CS 766 Project Midterm Report

Chen Wang, Xiahe Liu

May 8, 2017

1 Introduction to the Problem

Semantic segmentation is a hot topic recently in the field of computer vision. The main task of semantic segmentation [Tho16] is to cluster parts of the images together which belong to the same object class. Semantic segmentation has a broad range of applications, for example, the autonomous driving car system will use this technology to distinguish different road signs [MBLAGJ+07]; Medical systems will use this technology to detect tumors from brains[MBVLG02], etc. And these interesting apps motivate us to investigate this topic. Semantic segmentation is definitely not a new topic and people have studied it decades ago, developing various of different methods revolving around this topic. Recently, more and more researchers use approaches with convolutional neural networks to solve semantic segmentation problems and have got quite big advance in accuracy. In this project, we would like to investigate both the traditional approaches and convolutional neural networks for this problem.

2 Current Progress

First of all, we have tried some traditional methods (K-means and mean shift) to do segmentation, however the performance is not that satisfying. Then we turn to convolutional neural networks to solve this problem. The details are as follows.

2.1 Traditional methods

For the first step, we tried two traditional ways of image segmentation, k-means and mean shift, to compare the performance and try to figure out if there any problems within these methods.

2.1.1 K-Means

We tried K-means in Matlab. The key idea of the K-means algorithm is to classify each single point in the image to one of the k clusters according to its distance (Euclidean space) to the mean of a cluster. We use the color (RGB) and spatial information of a image to constitute a Euclidean space. The key steps are shown as follows,
step 1: initialize the central points of each cluster.
step 2: assign each feature points to the cluster of the nearest central.
step 3: recalculate the central point for each cluster.
repeat step 2 and 3 until convergence.

2.1.2 Mean Shift

Different from K-Means, Mean Shift does not need to specify the total number of clusters at the beginning. Each hill (peak) is considered as a central point of a cluster. Every pixel in the image is assigned to the hill that it climbed to. The performance may depend on the size of the window. The key steps are as follows,
step 1: initialize a mode for each pixel, which represents the cluster this pixel belongs to.
step 2: for each mode, place a window around it with size w.
step 3: compute the mean of this window, and set the new mean value to the mode.
step 4: stop shifting when the difference of the new mode and previous mode is less than some threshold.
repeat step 2 to 4 for each mode. step 5: label the node with the same mode with the same cluster

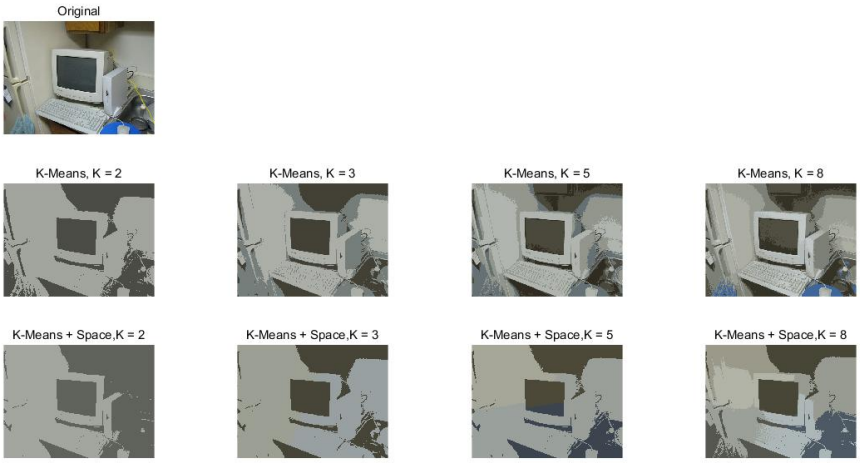

Figure 1: K-Means results for picture 1.

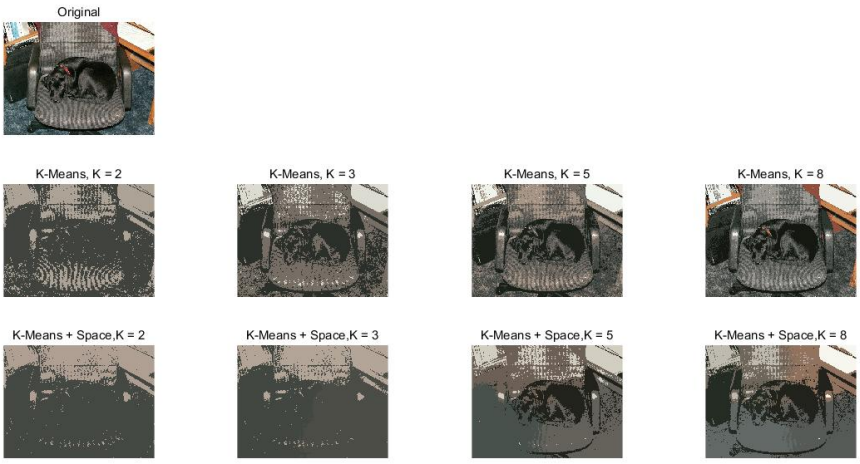

Figure 2: K-Means results for picture 2.

2.2 Simulation results

2.2.1 K-Means

For the second rows of Fig. 1 and 2, we use color information (RGB) as feature vectors, and we can see that smaller number of clusters leads to loss of some details of the images. However, larger number of clusters may split single object into unrelated pieces. In our simulations, the cluster number between 3 to 5 could be a good choice, but it varies for different images. For instance, $k = 3$ is better for picture 1, but $k = 5$ works better for picture 2. The third rows of Fig. 1 and 2 show the results of taking both RGB and spatial information into account. From the result, we can see that adding spatial information can improve the segmentation results, especially for the case that the color of objects are similar. In that kind of cases, using only color information cannot segment the objects very well. However, the main drawback of K-Means is that we must fix the number of clusters, and it is very sensitive to outliers, which can be seen from the results. There are lots of discrete segmented areas in the images.

