ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
имени М. В. ЛОМОНОСОВА»

ФИЗИЧЕСКИЙ ФАКУЛЬТЕТ

КАФЕДРА МАТЕМАТИЧЕСКОГО МОДЕЛИРОВАНИЯ И ИНФОРМАТИКИ

КУРСОВАЯ РАБОТА

«Изучение различных методов сегментации и оценка их применимости к морфологическому анализу изображений»

Выполнил студент
202 Группы
Языков Юрий

Научный руководитель:
доктор физико-математических наук
Чуличков А.И.

МОСКВА
2025

1 Введение:

Аннотация:

В этой работе рассматриваются различные методы сегментации изображений на области примерно постоянного цвета. Результаты, полученные различными способами будут сравнены при помощи морфологического анализа изображений, а именно поиска отличий между изображениями, состоящих из областей постоянного цвета. Таким образом, мы, сравнив результаты с тем, что размечено человеком, оценим применимость этих методов для поиска отличий между изображениями.

Актуальность:

Задачи сегментации изображений остаются одними из наиболее актуальных в области компьютерного зрения. В настоящее время не существует модели, которая могла бы выдавать универсальную и подходящую для всех сфер сегментацию. Многие архитектуры местами показывают результат, к примеру, в задачах классификации по Imagenet, лучше чем человек, но их универсальность и способность к распознаванию, далека от человеческого зрения. Несмотря на меньшую универсальность, сегментация моделями в сравнении с человеком позволяет быстрее анализировать больший объем данных . А также интерес представляют методы морфологического анализа изображений, которые дают математическую интерпретацию изображений. Особый интерес вызывает подход Пытьева к сравнению изображений, который демонстрирует устойчивость к некачественной сегментации, а результатом является визуально понятное человеку отличие между изображениями . В этой работе будут приведены 4 методов сегментации 2 обычных и 2 нейросетевых.

Цель работы

Целью данной работы является реализация и сравнительный анализ различных подходов к сегментации изображений с использованием методов морфологического анализа.

Решаемые задачи:

1. Реализация различных методов по сегментации цветных изображений на области постоянного цвета.
2. Анализ и выделение лучшего метода сегментации при поиске отличий между изображениями.

2 Теоретическая часть работы

Основные задачи и математическая модель которую мы будем использовать для изображений

2.1 Математическая модель цветного изображения:

Основные представления о модели, и о том, что является изображением с точки зрения математики и как мы будем искать отличия между ними по форме.

2.1.1 Цветное изображение в математике

Цветное изображение может быть представлено как функция, отображающая координаты пикселей в пространстве векторов цвета.

$$f(x) = \sum_{j=1}^{N} a_j(x), \bar{\phi}_j \tag{1}$$

где x- координата изображения ,N-количество цветовых базисных векторов , a_j -яркость вектора,ϕ- базисный цветовой вектор. В вопросах компьютерного зрения превалирует система RGB для хранения цветов изображений, но в данной работе будет использована HSV система для хранения векторов цвета. Тогда

наше изображение можно записать как :

$$f(x) = V(x)\bar{\phi}_v + H(x)\bar{\phi}_H + S(x)\bar{\phi}_S \quad (2)$$

Также цветовое пространство HSV удобно интерпретировать в цилиндрической системе координат к примеру:

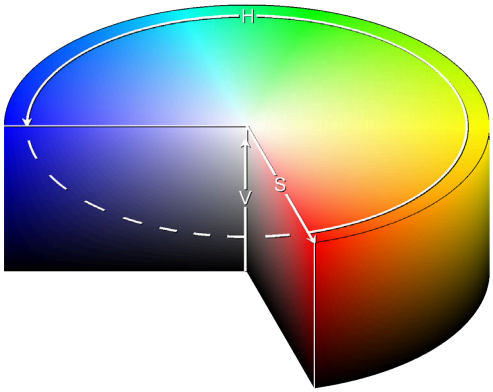

Рис.1 HSV цветовое пространство визуальная интерпретация.

Сегмент изображения определяется как множество координат, принадлежащих области, выделенной по заданному критерию (например, по цветовой однородности).Сегмент изображения задается как:

$$S(f, A) = f(x) * \chi(x, A) \quad (3)$$

,где χ это принадлежность к сегменту.

$$\chi(x, A) := \begin{cases} 1, & x \in A, \\ 0, & x \notin A. \end{cases}$$

2.1.2 Поиск отличий по форме

Введем такое значение как пересечение двух сегментов

$$I(A_i, B_j) = \chi(x, A_i) * \chi(x, B_j) \quad (4)$$

где A_i, B_j-сегменты изображений a,b Для формализации различий между сегментами изображений используется метрика, основанная на разности средних значений цвета в соответствующих областях. :

$$\Delta(x)_{ij} = \sum_{i=0}^{N} \sum_{j=0}^{M} [I(A_i, B_j)(g_A(x) - g_B(x)) + \chi(x, A_i)(g_A(x) - g_B(x)) \frac{\sum I(A_i, B_j)}{\sum \chi(A_i)}] \quad (5)$$

где $\Delta(x)_{ij}$-отличие между двумя сегментами. а g_A-среднее значение цвета по области

$$g_A(x) = \frac{\sum_{i=0}^{N} f(x_i) * \chi(x_i, A)}{\sum_{i=0}^{N} \chi(x_i, A)} \quad (6)$$

Новое изображение отличия всех сегментов изображения записывается как:

$$f'(x) = \sum_{i=0}^{N} \sum_{j=0}^{M} \Delta(x)_{ij} \quad (7)$$

такой подход позволяет наглядно показать отличие между формами изображений.

