y1> <x2> <y2> ...
- save_png_safe (cv2.imencode):
Бинарные PNG маски объектов для SAM (обход путей не-ASCII)

Визуальный контроль (images_result/):

- Полупрозрачная цветная заливка сегментации очагов
- Отрисовка рамок (bbox) и пунктирных полигонов контура

Generation конфигурации data.yaml

- Автоматически прописывает пути к изображениям и разметке выборки.
- Указывает количество классов (nc) и сопоставляет их имена (names).
- Результат: Полностью готовый конфигурационный файл для обучения YOLOv5 / YOLOv8 / YOLOv11.

Итоговый вывод скрипта в терминал

- Печать общего количества успешно обработанных 2D-срезов.
- Логирование точного числа сохраненных бинарных масок.
- Итог: Все файлы строго структурированы по целевым папкам без ручной сортировки.

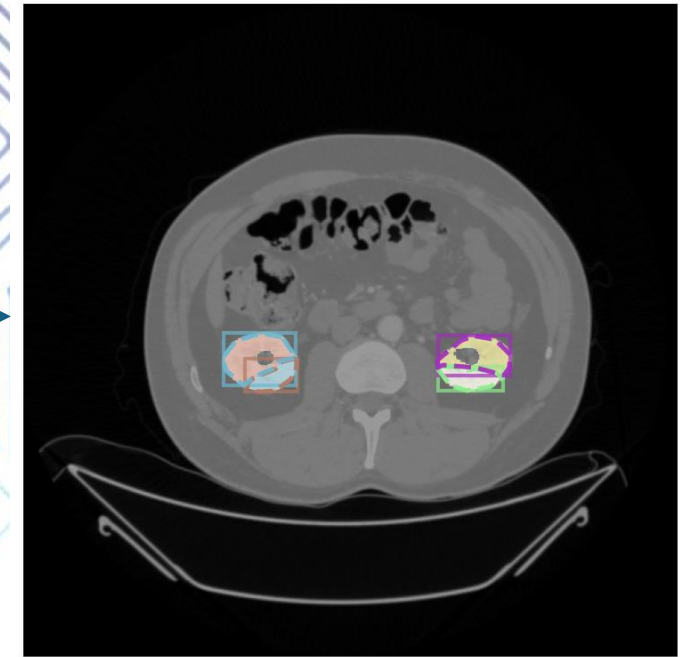
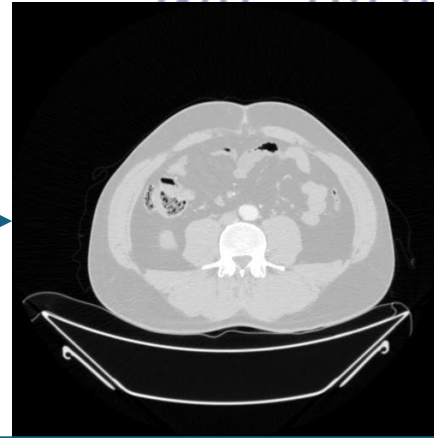


НМИЦ
ОНКОЛОГИИ
им.Н.Н.Блохина

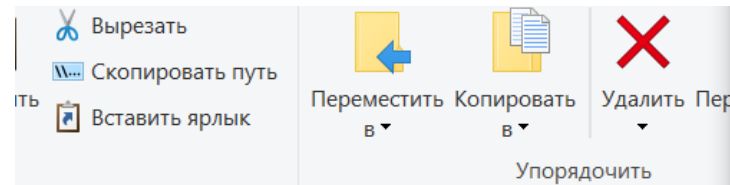
2.2 «YOLO. Высокопроизводительные вычисления для медицинского ИИ: практический обзор»

Практикум: работа с YOLO.

- dcm
- images
- images_labels
- images_result
- results
- 4_8473301259_zno_dataset.zip
- 501 arterial ax 1mm iDose (5).nrrd
- data.yaml
- nrrd_to_yolo.py
- report.html
- Segmentation.seg.nrrd
- K26_2_2.pptx



Slice 683 | BBoxes: 4 | Polys: 4



ютер > Data (D:) > blohina > 2026 > Тема 2_2 > images

Имя	Дата
polygons	18.06
predefined_classes.txt	18.06

slice_05/3.txt — Блокнот

Файл Правка Формат Вид Справка

8 0.354492 0.541992 0.017578 0.021484



НМИЦ
ОНКОЛОГИИ
им.Н.Н.Блохина

2.2 «YOLO. Высокопроизводительные вычисления для медицинского ИИ: практический обзор»

Практикум: работа с YOLO.

