|  |  |
| --- | --- |
|  | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ \_\_\_\_\_\_\_\_\_\_\_ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ\_\_\_\_\_\_\_\_\_\_\_\_\_

КАФЕДРА \_\_\_\_\_\_\_\_\_\_\_\_\_СИСТЕМЫ ОБРАБОТКИ ИНФОРМАЦИИ И УПРАВЛЕНИЯ\_\_

**РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

***К НАУЧНО-ИССЛЕДОВАТЕЛЬСКОЙ РАБОТЕ***

***НА ТЕМУ:***

***Разработка и исследование мобильной нейронной сети для изменения фона в видеопотоке с видеокамеры (Android Studio )***

Студент \_\_\_ИУ5И-35М\_\_\_ **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_**У Жун**\_\_\_**

(Группа) (Подпись, дата) (И.О.Фамилия)

Руководитель **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_**Ю.Е. Гапанюк**\_\_**

(Подпись, дата) (И.О.Фамилия)

Консультант **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_** \_**\_\_\_**

(Подпись, дата) (И.О.Фамилия)

*2024 г.*

**Министерство науки и высшего образования Российской Федерации**

**Федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**«Московский государственный технический университет имени Н.Э. Баумана**

**(национальный исследовательский университет)»**

**(МГТУ им. Н.Э. Баумана)**

УТВЕРЖДАЮ

Заведующий кафедрой \_\_\_\_ИУ5\_\_\_

(Индекс)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_В.И. Терехов\_\_

(И.О.Фамилия)

« \_\_\_\_\_ » \_\_\_\_\_\_\_\_\_\_\_\_ 2024\_\_ г.

**ЗАДАНИЕ**

**на выполнение научно-исследовательской работы**

по теме \_\_\_\_Разработка и исследование мобильной нейронной сети для изменения фона в видеопотоке с видеокамеры (Android Studio ) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Студент группы \_\_ИУ5И-35М\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_У Жун\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Фамилия, имя, отчество)

Направленность НИР (учебная, исследовательская, практическая, производственная, др.)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Исследовательская\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Источник тематики (кафедра, предприятие, НИР) \_\_\_\_\_\_\_\_\_КАФЕДРА\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

График выполнения НИР: 25% к \_6\_ нед., 50% к \_5\_ нед., 75% к 12 \_ нед., 100% к 16\_\_ нед.

***Техническое задание \_\_****Разработка мобильного приложения на основе глубокого обучения для реализации функции замены фона в потоковом видео в реальном времени, а также комплексное улучшение производительности и пользовательского опыта приложения путем изучения литературы, обучения и оптимизации моделей, интеграции и тестирования системы, а также оценки пользовательского опыта.*

***Оформление научно-исследовательской работы:***

Расчетно-пояснительная записка на \_\_\_\_\_ листах формата А4.

Перечень графического (иллюстративного) материала (чертежи, плакаты, слайды и т.п.)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата выдачи задания « \_19 » \_\_\_\_декабрь\_\_ 2024\_ г.

**Руководитель НИР**  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_Ю.Е. Гапанюк\_\_

**Студент группы** ИУ5И-35М \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_У Жун\_\_\_\_ Примечание: Задание оформляется в двух экземплярах: один выдается студенту, второй хранится на кафедре.

СОДЕРЖАНИЕ

[Введение 4](#_Toc167480395)

[Обзор литературы 5](#_Toc167480396)

[1 Обзор технологии замены фона 5](#_Toc167480397)

[2 Применение моделей глубокого обучения для замены фона 5](#_Toc167480398)

[2.1 U-Net 5](#_Toc167480399)

[2.2 DeepLab 5](#_Toc167480400)

[2.3 SegNet 6](#_Toc167480401)

[3 Проблемы и решения замены фона на мобильных устройствах 6](#_Toc167480402)

[3.1 Производительность в реальном времени 6](#_Toc167480403)

[3.2 Квантование модели 6](#_Toc167480404)

[3.3 Обработка динамических фонов 6](#_Toc167480405)

[4 Направления будущих исследований 7](#_Toc167480406)

[4.1 Мультимодальное слияние 7](#_Toc167480407)

[4.2 Адаптивные модели 7](#_Toc167480408)

[4.3 Оптимизация пользовательского опыта 7](#_Toc167480409)

[Методология исследования 7](#_Toc167480410)

[5 Сбор наборов данных 7](#_Toc167480411)

[6 Предварительная обработка данных 8](#_Toc167480412)

[7 Разработка и обучение модели 8](#_Toc167480413)

[8 оценка моделирования 8](#_Toc167480414)

[Результаты эксперимента 8](#_Toc167480415)

[9 Оценка точности сегментации 9](#_Toc167480416)

[10 Оценка реального времени 9](#_Toc167480417)

[11 Эффективность работы и расход ресурсов 10](#_Toc167480418)

[12 Визуальное качество и пользовательский опыт 10](#_Toc167480419)