2.1.3 Ортогональная проекция

С целью минимизации различий, обусловленных освещением или контрастом, производится ортогональное проецирование одного изображения на другое. Это позволяет нивелировать эффекты, связанные с яркостными характеристиками снятого изображения. для этого посчитаем среднее значение по параметрам s,v в hsv пространстве и сделаем проекцию по яркости и контрасту изображения 1 на изображение 2 ;

$$H'_{f1}(x) = H_{f1}(x) * \frac{\sum_{i=0}^{N} H_{f2}(x_i) * M}{\sum_{j=0}^{M} H_{f1}(x_j) * N} \quad (7)$$

аналогично с цветовым вектором V.

2.2 Методы сегментации изображений на области примерно постоянного цвета

В данной работе использованы следующие методы : 1) Каждый пиксель изображения является областью постоянного цвета. 2) Сегментация изображения на области методом суперпикселей. 3) Сегментации на основе человеческого

зрения для валидации. 4) Метод кластеризации К-средних. 5) Свёрточная нейронная сеть обученная на поиск границ областей постоянного цвета. 6) Свёрточная нейронная сеть обученная для поиска областей постоянного цвета близких к цветовым векторам из заранее заданных.

2.2.1 Методы без нейронных сетей

2.2.1.1 Каждый пиксель область постоянного цвета

В этом подходе каждый пиксель рассматривается как отдельный сегмент. Соответственно, различия между изображениями вычисляются напрямую по пиксельным значениям.

2.2.1.2 Суперпиксели

Задача : реализовать алгоритм, который получает на вход изображение, а возвращает набор сегментов изображения.

Суперпиксели — это компактные группы пикселей, сохраняющие границы объектов на изображении. Они упрощают обработку, сокращая вычислительные затраты, и используются в задачах. В данной работе используется метод кластеризации изображения суперпикселями SLIC.

SLIC (Simple Linear Iterative Clustering) Метод, основанный на кластеризации **k-средних** в 5 мерном-пространстве (цвет LAB + координаты XY).

Алгоритм состоит из следующих шагов: 1 Инициализация центроидов: $C_k = [l_k, a_k, b_k, x_k, y_k]$ с шагом $S = \sqrt{\frac{N}{K}}$.

Для каждого пикселя вычисляется расстояние

$$D = \sqrt{d_{lab}{}^2 + \left(\frac{m}{S}\right)^2 d_{xy}^2} \tag{8}$$

d_{lab}- цветовое расстояние d_{xy}- геометрическое расстояние m- параметр компактности N - число пикселей, K - число суперпикселей. Новый центроид

считается таким образом:

$$C_k^{new} = \frac{1}{|S_k|} \sum_{i \in S_k} \begin{bmatrix} l_i \\ a_i \\ b_i \\ x_i \\ y_i \end{bmatrix}. \quad (9)$$

Критерий остановки алгоритма:

$$\|C_k^{new} - C_k^{old}\| < \epsilon. \quad (10)$$

2.2.1.3 Человеческое зрение

Сегментация, выполненная вручную на основе зрительного восприятия и семантической интерпретации сцен. Такой подход используется в работе в качестве эталонной разметки.

2.2.1.4 Метод К-средних

Популярный метод кластеризации для данных. Метод K-средних реализует кластеризацию на основе минимизации внутрикластерной дисперсии. Число кластеров K задаётся заранее, и центры обновляются с помощью итеративной процедуры минимизации.

$$V = \sum_{i=1}^{k} \sum_{x \in A} (x - \mu_i)^2 \quad (11)$$

В нашем случае мы кластеризуем цвета и выбираем K векторов цвета.

2.2.2 Нейронные сети

В этой работе для обучениях обеих моделей будем использовать архитектуру свёрточной нейронной сети UNET. В оригинальной, авторской статье 2015 года, UNET модель применялась для сегментации медицинских изображений, но модель также нашла применение и для сегментации другого рода изображений. Несмотря на давность релиза модели она все еще показывает хорошие результаты и на основе ее архитектуры создаются новые улучшенные версии. В

нашем случае мы возьмем ее классическую архитектуру из первой статьи и обучим модели.

