Image Foreground/Background Segmentation using Max-Flow

Group Member：

Chengeng Liu	**BUID:U14927368**	**SCC uer namelcg518**
Shaowei Lin	**BUID: U17616063**	**SCC user name :linsw**
Fei Gao	**BUID: U74859566**	**SCC user name: gaof**
Siyuan Zhang	**BUID:U74771707**	**SCC user name: siyuanz**
Runchuan Feng	**BUID:U01547256**	**SCC user name: fengrc**

Abstract

Image segmentation is a process of partitioning a digital image into multiple segments according to some certain features. And in this project we are gonna focusing on selecting a region of background pixels and another region of foreground pixels from the given gray scale images. The key concept is to utilize the disparity between intensity of adjacent pixels to construct energy flow. Each pixel will be modeled as a node, and there is always a edge between neighboring pixels. Besides, there are two imaginary terminal pixels representing the source and sink of the flow. In this way, image segmentation is formulated as a max-flow problem, and the min-cut of the residual graph will be used as contour that separates image foreground and background.

In order to solve the max-flow problem, People have established a relatively perfect theoretical system and developed a large number of algorithms, such as K-means, Edmonds-Karp, Gaussian model, We implement the EK algorithm, K-means, Gaussian mixture algorithm and compare their accuracy and efficiency.

Implementation

Transform Image to Graph

In order to perform maximum flow segmentation on the image, we firstly transform the input image into network flow graph performing the following steps:

1.Each pixel in the image is considered as a node. Two more specially designated terminal nodes: S(source) and T(sink) are added.

2. There are two kinds of edges in the graph, one of them is n-links which connecting neighboring nodes in a regular grid-like fashion. And another kinds of edges are used to connect nodes to the terminals(t-link).

3.All edges are assigned some different nonnegative weight(cost).

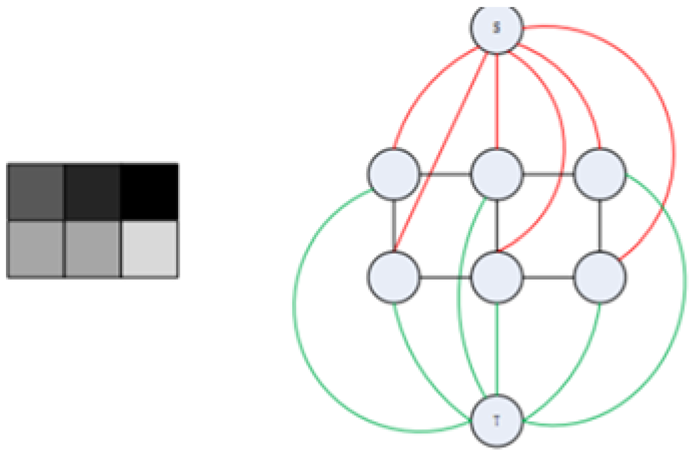


Fig.1 A image with 3*2 pixels and its graph

As for the Calculation of Edge Weight, there are several different situations as follows:

1. the weight of t-links connecting a terminal with another terminal will set to be 0.
2. the weight of t-links connecting a terminal with itself will set to be a large value (1000 in our code).
3. the weight of other links are based on gray similarity, using the following function. ***Ip*** and ***Iq*** are the intensity of the neighboring pixels, and σ is chosen to be 30.

As we can see in the function, larger differences in gray level results in smaller values assigned to edges and the converse for similar gray levels. Thus, we have transform a image to a graph.

Maxflow algorithm

Ford-Fulkerson method with Edmonds-Karp algorithm

Let's define one more thing. A residual capacity of a directed edge is the capacity minus the flow. It should be noted that if there is a flow along some directed edge(u,v), then the reversed edge has capacity 0 and we can define the flow of it as f((u,v)) = -f((u,v)).This also defines the residual capacity for all the reversed edges. We can create a residual network from all these edges, which is just a network with the same vertices and edges, but we use the residual capacities as capacities.