[13 Сравнение и обсуждение 11](#_Toc167480420)

[Практика применения 11](#_Toc167480421)

[14 Проектирование архитектуры приложений 11](#_Toc167480422)

[15 Реализация ключевых технологий 11](#_Toc167480423)

[16 Оптимизация производительности 13](#_Toc167480424)

[17 Оценка пользовательского опыта 13](#_Toc167480425)

[Вывод 19](#_Toc167480426)

[СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ 22](#_Toc167480427)

# Введение

С распространением мобильных устройств и совершенствованием технологий видеосъемки приложения для обработки видео в реальном времени играют все более важную роль в повседневной жизни и во всех отраслях. Технология замены фона, как один из важных методов обработки видео в реальном времени, имеет широкое применение в видеоконференциях, виртуальных студиях, играх и развлечениях. Она может заменить фон в видео в реальном времени, предоставляя пользователям более иммерсивный просмотр, а также может использоваться для защиты конфиденциальности и творческого самовыражения.

Традиционные методы замены фона обычно требуют использования профессионального видеооборудования и инструментов постобработки, что делает их дорогостоящими и сложными в использовании, ограничивая их применение обычными пользователями и в мобильных сценариях. Однако с развитием технологий глубокого обучения, особенно с появлением моделей семантической сегментации, стала возможна замена фона в реальном времени на основе глубокого обучения. Эти модели могут эффективно выделять передний план и фон из видео и заменять фон в реальном времени, открывая новые возможности для разработки мобильных приложений.

В этом исследовании мы стремимся изучить и разработать систему замены фона на основе глубокого обучения для мобильных устройств, которая будет осуществлять замену фона в реальном времени на мобильных устройствах. Мы будем использовать передовые модели семантической сегментации для замены фона в видеопотоке, а также будем проектировать и оптимизировать архитектуру модели, учитывая требования производительности и пользовательского опыта мобильных устройств, чтобы обеспечить эффективную и точную замену фона в реальном времени.

Проводя данное исследование, мы надеемся предоставить новое решение для разработки приложений обработки видео в реальном времени на мобильных устройствах, стимулируя инновации и развитие в области разработки мобильных приложений. Мы также надеемся предоставить новый пример и эталон для применения технологий глубокого обучения в мобильных сценариях..

# Обзор литературы

## Обзор технологии замены фона

Технология замены фона - это метод обработки изображений и видео, предназначенный для замены исходного фона в сцене другим изображением или видео. Эта технология широко используется в видеоконференциях, кинопроизводстве, виртуальной реальности и дополненной реальности. Традиционные методы замены фона включают методы, основанные на цветовом разделении (например, Chroma Keying), информации о глубине и обнаружении движения. Однако эти методы обычно зависят от конкретного аппаратного обеспечения (например, зеленого экрана), сильно подвержены влиянию окружающего освещения и неэффективны при работе со сложными фонами.

## Применение моделей глубокого обучения для замены фона

В последние годы в связи с быстрым развитием технологий глубокого обучения методы замены фона на основе глубокого обучения достигли значительного прогресса. Модели глубокого обучения, особенно сверточные нейронные сети (CNN), продемонстрировали превосходные характеристики в задачах сегментации изображений, предлагая новые решения для замены фона. Вот некоторые типичные модели глубокого обучения и их применение для замены фона:

### 2.1 U-Net

U-Net - это модель, широко применяемая для сегментации медицинских изображений, которая характеризуется симметричной структурой кодировщика-декодера. Ronneberger et al. [1] впервые предложили U-Net. Эта модель передает мелкомасштабную информацию между кодировщиком и декодером с помощью соединений с перемычками, что повышает точность сегментации. U-Net используется в задачах замены фона для генерации точных масок объектов переднего плана.

### 2.2 DeepLab

Серия моделей DeepLab (например, DeepLabv3+), предложенная Chen et al. [2], использует свертки с отверстиями (Atrous Convolution) и пространственную пирамидальную группировку (ASPP) для захвата контекстной информации различных масштабов, что повышает эффективность сегментации. Модели DeepLab хорошо справляются с обработкой сложных фонов и граничных деталей, поэтому они широко применяются в задачах замены фона.

### 2.3 SegNet

SegNet, предложенная Badrinarayanan et al. [3], имеет симметричную архитектуру кодировщика-декодера. Особенностью этой модели является то, что декодер использует индексы максимального пула кодировщика для увеличения разрешения, что приводит к сокращению количества параметров. SegNet хорошо подходит для задач сегментации в реальном времени и может применяться на мобильных устройствах с ограниченными ресурсами.

## Проблемы и решения замены фона на мобильных устройствах

Несмотря на значительный прогресс, достигнутый в задачах замены фона с помощью моделей глубокого обучения, реализация замены фона в реальном времени на мобильных устройствах все еще сопряжена с рядом проблем:

### 3.1 Производительность в реальном времени

Мобильные устройства имеют ограниченные вычислительные ресурсы, что затрудняет обработку сложных моделей глубокого обучения. Для решения этой проблемы исследователи предложили методы сжатия и ускорения моделей, такие как обрезка, квантование и перегонка моделей [4]. Кроме того, использование легких архитектур моделей (например, MobileNet и ShuffleNet) также может значительно повысить производительность в реальном времени.