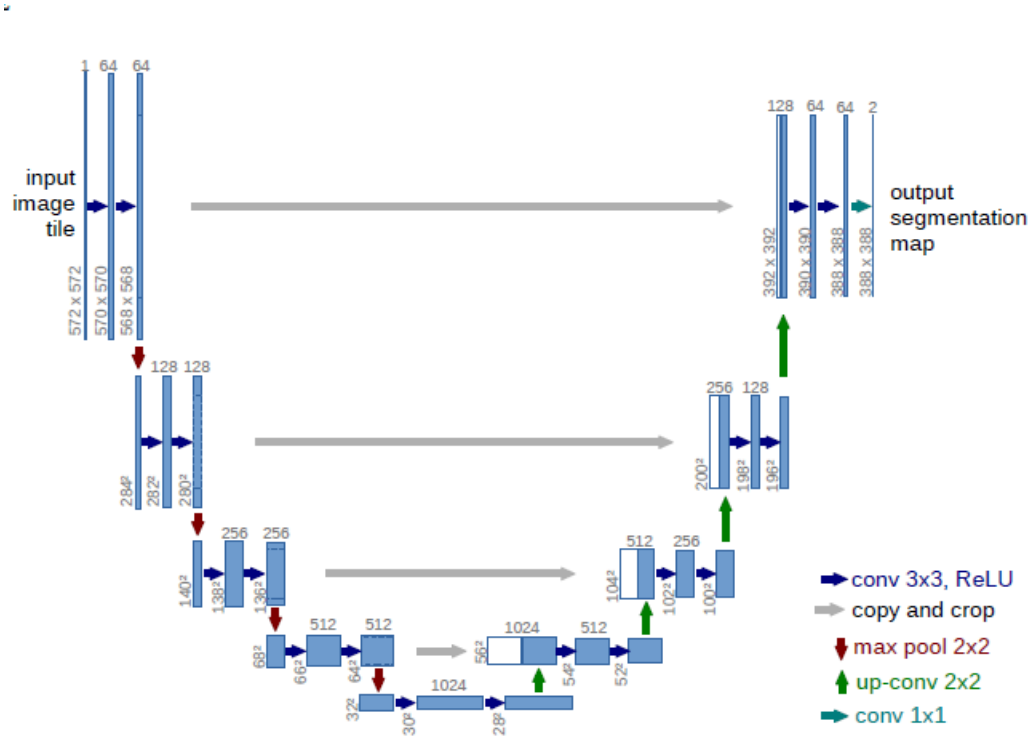

Рис. 2 Архитектура модели Unet.

2.2.2.1 Модель обученная на поиск границ

Задача : Получить модель, получающую на входе изображение и возвращающую бинарную маску с размеченными границами сегментов постоянного цвета.

Основная идея: Получив на выходе маску с границами инвертировать её, и получить области примерно постоянного цвета. После чего выделить отдельные сегменты, а также из-за, довольно большой толщины, границы тоже выделить как сегменты, но выбирать в сегмент не все что находится в границе, а только в пределах некоторого заранее заданного расстояния.

Обучающий датасет: Для обучения модели, сгенерируем синтетический датасет, состоящий из примитивов(многоугольники, элипсы, полосы) различного цвета, поверх которых добавлен шум по гауссу в качестве входных данных, а таргетами будет бинарная маска где выделены границы примитивов. Также при обучении будут использоваться линейные преобразования как поворот и отражения для лучшего обучения. Количество картинок в обучающем датасете 2900 примерно, размер 512x512x3.

Обучение: Для обучения моделей использовалась классическая архитектура U-Net с числом фильтров по уровням энкодера: 32,64, 128, 256,512,1024. Размер входного изображения — 512×512×3 (RGB).А итоговое количество параметров составило: примерно 51 млн параметров.

Для повышения устойчивости модели к различным преобразованиям применялись пространственные аугментации, сохраняющие связность объектов: случайные повороты изображения на произвольный угол и горизонтальные/ вертикальные отражения. Это позволяло модели лучше обобщать форму объектов независимо от их положения.

В качестве функции потерь использовалась **Dice Loss**, поскольку при сегментации нас в первую очередь интересует точность совпадение предсказанных и истинных масок:

$$\mathrm{DiceLoss}(y, p) = 1 - \frac{2 \cdot y \cdot p + \epsilon}{y + p + \epsilon} \tag{12}$$

где y — истинная маска, p — предсказанная, ϵ=0.001 — малая поправка, предотвращающая деление на ноль. Более подходящей была бы функция потерь, учитывающая непрерывность границ, однако Dice Loss показал приемлемые результаты.

Валидация производилась с использованием метрики **IoU (Intersection over Union)**, что позволяло объективно оценить качество перекрытия сегментированных областей с эталоном.

В качестве оптимизатора применялся **Adam**, так как он популярный, простой и обеспечивает быструю сходимость и стабильно работает на практике при обучении сверточных сетей. Скорость обучения (learning rate) устанавливалась на уровне 10^{-3} с последующим снижением при стабилизации функции потерь.

Также, для стабилизации обучения и ускорения сходимости, после каждой свёртки в сети применялась **Batch Normalization**, а входные изображения были предварительно нормализованы по каждому каналу, что ускоряло обучение и уменьшало влияние колебаний входных данных.

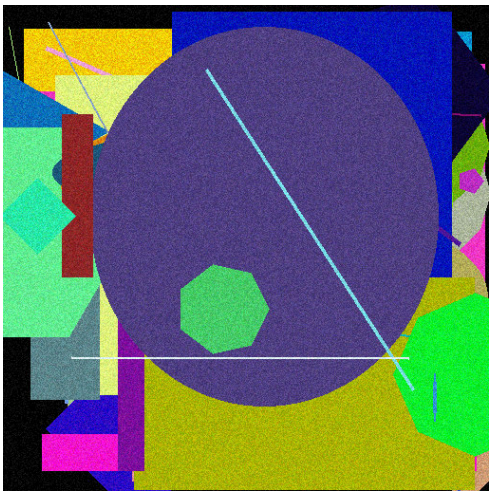

Рис.3 Пример изображения из обучающего датасета.

2.2.2.2 Модель обученная на поиск областей постоянного цвета

Задача : Получить модель , получающую на вход изображение и возвращающую маску с классами к которым принадлежат пиксели и чтобы это были области постоянного цвета.