The Ford-Fulkerson method works as follows. First, we set the flow of each edge to zero. Then we look for an augmenting path form s to t. An augmenting path is a simple path in the residual graph, i.e. along the edges whose residual capacity is positive. If such a path is found, then we can increase the flow along these edges. We keep on searching for augmenting paths and increasing the flow. Once an augmenting path doesn't exist anymore, the flow is maximal.

It should be noted that the Ford-Fulkerson method doesn't specify a method of finding the augmenting path. Possible approaches are using DFS or BFS which both work in O(E). If all the capacities of the network are integers, then for each augmenting path the flow of the network increases by at least 1 (for more details see Integral flow theorem). Therefore, the complexity of Ford-Fulkerson is O(EF), where F is the maximal flow of the network. In the case of rational capacities, the algorithm will also terminate, but the complexity is not bounded. In the case of irrational capacities, the algorithm might never terminate, and might not even converge to the maximal flow.

Edmonds-Karp algorithm is just an implementation of the Ford-Fulkerson method that uses BFS for finding augmenting paths. The intuition is, that every time we find an augmenting path one of the edges becomes saturated, and the distance from the edge to s will be longer if it appears later again in an augmenting path. The length of the simple paths is bounded by V.

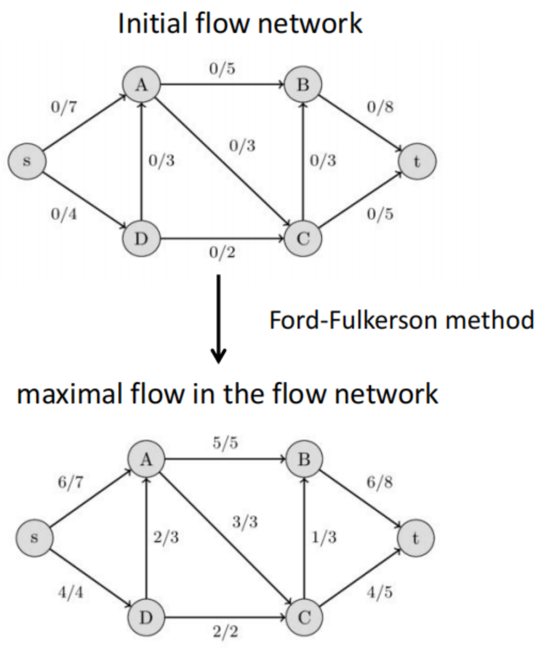


Fig2: The flow of Edmonds-Karp algorithm

Other clustering algorithms

Kmeans

Clustering is a method to divide a set of data into a specific number of groups. It's one of the most commonly used methods is k-means clustering. The k-Means algorithm is a clustering algorithm based on distance similarity, it partitions a given set of data into k-disjoint clusters. The K-means algorithm consists of two separate stages. In the first stage, it computes k centroids, and in the second stage, it takes each point to the closest cluster to the centroid of the respective data point [1]. Euclidean distance was usually used to find the nearest centroid.

The basic process of the K-Means algorithm is: Firstly, randomly selected K initial clustering centers. Secondly, calculate the distance between each sample and each cluster center, and assign each sample to the nearest cluster center. Thirdly, for each cluster, the mean value of all samples is taken as the new clustering center of the cluster. Finally, repeat steps 2 and 3 until the clustering center no longer changes or the set number of iterations is reached.

```
Algorithm 1 k-means algorithm
1: Specify the number $k$ of clusters to assign.
2: Randomly initialize $k$ centroids.
3: repeat
4:     expectation: Assign each point to its closest centroid.
5:     maximization: Compute the new centroid (mean) of each cluster.
6: until The centroid positions do not change.
```

Figure 3: K-means pseudocode

Each pixel in an image is a point in three-dimensional space consisting of the intensities of the red, blue, and green channels, and the segmentation algorithm simply treats each pixel in an image as an individual data point [2]. So for each image, take each red, blue, and green pixel value as the input vectors and cluster these vectors by using the K-means algorithm, for any particular value of K, by re-drawing the image replacing each pixel vector with the {R, G, B} intensity triplet given by the center μk to which that pixel has been assigned.

