U-Net Based Multi-instance Video Object Segmentation

Heguang Liu
Stanford University
Uber Technologies, Inc
heguangl@stanford.edu

Jingle Jiang
Stanford University
Uber Technologies, Inc
jxj142@stanford.edu

Abstract

Multi-instance video object segmentation is to segment specific instances throughout a video sequence in pixel level, given only an annotated first frame. In this paper, we implement an effective fully convolutional networks with U-Net similar structure built on top of OSVOS fine-tuned layer. We use instance isolation to transform this multi-instance segmentation problem into binary labeling problem, and use weighted cross entropy loss and dice coefficient loss as our loss function. Our best model achieves F mean: 0.467 and J mean: 0.424 on DAVIS dataset, which is a comparable performance with the State-of-the-Art approach. But case analysis shows this model can achieve a smoother contour and better instance coverage, so it's better for recall focus segmentation scenario. We also did many experiments on other convolutional neural networks, including SegNet, Mask R-CNN, and provide insightful comparison and discussion.

1. Introduction

Video object segmentation targets at segmenting a specific object throughout a video sequence, given only an annotated first frame [17]. It requires labeling each instances in pixel level accuracy. Multi-instance video object segmentation is very challenging because when instances occlude each other often causing failure in tracking. In recent years, it has gained increasing popularity due to its wide usage in autonomous driving vehicles, motion tracking, video summarization, etc.

In this project, DAVIS (Densely Annotated VIdeo Segmentation) 2017 Dataset [14] will be used for training and evaluating. Specifically, the input is a sequence of RGB images, and an annotated first frame to indicate the object of interest. The output is the annotation of each instance for each frame. This paper mainly focus on approaches which doesn't use temporal information, to meet the real-time processing need.

Figure 1. Example annotations of DAVIS 2017 dataset

After experimenting many approaches, including SegNet [1], U-Net and Mask R-CNN [15], we identified a fully convolutional neural networks with an U-Net similar architecture, on top of an OSVOS (One-Shot Video Object Segmentation) [2] fine-tuned layer achieves the best result. This model has a comparable performance with the non-temporal State-of-the-Art OSVOS on DAVIS 2017 dataset. From the case analysis, we found the annotation produced by this model is more complete and with a smoother contour, compared with OSVOS. Thus it's a better model for recall focus scenario whereas OSVOS for contour accuracy focused scenario.

This paper also includes insightful discussion about choice of object isolation, loss function, model structure as well as failure analysis. We hope it can be useful for future researcher.

2. Related Work

2.1. Image Segmentation

The State-of-the-Art image semantic segmentation solution typically takes a encoder-decoder architecture, as popularized in Long, et al's work [10]. Briefly, the encoder is a pre-trained classification network like VGG [5]/ ResNet [7]; and decoder's task is to semantically project the discriminative features (lower resolution) learned by the encoder onto the pixel space (higher resolution) to get a dense classification.

Few different decoding mechanism has been proposed over the years. When the first breakthrough of semantic seg-

