Домашнее задание №4: «Comic-Con и k-means»

Дедлайн 1 (20 баллов): 13 марта, 23:59
Дедлайн 2 (10 баллов): 20 марта, 23:59

Домашнее задание нужно написать на Python и сдать в виде одного файла. Правило именования файла: `name_surname_4.py`. Например, если вас зовут Иван Петров, то имя файла должно быть: `ivan_petrov_4.py`.

На этот раз мы решили отправиться на Comic-Con. В рамках подготовки к нему нужно напечатать футболки с изображением Бэтмена и Супермена. В нашем распоряжении оказалась только одна картинка, на которой присутствуют необходимые персонажи[1] и принтер для печати по ткани, печатающий в 16-цветном режиме. С помощью алгоритма k-средних нужно сделать новое изображение наших героев с использованием только 16 цветов.

1 Каждый пиксель изображения несет информацию о своём цвете из модели RGB (цветовая модель изображения, которая состоит из трех компонентов R — red, G — green, B — blue). Значение каждой компоненты RGB может быть в пределах 0 … 255. Это дает возможность закодировать 255*255*255 цветов.
Для работы с изображениями в Python есть библиотека-обёртка над OpenCV [2]. С ее помощью выполнить задание будет проще.
Первым делом нам необходимо реализовать функцию, которая читает файл и преобразует его в двумерную матрицу размерности $(M \times N, 3)$. Сигнатура функции следующая:

```
def read_image(path):
    # ...
    # подсказка: если вы используете библиотеку cv,
    # не забудьте перевести изображение в формат rgb
    return image
```

2 Следующий шаг – реализовать функцию `k_means(X, n_clusters, distance_metric)`, которая принимает матрицу X размерности `(n_samples, n_features)`, количество кластеров, на которые мы хотим разбить изображение и метрику. Результатом функции является пара из вектора размера `n_samples`, где в i-й ячейке содержится кластер, соответствующий i-му пикселю, и вектора размера `(n_clusters)` с центрами кластеров

3 Для оценки результата работы реализуйте две функции.
Первая функция `centroid_histogram(labels)` строит гистограмму на основе количества пикселей, приписанных каждому кластеру и возвращает ее в виде вектора. Вторая функция `plot_colors(hist, centroids)` принимает построенную гистограмму и список центров кластеров и строит bar chart, показывающий относительную частоту каждого цвета.

---

[1] https://gist.github.com/ktisha/a898e6a7a7d45b4183b0

[2] http://docs.opencv.org/trunk/doc/py_tutorials/py_gui/py_image_display/py_image_display.html

Пример:

Функция может выглядеть так:

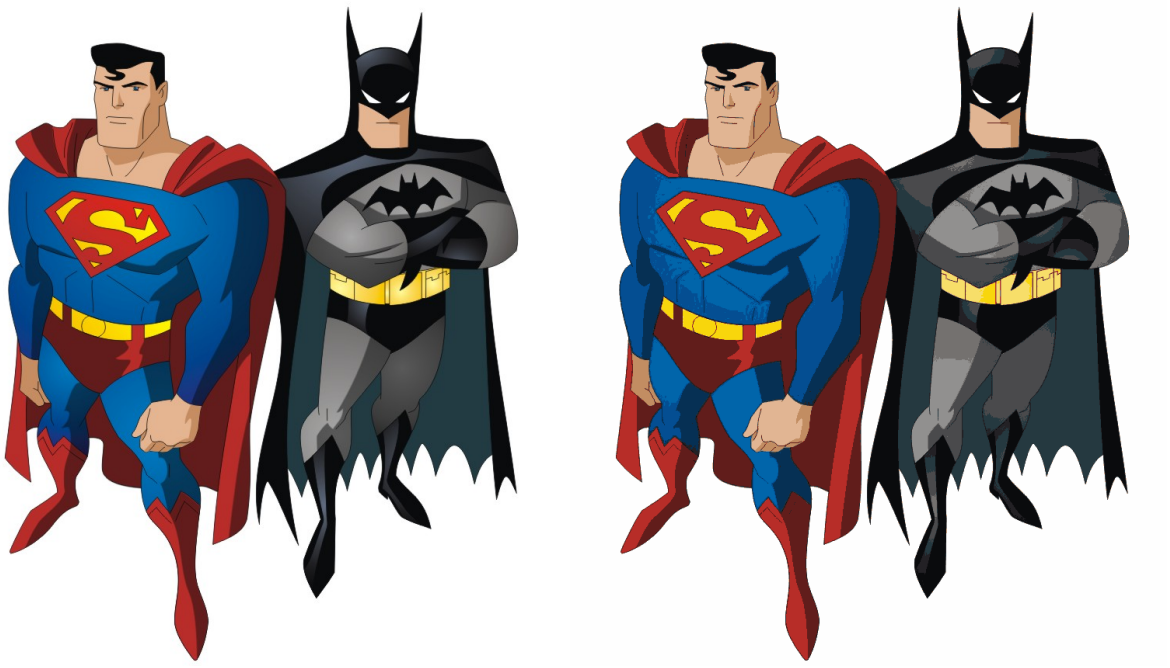
```
def plot_colors(hist, centroids):
    # инициализировать переменные bar и start_x

    for (percent, color) in zip(hist, centroids):
        # вычислить end_x
        cv2.rectangle(bar, (int(start_x), 0), (int(end_x), 50),
            color.astype("uint8").tolist(), -1)
        # обновить значение start_x

    return bar
```

4 Последняя фунция `recolor(image, kmeans, n_colors)` принимает изображение, результат работы `k_means` и количество цветов, и перекрашивает каждый пиксель изображения в тот цвет, к которому его отнес метод `k_means`.

5 Результатом работы программы должно быть изображение, в котором присутствуют только 16 цветов. Его надо прикрепить к письму.



(a) Исходное изображение (b) 16-цветное изображение