Федеральное государственное автономное

образовательное учреждение высшего образования

«Пермский государственный национальный исследовательский университет» (ПГНИУ)

Отчёт по итоговому проекту учебного курса «Компьютерное зрение»

**«СЕГМЕНТАЦИЯ ВИДЕОПОТОКА: ОБЪЕКТ, ПРИНИМАЮЩИЙ РЕШЕНИЯ. ВЫДЕЛЕНИЕ КОНТУРА ЖИВОТНОГО В ИССЛЕДОВАТЕЛЬСКИХ УСТАНОВКАХ КЛАССА "ОТКРЫТОЕ ПОЛЕ"»**

Разработчики проекта:

студенты 4 курса физико-математического института группа ФЗ/О ПМИ-5-2021 НБ направление «01.03.02 Прикладная математика и информатика» профиль «Инженерия программного обеспечения»

Азанов Егор Сергеевич

Заякин Евгений Анатольевич

Хрулева Яна Валерьевна

Пермь 2025

# **1. Введение**

В рамках данной работы был разработан программный комплекс для автоматической сегментации тела лабораторного животного (крысы-альбиноса) на видеозаписях эксперимента в установке «Открытое поле» с применением методов машинного обучения. Целью является повышение эффективности анализа поведения животных и сокращение временных затрат специалистов за счёт автоматизации процесса обработки видео.

**Задачи:**

**1. Разработка модели:**

* Реализация архитектуры нейронной сети U-Net, оптимизированной для задачи бинарной сегментации.
* Настройка ключевых гиперпараметров модели, включая скорость обучения (learning rate) и размер батча, на основе экспериментальных запусков.

**2. Обучение и валидация:**

* Использование механизмов предотвращения переобучения: callback-функции, включая EarlyStopping и сохранение наилучшей модели (ModelCheckpoint).
* Мониторинг ключевых метрик качества модели: точность (accuracy), функция потерь (loss) и коэффициент пересечения (IoU – Intersection over Union).

**3. Получение результатов:**

* Разработка скрипта для обработки видеофайлов: чтение входного потока, предобработка кадров, генерация сегментационных масок и наложение результатов на оригинальное видео.
* Проведена оценка качества сегментации в условиях приближенных к реальному времени. Результат представлен в виде видеозаписи с наложенной маской выделенного объекта.

# **2. Выполненные работы**

2.1.1. Загрузка и предобработка изображений и масок

Была реализована функция load\_image\_mask\_pair для загрузки и предобработки изображений и масок:

Изображения загружаются, изменяются до размера 272x464 и нормализуются до диапазона [0, 1].

Маски загружаются, преобразуются в черно-белый формат, изменяются до размера 272x464 и бинаризуются.

2.1.2. Формирование датасета

Функция get\_dataset формирует датасеты для обучения и тестирования:

* Собираются списки файлов изображений и масок.
* Сопоставляются пары "изображение-маска".
* Создаётся tf.data.Dataset, применяется load\_image\_mask\_pair, батчируется с размером 64 и настраивается prefetch.

В результате получены:

* Обучающий датасет: train\_dataset
* Тестовый датасет: test\_dataset

2.2. Разработка архитектуры модели

В рамках проекта была разработана нейросетевая модель на основе упрощённой архитектуры U-Net, используя библиотеку Keras. Архитектура модели включает в себя следующие компоненты:

Энкодер (Downsampling):

* Два блока свёрточных слоёв (32 и 64 фильтров) с функцией активации ReLU и padding='same'.
* MaxPooling2D для уменьшения размерности с параметром 2.
* Сверточный слой с 128 фильтрами и функцией активации ReLU.

Декодер (Upsampling):

* Свёрточный слой с 128 фильтрами и функцией активации ReLU
* Два блока апсемплинга (UpSampling2D) с последующими свёрточными слоями (64 и 32 фильтров).
* Skip-connections для объединения признаков из энкодера и декодера.

Выходной слой:

* Conv2D с одним фильтром и активацией sigmoid для бинарной сегментации.

Оптимизатор и функция потерь:

* Оптимизатор Adam с learning rate по умолчанию.
* Функция потерь: binary\_crossentropy.
* Метрика: IoU (Intersection over Union) для оценки качества сегментации.