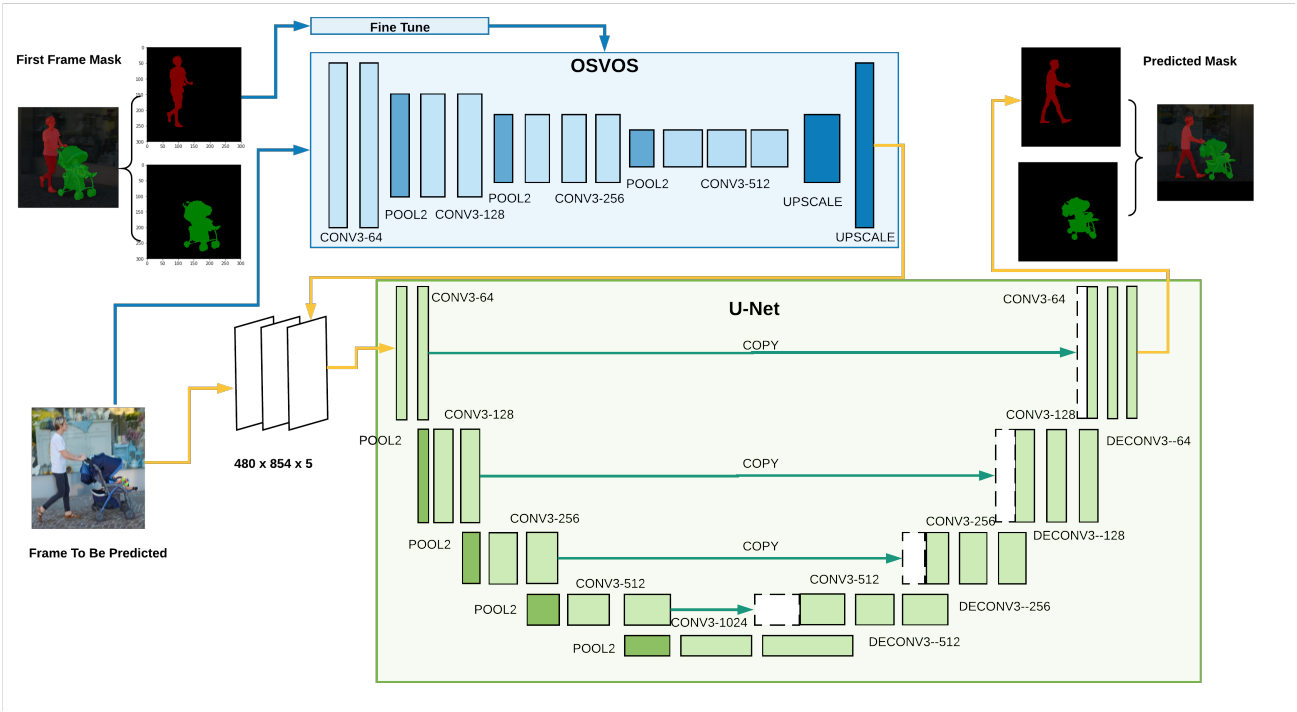

Figure 2. U-Net based Fully Convolutional Networks Architecture

mentation arrives with Long, et al [10] they only employed a coarse up-sampling for the decoder architecture. Although a good start, the heatmap produced using this technique is quite coarse. Later, SegNet [1] introduced transposed convolutional layers into the decoder work. Built on similar idea, U-Net [15] further introduced skip connection to improve.

Another different approach is Region-Based Semantic Segmentation, where a detection is performed first followed by segmentation. It often relies on a two-stage processing: with Region of Interest (ROI) proposal followed by a heavy-lifting working network. Earlier effort on the first stage has been focusing on "segment candidate proposal", e.g. DeepMask [13]. But it turns out that these methods are slow and less accurate because segmentation *precedes* recognition. Most recently, He, et al proposed an elegant solution Mask R-CNN [4]. It extends Faster R-CNN by adding a branch for predicting segmentation masks in parallel with the existing branch for classification and bounding box recognition. Mask R-CNN is an end-to-end model, outperforms all existing, single-model entries on the architecture for Pascal VOC [3] and MSCOCO [9] challenges.

2.2. Video Object Segmentation

There are three leading approaches to Video Object Segmentation, OSVOS [2], MaskTrack [11] and Recurrent mask propagation [17].

The current non-temporal State-of-the-Art is OSVOS. OSVOS approach first converts a pre-trained image classification CNNs, eg VGG-16, to a fully convolutional network by removing the FC layer and insert new loss, then train this fully convolutional network on the DAVIS dataset. By fine-tuning this network with the first frame of the video, OSVOS generates a one-time model and test it on that entire sequence, using its new weight. This approach shows that we can achieve great result without using temporal information of the video. The model we developed is on top of this approach.