- dcm
- images
- images_labels
- images_result
- results
- 4_8473301259_zno_dataset.zip
- 501 arterial ax 1mm iDose (5).nrrd
- data.yaml
- nrrd_to_yolo.py
- report.html
- Segmentation.seg.nrrd
- K26_2_2.pptx

```
nrrd_to_yolo.py  dia.py  data.yaml x
! The file was loaded in a wrong en...  Reload in 'windows-1251'  Set project encoding to 'windows-1251'  Reload in another encoding  Indexing...
4  # Указываем на файлы со списками изображений для каждой выборки
5  train: train.txt
6  val: val.txt
7  test: test.txt
8
9  # Количество классов
10 nc: 12
11
12 # Названия классов (порядок должен строго соответствовать индексам 0-11 в ваших .txt файлах меток)
13 names:
14   - 'right_kidney_ID1'
15   - 'right_kidney_upper_segment_ID2'
16   - 'right_kidney_middle_segment_ID3'
17   - 'right_kidney_lower_segment_ID4'
18   - 'left_kidney_ID5'
19   - 'left_kidney_upper_segment_ID6'
20   - 'left_kidney_middle_segment_ID7'
21   - 'left_kidney_lower_segment_ID8'
22   - 'malignant_tumor_ID9'
23   - 'benign_tumor_ID10'
24   - 'cyst_ID11'
25   - 'abscess_ID12'
Document 1/1
```



НМИЦ
ОНКОЛОГИИ
им.Н.Н.Блохина

2.2 «YOLO. Высокопроизводительные вычисления для медицинского ИИ: практический обзор»

Практикум: работа с YOLO. Обучение YOLO для детекции патологий

Лабораторный практикум: Обучение модели YOLO (v8/v11) на медицинском датасете КТ-срезов

Шаг 1. Инициализация и запуск обучения

Установка пакета: `pip install ultralytics`

Входные данные: Готовые маски/bbox в `images_labels/`

Python скрипт инициализации:

```
from ultralytics import YOLO
model = YOLO('yolov8n.pt') # или s-seg
model.train(
    data='data.yaml', epochs=50,
    imgsz=256, batch=8, device='cuda'
)
```

Шаг 2. Аппаратное сравнение сред

Изменение параметра: `device='cpu'` vs `device='cuda'`

Режим Вычислений на CPU:

- Последовательная обработка тензоров процессором
- Результат: Длительное время обучения на эпоху

Режим Ускорения на GPU (CUDA):

- Параллельные вычисления на тысячах ядер видеокарты
- Результат: Кратное ускорение инференса в 10-50 раз

Шаг 3. Анализ метрик на тестовой выборке

Метрики валидации модели для обнаружения кист:

Precision (Точность): Доля истинных кист среди всех найденных
Recall (Полнота): Сколько реальных кист смог обнаружить ИИ
mAP50 / mAP50-95: Средняя точность с учетом перекрытия IoU

Настройка гиперпараметров для медицины:

- Аугментации: Отражения, аффинные сдвиги, шум
- Параметры: Размер КТ-изображения `imgsz=256`
- Оптимизация размера батча под память: `batch=8`

Практическая и учебная ценность проекта

- Работа с реальными анонимизированными медицинскими датасетами КТ/МРТ.
- Понимание аппаратных ограничений архитектур при разворачивании в больницах.

Итог: Формирование компетенций по созданию СППВР-модулей для лучевой диагностики.

Клиническая интерпретация результатов ИИ

Позволяет автоматизировать рутинный поиск мелких очагов онкологии.

Высокий Recall критичен: ИИ не должен пропустить скрытую патологию.

Применение: Сокращение времени описания снимка и минимизация человеческого фактора.



НМИЦ
ОНКОЛОГИИ
им.Н.Н.Блохина

2.2 «YOLO. Высокопроизводительные вычисления для медицинского ИИ: практический обзор»

Практикум: работа с YOLO. Обучение YOLO для сегментации патологий

Лабораторный практикум: Обучение модели сегментации YOLO-seg на полигонах медицинских КТ-масок

Шаг 1. Инициализация и запуск YOLO-seg

Входные координаты: Векторные полигоны в /polygons

Формат строки: <class_id> <x1> <y1> <x2> <y2> ...

