

Отчет о лабораторной работе №3

Обработка и распознавание изображений, ММП ВМК МГУ.

Аристархов Данила Дмитриевич.

Май 2024.

Содержание

1 Постановка задачи	2
2 Описание данных	2
3 Описание метода решения	2
3.1 Сегментация изображения	2
3.2 Построение скелета изображения	3
3.3 Построение признакового описания структуры графа	3
3.4 Классификация	3
4 Описание программой реализации	3
5 Эксперименты	3
6 Выводы	4

1 Постановка задачи

Необходимо разработать и реализовать программу для работы с изображениями для классификации изображений моделей графов, построенных из магнитной головоломки. Программа должна производить сегментацию изображения, генерацию признаковых описаний структуры графов и классификацию изображения в соответствии с заданным набором эталонов. Также необходимо разработать пользовательский интерфейс для работы с программой, обеспечивающий выбор изображения, выполнение операций преобразования и выдачу результата.

2 Описание данных

Данные представляют из себя растровые изображения, на которых изображены модели графа, построенных из деталей магнитной игры-головоломки.

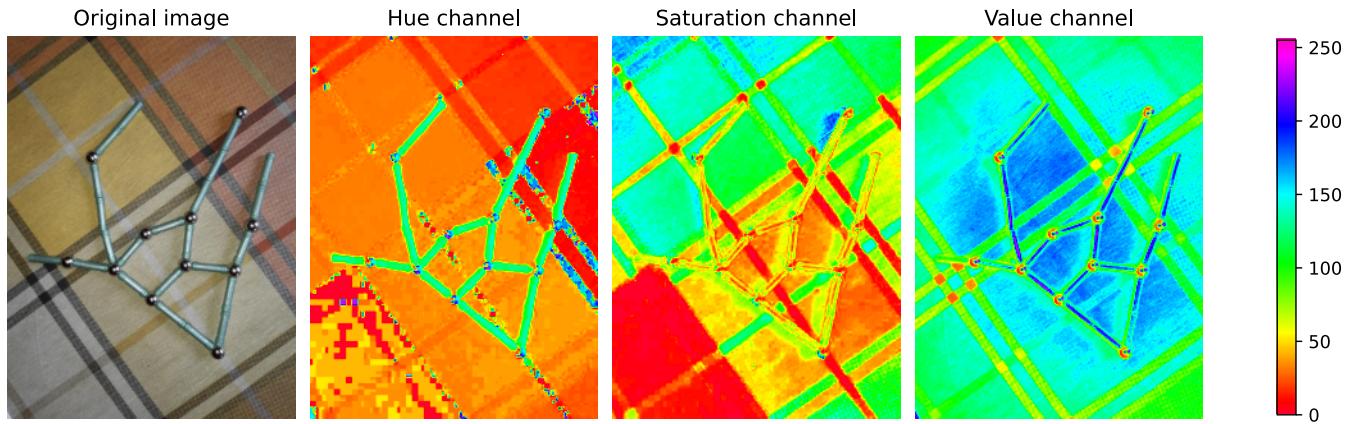


Рис. 1: Разложение изображения по параметрам HSV

3 Описание метода решения

3.1 Сегментация изображения

Первым этапом алгоритма является выделение маски ребер графа.

1. Сначала изображение переводится в HSV (Hue, Saturation, Value) представление. В таком формате проще сегментировать изображение.
2. Далее проводится бинаризация изображение по HSV параметрам. Данные параметры подбирались путем анализа изображений по отдельным компонентам (см. [рис. 1](#)), а также путем экспериментов. Оптимальные значения могут отличаться в зависимости от изображения. Для данной задачи были выбраны следующие пороги:
 - (a) Изображения на белом фоне:
 - i. Hue $\in [50, 140]$
 - ii. Saturation $\in [0, 255]$
 - iii. Value $\in [150, 255]$
 - (b) Изображения на пестром фоне:
 - i. Hue $\in [30, 100]$
 - ii. Saturation $\in [30, 255]$

- iii. Value $\in [0, 255]$

Замечание: Все параметры HSV были приведены к значениям $[0, 255]$.

3. Для удаления шумов использовался медианный фильтр с ядром $(7, 7)$.
4. Также используется открытие с ядром $(7, 7)$.