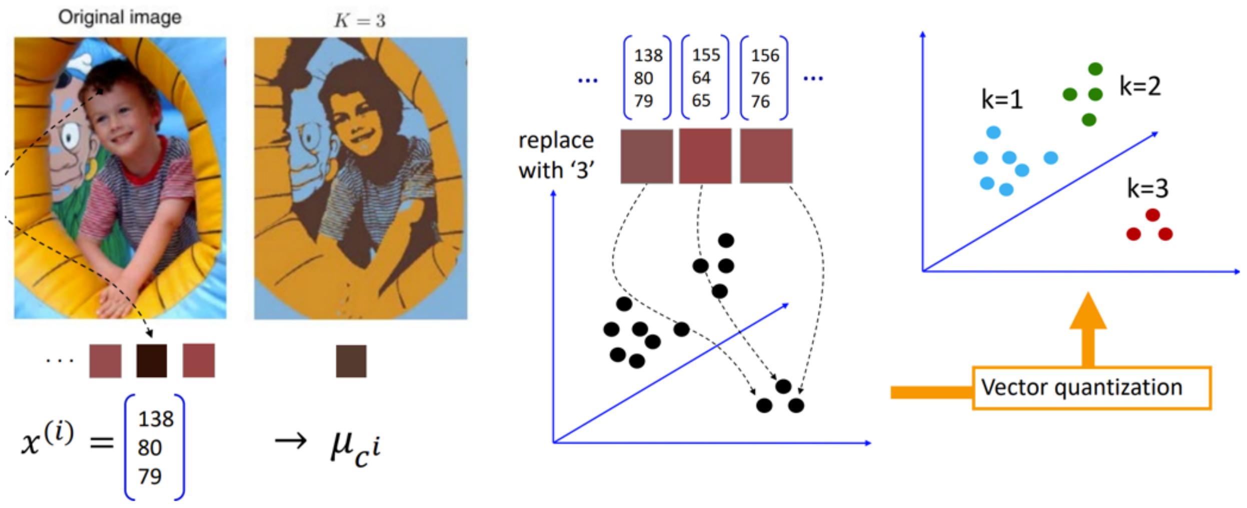


Figure 4: One example of the K-means clustering algorithm to image segmentation.

Gaussian Mixture

Similar to the K-means method, the Gaussian mixture model is also a clustering algorithm, which can be used to do image segmentation. The main idea of the Gaussian mixture model is to assume all data are generated from one of the Gaussian distributions. That means each point of the data, or every pixel can be represented as a random number from the normal distribution featured with a certain mean and a certain variance.

To construct the model, first we assume there are k components. Each component generates data from a multivariate Gaussian with mean and covariance matrix. Each data point is sampled from a generative process: 1. Choose component I with probability P(y=i) 2. Generate data points.

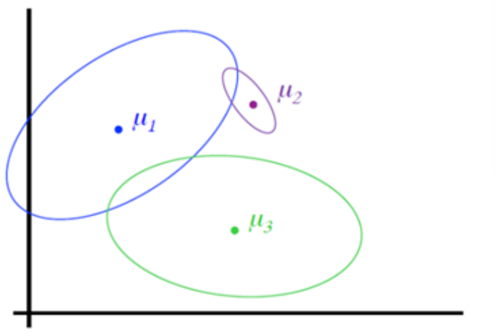


Figure 5: Gaussian mixture model (GMM)

A density model p(x) may be multi-modal. We may be able to model it as a mixture of unimodal distribution. And each mode may correspond to a different sub-population(in this case, the corresponding RGB values). Here are two examples of uni-modal and multi-modal.