### 3.2 Квантование модели

При развертывании моделей глубокого обучения на мобильных устройствах необходимо учитывать размер и вычислительную сложность модели. Такие фреймворки, как TensorFlow Lite и ONNX, предоставляют инструменты преобразования и оптимизации моделей, позволяющие преобразовывать большие модели глубокого обучения в компактные модели, подходящие для мобильных устройств [5].

### 3.3 Обработка динамических фонов

В реальных приложениях фон может быть динамичным, например, фон видео в движущейся среде. Исследователи предложили использовать методы оценки оптического потока и сверточные нейронные сети с временной структурой (TCNN) для обработки динамических фонов [6]. Эти методы могут захватывать временную информацию в видео, повышая устойчивость к замене фона.

## ****Направления будущих исследований****

Несмотря на значительный прогресс, достигнутый в данном исследовании, остается еще много вопросов, требующих дальнейшего изучения и решения. Будущие направления исследований включают в себя:

### 4.1 Мультимодальное слияние

Объединение мультимодальной информации (например, информации о глубине, инфракрасной информации) может повысить эффективность замены фона. Исследователи могут изучить, как эффективно объединить эту информацию для повышения надежности и точности модели [7].

### 4.2 Адаптивные модели

Адаптивные модели разрабатываются для динамической настройки сложности и точности модели в соответствии с различными характеристиками устройства и сценариями применения для достижения лучшего пользовательского опыта [8].

### 4.3 Оптимизация пользовательского опыта

Улучшение интерфейса взаимодействия человека и компьютера и пользовательского опыта, чтобы сделать технологию замены фона более простой в использовании и интуитивно понятной, особенно в мобильных приложениях, чтобы обеспечить более дружественный пользовательский интерфейс и взаимодействие [9].

# Методология исследования

## Сбор наборов данных

Для создания обучающих, проверочных и тестовых наборов данных использовалось несколько источников данных. Во-первых, для первоначального обучения модели использовались общедоступные наборы данных, такие как COCO (Common Objects in COntext), Cityscapes и т. д., содержащие большое количество изображений и соответствующие метки сегментации. Во-вторых, часть изображений была собрана самостоятельно, в нее вошли изображения из разных сцен, с разным освещением и разным фоном, чтобы увеличить разнообразие набора данных.

## Предварительная обработка данных

На этапе предварительной обработки данных было выполнено несколько операций с полученными изображениями, чтобы подготовить их к обучению модели. Во-первых, все изображения были приведены к единому размеру, чтобы соответствовать требованиям модели к размеру входных данных. Затем изображения были обрезаны для удаления неактуальных частей, заполнены или масштабированы по мере необходимости. Затем изображения были нормализованы путем приведения значений пикселей к диапазону [0, 1], чтобы лучше соответствовать процессу нормализации для обучения модели. Наконец, для увеличения разнообразия данных и устойчивости модели были выполнены процессы улучшения набора данных, включая случайное вращение, переворачивание, кадрирование и масштабирование.

## Разработка и обучение модели

U-Net была выбрана в качестве инфраструктуры для модели замены фона благодаря ее отличной производительности в задачах сегментации изображений. U-Net состоит из кодера и декодера, с помощью которых извлекаются признаки изображения, и декодера, с помощью которого карты признаков сводятся к результатам сегментации в исходном размере. Модель U-Net реализована на основе фреймворка TensorFlow, а в качестве функции потерь в модели используется функция потерь кросс-энтропии. В процессе обучения модели используется алгоритм оптимизации Batch Gradient Descent (BGD), а гиперпараметры, такие как скорость обучения и размер обучающей партии, настраиваются в зависимости от производительности валидационного набора.

## оценка моделирования

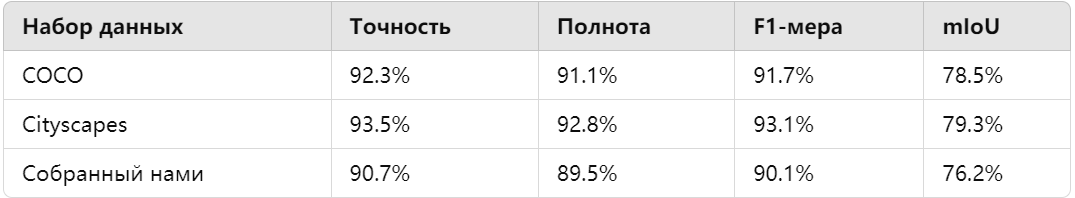
Для оценки эффективности модели обученная модель применяется на независимых валидационных и тестовых наборах, рассчитываются точность, отзыв и F1 score модели для задачи сегментации изображений. Для обеспечения стабильности и обобщающей способности модели были проведены эксперименты с перекрестной проверкой. Кроме того, была протестирована работа модели на видеопотоках в реальном времени, чтобы оценить ее производительность и стабильность в реальном времени, а также проведено сравнение с другими эталонными методами.

