Improved Image Labeling

Jeremy Deng
Erik Maldonado

Introduction

Image annotation is the process of computationally assigning labels to an image. Automatic image annotation can be used to create valuable data sets in a fraction of the time it would take humans to do so. Known limitations regarding image annotation include: image retrieval is valid for only one language; subjectivity of human perspective, image abstractions. We address those limitations with the objective to identify if image segmentation improves image labeling when compared to the same non-segmented image. We use an approach similar to a class probability map where we look at isolated chunks of the original image, obtained using k-means and grabcut, and labeled independently using Google's Cloud Vision API; Those labels are then aggregated to produce a single output. We expect that more labels will be generated using segmentation, each image is looked at as multiple individual pieces instead of a whole, which should provide a more fine-grain description of each image.

Methodology

1. K-means clustering algorithm

For the purpose of splitting images into parts that share the same identity, our group took an approach to use k-means clustering algorithm in OpenCV. For instance, this algorithm segments image based on the euclidean distance between each pixel and centroid mean. The segmented images are then output into the directory:

“./segments/<imageName>”, for sending it into Google's cloud vision API to generate labels. Each image was tested using this method and tested with different cluster size.

Convert image pixel value to float point

By default in OpenCV each pixel is described with three color value vector in the alternation style of RGB color model, such that {BLue, Green, Red}. For every color value of the pixel, it ranges from 0 - 255. However, k-mean clustering requires multiplication and division that would alter the integer color value into floating point. Thus, we convert the data type to a float point by the following function call in OpenCV into floating point range from 0-1 for more defined calculation. The conversion is reversible by multiplying 255 into the floating point.

```
Mat data;
img.convertTo(data, CV_32F);
```

Figure 1: Code for converting image pixel type to float point

Reshaping the image matrix

Since each image is stored as a [M x N] matrix where M is the column of pixel and N is the row of the pixel. Instead of performing calculation on a 2D matrix,we decided to reshape the matrix into [1 x M*N] matrix. This is beneficial to generate label on each location inside the matrix.

Generate random cluster mean

For a given cluster size, we want to generate a mean for each cluster size. Thus, we set the flag[1] of the k means function to initialize cluster size of random mean and store in a [1 x N] matrix.

Calculate the distance and update mean

For a given iteration, the kmeans function is then used to calculate the euclidean distance(See Equation 1) between pixels to each mean and assign a label to the closest mean value. After all the pixel value has labeled with a cluster. We further calculate the mean of the cluster and update the new mean. In each iteration. This step is repeated and it would stop if one of the two conditions is met: (1) The algorithm have proceed max iteration time. (2) No changes were made through the last iteration.

$$\sqrt{\sum_{i=1}^{n}(q_i - p_i)^2}$$

Equation 1: Euclidean Distance Function

2. GrabCut Algorithm

GrabCut is a different approach for segmenting images in computer vision. Our group uses the pre-implemented model in OpenCv. The GrabCut algorithm is implemented based on graph cut[2]. This method requires the user to specify a bounded window inside the image that contains the ideal object. Then it would estimate the color distribution between the foreground(inside the bounded window) and the background. If the pixel is likely belongs to background, the bounded region would shrink towards inside.

User Specify Window

We decided to crop the image into four quadrants, and inside each quadrant perform the grabCut with a windows size 10 pixel away from the boundary. Then we perform a grabCut algorithm again in the center of the image. We used the rectangle method in OpenCv to generate the bounded region for grabCut method[3].

3. Google's Cloud Vision API

Segmented images were encoded as jpeg images then encoded again using base64. The base64 encoded images were inserted into json objects matching the API's schema. Then, sent off to the API for a response via an HTTP POST.

```
for (const auto& json_object: json_objects)
{
        // setup request
        http::http_request post(http::methods::POST);
        post.set_body(json_object);

        // async request
        pplx::task<void> async_chain = api.request(post)

        // handle http_response from api.request
        .then([&group, &label](http::http_response response)
        {
                json::value result = response.extract_json().get();
                for (auto object : result[L"responses"][0][L"labelAnnotations"].as_array())
                {
                        label = to_string(object[L"description"].serialize());
                        label.erase(remove(label.begin(), label.end(), '\"'), label.end());
                        group.insert(label);
                }
        });

        // wait for outstanding I/O
        try { async_chain.wait(); }
        catch (const exception& e) { printf("request exception:%s\n", e.what()); }
}
```

Figure 2: Code for API communication

Results

By using the k-mean cluster and grab cut algorithm, we would be able to generate a good amount of segments on different images. In each image we used k-mean clustering algorithm to segment the image, different cluster value is tested to output different label. Figure 2 presents an example of directory view on segmented maps of k means clustering.