Основная идея Выделить конечное количество векторов цвета и сегментировать изображение на области постоянного цвета где области будут иметь цвет одного из списка цветов. В этой работе я выделил 32 конечных вектора (27 цветных и 5 оттенков серого). По причине того что конечных цветовых векторов не так много, на выходе мы получим не слившиеся в один, отдельные области примерно постоянного цвета, при этом будет не так много отдельных областей на предметах которые можно было бы назвать областью одного постоянного цвета.

Рис.4 Заранее выделенные 32 цвета.

Обучающий датасет: Для обучения этой модели сгенерируем синтетический датасет, состоящий из примитивов(многоугольники, эллипсы , звезды), поверх

которых добавим шум по гауссу и градиент по SV(Яркость и контраст) координатам вектора цвета. Таргетом будет маска где на месте примитива будут находится метки класса ближайшего к исходному вектору цвета. Количество картинок в обучающем датасете примерно 2000 , размер 512x512x3

Обучение: Во многом повторяет предыдущую модель за исключением: Для обучения моделей использовалась классическая архитектура U-Net с числом фильтров по уровням энкодера: 64, 128, 256, 512, 1024 — аналогично оригинальной статье [1]. Размер входного изображения — 256×256×3 (RGB), при 512x512 обучение было слишком долгим и модель становилась слишком тяжелой. Число параметров примерно 50млн. Функция потерь diceloss для нескольких классов чтобы модель могла учитывать как область так и предсказание лишних классов в области.

$$\mathrm{DiceLoss}(y, p) = 1 - 2 * \frac{\sum_l \sum_i (y_i^l * p_i^l) + \epsilon}{\sum_l \sum_i (y_i^l + p_i^l) + \epsilon} \tag{13}$$

Остальная часть как в предыдущей модели

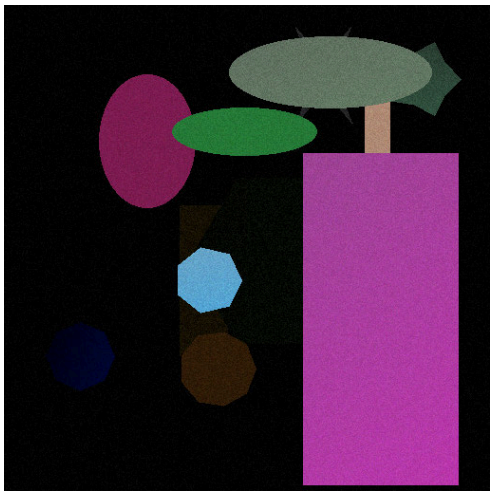

Рис. 5 Пример изображения из обучающего датасета.

3. Реализация методов

Был использован язык программирования python, а также ряд библиотек (numpy,albumentations,torch,cv2,PIL). Также для обучения модели была использована видеокарта с поддержкой CUDA.