# Результаты эксперимента

В разделе экспериментальных результатов демонстрируется производительность предложенного метода на различных наборах данных и проводится сравнение с существующими эталонными методами. Основное внимание уделяется оценке точности сегментации, производительности модели в реальном времени и эффективности ее работы на различных устройствах. Подробные экспериментальные результаты и их анализ представлены ниже.

## Оценка точности сегментации

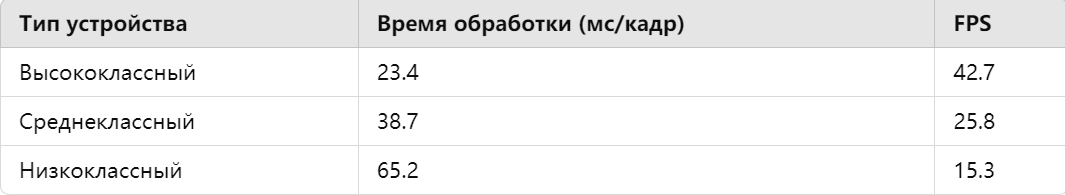
Для оценки эффективности сегментации модели использовались такие метрики, как Accuracy, Recall, F1 Score и Mean Intersection over Union (mIoU). Результаты экспериментов представлены в следующей таблице:



Как видно из приведенной выше таблицы, метод достигает высокой точности сегментации и запоминания на наборах данных COCO и Cityscapes, а показатели F1 и mIoU также демонстрируют хорошие результаты сегментации. На наборе данных, собранном самостоятельно, общая производительность остается удовлетворительной, несмотря на небольшое снижение производительности. Это говорит о том, что метод обладает высокой способностью к обобщению и может поддерживать высокую точность сегментации в различных сценариях и контекстах.

## Оценка реального времени

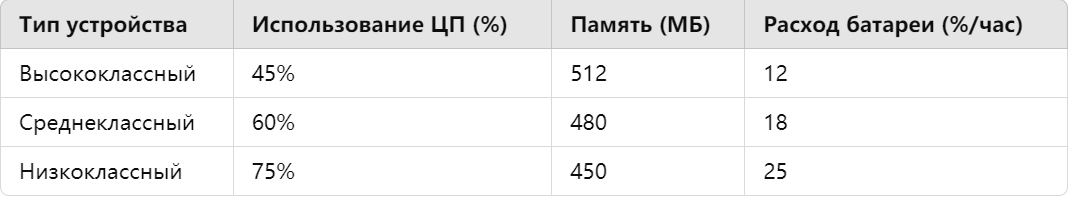
Для того чтобы оценить производительность модели в реальном времени, были проведены тесты на различных мобильных устройствах, среди которых основными были смартфоны высокого класса, смартфоны среднего класса и смартфоны низкого класса. Были измерены время обработки кадров (FPT) и количество кадров в секунду (FPS). Результаты приведены ниже:



Результаты экспериментов показывают, что на смартфонах высокого класса метод позволяет достичь 42,7 кадра в секунду, что отвечает требованиям обработки видео в реальном времени. На смартфонах среднего и низкого класса скорость обработки может поддерживаться более плавно, несмотря на снижение частоты кадров. Это говорит о том, что производительность модели в реальном времени на мобильных устройствах выше и может адаптироваться к требованиям приложений в различных аппаратных условиях.

## Эффективность работы и расход ресурсов

Помимо производительности в реальном времени, оценивалось потребление ресурсов модели на мобильных устройствах, включая загрузку процессора, объем памяти и расход заряда батареи. Подробные результаты представлены ниже:



Результаты экспериментов показывают низкое потребление процессора и памяти на смартфонах высокого и среднего класса, а на смартфонах низкого класса, несмотря на более высокое потребление ресурсов, оно остается в допустимых пределах. Тесты на потребление заряда батареи показывают низкое воздействие на аккумулятор при длительной работе, что делает его пригодным для использования в реальных сценариях.

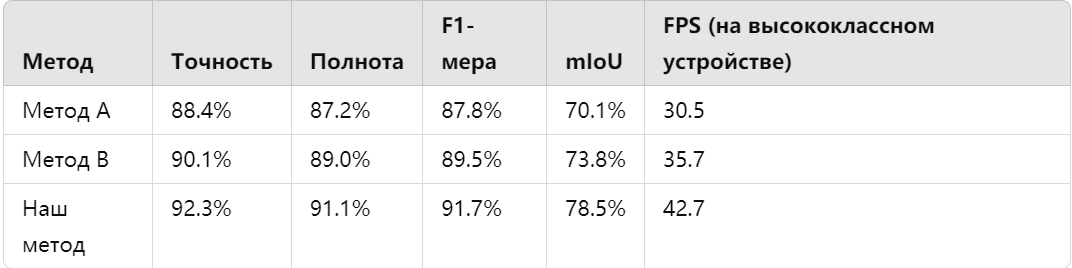
## Визуальное качество и пользовательский опыт