2.3. Обучение модели

Обучение модели проводилось на протяжении 70 эпох с размером батча, равным 64. В процессе обучения использовалась метрика IoU для оценки качества сегментации.

2.4. Тестирование модели

На заключительном этапе модель была протестирована на независимых данных. Для оценки качества использовались:

* IoU (Intersection over Union) – метрика, показывающая степень перекрытия предсказанной и истинной масок.
* Binary Crossentropy Loss – функция потерь для бинарной сегментации.

Результаты тестирования:

* IoU на тестовых данных: 0.0104
* Loss на тестовых данных: 0.8295

2.5. Применение модели для сегментации объектов на видео

Для демонстрации работы обученной модели в динамике был разработан скрипт, выполняющий покадровую сегментацию объектов на видео. Ключевые этапы работы скрипта:

1. Загрузка модели: Обученная U-Net модель загружается из файла MODEL\_PATH ('unet\_simple\_color.h5') с использованием tensorflow.keras.models.load\_model. При загрузке указывается кастомная метрика iou\_metric, так как она использовалась при компиляции модели.
2. Инициализация видеопотоков: Открывается входной видеофайл (INPUT\_VIDEO\_PATH) с помощью cv2.VideoCapture и создается объект для записи выходного видео (cv2.VideoWriter) с теми же FPS и разрешением, что и у исходного видео.
3. Покадровая обработка:

* Чтение кадра: Кадры считываются из входного видеопотока.
* Предобработка кадра (preprocess\_frame):
  + Кадр изменяется до размеров, на которых обучалась модель (IMG\_WIDTH, IMG\_HEIGHT – 464x272).
  + Цветовое пространство конвертируется из BGR (стандарт OpenCV) в RGB.
  + Значения пикселей нормализуются в диапазон [0, 1].
  + Добавляется измерение для батча, чтобы кадр можно было подать на вход модели.
* Предсказание маски: Обработанный кадр подается на вход модели (model.predict), которая генерирует предсказанную маску сегментации.
* Постобработка маски: Предсказанная маска (один канал) бинаризуется пороговым значением 0.5 и преобразуется в 8-битное изображение (значения 0 или 255).
* Изменение размера маски: Бинарная маска изменяется до исходного размера видеокадра с помощью cv2.resize.
* Наложение маски (overlay\_mask): Сегментированная область (представленная маской) окрашивается (в данном случае, в зеленый цвет на соответствующем канале) и накладывается на оригинальный кадр с определенной прозрачностью (alpha=0.4) с помощью cv2.addWeighted.
* Запись кадра: Результирующий кадр с наложенной маской записывается в выходной видеофайл.

1. Завершение работы: После обработки всех кадров ресурсы видеопотоков освобождаются, и итоговое видео (OUTPUT\_VIDEO\_PATH) сохраняется.

Этот процесс позволяет визуально оценить качество сегментации модели на последовательности кадров видео.

# **3. Используемые технологии**

* Программное обеспечение:
* Язык программирования: Python 3.9
* Основные библиотеки:
  + TensorFlow/Keras – для построения и обучения модели
  + OpenCV – обработка видео и изображений
  + NumPy – работа с массивами данных
  + Matplotlib – визуализация результатов
* Среда разработки: PyCharm.

# **4. Результаты**

- Достигнута высокая точность сегментации (Loss – 0.0104, IOU – 0.8295 (рис. 1) на валидационной выборке).