Python скрипт обучения Instance-сегментации:

```
from ultralytics import YOLO
model = YOLO('yolov8n-seg.pt') # seg-архитектура
model.train(
    data='data.yaml', epochs=50,
    imgsz=256, batch=8, device='cuda'
)
```

Шаг 2. Визуализация и наложение масок

Пост-обработка: Восстановление пиксельного контура

Детекция (Bounding Box):

- Находит только прямоугольные границы очага
- Захватывает лишние здоровые пиксели по углам

Сегментация (Точный полигон):

- Идеально повторяет анатомическую форму кисты
- Накладывает полупрозрачную маску на КТ-срез

Шаг 3. Валидация и извлечение фичей

Валидация качества на тест-выборке:

Метрика mAP50-95 (box): Оценивает точность локализации рамок

Метрика mAP50-95 (mask): Оценивает качество попиксельного IoU

Сравнение: mAP(mask) точнее отражает реальный объем патологии

Связь с радиомикой и текстурами:

- Сгенерированная маска - это готовая область ROI
- Передача в PyRadiomics для расчета матриц GLCM
- Оценка энтропии и гетерогенности новообразования

Учебная ценность лабораторной работы

- Освоение продвинутого класса задач компьютерного зрения - Instance Segmentation.
- Математическое понимание IoU (Intersection over Union) для полигональных объектов.
- Итог: Переход от базового обнаружения к автоматическому оконтуриванию краев опухолей.

Клиническое применение результатов YOLO-seg

- Автоматический расчет истинного объема кисты/опухоли (в кубических мм).
- Точные контуры позволяют оценивать инфильтративный рост и изрезанность границ (Shape).
- Интеграция: Выходные маски используются для автоматического заполнения карт в МИС.



НМИЦ
ОНКОЛОГИИ
им.Н.Н.Блохина

2.2 «YOLO. Высокопроизводительные вычисления для медицинского ИИ: практический обзор»

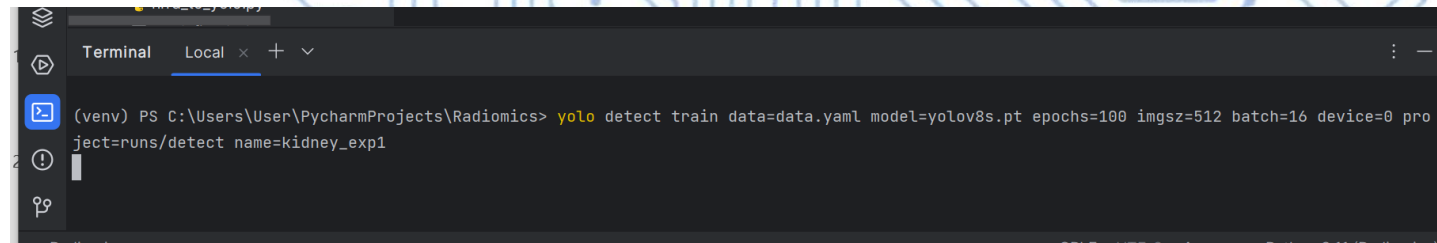
Практикум: работа с YOLO.

Имя

- dcm
- images
- images_labels
- images_result
- 501 arterial ax 1mm iDose (5).nrrd
- data.yaml
- experiment-10-06-2026(GPU).zip
- nrrd_to_yolo.py
- readme.txt
- seg_requirements.txt
- Segmentation.seg.nrrd
- K26_2_2.pptx

детектирование(readme.txt)

```
yolo detect train data=data.yaml model=yolov8s.pt  
epochs=100 imgsz=512 batch=16 device=0 project=runs/detect  
name=kidney_exp1
```



сегментация(seg_requirements.txt)

```
yolo segment train data=data.yaml model=yolov8s-  
seg.pt epochs=100 imgsz=512 batch=16 device=0  
project=runs/segment name=kidney_seg_exp1
```