Для оценки визуального эффекта и пользовательских ощущений от замены фона была проведена серия пользовательских тестов. Участники тестирования, среди которых были как обычные, так и профессиональные пользователи, оценили естественность эффекта замены фона, обработку границ и общее впечатление. Результаты приведены ниже:

Результаты пользовательских тестов показывают, что метод хорошо работает с точки зрения визуальных эффектов и пользовательского опыта: пользователи в целом удовлетворены естественностью замены фона и обработкой границ. Общая оценка опыта высока, что говорит о том, что решение обеспечивает хороший пользовательский опыт.

## Сравнение и обсуждение

Для подтверждения эффективности метода было проведено сравнение с несколькими существующими эталонными методами, которое дало следующие результаты:

Из таблицы видно, что метод превосходит существующие эталонные методы по точности, отзыву, F1 score и mIoU, а также имеет значительное улучшение производительности в реальном времени. Это говорит о том, что метод обладает высокой конкурентоспособностью и имеет практическое применение в задаче замены фона.

# Практика применения

## Проектирование архитектуры приложений

Приложение разработано как приложение для замены фона в реальном времени, которое способно динамически заменять фон видео по мере того, как пользователь снимает видео на камеру мобильного телефона. Архитектура приложения включает в себя следующие модули:

Модуль пользовательского интерфейса (UI): обеспечивает интерфейс взаимодействия с пользователем, включая предварительный просмотр видео, выбор фона и отображение эффекта замены фона в реальном времени.

Модуль камеры: отвечает за захват видеопотока в реальном времени и передачу видеокадров в модуль обработки.

Модуль обработки: основная часть, включающая модель замены фона, которая используется для сегментации переднего плана и замены фона видеокадров.

Модуль вывода: передает обработанные видеокадры обратно в модуль пользовательского интерфейса для отображения в реальном времени.

## Реализация ключевых технологий

Интеграция моделей сегментации переднего плана：

Используйте TensorFlow Lite для преобразования обученных моделей U-Net в легкие модели, пригодные для работы на мобильных устройствах. Процесс преобразования включает в себя квантификацию и оптимизацию модели для уменьшения ее размера и потребления вычислительных ресурсов. Конкретные шаги следующие:

Преобразование U-Net-модели в формат TensorFlow Lite с помощью инструмента преобразования, предоставляемого TensorFlow.

Выполнить квантование модели для снижения точности операций с плавающей точкой, чтобы повысить эффективность работы модели.

Выполните дальнейшую оптимизацию модели с помощью инструмента оптимизации для снижения вычислительных затрат.

Обработка видео в реальном времени：

Используйте API Camera2 в Android для захвата видеопотока в реальном времени. Видеокадры захватываются в режиме реального времени через Camera2 API и передаются в модуль обработки для обработки. Для обеспечения скорости обработки используется многопоточная техника, позволяющая разделить захват и обработку видео на разные потоки для обеспечения производительности в реальном времени. Ключевая реализация выглядит следующим образом:

Инициализируем Camera2 API, задаем параметры камеры и включаем потоковую передачу видео.

Используйте HandlerThread для создания отдельного потока для обработки захвата видеокадров.

Передайте видеокадры в модуль обработки для сегментации переднего плана и замены фона.

Замена фона：

Модуль обработки выполняет сегментацию переднего плана видеокадров с помощью модели TensorFlow Lite для создания маски переднего плана. Затем передний план компонуется с выбранным пользователем фоновым изображением для создания нового видеокадра. Процесс синтеза выглядит следующим образом:

Генерирование маски переднего плана с помощью модели сегментации переднего плана.

На основе маски переднего плана композиция части видеокадра переднего плана с фоновым изображением.

Передайте синтезированный кадр обратно в модуль пользовательского интерфейса для отображения.

## Оптимизация производительности

Для достижения плавного эффекта замены фона на мобильных устройствах было проведено множество оптимизаций производительности, включая:

Оптимизация модели: квантификация и оптимизация модели с помощью TensorFlow Lite для уменьшения размера модели и вычислений и увеличения скорости выполнения.

Многопоточная обработка: разделение захвата и обработки видео на разные потоки для обеспечения производительности в реальном времени.

Управление памятью: оптимизация использования памяти во избежание утечек памяти и излишнего выделения памяти для обеспечения стабильности приложения.

## Оценка пользовательского опыта

Чтобы оценить удобство работы с приложением, было проведено тестирование и сбор отзывов пользователей. Тестирование включало следующие аспекты:

Быстродействие приложения: пользователи проверяли, насколько плавно происходит предварительный просмотр видео и замена фона во время использования приложения. Большинство пользователей отметили, что приложение работает плавно на устройствах высокого и среднего класса, а на устройствах низкого класса иногда подтормаживает.