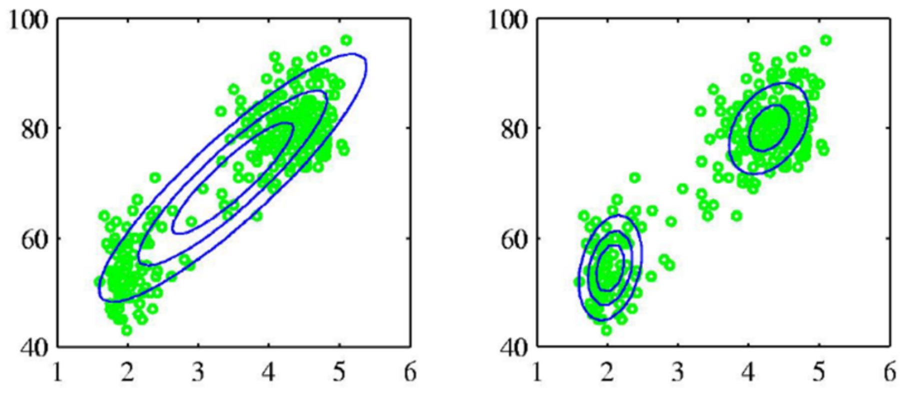


Figure 6: Single gaussian and mixture of two gaussians

Given a mixed gaussian distribution, we can use the EM(expectation maximization) algorithm to derive the original distributions. The steps are (1) Given a set of parameters and training data (2)estimate the class if each training example using the parameters yielding new (weighted) training data (3) relearn the parameters based on the new training data, repeat from (1).

GMM is a cluster algorithm similar to K-means so it for sure can be implemented in image segmentation by a similar transition format and methods to K-means.

Instructions

Maxflow

Edmonds-Karp algorithm is implemented by c.The role of the *main.cpp* file is to run.
The function of the *EK.hpp* file is to initiate. The function of initiate is the core function to cut the flow graph with maxflow.
The core function of *IG.hpp* is to convert the gray value matrix into a network flow graph, and use the weight= formula (gray value difference).

K-means

K-means algorithm is implemented by python. The code file for K-means is called ”*kmeans.ipynb*”. It can be easily run on Google Colab, it is a free Jupyter notebook environment that runs entirely in the cloud. To run this code, firstly, open the “*kmeans.ipynb”* and click “*Open in Colab”* button on the top left concern, which will directly launch the Google Colab on your computer. Next, download the image samples from GitHub and upload them to the Google colab. There is a file button on the left column where you can upload the file and please make sure that the image you upload is in the same category as the code. If you want to use different images to test, there is one line called `image = cv2.imread('xxx.jpg')` on the first code segment, which is used to read the image. You can change the different image names to test different image results.