MaskTrack, on the other hand, feed the predicted mask of the previous frame as additional input to the network, to make the input 4 channels. Then train a CNN eg VGG-16, from scratch on a combination of semantic segmentation and image saliency datasets. Finally it adds an identical second stream network, based on optical flow input. Variation approaches like Online Adaptation [16] and re-identification [8] has also gain great result.

A new released Recurrent Mask Propagation [17] approach formulates a deep recurrent network that is capable of segmenting and tracking objects in video simultaneously by their temporal continuity, so it's able to re-identify them when they re-appear after a prolonged occlusion and achieve great result. This approach uses temporal info, which is not the focus area of this paper.

3. Methods

3.1. Architecture

For this paper, we propose an U-Net based fully convolutional networks architecture, as shown in Figure 2.

3.2. Key Components

3.2.1 Instance Isolation

To isolate the instance of the multi-instance mask, we explored 3 approaches:

- Using the raw multi-instance mask for prediction. The input dimension is H x W x 1, value ranges [0,1,2...N]
- Project the multi-instance mask to a input of dimension H x W x N, where each layer is the binary labeling of a instance, value ranges [0, 1]
- Isolate the multi-instance mask image to N binary label inputs, each with dimension H x W x 1, values ranges [0, 1]

After experiments and analysis, we found the third approach is the most effective one. Because it simplifies the multi-instance segmentation problem to a single-instance binary segmentation problem.

3.2.2 OSVOS

OSVOS is using a VGG-16 pre-trained on ImageNet [6] as a backbone, removes the FC layer to make it a fully convolutional network, and then trained on DAVIS dataset. After getting this parent model, we use the first frame of the video to tune this model for 500 iterations, which generates a customized model for each video sequence. This fine-tune step turns out to be very important. We used open source OSVOS code to accomplish this step. [2]

3.2.3 U-Net based Fully Convolutional Networks

DAVIS dataset only has 4219 training images, so we use a fully convolutional networks, U-Net, to tackle small training data size problem. We first used a series of convolutional layer and max pooling layer to construct a contracting path, in order to capture enough context from the image. Then we used up-sampling layer to replace pooling layer to increase the output resolution. This is a symmetric expanding path of the contracting path. In order to localize, we crop and merge the high resolution feature from the contracting path with these up-sampled output, then send them to successive convolution layer to assemble a more precise output. For this U-Net, we experiment different number of filters in each layer to get the best result.

3.2.4 Layer: Loss function

We mainly explored two types of loss function - weighted cross entropy loss and dice coefficient loss - both of which are typically used for segmentation task.

- Weighted Cross Entropy Loss

$$\boldsymbol{L} = -\sum_{\boldsymbol{x}} \boldsymbol{\omega}(\boldsymbol{x})\, \boldsymbol{p}(\boldsymbol{x}) \log \boldsymbol{q}(\boldsymbol{x}).$$

  where $p(x)$ denotes the true distribution. $q(x)$ denotes the predicted distribution from the neural network. $\omega(x)$ is a weight coefficient.

  Weight coefficient is to scale up the contribution from foreground class when foreground only constitute small area comparing to background. Without the weight, the model would simply predict all pixels as background. We set $\omega(x)$ as the ratio of background pixel count and foreground pixel count.

- Dice Coefficient Loss

$$\boldsymbol{L} = \mathbf{1} - \frac{\mathbf{2}|\boldsymbol{X} \cap \boldsymbol{Y}|}{|\boldsymbol{X}| + |\boldsymbol{Y}|}$$

  where $|.|$ denotes $L1$ norm. Dice loss is bound between 0 and 1 where 0 suggests no similarity and 1 suggests completely overlapping. It's a more interpretable loss compared with cross entropy.

During experiment, we observed these two loss functions are positive associated, and weighted cross entropy loss is more numerical stable than dice coefficient loss. So we use the weighted cross entropy loss to train the model and use both during model evaluation.