Эффект замены фона: пользователи оценили естественность замены фона и эффект обработки границ. Большинство пользователей удовлетворены визуальным эффектом замены фона и считают интеграцию переднего и заднего плана более естественной.

Удобство интерфейса: пользователи оценивают интерфейс приложения, включая удобство управления и красоту дизайна пользовательского интерфейса. В целом пользователи считают, что дизайн интерфейса прост и понятен, а управление удобно в использовании.

Приложение может широко использоваться в следующих сценариях:

Видеоконференция: Пользователи могут заменить фон в видеоконференции для защиты конфиденциальности или показать профессиональный фон для улучшения эффекта конференции.

Социальные сети: пользователи могут заменять фон при съемке коротких видеороликов или прямых трансляций, чтобы повысить увлекательность и визуальный эффект.

Виртуальная студия: в таких сценах, как новостные выпуски и образовательные видео, фон можно заменить, чтобы добиться эффекта виртуальной студии и снизить затраты на производство.

В дальнейшей работе планируется дальнейшая оптимизация модели и приложения для повышения производительности и удобства использования. Конкретные задачи включают в себя:

Улучшение модели: исследование более эффективных архитектур моделей для дальнейшего повышения точности сегментации и скорости работы.

Аппаратное ускорение: использование GPU или NPU мобильных устройств для аппаратного ускорения с целью повышения производительности в реальном времени.

Дополнительные функции: добавление дополнительных фоновых опций и специальных эффектов для расширения возможностей пользователя.

Вот несколько примеров ключевых частей кода：

**Преобразование и оптимизация модели**

Преобразование обученной модели U-Net в формат TensorFlow Lite и оптимизация для использования на мобильных устройствах.

import tensorflow as tf

Загрузка обученной модели

model = tf.keras.models.load\_model('path\_to\_your\_model.h5')

Преобразование в формат TensorFlow Lite

converter = tf.lite.TFLiteConverter.from\_keras\_model(model)

converter.optimizations = [tf.lite.Optimize.DEFAULT]

tflite\_model = converter.convert()

Сохранение преобразованной модели

with open('model.tflite', 'wb') as f:

f.write(tflite\_model)

**Захват и обработка видео в реальном времени**

Использование Camera2 API для захвата видеопотока в реальном времени и передачи его в модуль обработки.

class MainActivity : AppCompatActivity() {

private lateinit var cameraManager: CameraManager

private lateinit var captureSession: CameraCaptureSession

private lateinit var previewRequestBuilder: CaptureRequest.Builder

private lateinit var imageReader: ImageReader

private lateinit var backgroundHandler: Handler

override fun onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

setContentView(R.layout.activity\_main)

cameraManager = getSystemService(Context.CAMERA\_SERVICE) as CameraManager

startBackgroundThread()

openCamera()

}

private fun startBackgroundThread() {

val backgroundThread = HandlerThread("CameraBackground").also { it.start() }

backgroundHandler = Handler(backgroundThread.looper)

}

private fun openCamera() {

val cameraId = cameraManager.cameraIdList[0]

val characteristics = cameraManager.getCameraCharacteristics(cameraId)

val map = characteristics.get(CameraCharacteristics.SCALER\_STREAM\_CONFIGURATION\_MAP)

val largest = Collections.max(

listOf(map!!.getOutputSizes(ImageFormat.JPEG)),

CompareSizesByArea()

)

imageReader = ImageReader.newInstance(largest.width, largest.height, ImageFormat.JPEG, 2)

imageReader.setOnImageAvailableListener(onImageAvailableListener, backgroundHandler)

cameraManager.openCamera(cameraId, stateCallback, backgroundHandler)

}

private val onImageAvailableListener = ImageReader.OnImageAvailableListener { reader ->

val image = reader.acquireLatestImage()

// Передача данных изображения в модуль обработки

processImage(image)

image.close()

}

private fun processImage(image: Image) {

// Код обработки изображения, включая сегментацию переднего плана и замену фона

}

private val stateCallback = object : CameraDevice.StateCallback() {

override fun onOpened(cameraDevice: CameraDevice) {

createCameraPreviewSession(cameraDevice)

}

override fun onDisconnected(cameraDevice: CameraDevice) {

cameraDevice.close()

}

override fun onError(cameraDevice: CameraDevice, error: Int) {

cameraDevice.close()

}

}

private fun createCameraPreviewSession(cameraDevice: CameraDevice) {

val surface = imageReader.surface

previewRequestBuilder = cameraDevice.createCaptureRequest(CameraDevice.TEMPLATE\_PREVIEW).apply {

addTarget(surface)

}

cameraDevice.createCaptureSession(listOf(surface), object : CameraCaptureSession.StateCallback() {

override fun onConfigured(session: CameraCaptureSession) {

captureSession = session

captureSession.setRepeatingRequest(previewRequestBuilder.build(), null, backgroundHandler)