3.1 Результат сегментации

3.1.1 Модель обученная на поиск границ

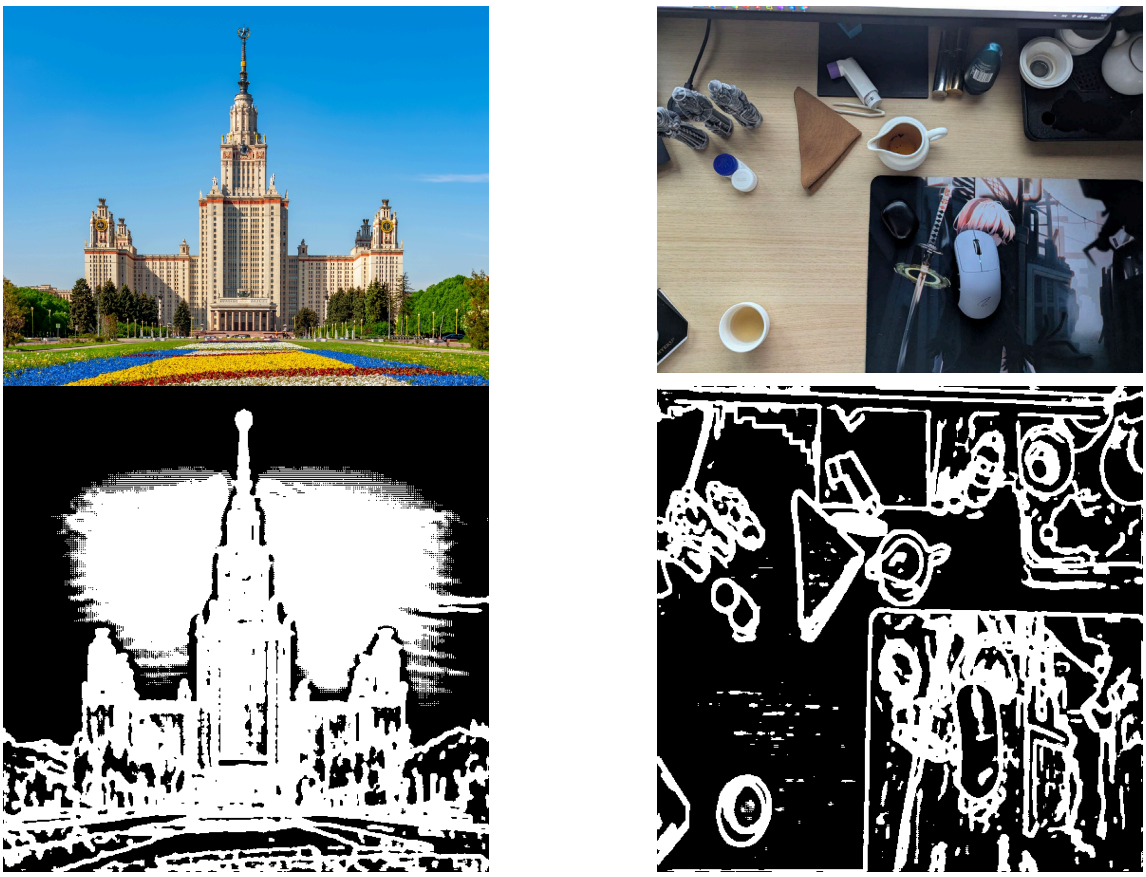

Рис. 6 Пример работы модели.

Как видно из результатов она демонстрирует низкое качество сегментации сложных изображений, с более простыми сценами модель справляется лучше, но для нашей цели выделяет немного лишние границы.

3.1.2 Модель с заранее выделенными векторами цвета

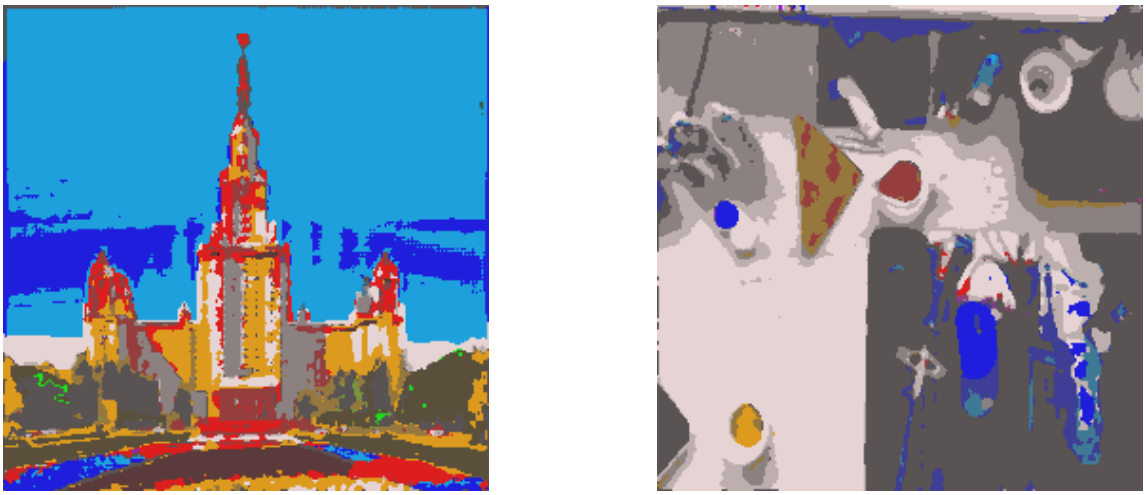

Рис. 7 Пример работы модели.

В случае сегментации на области постоянного цвета модель, в отдельных случаях модель некорректно определяет области постоянного цвета, к примеру небо на фото слева. На простых сценах справляется визуально хорошо.

3.1.3 Суперпиксели

Рис. 8 Пример работы метода.

Метод SLIC демонстрирует склонность к фрагментации границ объектов. Полученные суперпиксели нередко теряют структурную целостность, особенно при наличии текстур или шумов.

3.1.4 K средних

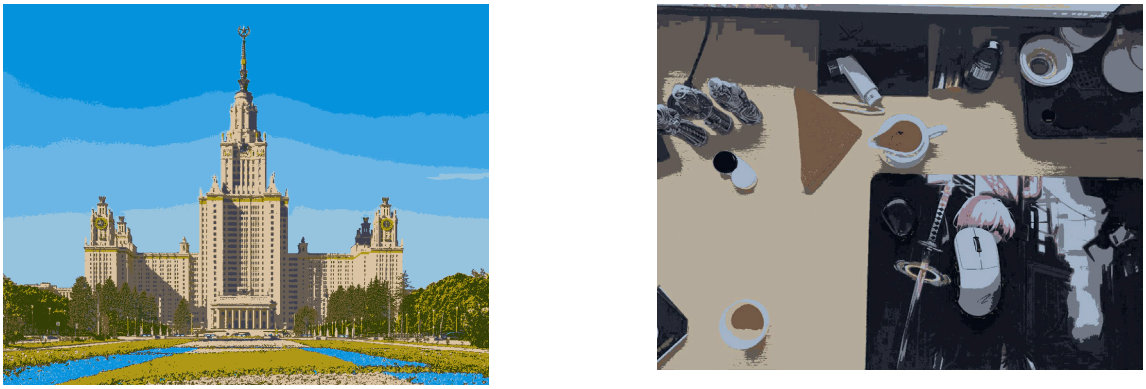

Рис. 9 пример работы алгоритма.

При учете того что метод не требует предварительного обучения или еще чего получается , хорошая кластеризация и разделение на области примерно постоянного цвета. В работе K=10.

3.2 Поиск отличий по форме

Применив различные методы, получим сегментации на области постоянного цвета исходных изображений. Затем найдем отличия по форме между

изображениями.