4. Dataset and Metrics

4.1. DAVIS Dataset

DAVIS 2017 Challenge [14] on Video Object Segmentation was used for this project. DAVIS dataset spans multiple occurrences of common video object segmentation challenges, such as occlusions, motion blur and appearance changes. On top of DAVIS 2016 dataset [12], the 2017 dataset added multi-instance challenge. This requires semantic/instance-level segmentation. A detailed summary of the dataset is shown in Table 1.

4.2. Evaluation Metrics

We used official evaluation code from DAVIS-2017 to report our model performance. Specifically, we evaluated the following:

- **Region Similarity**: the intersection-over union between the predicted mask M and ground-truth G. Note

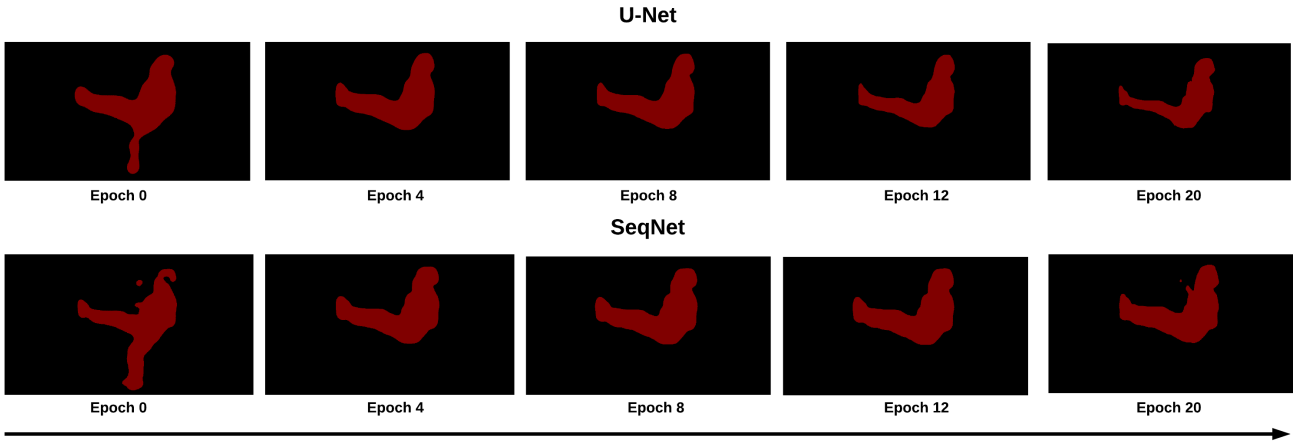

Figure 3. Comparison of SegNet and U-Net over training epochs

Table 1. DAVIS 2017 Dataset

	Train	Val	Test-Dev
Number of sequence	60	30	30
Number of images	4219	2023	2037
Mean number of objects	2.3	1.97	2.97

the similarity between this metric and the Dice coefficient loss we introduced earlier.

$$J = \frac{|M \cap G|}{|M \cup G|}$$

- **Contour Accuracy**: the Harmonic mean of contour's precision and recall

$$F = \frac{P_c * R_c}{P_c + R_c}$$

4.3. Training Setup

The model was built on tensorflow 1.8 framework and implemented using Python 3.6. We've written 3000+ lines of code from scratch to pre-process image, isolate objects, construct U-Net, SegNet graph, track training progress and evaluate model. Besides original code, we used the davis-2017 code repository for model evaluation and OSVOS code repository for constructing OSVOS layer.

Our model was trained on N1-HighMem-8 instance, with v8CPU, 52 GB memory, on Google Cloud Compute Platform. We've also attached NVIDIA Tesla P100 GPU with 16GB memory to the virtual machine. Due to the sheer amount of model structure/parameters we have experimented upon, in total 5 GPUs are used for this project.

5. Experiments and Results

5.1. Model Performance Comparison

We experimented a few different architectures. The results of the two most promising architecture, SegNet and U-Net, are summarized on Table 2. It is observed that best U-Net has a J-mean comparable with the State-of-the-Art OSVOS, in spite of a slightly worse F-mean. Figure 3 shows the convergence pattern of the two architectures on an example training image.