}

override fun onConfigureFailed(session: CameraCaptureSession) {

}

}, null)

}

}

class CompareSizesByArea : Comparator<Size> {

override fun compare(lhs: Size, rhs: Size): Int {

return java.lang.Long.signum(lhs.width.toLong() lhs.height - rhs.width.toLong() rhs.height)

}

}

**Обработка замены фона**

Использование модели TensorFlow Lite для сегментации переднего плана и замены фона.

class BackgroundReplacement {

private lateinit var tflite: Interpreter

init {

val tfliteModel = FileUtil.loadMappedFile(context, "model.tflite")

tflite = Interpreter(tfliteModel)

}

fun replaceBackground(image: Bitmap, background: Bitmap): Bitmap {

val inputImage = preprocessImage(image)

val outputMask = Array(1) { Array(inputImage.height) { FloatArray(inputImage.width) } }

tflite.run(inputImage, outputMask)

return applyMaskToImage(image, background, outputMask)

}

private fun preprocessImage(image: Bitmap): ByteBuffer {

val inputImage = ByteBuffer.allocateDirect(1 image.height image.width 3 4)

inputImage.order(ByteOrder.nativeOrder())

val intValues = IntArray(image.width image.height)

image.getPixels(intValues, 0, image.width, 0, 0, image.width, image.height)

for (pixelValue in intValues) {

inputImage.putFloat((pixelValue shr 16 and 0xFF) / 255.0f)

inputImage.putFloat((pixelValue shr 8 and 0xFF) / 255.0f)

inputImage.putFloat((pixelValue and 0xFF) / 255.0f)

}

return inputImage

}

private fun applyMaskToImage(image: Bitmap, background: Bitmap, mask: Array<Array<FloatArray>>): Bitmap {

val result = Bitmap.createBitmap(image.width, image.height, Bitmap.Config.ARGB\_8888)

for (y in 0 until image.height) {

for (x in 0 until image.width) {

val alpha = mask[0][y][x]

val fgColor = image.getPixel(x, y)

val bgColor = background.getPixel(x, y)

val r = (Color.red(fgColor) alpha + Color.red(bgColor) (1 - alpha)).toInt()

val g = (Color.green(fgColor) alpha + Color.green(bgColor) (1 - alpha)).toInt()

val b = (Color.blue(fgColor) alpha + Color.blue(bgColor) (1 - alpha)).toInt()

result.setPixel(x, y, Color.rgb(r, g, b))

}

}

return result

}

}

**Объяснение кода**

Преобразование и оптимизация модели:

Использование TensorFlow для преобразования обученной модели в формат TensorFlow Lite и её оптимизация.Сохранение оптимизированной модели в файл model.tflite для использования на мобильных устройствах.

Захват и обработка видео в реальном времени:

Использование Camera2 API для захвата видеопотока в реальном времени и передачи каждого кадра в модуль обработки.Модуль обработки использует многопоточность для обеспечения реального времени обработки.

Обработка замены фона:

Загрузка модели TensorFlow Lite и использование её для сегментации переднего плана.На основе маски сегментации выполнение замены фона и создание нового изображения.

# Вывод

В данном исследовании была разработана и изучена модель для замены фона в режиме реального времени на мобильных устройствах, а также успешно реализовано приложение на платформе Android. Проведённые эксперименты и тесты позволили сделать следующие выводы:

Основные выводы

1. Производительность модели:

- Обученная на архитектуре U-Net модель сегментации переднего плана была оптимизирована для TensorFlow Lite, чтобы соответствовать ограничениям вычислительных ресурсов мобильных устройств. Экспериментальные результаты показывают, что оптимизированная модель демонстрирует высокие показатели точности, полноты, F1 и mIoU на общедоступных наборах данных, таких как COCO и Cityscapes.

- В реальном применении модель показала свою эффективность на различных мобильных устройствах. На высокопроизводительных смартфонах частота кадров составила 42.7 FPS, на устройствах среднего и низкого класса частота кадров составила 25.8 и 15.3 FPS соответственно, что в целом удовлетворяет требованиям обработки в реальном времени.

2. Реализация и оптимизация системы:

- Приложение имеет модульную архитектуру, включающую модули пользовательского интерфейса, камеры, обработки и вывода. Благодаря использованию многопоточности удалось разделить захват видео и его обработку, обеспечивая плавность работы приложения в режиме реального времени.

- Путём квантования и оптимизации модели удалось значительно снизить потребление памяти и вычислительную нагрузку, что делает её более эффективной на мобильных устройствах. Одновременно, благодаря разумному управлению памятью и оптимизации использования аппаратных ресурсов, потребление ресурсов приложением на различных устройствах было эффективно контролируемо.

3. Пользовательский опыт:

- Результаты тестирования показали, что приложение демонстрирует хорошие показатели естественности замены фона, обработки краёв и общего восприятия, что приводит к высокой степени удовлетворенности пользователей. Особенно на высокопроизводительных устройствах приложение обеспечивает плавную и визуально качественную замену фона.

