One Shot Object Segmentation

Jagpreet Singh Chawla
School of Informatics, Computing & Engineering
Indiana University, Bloomington
Email: jchawla@iu.edu

Amit Kumar Yadav
School of Informatics, Computing & Engineering
Indiana University, Bloomington
Email: yadavam@iu.edu

Bivas Maiti
School of Informatics, Computing & Engineering
Indiana University, Bloomington
Email: bmaiti@iu.edu

***Abstract*—Semantic segmentation is a classic vision problem which has many applications which include, but is not limited to robotics, autonomous driving, virtual reality, etc. Unlike detection, segmentation provides a more precise object boundary which is especially useful in the field of robotics. Semantic segmentation is a highly researched topic, and most of the proposed methods require a large amount of data to train accurate models. While getting image data is not very difficult because of the internet, getting a large amount of annotated data requires a lot of resources. Therefore, in this project we are focusing on the problem of one-shot object segmentation, i.e. Segmenting an object using only one annotated example per class. We implemented approaches from two papers, compared and evaluated them on RGB-D Object Dataset, and improved the results by incorporating depth information using MRF. We also discuss the limitations of these models as seen on this dataset.**

1. Introduction

Semantic Segmentation is the per-pixel association of an image with different classes. This is a classic computer vision problem which has a wide range of applications, which includes, but is not limited to robotics, autonomous driving, virtual reality, etc. This is a widely researched topic, and strides of progress have been made in this area since the popularity of deep-learning. While we live in the age of data deluge with the number of available images in the internet growing exponentially, per-pixel level annotated data remains very difficult and expensive to obtain. This is the motivation for One-Shot Semantic Segmentation, which requires only one image per class. In this project, we are implementing two existing one-shot semantic segmentation models but on a different dataset.

- OSLSM [8] - This is one of the first works on this topic. This model uses a meta - learner approach to find the segmentation.
- Revolver[11] - This uses guided networks and a late fusion policy for support object mask.

We implemented these models on the RGB-D Object Dataset [6] to segment an object. We used only RGB information for the model inputs and then incorporated the depth-map at the post-processing step to obtain significant improvement over results.

2. Background and Related Work

Semantic Segmentation is a well researched topic in the Computer Vision community. After the popularity of deep learning, the state of the art in semantic segmentation has increased considerably. In [7], after the introduction of Fully Convolutional Networks, a number of segmentation networks like Segnet [1] and U-Net [9] were introduced with very good performances. All of these networks use transfer learning, with pre-trained networks like VGG [12] to extract features. For one-shot object segmentation, we extract features from a support image and segment the object in query image by finding similarities. Various approaches have been taken in the the past for one shot semantic segmentation. OSVOS[2] tackles the problem of segmenting all frames of a video into object and background given only the first frame. The approach taken by them is using pre-trained networks on large datasets and fine tuning them to discriminate the query object. In [5], Koch et al. explores a Siamese Network based approach to find similarity between inputs and segments the image based on that. In OSLSM[11], the authors introduce a novel experimental setup which has a two branch model where one branch extracts features from query image and other uses masked support image to generates model parameters to produce final segmentation. Continuing on the same experimental setup, Kate Rakelly et al. introduced a similar two-branched approach, using Guided Conditional Networks [8] and Late fusion. This model, named Revolver, should ideally work on sparse annotations too, i.e. in the extreme case, we have only one pixel denoting positive and one pixel denoting negative classes. Both OSLSM and Revolver were trained and tested on the Pascal VOC Datset[3]. We have implemented OSLSM and Revolver on the RGB-D Object Dataset[6].