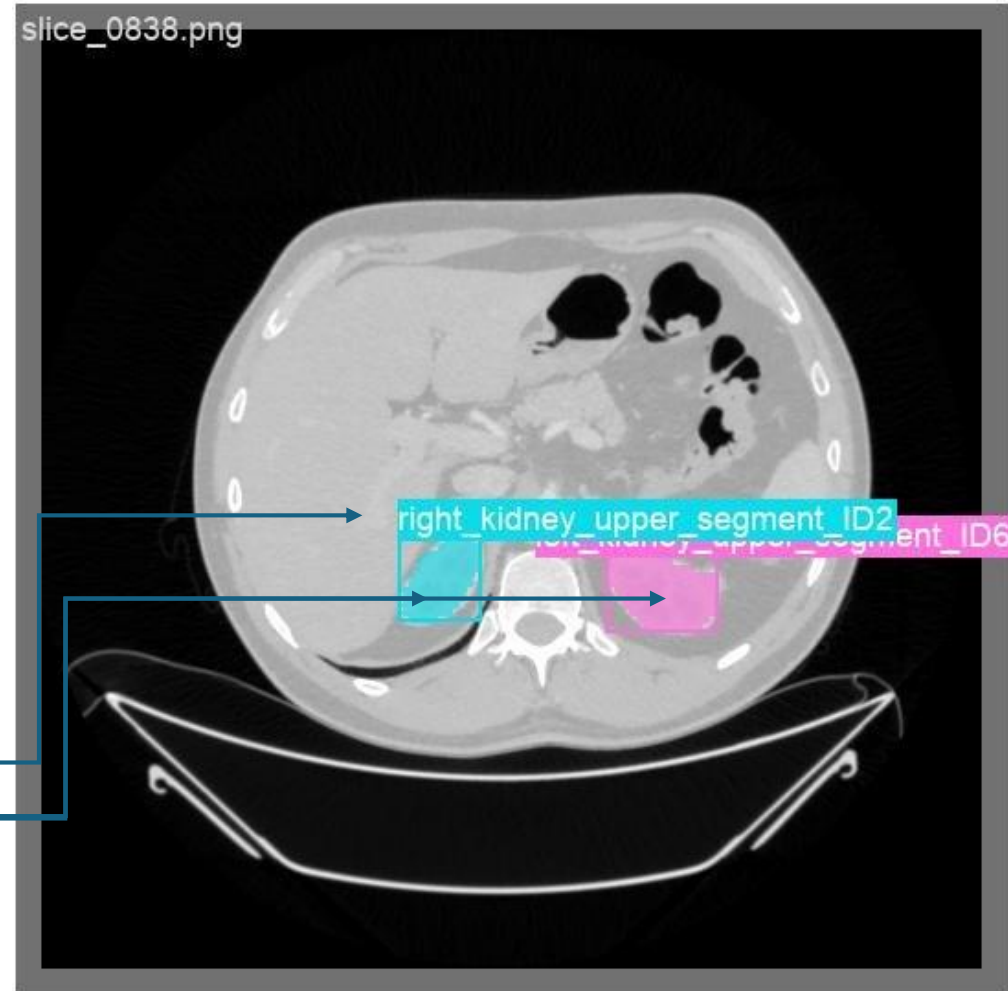
НМИЦ
ОНКОЛОГИИ
им.Н.Н.Блохина

2.2 «YOLO. Высокопроизводительные вычисления для медицинского ИИ: практический обзор»

Практикум: работа с YOLO. Результат: Детектирование+сегментация (наложение)

Имя

- dcм
- images
- images_labels
- images_result
- 501 arterial ax 1mm iDose (5).nrrd
- data.yaml
- experiment-10-06-2026(GPU).zip
- nrrd_to_yolo.py
- readme.txt
- seg_requirements.txt
- Segmentation.seg.nrrd
- K26_2_2.pptx





НМИЦ
ОНКОЛОГИИ
им.Н.Н.Блохина

2.2 «YOLO. Высокопроизводительные вычисления для медицинского ИИ: практический обзор»

Инференс (предсказание) с замером производительности

Лабораторный практикум: Пакетный инференс моделей YOLO и аппаратный бенчмаркинг (CPU vs GPU)

Шаг 1. Пакетная обработка КТ-срезов

Источник данных: Потокное чтение папки images/

Потоковый инференс: Проход по всем PNG файлам

Python скрипт логики прогнозирования:

```
results = model.predict(  
    source='images/', # путь к срезам  
    device='cuda',   # или 'cpu'  
    conf=0.25, save_txt=True  
)  
# Экспорт координат боксов и масок в JSON
```

Шаг 2. Точный хронометраж и расчет FPS

Вычисление perf_counter():

```
import time  
t_start = time.perf_counter()  
total_time = time.perf_counter() - t_start
```

Математический расчет производительности:

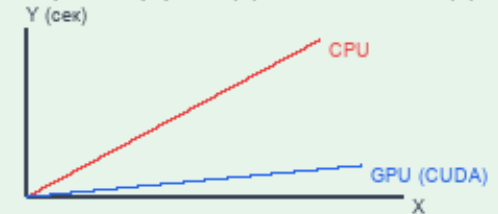
Формула FPS = (Количество снимков) / total_time

Структурирование результатов прогноза ИИ:

- Сохранение предсказанных масок и рамок в .JSON
- Экспорт визуализаций детекции в images_predict/

Шаг 3. Анализ аппаратной зависимости

График: Время инференса (Y) vs Число снимков (X):



Учебная ценность лабораторного практикума

- Понимание практической разницы в скоростях матричной обработки медицинских данных.
- Получение прикладных навыков оценки ресурсоемкости глубоких сверточных сетей.