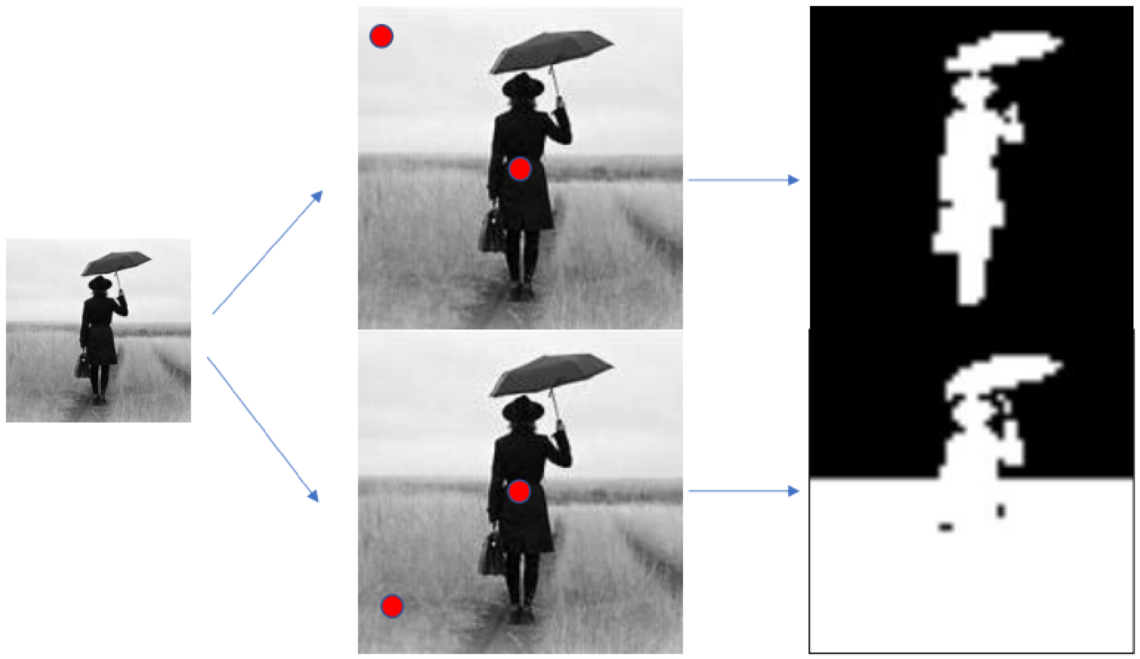
Gaussian Mixture

Simply run the ec504_GMM.ipynb in our github link.
To change picture: change the line *image_path = 'horse.jfif'* to your own image paths.
To change the number of clusters: change the line *ncomp=5* to a number between 2 to 5. If you want to enable more clusters, you should also change the color labels COLORS[] in the second code block.

Sample results with discussion

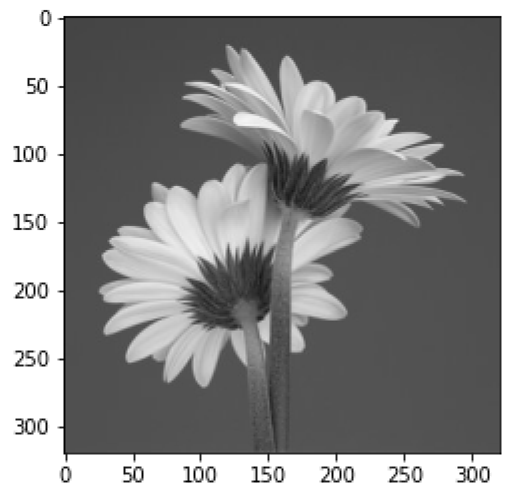
Maxflow

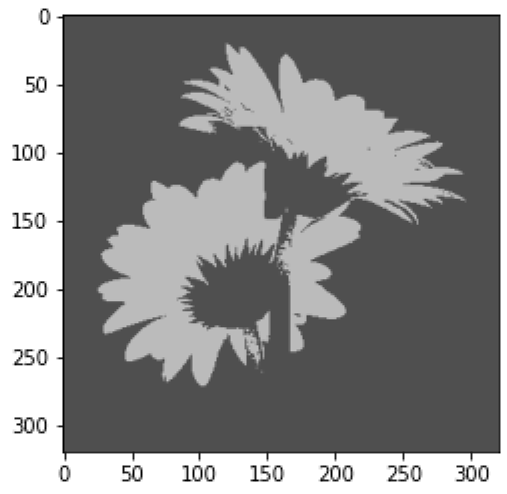
We choose two side of the picture, one as foreground and the other as the background. Left side as source, middle side as sink.



Kmeans

For grayscale images, let k = 2, and the result for kmeans algorithm is like:

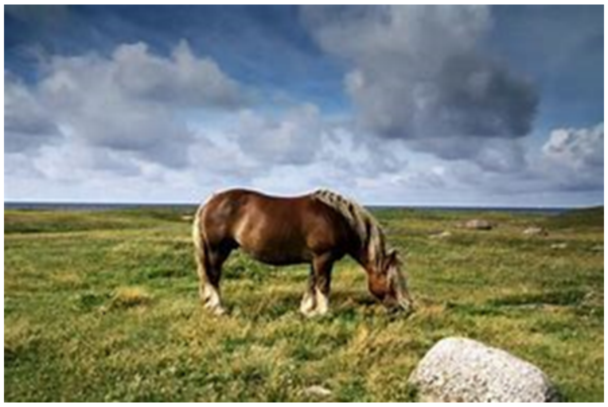




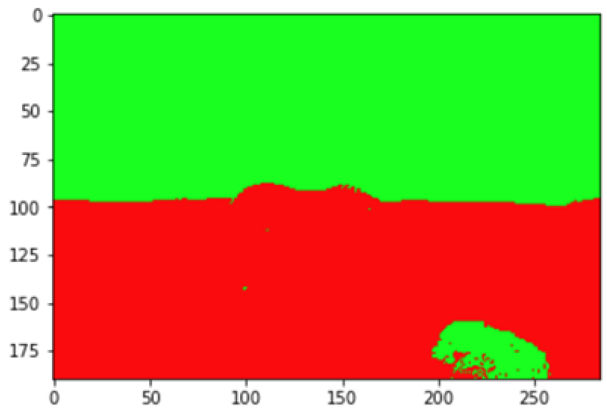
Left one is the original image and right is the result produced by kmeans algorithm.

Gaussian mixture

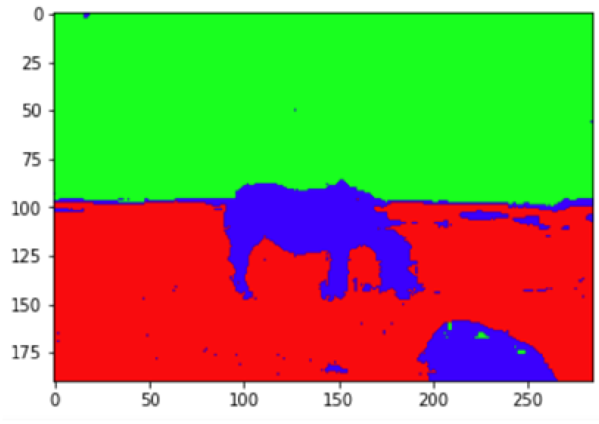
Here is the original test image.



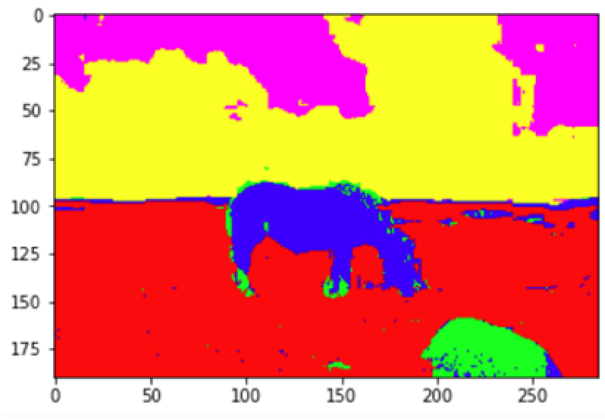
Here are segmented images using 2,3,5 gaussian distributions.



(K=2)



(K=3)



(K=5)

The results show that Gaussian distribution can nicely segment images in color.

References

[1] Nameirakpam Dhanachandra, Khumanthem Manglem, and Yambem Jina Chanu, Image Segmentation Using K -means Clustering Algorithm and Subtractive Clustering Algorithm, Procedia Computer Science, Volume 54, 2015, Pages 764-771, ISSN 1877-0509, https://doi.org/10.1016/j.procs.2015.06.090.
[2] Bishop, C. M. *Pattern Recognition and Machine Learning* (Springer-Verlag New York Inc., New York, NY, 2006)
[3]Boykov, Y., Kolmogorov, V. (2004). An experimental comparison of min- cut/max- flow algorithms for energy minimization in vision. IEEE Transactions on Pattern Analysis and Machine Intelligence, 26(9), 1124– 1137.
[4]Edmonds, J., Karp, R.M. (1972). Theoretical improvements in algorithmic efficiency for network flow problems. J. ACM, 19(2), 248–264.
[5]Ford, L.R., Fulkerson, D.R. (1955). Maximal flow through a network. Canadian Journal of Mathematics, 8, 399–404.