3.2 Построения скелета изображения

Далее по полученной маске строится скелет изображения. Из маски удаляются связные компоненты размером менее заданного порога. Данный параметр выбирается в зависимости от изображения и определяет чувствительность алгоритма к шумам. Для данной задачи порог был выбран равным 20 пикселям. Таким образом, удается получить отрезки толщиной в один пиксель, являющимися очертаниями ребер графа.

3.3 Построение признакового описания структуры графа

В качестве признакового описания графа был выбран вектор в котором, k -я компонента есть число вершин степени k в представленном графе. Построение этого вектора происходит в 2 этапа.

1. Выделение концов отрезков, соответствующим ребрам. Для этого выделяются пиксели, у которых в окрестности $(3, 3)$ присутствуют ровно 2 пикселя (включая сам этот пиксель). Этим пикселям как раз и соответствуют концы отрезка.
2. Подсчет степеней вершин. Для этого концы отрезков, лежащие на расстоянии менее некоторого порога, объединяются в одну вершину. Для данной задачи это расстояние было выбрано равным 60 пикселям.

3.4 Классификация

Из-за возможных неточностей модели, не ищется точное совпадение с эталоном класса. Полученный вектор признакового описания структуры графа сравнивается с векторами эталонов и вычисляется средний квадрат разницы этих векторов. В качестве ответа выбирается тот класс, где разница наименьшая.

4 Описание программой реализации

Программа была написана на языке программирования Python. Для работы с изображениями использовалась библиотеки OpenCV и scikit-image. Интерфейс был реализован в виде веб-сервера с помощью библиотеки Flask. Для удобство решение было обернуто в docker-контейнер, однако возможна установка программы и всех зависимостей вручную.

Работа с программой происходит следующим образом (см. [рис. 2](#)): сначала пользователь выбирает изображение для обработки и задает параметры программы. Далее происходит выдача результата в виде маски сегментации исходного изображения, скелета изображения, признакового описания графа и результата классификации.

5 Эксперименты

Алгоритм хорошо себя показал на большинстве изображений, корректно составил бинаризованную маску, построил скелет из отрезков, соответствующим ребрам графа и построил признаковое описание изображенного графа. Таким образом, удается получить правильный класс изображения.

Graph

Select image: 28.jpg

Hue bounds: 100

Saturaion bounds: 255

Value bounds: 255

Min size for component:

Radius for searching vertices:

(a) Выбор изображения и параметров

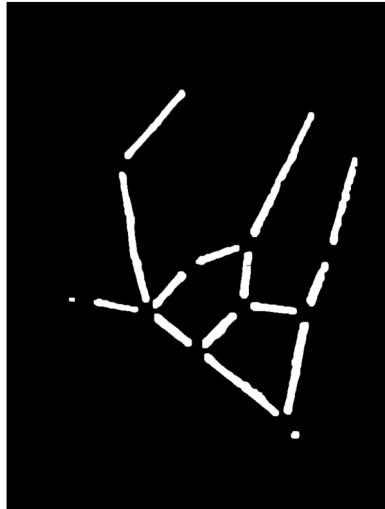
[Return to main page](#)

Sizes: [4, 4, 4, 1]