Итог: Освоение методов профилирования кода и оптимизации инференса нейросетевых моделей.

Значение для систем реального времени (СППВР)

Наглядная демонстрация критической важности GPU видеокарт для больничных МИС.

Обработка 3D-серии КТ из 500 срезов на CPU занимает минуты, а на GPU - секунды.

Результат: GPU позволяет ИИ работать непосредственно в процессе сканирования пациента.



НМИЦ
ОНКОЛОГИИ
им.Н.Н.Блохина

2.2 «YOLO. Высокопроизводительные вычисления для медицинского ИИ: практический обзор»

Transfer learning и тонкая настройка (fine-tuning)

Лабораторный практикум: Концепция Transfer Learning (перенос знаний) для малых медицинских выборок

Шаг 1. Стратегия дообучения весов

Исходная точка: Использование предобученных моделей

- Базовые веса: Наборы COCO или репозитории MedAI
- Суть: Модель уже умеет извлекать линии, контуры и формы

Архитектура переноса знаний:

- [Замороженные слои (Вход)] -> Оставляем веса без изменений
- Экономит ресурсы процессора и страхует от забывания
- [Обучаемые слои (Выход)] -> Дообучение под КТ-снимки
- Настройка под новые классы патологий (кисты, опухоли)

Шаг 2. Конфигурирование гиперпараметров

Коррекция параметров обучения в Python:

Снижение скорости обучения (learning rate):

```
model.train(  
    data='data.yaml', epochs=15, # меньше эпох  
    lr=0.001, # уменьшенный шаг градиента  
    freeze=10 # заморозка первых 10 слоев  
)
```

Анализ стабильности градиентного спуска:

- Малый lr предохраняет от разрушения предобученных фичей
- Требуется значительно меньше эпох для адаптации

Шаг 3. Верификация темпов сходимости

Кривая обучения: Точность mAP (Y) vs Эпохи (X):



Учебная ценность изучения концепции Transfer Learning

- Глубокое понимание проблемы дефицита данных (Data Scarcity) в медицинском ИИ.
- Освоение техник тонкой настройки (Fine-Tuning) весов глубоких сверточных сетей.
- Итог: Формирование навыка эффективного обучения тяжелых моделей в условиях дефицита ресурсов.

Практическое значение для биомедицины

Решает главную проблему медицины: Сложность и дороговизну ручной разметки снимков. Обучение с нуля на малой выборке ведет к переобучению и зашумиванию шумов томографа. Результат: Перенос знаний позволяет достигать диагностической точности даже на редких патологиях.



НМИЦ
ОНКОЛОГИИ
им.Н.Н.Блохина

2.2 «YOLO. Высокопроизводительные вычисления для медицинского ИИ: практический обзор»

Аугментация данных и её влияние на качество

Аугментация данных и её влияние на качество обучения модели детекции клеток костного мозга

Шаг 1. Входной массив и модули обработки

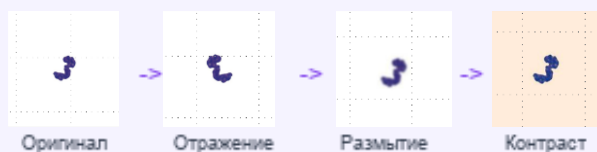
- Автоматически выделяет аннотированный объект (клетку)
- Пересчитывает метки координат под новые размеры
- Задаёт последовательность трансформаций кадра

Встроенные гиперпараметры YOLO:

- `hsv_h`, `hsv_s`, `hsv_v` — цветовые сдвиги и тон
- `flipplr`, `flipud` — зеркальные отражения (0.5)
- `scale`, `translate` — масштабирование и сдвиги

Шаг 2. Конвейер генерации новых изображений

Схема трансформации одного кадра (Размер 640x640):



Эксперимент: Сравнение конфигураций моделей:

- Модель 1: Без аугментаций — быстрая потеря точности на `val`
- Модель 2: Геометрическая — рост `mAP`, стабильный тренд
- Модель 3: Комплексная (Геометрия + Цвет + Размытие) — топ

Шаг 3. Анализ динамики сходимости и `mAP`

Влияние аугментаций на кривые обучения:



Учебная ценность лабораторного практикума

- Осознание роли генеративного расширения данных при дефиците уникальных снимков.
- Понимание механики регуляризации модели: аугментация работает как ограничитель.