3. Dataset

In this project, we have used the RGB-D Object Dataset [6]. The motivation for using this dataset is two-fold. We

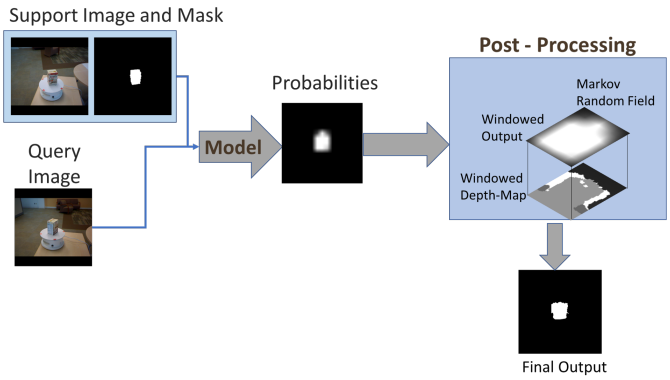

Figure 1. Experimental Setup

Figure 2. Semantic Segmentation includes Semantic Segmentation, Interactive Segmentation and Video Object Segmentation(Reprinted from [8])

wanted to implement the two models on a different dataset to check how well it works. In that regard, this dataset is vastly different from the Pascal VOC. It contains images of 300 common household objects organized into 51 categories. The second reason for using this dataset is that for each image, there is a depth map available which we are using for post-processing of the data. We also have the binary segmentation mask of each image. The entire dataset was created using a Kinect Style 3D Camera. The objects are kept on a turntable and spun around at a constant speed. The images are essentially frames from video sequence recorded from cameras placed about one meter from the turntable. Therefore, the dataset is divided into different scenes. We are only using 4 frames from each scene, generating unique pairs as our dataset. Two object classes and a few scenes are kept for testing. A total of 3408 image pairs are used for training.

4. Methods

In this section, we discuss the different approaches that we employed to achieve our segmentation goal.

4.1. Experimental Setup

In this project, we are using the experimental setup that was introduced by [11]. Unlike classical segmentation setup, where we have only one input image and one output image, in this case, we have 2 branches of input. In the top branch, we have the support image S, and its corresponding mask and in the bottom branch we have the Query Image to segment. Both OSLSM and Revolver has two branches. In case of OSLSM, we give the multiplied value of the mask and the actual image as the support input, whereas in Revolver we have the entire image and some annotations- either dense or sparse, as support input. Both the models give us probabilities of being in the same class as the support on all the pixels. We add a final post processing step after this to further refine the results. In post-processing, we add the depth information by Markov Random Fields using graph-cut method. The entire process is illustrated in Figure 1.

4.2. OSLSM

One-Shot Learning for Semantic Segmentation[11] is a two-branched meta-learner approach to segment out an object of interest using just one or a few annotated examples. This model takes 2 inputs, masked support image which acts as an example and a query image where it tries to segment the object. There are two branches in the model, each takes one of the inputs. The top branch is a VGG16[12] based meta-learner which takes the 224x224x3 masked support image and generates weights for a pixel level logistic regression which will be applied as 1x1 convolution to the features generated by bottom branch. Since VGG16[12] gives a 1000 dimensional output whereas we require 4097 parameters, a constant weight matrix is used as a hashing method to convert 1000 dimensional output into 4097 dimensional output. As per the original paper[11], this reduces the variance and over-fitting and increases dimensionality while avoiding any increase in number of parameters. Their hash function can be defined as:

$$\theta(i) = s(i)v(x(i))$$

where $s(i) \in \{-1, +1\}$ and $x(i)$ is a random index between $[1, 1000]$ and $v(i)$ is i^{th} unnormalized final layer output of VGG16 based model. This hash can be pre-calculated as a constant weight matrix using a discreet Dirac delta function.

The bottom branch of the model uses final convolutional layer (ignoring deconvolutions) of FCN32[7] to generate features. It takes 500x500x3 query image as input and generates 16x16x4096 feature vectors. Final output is generated by classifying each pixel using hashed output from top branch as parameters for a 1x1 convolution with sigmoid activation. The model can be further adapted for few-shot segmentation by producing separate segmentation using each support image and then taking the average.