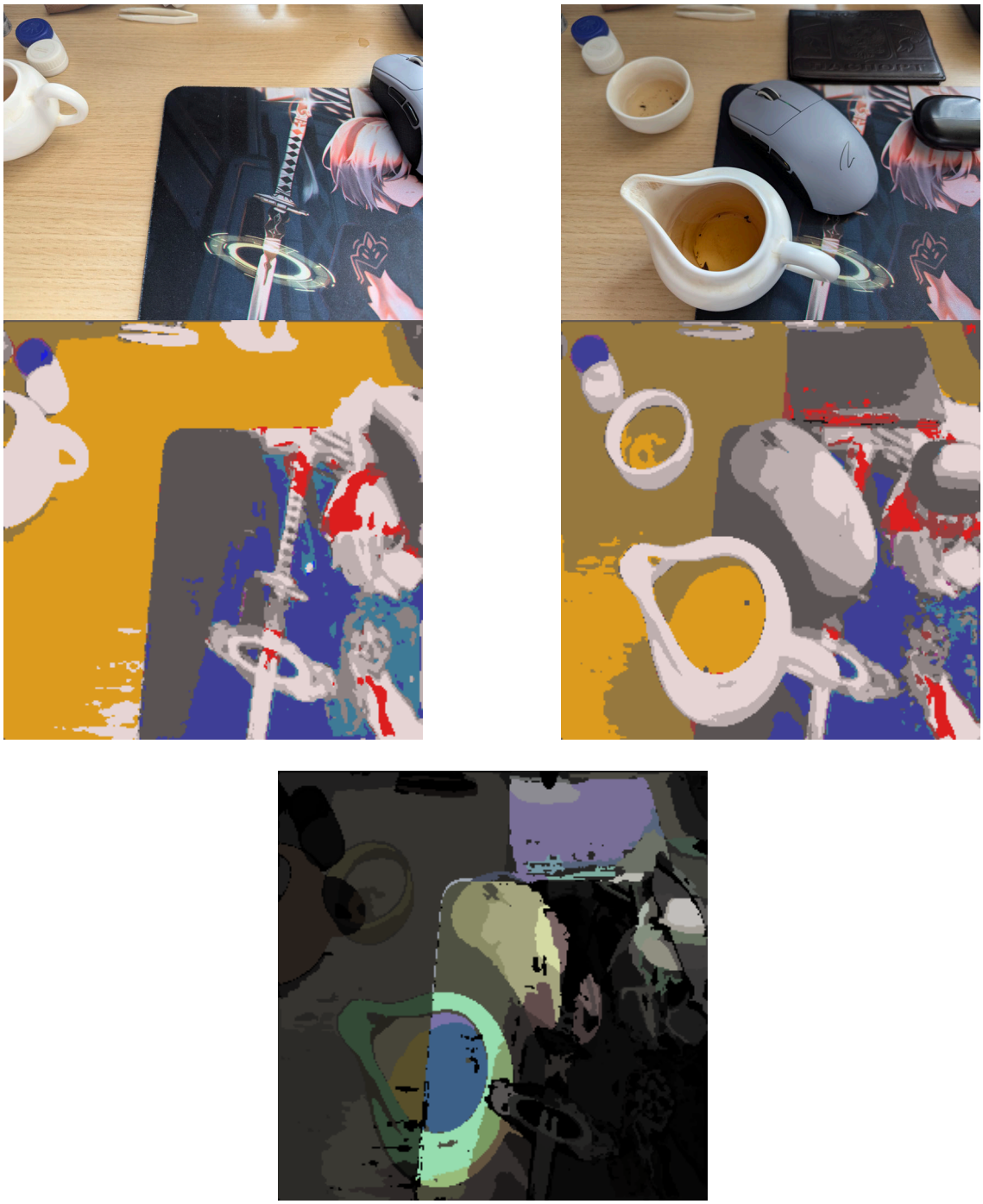

Рис. 10 результат использования метода заранее выделенных векторов цвета.