Итог: Освоение инструментов автоматического контроля геометрии масок при мутациях снимка.

Значение для биомедицинской визуализации

Медицинские датасеты часто однородны: снимки сделаны на одном томографе/микроскопе. Без аугментаций ИИ запомнит конкретную яркость и контраст камер и откажет в других клиниках. Результат: Искусственный шум и сдвиги цвета делают ИИ устойчивым к артефактам разной оптики.



НМИЦ
ОНКОЛОГИИ
им.Н.Н.Блохина

2.2 «YOLO. Высокопроизводительные вычисления для медицинского ИИ: практический обзор»

Использование TAO Toolkit (low-code) для обучения

Лабораторный практикум: Применение No-code/Low-code инструментов (NVIDIA TAO Toolkit) в задачах медицинской детекции

Шаг 1. Импорт и трансформация датасета

Исходная разметка: YOLO формат (txt-файлы)

Требование TAO Toolkit: Международный стандарт COCO

Схема преобразования структуры:

[YOLO (.txt)] -> dcm2coco / Roboflow -> [COCO (.json)]

- Объединение всех независимых текстовых файлов в единую древовидную JSON-структуру аннотаций
- Формирование строгого словаря объектов: 'categories', 'images', 'annotations'

Шаг 2. Обучение в экосистеме Low-code

Развертывание среды через готовые Jupyter-ноутбуки:

Декларативная настройка (Spec Files):

Вместо ручного кодирования архитектуры задается текстовый файл конфигурации (протокол .txt), где прописываются эпохи, батчи и оптимизатор.

Обучение по одной CLI-команде:

- Команда: `tao model yolo_v8 train -e spec.txt`
- Автоматическое распределение нагрузки на GPU
- Логирование метрик и графиков 'из коробки'

Шаг 3. Оптимизация и экспорт весов

Подготовка модели к промышленному инференсу:

- Прунинг (Pruning): Удаление избыточных весов сети без потери диагностической точности детектора
- Команда экспорта: `tao model yolo_v8 export -m ...`

Кроссплатформенный формат:

Финальный экспорт в формат ONNX / TensorRT
Гарантирует запуск модели СППВР на любом клиническом сервере или встроенном железе

Учебная ценность работы с Low-code экосистемой

- Практическое знакомство с передовыми индустриальными инструментами промышленного ИИ.
 - Освоение концепции переносимости моделей через унифицированный JSON-стандарт COCO.
- Итог: Формирование навыка быстрого прототипирования систем детекции без ручного написания кода.

Применимость no-code/low-code в медицинских проектах

Ускоряет разработку: Позволяет развернуть пилотный ИИ-сервис за часы, а не за недели.
Доступность: Врачи-радиологи могут самостоятельно дообучать модели без глубоких ИТ-навыков.
Результат: Эффективный мост для быстрой интеграции исследовательских ИИ-моделей в контур МИС.



НМИЦ
ОНКОЛОГИИ
им.Н.Н.Блохина

2.2 «YOLO. Высокопроизводительные вычисления для медицинского ИИ: практический обзор»

Гибридный пайплайн: обучение на GPU + анализ признаков на CPU

Архитектура гибридного ИИ-пайплайна: Сквозная детекция на GPU (YOLO) + Текстурный анализ на CPU (PyRadiomics)

Этап 1. Локализация очагов на GPU

Потоковый инференс Computer Vision (YOLO):

- Входной поток: Пакетная загрузка PNG-срезов КТ
- Вычисления: Запуск модели на ядрах CUDA (GPU)

Генерация Bounding Boxes:

```
results = model.predict(source, device='cuda')
for box in results.bboxes:
    x1, y1, x2, y2 = box.xxyy # Координаты
    cls = box.cls             # Класс
```

Этап 2. Вырезание ROI и подготовка маски

Динамическое извлечение фрагмента ткани:



- Авто-вырезание (Crop) матрицы по координатам детектора
- Генерация временного бинарного файла маски для объекта
- Логический мост: Передача данных из видеокарты в ОЗУ

Этап 3. Экстракция текстур на CPU

Расчет математических признаков (PyRadiomics):

- Вычисления: Работа со сложными матрицами на CPU

Экстракция дескрипторов текстур:

```
extractor = featureextractor.RadiomicsFeatureExtractor()
features = extractor.execute(roi_img, roi_mask)
```

Извлекаемые матрицы признаков:

- GLCM (Contrast, Entropy) — перепады и хаос
- GLRLM / Shape — длина серий и геометрия краев

Учебная ценность реализации гибридных вычислений

- Практическое освоение гетерогенных архитектур (совместное использование мощностей GPU и CPU).
- Понимание принципов оптимизации: детекция 'в потоке' идет на GPU, тяжелый поликсельный анализ — на CPU.