We are using VGG16[12] pretained on ImageNet and FCN32[7] pretrained on VOC2011[3].

4.3. Revolver

Revolver is two-branched fully convolutional neural network, developed by Rakelly et. al.[8], used for segmentation.In this model, the task inputs are

$$\tau = \{\{(x_1, L_1), \ldots, (x_S, L_S)\}, x_1, \ldots, x_Q; x_s, x_q P_l(R^N)\}$$

$$L_s = \{(p_j, l_j) : j\{1 \ldots P\}\}, l \in \{1 \ldots K\} \cup \{\phi\}$$

where S is the number of support images x_s, in a support set, Q is the number of unannotated query images x_q, and

L_s are the support annotations. The annotations are sets of point-label pairs (p, l) with $|LS|$ = P per image, where every label l is one of the K classes or unknown (ϕ). The task outputs, that is the targets for the support-defined segmentation task on the queries, are

$$Y = (y_1, \ldots, y_Q), y_q = \{(p_j, l_j) : jx_q\}$$

The authors have decided to consider each task to be binary, that is, either a pixel belongs to the class (+ve) or it belongs to the background (-ve). This can be easily extended to multi-class problem by taking union of the binary sub-problems. Similar to the authors, in our implementation of Revolver, we are considering each task as a binary problem. Since the dataset that we are using contains object belonging to only one class in an image, we do not extend the tasks to multi-class problems. Moreover, our implementation works only under the one-shot condition. At inference time, similar to the original work, we are considering only one query image at a time since inference is independent of other images in the query set. Finally, we have only tested our implementation with dense annotations and can't comment on the performance of model when only sparse annotations are available.

The network is a fully convolutional network with two branches, a guide branch and an inference branch, and can be trained end-to-end. The guide branch learns the latent representation from the support images which helps guide the inference branch during segmentation of query images. Revolver uses VGG16[12], pretrained on ILSVRC[10], cast to its fully-convolutional form, as its backbone. The VGG-16 network acts as a feature extractor, in both the branches, to extract meaningful feature maps from an input image. Unlike previous works, in which the support mask is used as an extra channel for support image, in revolver, the support mask is fused with the feature maps as produced by the feature extractor, of support image by downsampling support mask to the size of feature maps using bi-linear interpolation. Global Average Pooling is applied on the resultant feature maps to make latent representation of a task space and time invariant. Under the conditions of few-shot learning, the latent representation from each support image is merged using m_s(averaging, authors and ours choice), such that m_s is differentiable. During inference time, the guides extracted by the guide network are fused with the feature maps, produced by the feature extractor, of query image. The fusing is done by tiling the guide to query image's feature maps size and concatenating it to channel dimension. A small fully convolutional network then produces a binary segmentation mask from the fused query-support features. In case of multi-class setting, the various binary mask can be squashed into a single mask to produce a multi-class mask for the query image. The resultant mask is then up-sampled using bi-linear interpolation to the original size of query image. Revolver has proven to work well in three domains of segmentation tasks: semantic segmentation, interactive segmentation and video-object segmentation. The network is designed to work well under the conditions of few-shot learning, that is, when there are only a few number of examples for a given instance(referred to as task in the paper), and when only sparse annotations are available, that is, not all the pixels are annotated in the input. Their approach "propagates pixel annotations across space for interactive segmentation, across time for video segmentation, and across scenes for semantic segmentation." [8]. The proposed method can also be extended to work under one-shot condition, that is, when only one example per class is available. Extending and adopting notations from [4] and [8], few-shot semantic segmentation can be defined in terms of (τ, γ) pairs where τ and γ are sampled from examples of an instance P.