Class: 2



Original image



Segmentation mask

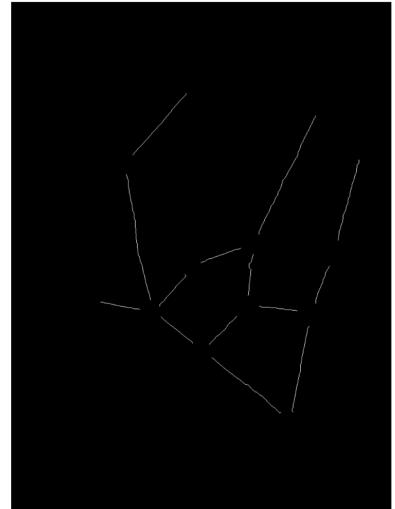


Image skeleton

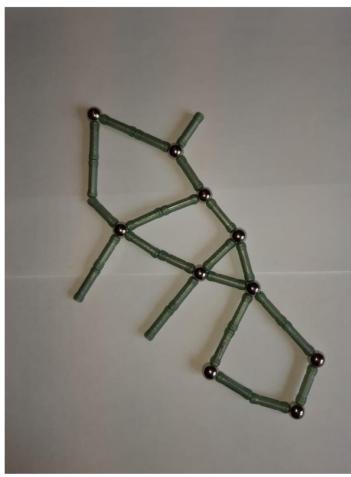
(b) Результат работы программы

Рис. 2: Этапы работы с программой

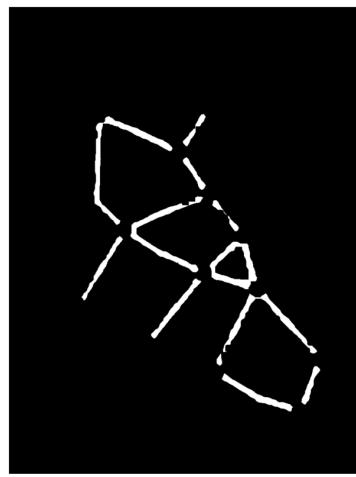
Однако на некоторых изображениях алгоритм не смог справиться с задачей. В первую очередь этого связано с неправильной сегментацией. На изображениях с пестрым фоном цвет магнитной головоломки совпадает с цветом фона, из-за чего трудно сегментировать изображение по цветам. Из-за неправильной сегментации, на скелете изображения возникают лишние «ребра» и строится неверное признаковое представление, и, как следствие, происходит неправильная классификация. Решением данной проблемы может быть более тонкий подбор параметров сегментации под каждое изображение.

6 Выводы

Предложенный алгоритм смог добиться высокой точности при построении признакового описания графов и классификации изображений. Однако на изображениях с неоднородным фоном требуется точная настройка параметров для улучшения качества сегментации.



Original image



Segmentation mask

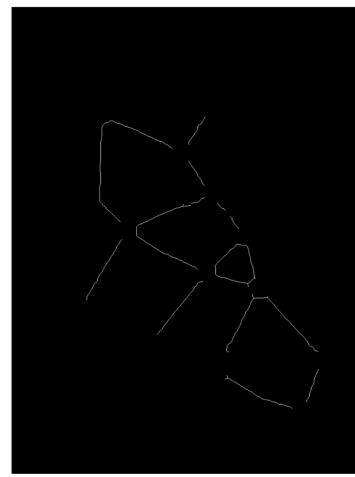


Image skeleton

(a) true label = 1, true vector = [3, 4, 3, 3]
pred label = 1, pred vector = [3, 4, 2, 1, 0, 0, 0, 0, 0, 1]



Original image



Segmentation mask

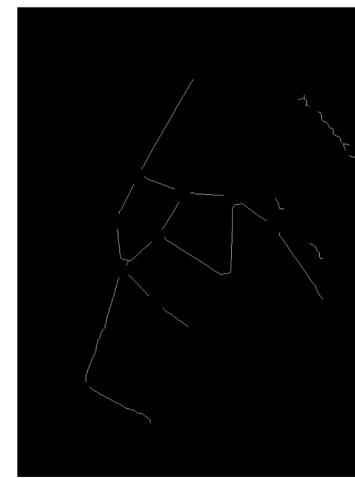
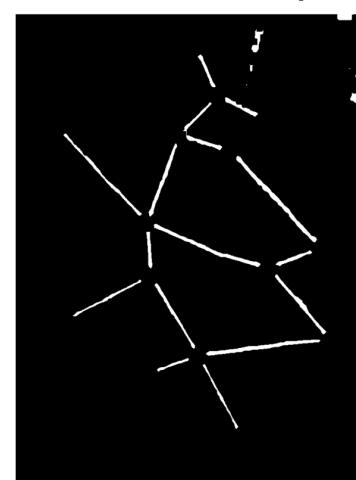


Image skeleton

(b) true label = 2, true vector = [4, 4, 4, 1],
pred label = 2, pred vector = [5, 4, 4, 2, 1]



Original image



Segmentation mask

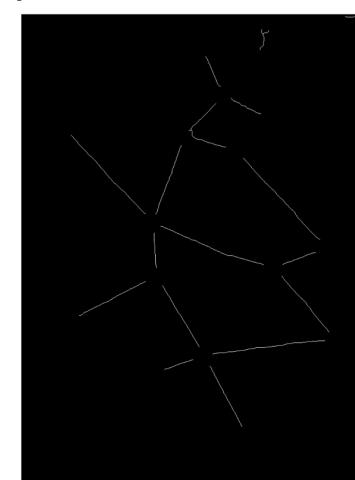


Image skeleton

(c) true label = 4, true vector = [6, 3, 3, 2]
pred label = 2, pred vector = [6, 6, 5, 2]

Рис. 3: Эксперименты