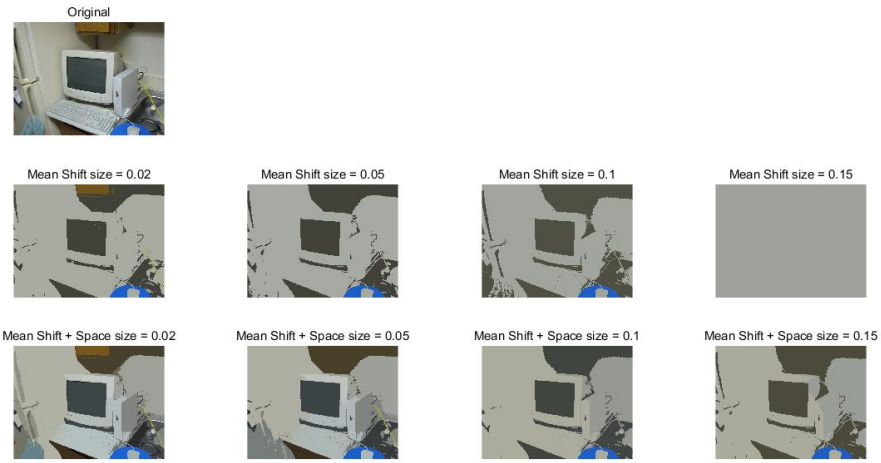

Figure 3: Mean Shift results for picture 1.

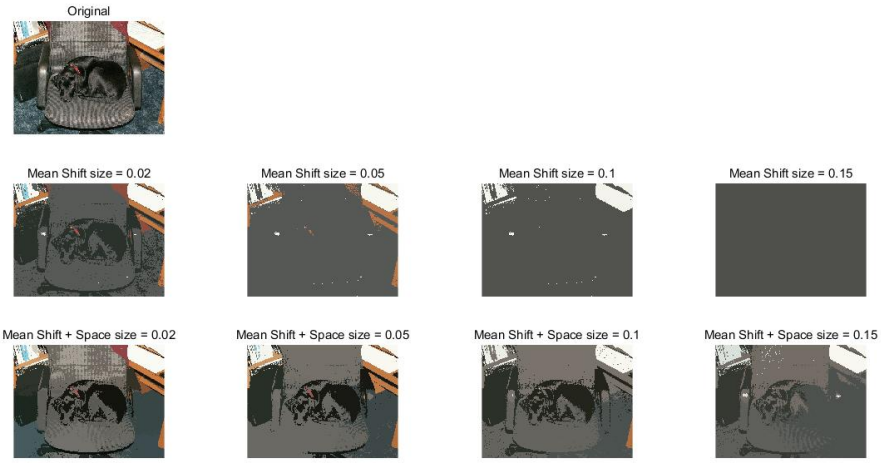

Figure 4: Mean Shift results for picture 2.

2.2.2 Mean shift

The second rows of Fig. 3 and 4 shows the result of mean shift algorithm using only color information as features. We can see that the performance is much better than that of K-Means. The number of clusters are not fixed, it depends on the image itself, the window size, as well as the threshold. As we increase the size of the window, the number of clusters could be reduced, we can get much clearer segments of objects. But different objects might be unified when the size of the window is large. What is more, Mean Shift is computational expensive, which spends much more time than K-Means. After adding the space information, the segmentation results can be further improved.

2.3 Convolution Neural Network

After reading through several papers related to this topic, we finally chose [LSD15] as our implementation reference for a couple of reasons. First, this paper was published in CVPR 2015 as best paper honorable mention, which is relatively new. Second, it is the first paper which trains fully convolutional neural network end-to-end for pixel-wise predictions and from supervised pre-training. Third, it proposes several upsampling methods to make sure that the output of the FCN is of the same dimension of the input image. What is more, it also proposes to use skip layers to produce accurate and detailed segmentations by combing semantic information from a deep, coarse layer with appearance information from a shallow, fine layer.

we planned to use caffe as our framework to build up our fully convolutional neural network model. It is not only because it is a very popular framework in the field of deep learning and computer vision, but also because that it is initiated and developed by Berkley Vision and Learning Center, where the authors of [LSD15] come from. Thus, they have a referenced implementation of their model written in caffe, which is a good resource for us to grasp caffe and FCN in semantic segmentation.

3 Discussion and Problems

When we came up with the idea of this topic at first time, we planned to implement and train our CNN based on the model proposed in the paper for we didn't expect that the paper has released a referenced implementation and made it open source. Though it is a pretty valuable learning resource, we are not quite sure whether we need to re-implement it using another framework or, we could do some further modifications based on their original model and do some comparable experiments? Another option would be to implement some other CNN models in other papers in order to make a comparison with [LSD15]. We will really appreciate your comments and suggestions on our future directions.

References

[LSD15] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015.

[MBLAGJ+07] Saturnino Maldonado-Bascon, Sergio Lafuente-Arroyo, Pedro Gil-Jimenez, Hilario Gomez-Moreno, and Francisco López-Ferreras. Road-sign detection and recognition based on support vector machines. *IEEE transactions on intelligent transportation systems*, 8(2):264–278, 2007.

[MBVLG02] Nathan Moon, Elizabeth Bullitt, Koen Van Leemput, and Guido Gerig. Automatic brain and tumor segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 372–379. Springer, 2002.

[Tho16] Martin Thoma. A survey of semantic segmentation. *arXiv preprint arXiv:1602.06541*, 2016.