4.4. Post Processing

After we get the output probabilities from the models, we do a post-processing step on it. Here we apply the depth Information by implementing Markov Random Fields using Graph Cuts. For this, we assume that any particular pixel is more likely not to be a boundary than a boundary. Thus we have defined a constant cost to each of the horizontal cuts. We also create a window of the object with the output probabilities that we received from the model after introducing a little slack. Our assumption is that the object will never be outside of this window. To create the window, we take the probabilities from the model and only take the rectangular window where the probabilities of being the object is more than a particular threshold. This significantly reduces our graph size and therefore computing time. Markov Random fields graph cut construction are given by the below equation:

$$E(x_1, x_2, x_3, ...x_n) = \sum_p D_p(x_p) + V_{p,q}(x_p, x_q),$$

where $E(x_1, x_2, x_3, ...x_n)$ is the cost of the entire cut, $V_{p,q}(x_p, x_q)$ is the cost of putting a segmentation boundary between pixel p and q, and $D_p(x_p)$ is the cost of assigning the pixel p to one of 2 different sets. In our case, we have the probability map and depth map as the two different costs of a pixel to be associated to any set. We first normalized the depth map to have values between 0 and 1 so that it would be comparable with the probability map. We also changed the cost function based on the depth map. If there exists considerable variance in the depth map, it would mean that there is important depth information which can help our segmentation. In that case we decrease the cost of cut to the depth map. On the other hand, if the depth map has little variance, we increase the cost of cut to the depth map, asking the MRF to gather most of its information from the probability map itself. It can be thought of as a tug of war between the depth map and the probability map based on the costs. After defining all the costs, we use the max flow min cut algorithm to solve for the segmentation. In some cases, due to bad depth image, or less than desirable probabilities, we didn't get a segmentation with the graph cut method, in which cases, segmentation was done by thresholding of the probability values. As it turns out, the post processing step improves the IOU values considerably.

Model	Unseen classes		Seen Classes	
	RGB	RGB-D	RGB	RGB-D
OSLSM	32.58%	42.64%	51.11%	63.73%
Revolver	14.76%	15.39%	18.76%	22.36%

TABLE 1. MEAN INTERSECTION OVER UNION OF OSLSM AND REVOLVER ON SEEN AND UNSEEN CLASSES

5. Results

As illustrated in Figure 1, we have probabilities as output from the models, then after post processing we have the final segmentation output. As a metric, we have chosen Intersection over Union(IOU), the standard metric for segmentation. We have performed our experiments on some seen classes and some never seen before classes. As illustrated in Table 1, the IOU for Seen Classes is more than that of unseen classes, and the OSLSM Model performs much better than Revolver for our Dataset. In all cases, the post processing step increases the IOU. Some examples of the results are illustrated in Figure 3, where we see the output from both the models for both seen and unseen classes. It is worth noting here that the "Results Without Depth" essentially means segmenting out all pixels with probability values above a particular threshold. These test results are based on our experiments involving only positive examples, i.e. examples where query image does contain the object which is annotated in support image. We also tested on negative examples, but the model doesn't perform well in these cases as it is sometimes try to segment an object instead of giving an empty mask. We discuss this further in next section.

6. Discussion

The Revolver model was supposed to be an improvement on the OSLSM model.But in our case, we obtain much better IOU values for the OSLSM model than the Revolver model. The reason for this could be that our dataset is vastly different from the one that the authors of Revolver trained and tested their model on. The classes of our model are much smaller in size compared to the classes of the PASCAL-VOC Database. There is a bilinear interpolation kernel which upsamples the output from a 6 x 6 output to a 224 x 224 output. Since in our dataset the object size is very small, it doesn't work very well, and produces a blocky output. We tried to mitigate this issue by changing the interpolation kernel by learnable deconvolution kernel and also by adding deconvolution layers before final layer, but it made the output worse and we got a grid-like output. Another interesting result we got from our OSLSM model is that it sometimes fails on negative examples and segments the wrong objects, i.e. when we give a query image which does not have the object from support image, it is still giving out a segmented output. This might be due to the the fact that the model was only trained using positive examples and the images in dataset always contain a single object. We plan to mitigate this issue by training the model with more negative examples and changing the loss function to impose high penalty on these cases. The original Revolver model should work for Few-Shot and sparse annotations, i.e. it should be able to take more than one support image, Our Revolver model currently works only for one-shot and dense annotations, we plan to enhance the model to work for the mentioned scenarios as well.

