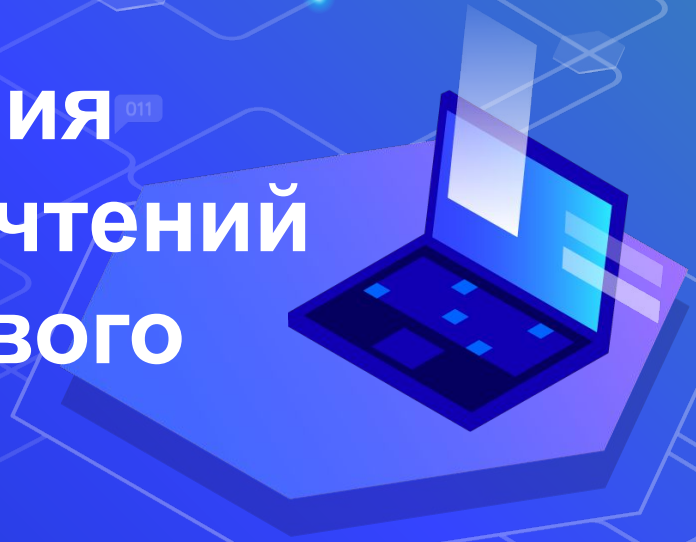
НИУ МАИ - 2019

Алгоритм определения зависимости предпочтений покупателя от цветового оттенка витрины



Пермяков Никита

Начальные предположения

- Фотография одного и того же продукта, с разным освещением и теплотой цвета, будет способствовать различному потребительскому спросу
- При этом потенциальные покупатели не должны иметь принципиального отношения к цветовому оттенку продукта
- **Вывод:** не меняя тип продукта, можно манипулировать спросом клиента, тем самым максимизируя прибыль от продаж по отношению к каждому покупателю в отдельности, затрачивая ресурсы только на этот анализ и персонализированное фото.

Существующие решения

Теоретические:

- Отсутствует реализация, даже псевдо-код
- Алгоритмы описаны в трудно интерпретируемой форме

Доступные примеры:

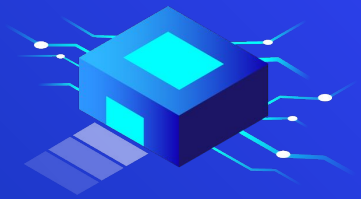
- статья Ермолаева
- статья Чугунова

Коммерческие реализации:

(API YANDEX, API GOOGLE, Поисковики (yandex, google и т.д.))

- Дорогая подписка
- Черный ящик

но высокое качество



Open source:

- Серый ящик (доверие другому)
- Потребность в вычислительных ресурсах (считать на своем железе)
- Качество ниже чем у коммерческих проектов

но доступ к коду и постоянные патчи

Идея решения основана на статье о сегментации

Особенности

- Учитываются три канала изображения RGB (а не Ч/Б)
- Результат отображен на графике scatterplot 3D
- Добавлен порог для весов яркости вертикальных и горизонтальных линий пикселей

- Сложность алгоритма ~ $O(n^2)$

Реализация

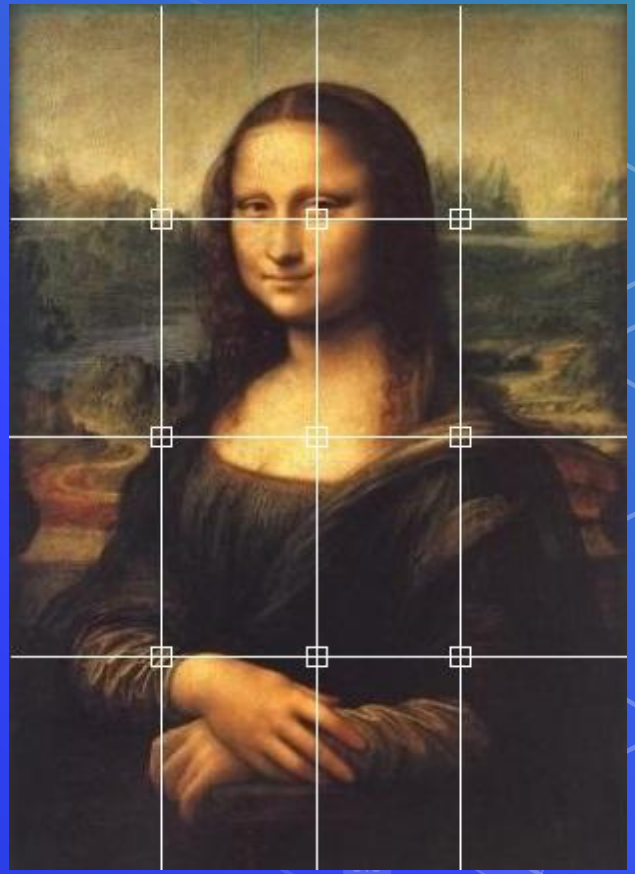
1. Изображение переводится в вектор размерности три. (Основание: кодирование RGB, максимальне значение 255).