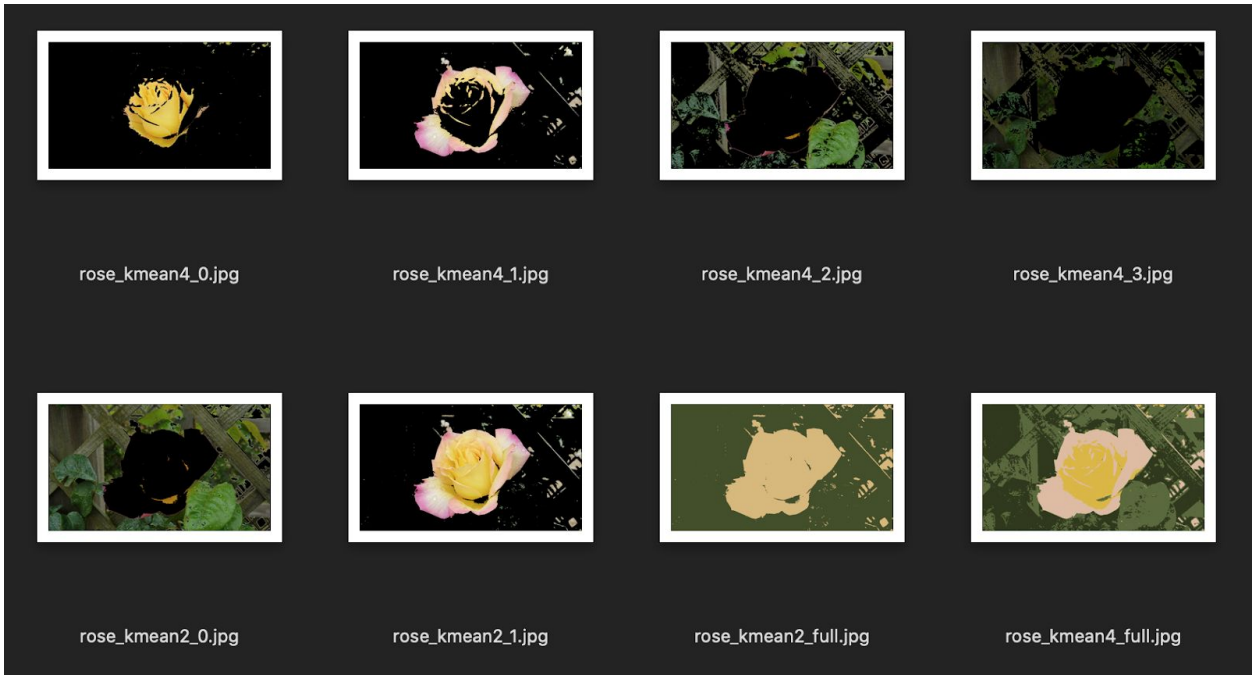

Figure 3: Directory view of K means clustering output with size 2 and 4.

For every image, the grab cut would generate 5 more images on the 4 quadrants and the center. Furthermore, some images are tested to have more bounded region to crop out the expected object. Figure 3 illustrates the results of grabcut for rose image.

Figure 4a(Upper Left): Original image. Figure 4b(Upper right): center cut using GrabCut. Figure 4c(Lower left): fourth quadrant cut using GrabCut. Figure 4d(Lower right): third quadrant cut using GrabCut.

Our testing showed that segmented images generated 46.39% more labels on average when compared to the labels generated using the unsegmented images.

Image	Base Labels	Segmented Labels	Percent Change
Background	13	30	+%56.66
Rose	23	36	+%36.11

Limitations

We could not control the minimum probability for returned labels when using the API. All

labels returned had a probability of %50 or greater.

Discussion

Our group initially used the k-mean clustering to segment the image. However, there are several limitations on k-mean clustering. Since the segmented class are grouped into the same cluster based on color distance. This method of segmentation algorithm fails to generate segment parts as we expected. Our specific aim is to segment the image into meaningful segments that contains information on the same object. Using the rose example demonstrated in figure 2 and 3, K-mean cluster successfully segments the rose buds and leaves. However, we wanted more segments that would separate leaves into one class, fence into other class and the rose as another one. If we initialize the k mean with the specific value of these three colors, although it would still provides good results on segments based on object class. The assumption that one object consistently constructed of same color is made. K-mean clustering algorithm fails to segment class object that contains two distinctive colors. For example, if a black and white color cat is used in the k-mean clustering segmentation process. No matter which cluster size is used(excepted 0 and 1), the color white and black would definitely fell into two clusters. This method would still be a good fit to unicolor objects, as if the object inside the image have only one color and it is distinctive than the background color. But even at best case scenario, k-mean clustering would still provide some noisy label such as the file rose_kmean2_full.jpg in Figure 2. Thus, we researched other segmentation method to pursue our aim. Inspired by the graph cut, we used the pre-implemented function grabCut in OpenCv. This method

works very well and generating expected output most of the time. Some constraints of using this method is that the user have to manually specify where to look for the object. Since image can appear anywhere inside an image. The assumption that we made to perform grabCut on four quadrants and the center is insufficient. Due to the time constraint, we were not able to test our method. We also did some research on using mean shift clustering. After segmenting the images into parts and send to Google's cloud AI, it returned many more labels as we wanted. Worth to mention, the label like “Art” is returned after sending the full image of k-mean clustering image after updating pixel value into the centroid mean. “Midnight”, “Shadow”, or “darkness” are returned due to the balck background. We also changed the used method to alter the background into white and transparency. As a result, this generates on average 46.5% more labels.

Future Work

Using a different cloud API could produce different results; A small test of Microsoft’s Vision API returned results with probabilities as low as 25%.

Testing different segmentation algorithms would potentially allow even greater results at the cost of more processing time. Processing time could also be reduced, and accuracy increased, if we crop resulting segments.

It would be possible to translate image descriptions from any language to a common language. If labels generated are all in a common language, image retrieval could be performed for any image in any language.

References

[1]https://docs.opencv.org/master/d0/de1/group__core.html#ga276000efe55ee2756e0c471c7b270949

[2]https://en.wikipedia.org/wiki/GrabCut

[3]https://docs.opencv.org/master/d6/d6e/group__imgproc__draw.html#ga07d2f74cadcf8e305e810ce8eed13bc9

[4]https://en.wikipedia.org/wiki/Automatic_image_annotation