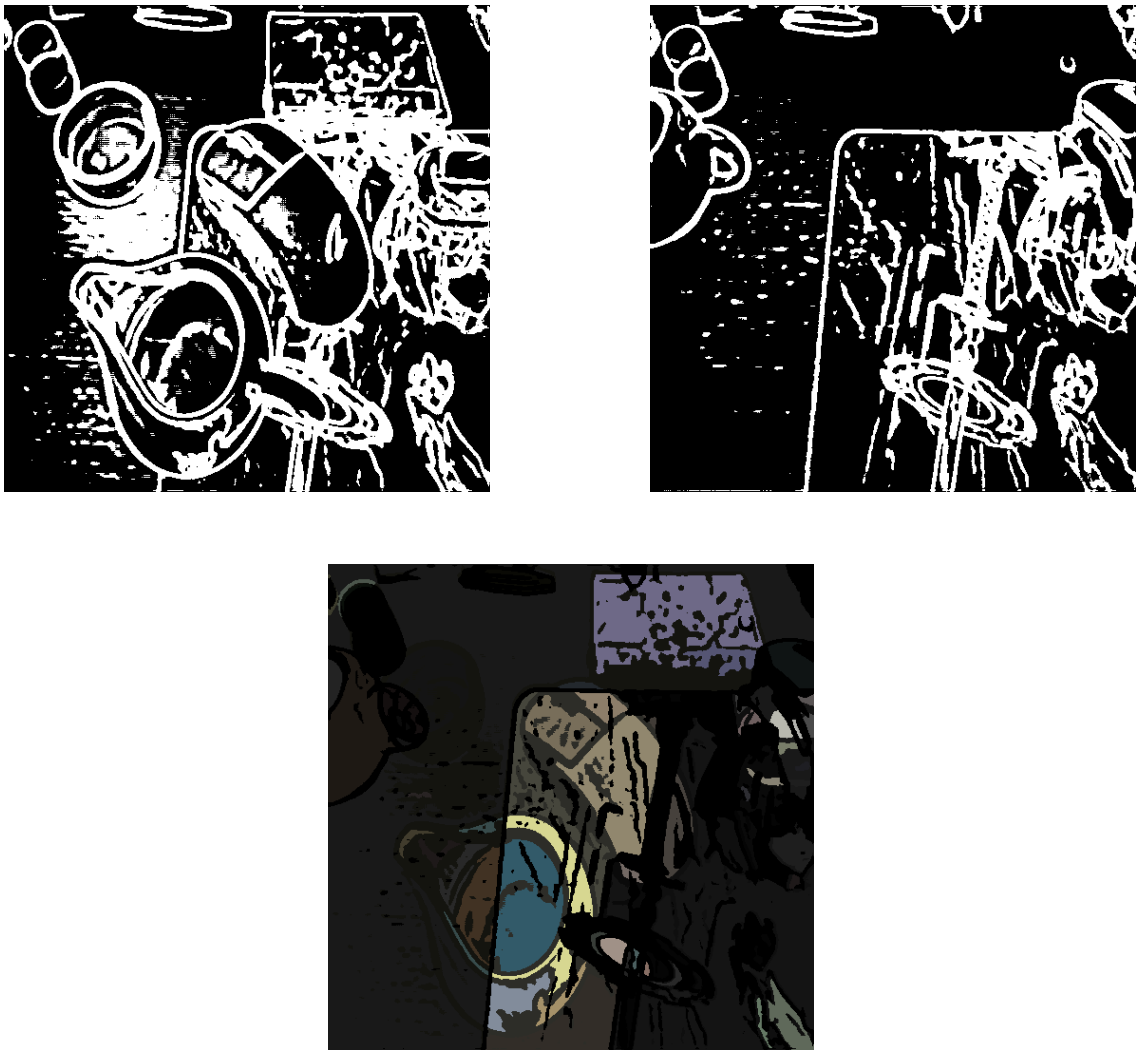

Рис. 11 результат применения модели обученной на поиск границ.

Рис. 12 Результат применения метода сегментации суперпикселями.

Рис. 13 Результат вычитания при использовании K-средних.

Рис. 14 Результат простого вычитания пикселей.

Из результатов поиска отличий видно, что модель, с заранее выделенными векторами, хорошо справляется с поставленной задачей и кроме цветовой информативности сохраняет еще и сегментационную.K-средних имеет много шумов, но неплохо справляется.

3.3 Анализ результата

Для численной оценки ошибки методов, сравним результат поиска отличий по форме между изображениями сегментированными на области постоянного цвета человеком и между сегментированными методами из этой работы.

3.3.1 Валидация методов с помощью простых сцен

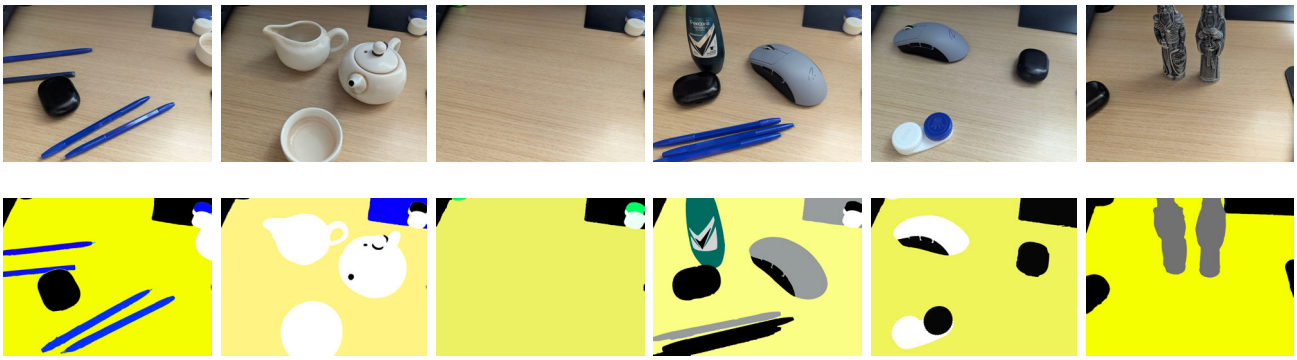

Рис. 15 Пример изображений по которым мы будем подбирать наилучшую модель

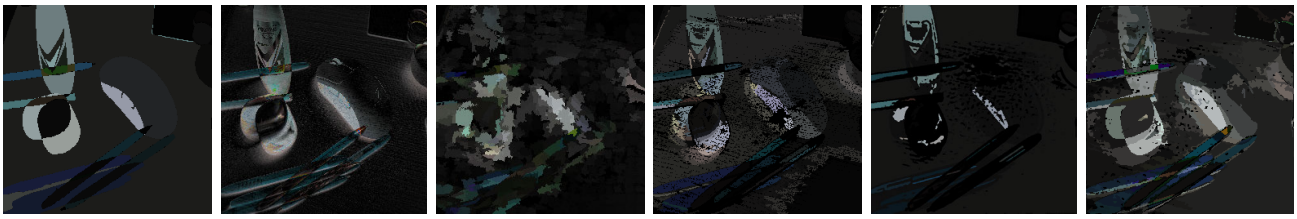

Рис. 16 Результаты поиска отличий между сценами 2 и 4. Методы(слева направо): Человеческая сегментация, по пикселям, суперпиксели, к-средних,модель обученная на границу, модель с заранее заданными цветами