2. Монотонные участки, близкие по значению, рассчитываются по строкам и столбцам и удаляются из матрицы изображения. В результате размерность матрицы уменьшается относительно исходной. (Оптимизация вычислений для монотонных изображений, выделен

3. Определяются узловые точки, количество которых берется произвольно, но чем их больше, тем точнее определяется характеристика изображения.

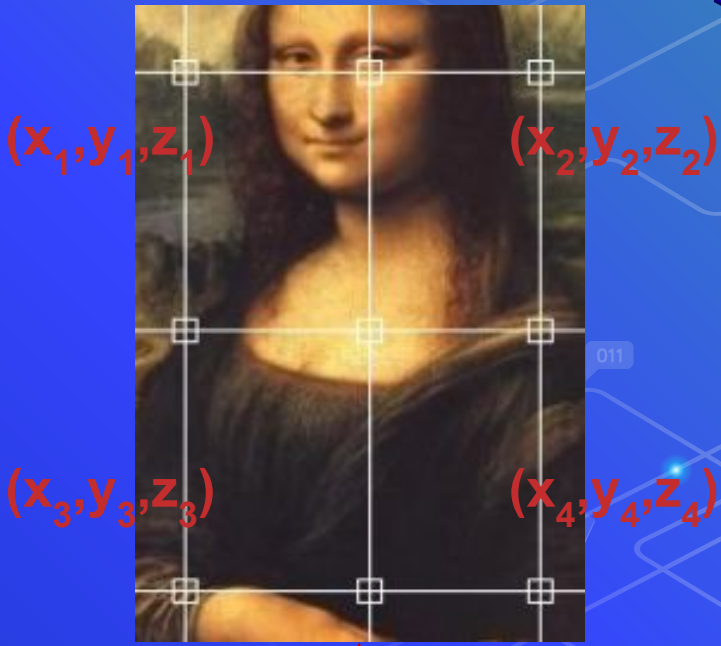
(При увеличении числа узловых точек сложность алгоритма удваивается).

Например, количество точек можно взять как квадратный корень из суммы разрешения изображения.



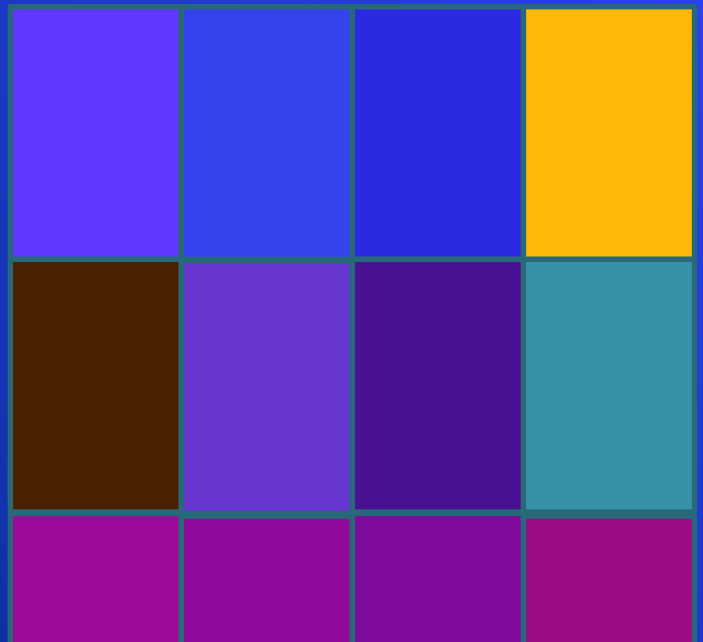
4. Рассчитываем среднюю яркость областей, разделенных этими точками. На выходе три числа для каждого треугольника.

5. Берем только области, не смежные с краями изображения, и считаем их разность между собой. Размер матрицы снова уменьшается.



6. Полученные 8 модулей для каждого прямоугольника складываются.

7. Затем все векторы изображения суммируются и делятся на (длинна средних * 3 * 256).