Table 2. Model Performance Comparison

Model	J Mean	F Mean	Dice Loss	CE Loss
OSVOS	0.499	0.592	-	-
SegNet	0.347	0.214	0.407	1.001
Best U-Net	0.424	0.467	0.289	1.029

Furthermore, a sample annotation comparison between the two is shown on Figure 4. From Figure 4 (c), we noticed U-Net tends to produce a more complete instance with a better coverage and smoother contour in exchange of contour accuracy (i.e. lower F value). As a comparison, OSVOS tends to have less coverage and a rigid contour. We believe our model works better for recall-focus scenario, like pedestrian segmentation.

SegNet, on the other hand, has a much coarse contour with lower precision. Its F-mean is much lower compared with U-Net. The reason is likely to be the absence of merging step, which connects the high resolution features from contracting path with the up-sampling features from the expanding path. Without this connection, SegNet tends to

”forget” high-resolution details by the end of the contracting process. We can see this clearly in Figure 4 (d).

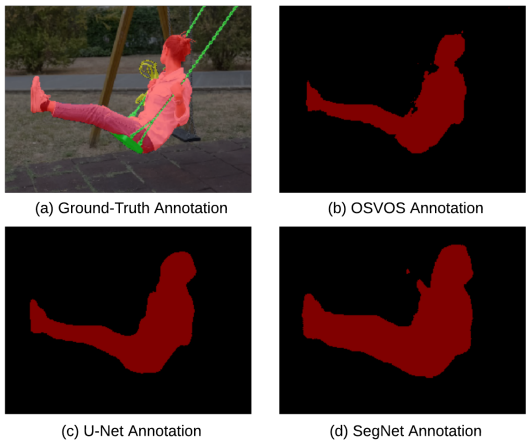

Figure 4. Comparison of Annotation Produced by Different Models

5.2. Hyper-parameters Tuning

For U-Net, we carried many hyper-parameters tuning experiments, including tuning on number of filters in each U-Net layer, batch size and learning rate. The experiment results are summarized in Table 3.

Table 3. U-Net Hyper-parameters Tuning Results

U-Net Filter	J	F	Params	Lr	Batch
16,32,64	0.314	0.163	700K	4e-5	20
32,64,128	0.345	0.325	1M	4e-5	20
64,128,256,512	0.419	0.430	31M	2e-5	8
64,128,256,512	0.424	0.467	31M	4e-5	8

Figure 5 shows the train loss (first row) and test loss (second row) over 10K iterations for two of our hyper-parameters tuning experiment. the periodic up and downs in training loss is because we were not able to do shuffling on training dataset, due to GPU memory limitation. Although learning rates were different, qualitatively they showed similar trend. The training loss keeps going down while the test loss reaches the best point and goes up afterwards.

Over simplified U-Net with 700K parameters wasn't able to achieve good performance (table 3). The best U-Net Model has convolutional filter number of 64, 128, 256, 512, contains 31M trainable parameters, with learning rate 4e-5 and batch size 8. This model is not over-fitting because both training and test loss is around 1.02. Due to the fact that U-Net has 31M+ parameters in the graph, the batch size can't go beyond 8 to prevent GPU out of memory. The training of the best U-Net takes around 1 hour per epoch and 8 hours to fully converge.

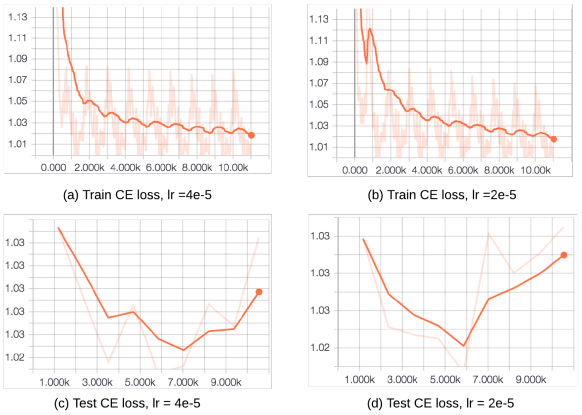

Figure 5. Train Loss and Test Loss with different learning rate

5.3. Case Analysis

We list 4 representative test annotations produced by our best U-Net model, as shown in Figure 6. Based on the case study, we have the following observations:

1. The model can handle intensive motion and appearance change gracefully.

   In Figure 6(a) drift chicane sequence, the annotation starts with a small and far car object, which makes multiple turns accompanied by drastic appearance change, distance change, and background fogs. The model keeps good and sharp track on the moving vehicle.

2. The model can handle multi-instances with similar motion very well, even with overlapping.

   In Figure 6(b) Horse Jump sequence, the rider and the horse has similar motion. The model can keep track of the two objects and draw relatively clear boundaries.

3. The model doesn't handle multiple object collusion very well.

   In Figure 6(c) Camel sequence, the annotation is very accurate in the first few frames, but when the target camel bypass another camel, the model starts tracking both camels afterwards.

4. The model lost track when object goes beyond image boundary and goes back.

   In Figure 6(d) Motocross jump, the annotation is very accurate in the first few frames, but when the rider moves out of the image boundary, the model lost track of the rider and start tracking only the motorcycle afterwards.

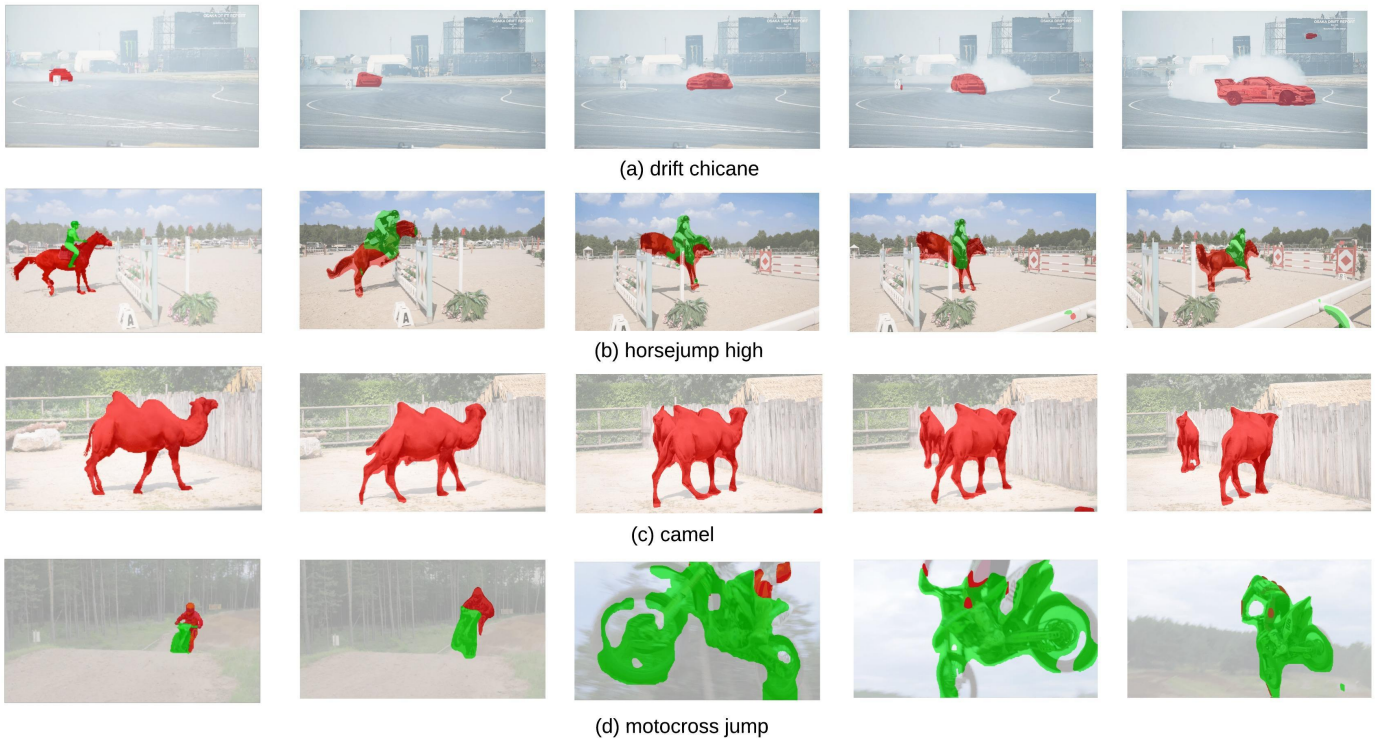

Figure 6. U-Net Predicted Annotations on DAVIS 2017 Test Set

5.4. Failed Experiments and Discussions

For this paper, we also explored few other approaches, which did not achieve a good result. We still want to report them and hope it can help with future research.

- **Mask R-CNN**: We noticed Mask R-CNN didn't produce reasonable result for semi-supervised video segmentation problem as in the case of DAVIS-2017 dataset, for two reasons. First, being a instance segmentation algorithm, Mask R-CNN requires *recognition* for *segmentation*. This caused difficulty when DAVIS dataset tracks object that Mask R-CNN model has not seen before. Figure 7 is a classic example where Mask R-CNN, trained with COCO dataset that did not include camel, was used to segment camel sequence. Consequently, Mask R-CNN mistakingly predict the camel as an horse with 0.998 confidence and further tried to correct the shape to be a horse, hurting the annotation accuracy.

Figure 7. Annotation Produces by Mask R-CNN