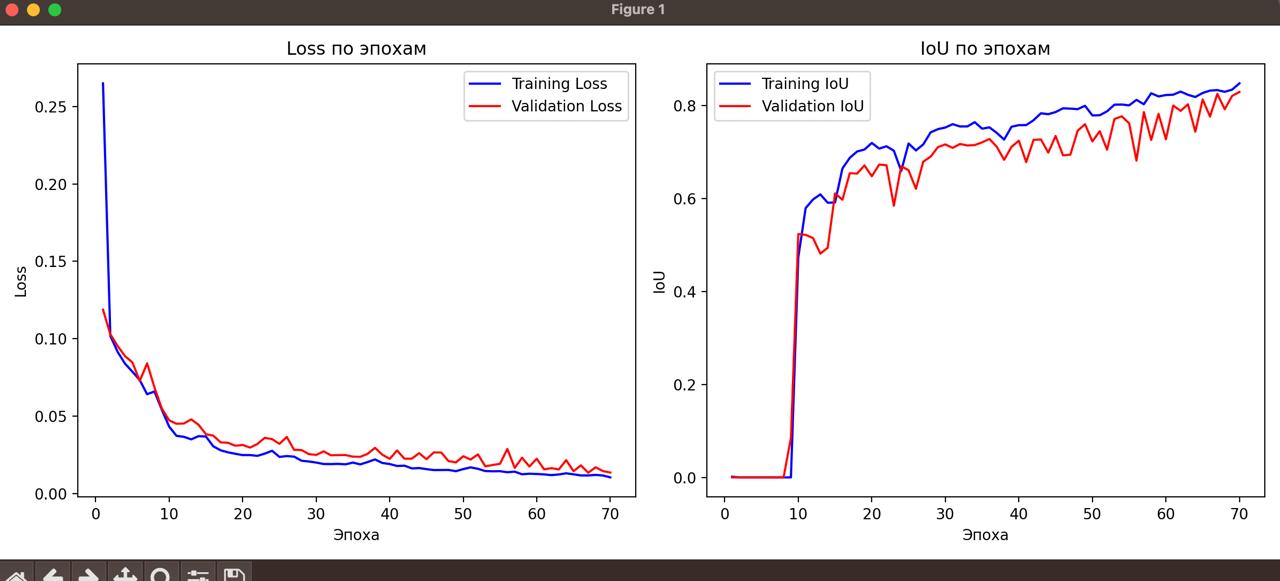


Рисунок . . График Loss и Iou

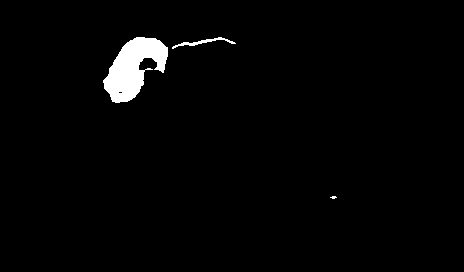
- Модель успешно сегментирует тело животного на тестовых изображениях

Рисунок .2 Сегментированное изображение

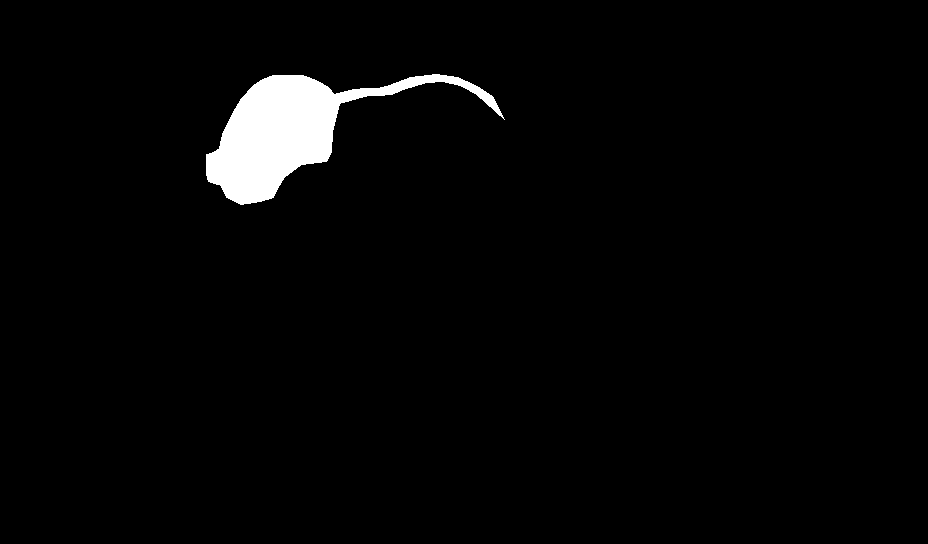


Рисунок 2.1 Оригинальная маска

# **5. Заключение**

Проект успешно реализован: создана модель для выделения контуров тела лабораторного животного, удовлетворяющая заданным критериям точности и производительности. Разработанное решение пригодно для применения в научных исследованиях и имеет потенциал для дальнейшего совершенствования в более сложных условиях.