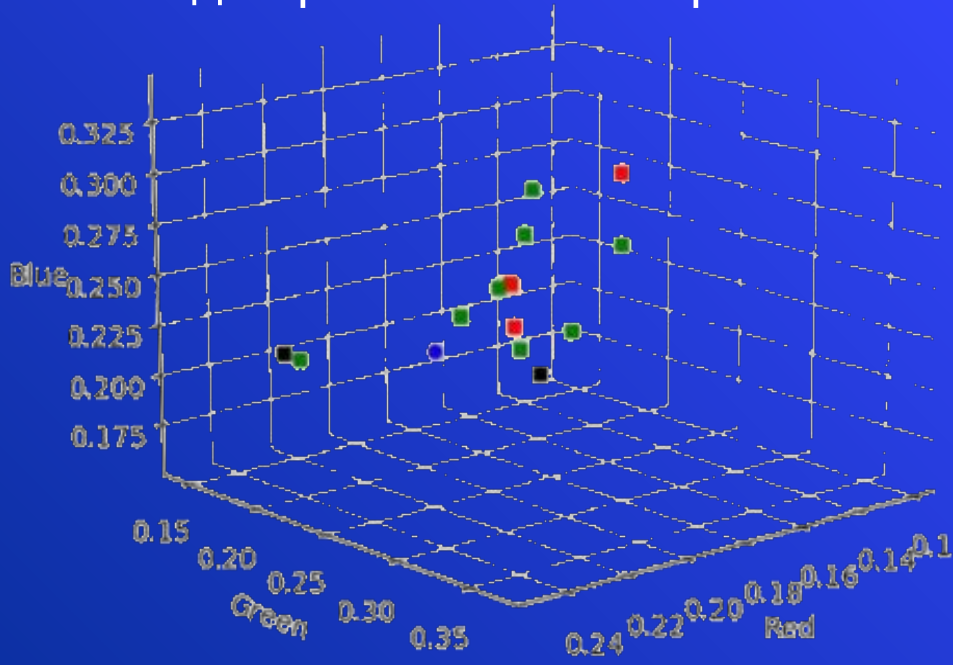


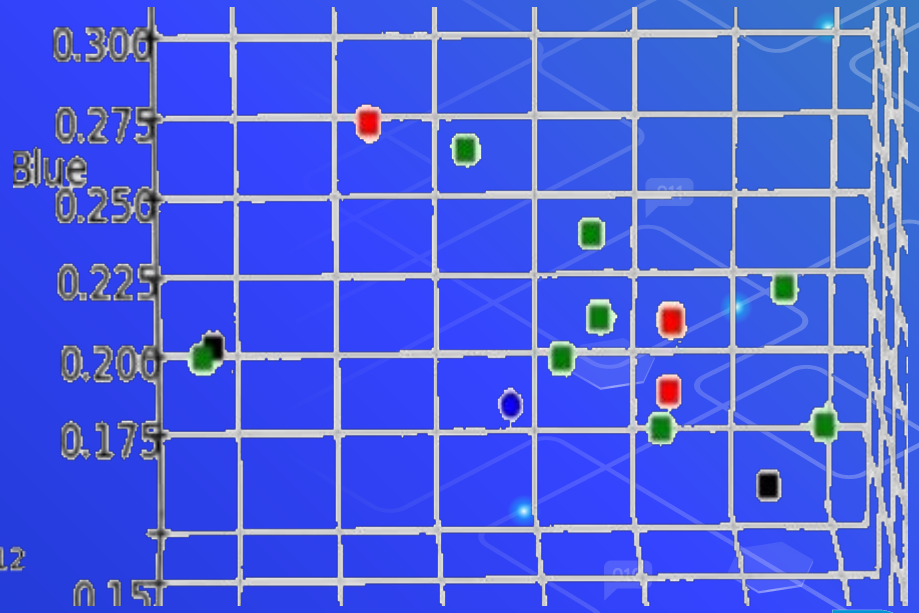
$$\text{feature\_vec} = \sum_{i=0}^{n} (x,y,z)/(\text{len}((x,y,z))*R*A)$$

где R - размерность вектора = 3,
A - максимальное значение пикселя = 256
len - длина вектора каждой области

Результат работы

- Получается нормализованный вектор, уникальный только для различных изображений.





Весь код, включая выгрузку данных из интернет-магазина Etsy, фильтр, и визуализацию доступен на Github:

https://github.com/nikit34/Etsy_analys_MY_ALGORITHM_DETECT_DIFFERENCE_IMG

Спасибо за внимание

Вопросы

Мои контакты:

https://www.linkedin.com/in/nikitapermikov/