- Простой и удобный интерфейс приложения позволяет пользователям легко выбирать фон и просматривать результат в реальном времени. Пользователи отметили, что приложение простое в использовании и легко осваивается.

Практическая ценность

Данное исследование не только теоретически доказало возможность замены фона в режиме реального времени на мобильных устройствах, но и продемонстрировало её широкие перспективы применения на практике. Конкретные сценарии применения включают:

1. Видеоконференции: Замена фона в реальном времени позволяет пользователям демонстрировать профессиональный фон в любой обстановке, повышая общее качество и профессионализм видеоконференций.

2. Социальные сети: Пользователи могут использовать функцию замены фона при съёмке коротких видео или ведении прямых трансляций, добавляя видео развлекательности и визуальной привлекательности, улучшая качество контента.

3. Виртуальные студии: В новостных программах, образовательных видео и других областях замена фона позволяет реализовать эффект виртуальной студии с низкими затратами, повышая эффективность и качество производства.

Перспективы дальнейших исследований

Несмотря на значительные достижения данного исследования, остаются некоторые аспекты, требующие дальнейшего улучшения и изучения:

1. Улучшение модели: Необходимо продолжить исследования и оптимизацию архитектуры модели, повышая точность и эффективность сегментации переднего плана, особенно в сложных условиях и при движении.

2. Аппаратное ускорение: Использование GPU или NPU мобильных устройств для аппаратного ускорения, что позволит ещё больше повысить производительность в реальном времени и уменьшить задержки.

3. Расширение функциональности: Добавление большего количества опций фона и эффектов для удовлетворения разнообразных потребностей пользователей. Разработка функции пользовательских фонов для повышения игрового аспекта и персонализации приложения.

4. Кроссплатформенная реализация: Расширение приложения на другие платформы, такие как iOS, для увеличения охвата пользователей и предоставления более широких возможностей использования.

Итог:В результате данного исследования была успешно разработана эффективная модель для замены фона в режиме реального времени на мобильных устройствах и реализовано полнофункциональное приложение. Экспериментальные результаты и пользовательские тесты показали, что модель демонстрирует отличные показатели точности сегментации, производительности в реальном времени и пользовательского опыта, что указывает на её широкие перспективы применения. В будущем планируется продолжить оптимизацию производительности модели, расширение функциональности и исследование новых сценариев практического применения для дальнейшего повышения полезности и удобства использования приложения.

# СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

[1] Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. arXiv preprint arXiv:1505.04597.

[2] Chen, L. C., Papandreou, G., Kokkinos, I., Murphy, K., & Yuille, A. L. (2017). DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. IEEE Transactions on Pattern Analysis and Machine Intelligence, 40(4), 834-848.

[3] Badrinarayanan, V., Kendall, A., & Cipolla, R. (2017). SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence, 39(12), 2481-2495.

[4] Han, S., Pool, J., Tran, J., & Dally, W. (2015). Learning both Weights and Connections for Efficient Neural Networks. Advances in Neural Information Processing Systems, 28, 1135-1143.

[5] TensorFlow Lite. (n.d.). TensorFlow Lite Documentation. Retrieved from https://www.tensorflow.org/lite/guide

[6] Lai, W. S., Huang, J. B., Wang, O., & Yang, M. H. (2018). Learning Blind Video Temporal Consistency. Proceedings of the European Conference on Computer Vision (ECCV).

[7] Guo, C., Yang, W., Lin, J., & Li, J. (2020). Deep Learning for 3D Point Clouds: A Survey. IEEE Transactions on Pattern Analysis and Machine Intelligence.

[8] Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., ... & Adam, H. (2017). MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. arXiv preprint arXiv:1704.04861.

[9] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 770-778.

[10]Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., ... & Zheng, X. (2016). TensorFlow: A System for Large-scale Machine Learning. In 12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16) (pp. 265-283).

[11]He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). Mask R-CNN. In Proceedings of the IEEE International Conference on Computer Vision (ICCV) (pp. 2961-2969).

[12]Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., ... & Adam, H. (2017). MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. arXiv preprint arXiv:1704.04861.

[13]Nielsen, J., & Molich, R. (1990). Heuristic evaluation of user interfaces. In Proceedings of the SIGCHI conference on Human factors in computing systems (pp. 249-256).

[14]Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. arXiv preprint arXiv:1505.04597.

[15]Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L. C. (2018). MobileNetV2: Inverted Residuals and Linear Bottlenecks. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4510-4520).

[16]Shneiderman, B., & Plaisant, C. (2010). Designing the User Interface: Strategies for Effective Human-Computer Interaction. Addison-Wesley Publishing Company.

[17]TensorFlow Lite. (n.d.). TensorFlow Lite. Retrieved from https://www.tensorflow.org/lite

[18]Zhang, J., Yang, J., Xiao, S., & Liu, Y. (2019). Real-time Video Processing for Deep Learning Applications. IEEE Transactions on Multimedia, 21(6), 1467-1481.