7. Conclusion

In this project, we have implemented two papers on a different dataset, and obtained interesting results. We learnt how these two models would perform on datasets vastly different from the one that they were trained on by the authors of the papers. We introduced a weighted cross entropy loss function to help the model by accounting for the imbalance in the number of negative and positive pixels. We also added a post processing step after we get the output probabilities from the models. For this, we used Markov Random Fields using Graph cut for further refining the results. While this does improve the IOU values, it is less desirable than an end to end CNN Network which would take a lot less time during inference. Post Processing of each image takes 1-4 second on average, and this is not desirable. In future, we plan to incorporate the Depth Channel directly into the network so that there is no need for any post processing, and the refined segmentation result can be obtained as a direct output from the network. We also need to mitigate the issue of segmenting out wrong images for negative examples, so in future we plan to change the loss function to account for the negative examples.

Acknowledgments

The authors would like to thank Dr. Md. Alimoor Reza for his invaluable guidance during our project.

References

[1] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. "Segnet: A deep convolutional encoder-decoder architecture for image segmentation". In: *IEEE transactions on pattern analysis and machine intelligence* 39.12 (2017), pp. 2481–2495.

[2] S. Caelles et al. "One-Shot Video Object Segmentation". In: *Computer Vision and Pattern Recognition (CVPR)*. 2017.

[3] M. Everingham et al. *The PASCAL Visual Object Classes Challenge 2011 (VOC2011) Results*. http://www.pascal-network.org/challenges/VOC/voc2011/workshop/index.html.

[4] Victor Garcia and Joan Bruna. "Few-shot learning with graph neural networks". In: *arXiv preprint arXiv:1711.04043* (2017).

[5] Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. "Siamese neural networks for one-shot image recognition". In: *ICML deep learning workshop*. Vol. 2. 2015.

[6] Kevin Lai et al. “A large-scale hierarchical multi-view rgb-d object dataset”. In: *2011 IEEE international conference on robotics and automation*. IEEE. 2011, pp. 1817–1824.

[7] Jonathan Long, Evan Shelhamer, and Trevor Darrell. “Fully Convolutional Networks for Semantic Segmentation”. In: *CoRR* abs/1411.4038 (2014). arXiv: 1411.4038. URL: http://arxiv.org/abs/1411.4038.

[8] Kate Rakelly et al. “Conditional networks for few-shot semantic segmentation”. In: (2018).

[9] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. “U-net: Convolutional networks for biomedical image segmentation”. In: *International Conference on Medical image computing and computer-assisted intervention*. Springer. 2015, pp. 234–241.

[10] Olga Russakovsky et al. “ImageNet Large Scale Visual Recognition Challenge”. In: *International Journal of Computer Vision (IJCV)* 115.3 (2015), pp. 211–252. DOI: 10.1007/s11263-015-0816-y.

[11] Amirreza Shaban et al. “One-Shot Learning for Semantic Segmentation”. In: *arXiv e-prints*, arXiv:1709.03410 (Sept. 2017), arXiv:1709.03410. arXiv: 1709.03410 [cs.CV].

[12] Karen Simonyan and Andrew Zisserman. “Very Deep Convolutional Networks for Large-Scale Image Recognition”. In: *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. 2015. URL: http://arxiv.org/abs/1409.1556.

Figure 3. Results: Rows- 1-3: OSLSM Seen Classes, 4-5: OSLSM Unseen Classes, 6-7: Revolver Seen Classes, 8-9: Revolver Unseen Classes, Columns- (a) Actual Image, (b) Output Probabilities, (c) Results Without Depth, (d) Results with depth, (e) Ground Truth