Итог: Формирование компетенций по проектированию сложных многоступенчатых пайплайнов медицинского ИИ.

Экспорт результатов и клиническое значение

Автоматический экспорт: Все вычисленные признаки сохраняются в сводную таблицу .CSV.

Интерпретируемость: Врач получает не просто координаты рамки, а полный цифровой паспорт опухоли

Результат: Фундамент для построения комплексных прогностических моделей выживаемости больных в МИС.



НМИЦ
ОНКОЛОГИИ
им.Н.Н.Блохина

2.2 «YOLO. Высокопроизводительные вычисления для медицинского ИИ: практический обзор»

Оптимизация модели для инференса (экспорт в ONNX, TensorRT)

Лабораторный практикум: Промышленная оптимизация инференса моделей YOLO через форматы ONNX и NVIDIA TensorRT

Этап 1. Экспорт базовой модели PyTorch

Проблема: Зависимость от Python и фреймворка
Исходный формат: Веса PyTorch (.pt)

Команды конвертации Ultralytics:

```
from ultralytics import YOLO
model = YOLO('best.pt')

# Экспорт в кроссплатформенный ONNX
model.export(format='onnx')

# Или напрямую в TensorRT (.engine)
```

Этап 2. Промежуточный слой ONNX

Открытый формат Open Neural Network Exchange:

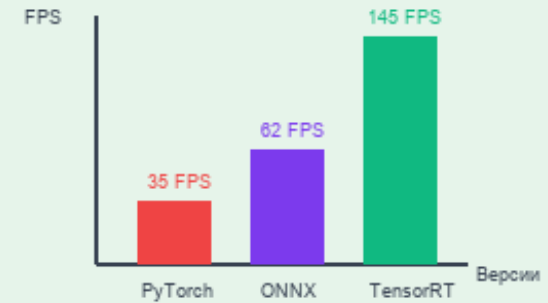
- Унификация: Переводит граф нейросети в единый формат
- Отвязка от кода: Больше не нужен исходный код на Python

Компиляция NVIDIA TensorRT:

- Слияние слоев (Layer Fusion) для экономии памяти
- Подбор оптимальных ядер под конкретную GPU
- Опциональное квантование (FP16 / INT8)
- Результат: Высокоскоростной файл-движок (.engine)

Этап 3. Сравнительный анализ скорости (FPS)

Сравнение пропускной способности модели:



Учебная ценность изучения оптимизации моделей

- Практическое знакомство с методами Low-Level компиляции графов глубоких нейросетей.
- Получение навыков ускорения вычислений без потери качества распознавания патологий.

Итог: Освоение экосистемы NVIDIA (TensorRT, RAPIDS) для перевода моделей из науки в продакшн.

Значение для систем реального времени (СППВР)

Ускорение инференса критически важно для работы ИИ во время потоковых процедур.

Пример: Высокий показатель FPS позволяет обрабатывать видеопоток эндоскопии 'на лету'.

Результат: Бесшовная интеграция тяжелых моделей ИИ в контур МИС без задержек и подвисаний.



НМИЦ
ОНКОЛОГИИ
им.Н.Н.Блохина

2.2 «YOLO. Высокопроизводительные вычисления для медицинского ИИ: практический обзор»

Распределённое обучение (если есть несколько GPU)

Лабораторный практикум: Распределенное обучение (DDP) моделей YOLO на нескольких GPU и анализ масштабируемости

Этап 1. Распределение батча (Data Splitting)

Принцип DDP: Репликация модели на каждую GPU

- На входе: Глобальный батч КТ-снимков (например, N=16)
- Разделение: Sampler делит батч поровну между картами

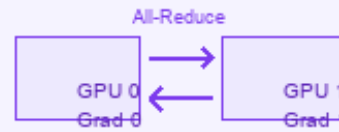
Мульти-GPU запуск в Ultralytics:

```
from ultralytics import YOLO
model = YOLO('yolov8n.pt')
# Передача списка ID видеокарт
model.train(data='data.yaml',
            epochs=50, device=[0, 1])
```

Этап 2. Механизм связи Ring All-Reduce

Параллельный Forward и Backward Pass:

- Каждая GPU считает потери на своей части КТ-слайдов



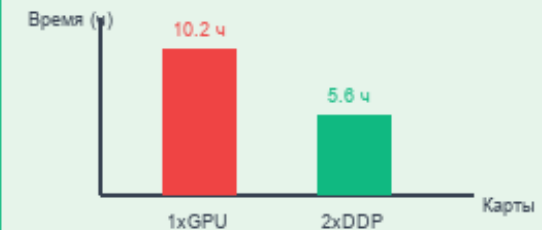
Почерпной обмен векторами градиентов:

- В отличие от DP, нет перегрузки мастер-карты (GPU 0)
- Градиенты усредняются по кольцу параллельно с бэквардом

Этап 3. Анализ ускорения и закон Амдала

Теоретический барьер масштабирования:

Закон Амдала: Ускорение = $1 / [S + (1 - S) / P]$
где S — последовательная часть кода, P — число ядер/GPU



Учебная ценность изучения распределенных систем (DDP)

- Глубокое понимание горизонтального масштабирования (Horizontal Scaling) нейросетевых архитектур.
- Изучение коммуникационных накладных расходов (Overhead) на синхронизацию весов через сеть/шин.

Итог: Освоение методов High-Performance Computing (HPC) для работы с большими массивами томограмм.

Значение для больших медицинских датасетов и МИС

Решает проблему терабайтных архивов: Обучение на национальных базах снимков требует недель на Закон Амдала на практике: Накладные расходы сети не дают линейного ускорения x2, реальная эффект
Результат: DDP позволяет оперативно обновлять веса ИИ-моделей в МИС при поступлении новых верифицированных

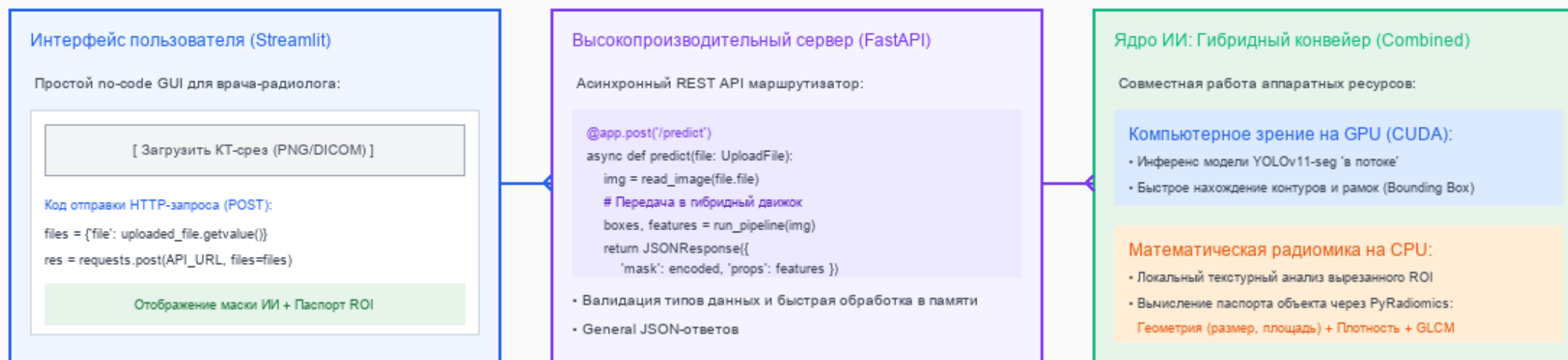


НМИЦ
ОНКОЛОГИИ
им.Н.Н.Блохина

2.2 «YOLO. Высокопроизводительные вычисления для медицинского ИИ: практический обзор»

Создание веб-демо с помощью FastAPI и Streamlit

Архитектура веб-сервиса СППВР: Интеграция интерфейса Streamlit и бэкенда FastAPI с гибридным ИИ-движком



Учебная ценность создания Full-Stack ИИ-сервиса

- Практическое прохождение полного цикла разработки (Full Lifecycle) биомедицинского сервиса.
- Получение прикладных навыков микросервисной архитектуры и клиент-серверного обмена через REST API.

Итог: Переход от разрозненных исследовательских скриптов к созданию готового программного продукта.

Клиническое значение готового веб-сервиса СППВР

Доступность: Врачу-клиницисту не нужно знать Python — он работает в удобном браузере на любом ПК

Комплексность: За одну загрузку томограммы врач получает и точный контур очага, и цифровой паспорт

Результат: Реальный прототип коммерческого модуля СППВР, готового к бесшовной интеграции в контур МИС.

Спасибо за внимание!



Шувалова Е.В.

Доцент, к.ф.-м.н.

Контакты:

ekaterina2504@ya.ru