Для численной оценки нормируем значения расстояния в промежуток [0,1], приведенная ниже нормировка дает больший вклад относительных погрешностей при малых глобально, но весомых отличиях. Также посчитаем ошибки и по обычной, линейной нормировке. Нормировка для малых погрешностей :

$$\Delta f(x) = \sum_{j=0}^{n} \left| \frac{f1_j(x) - f2_j(x)}{f1_j(x) + f2_j(x) + (5)} \right| \tag{14}$$

Обычная:

$$\Delta f(x) = \sum_{j=0}^{n} \left| \frac{f1_j(x) - f2_j(x)}{255} \right| \tag{15}$$

После сравнения получились такие ошибки по 15 изображениям отличий между сценами, при сравнении с эталонно размеченными.

Название метода	Относительная MAE(средняя абсолютная ошибка)	Относительная MSE(среднеквадратичная ошибка)	Абсолютная MAE	Абсолютная MSE
модель с заранее выделенными векторами цвета	0.28861	0.16445	0.06752	0.01094
модель обученная на границу	0.44361	0.33157	0.09613	0.02748
K-means	0.36043	0.24398	0.07872	0.01693
суперпиксели	0.43854	0.30205	0.09206	0.01813
простое вычитание	0.40765	0.27521	0.07876	0.01323

4 Выводы

Наименьшие значения ошибок были достигнуты моделью, сегментирующей изображение по заранее определённым векторам цвета. **Оценка эффективности методов**:

· Результат может быть частично не корректным, из-за ошибки разметки самих изображений человеческим зрением. Можно заметить блики и различные тени или отличия по яркости и контрасту, которые не были размечены.

· Наилучшие результаты по метрикам MAE (0.288) и MSE (0.164) продемонстрировала модель, обученная на выделение областей постоянного цвета. Это связано с её способностью учитывать цветовые паттерны и структурные особенности изображений, что минимизирует ошибки при сравнении форм.

· Метод суперпикселей (SLIC) и простое вычитание пикселей показали более высокие значения ошибок (MAE: 0.438–0.407, MSE: 0.302–0.275), что обусловлено их чувствительностью к шуму и отсутствием семантической адаптации.

· Плохой результат модели обученной на границу можно аргументировать тем, что модель имела не достаточной хороший синтетический датасет, и из-за толщины границы(по обучающей выборке примерно 3 пикселя, что много для изображения 512x512 и также дает вклад в ошибку), а также в целом видно, что не хватает реальных изображений или изображений с текстурами в датасете, для увеличения обобщающей способности.

· К-средних показал удовлетворительные результаты, не идеальный результат связан с несовершенностью алгоритма и отсутствием предварительной фильтрации для уменьшения шума и более гладкой сегментации.

· Также стоит отметить что результаты по абсолютной и относительной нормировке отличаются, простое вычитание пикселей 2ое по качеству в обычной нормировке. # 5 Дальнейшие перспективы 1. **Улучшение датасетов**: - Интеграция реальных изображений в синтетический датасет для повышения обобщающей способности моделей. - Генерация данных с вариациями освещения, текстуры и перспективы для обучения устойчивых моделей. 2. **Модернизация архитектур**: - Замена U-Net на более сложные и современные архитектуры (например U-Net++ или другие) для лучшего результата. - Можно попробовать добавить блок внимания на низких уровнях энкодера и декодера чтобы выделять более и менее важные признаки. - У лучшей модели(заранее заданные вектора) можно попробовать подобрать другое оптимальное количество векторов. 3. **Улучшение реализации:** - Возможным направлением улучшения является использование предобученных энкодеров (например, на ImageNet) с последующим дообучением декодирующей части. - Можно было разбить изображение на кусочки и применять модель на них вместо сжатия изображения до стандартного размера, а также обучить модель на входное изображение размером 512x512, что также бы улучшило качество. 4. **Применение к видео:** - Попробовать применить модель и метод к видео с неподвижной камерой и как либо применить его для анализа.

Приложение:

1. Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. *arXiv preprint arXiv:1505.04597*.
    - DOI: 10.48550/arXiv.1505.04597
    - PDF: arxiv.org/pdf/1505.04597.pdf
2. Goodfellow I., Bengio Y., Courville A. (2016). *Deep Learning*. MIT Press. 800 p.
3. Machine Learning Collection by Aladdin Persson (https://github.com/aladdinpersson/Machine-Learning-Collection/stargazers)
4. Пытьев Ю.П., Чуличков А.И.** Методы морфологического анализа изображений. — М.: Физматлит, 2010. — 336 с. **ISBN:** 978-5-9221-1210-7

ссылка На мой репозиторий с частью кода (https://github.com/pechenie8v8kletku/Course_work_phys_phac/tree/main)