Another failure reason on vanilla Mask R-CNN is it was not able to incorporate the semi-supervised information provided on the first image. This often lead to unnecessary objects being segmented in the following frames, as well as wrong segmentation orders for the original targets. However, this problem may be mitigated by training the Mask R-CNN backbone on the first image of each video sequence, similar as OSVOS's approach.

- **Unweighted Cross Entropy as Loss Function**: For DAVIS dataset, since most of the instance is very small compared with the background, when we tried to use unweighted cross entropy as loss function, almost all the pixels are predicted as background.
- **Direct Feed Multi-instance Image**: We also tried to feed multi-instance image to the networks directly and transform this problem into a multi-class classification problem. Because convolutional networks is not able to project the value of 1, 2, 3 to layer 1, 2, 3. The result produced by this approach is also very poor.

6. Conclusion

For this paper, we implement and compare a number of fully convolutional networks to tackle the multi-instances video object segmentation problem. Among SegNet, U-Net, Mask R-CNN, U-Net based architecture achieves the best result with **F mean: 0.467** and **J mean: 0.424** . This result is comparable to the current State-of-the-Art approach on DAVIS Dataset. Based on the model comparison, we found the annotation produced by this model is more complete and with a smoother contour, compared with OSVOS. Thus it works better under recall focus scenario.

From the case study, we noticed this model doesn't perform well on 2 cases: 1). Multi-instance occlusion 2). Instance lost tracking after it goes out of image boundary and goes back. In the future, we could try 1) Recurrent Neural network to better tracking each object by its temporary continuity to handle occlusion. 2) Adaptive object re-identification to prevent target lost.

7. Acknowledgements

We would like to thank Professor FeiFei Li, Instructor Justin Johnson and Serena Yeung and all the TAs for the great class experience and project setting. Also want to thank Google Cloud for sponsoring GPU instances for model training.

References

[1] V. Badrinarayanan, A. Kendall, and R. Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *CVPR*, 2015.

[2] S. Caelles, K.-K. Maninis, J. Pont-Tuset, L. Leal-Taixe, D. Cremers, and L. V. Gool. One-shot video object segmentation. *CVPR*, 2017.

[3] M. Everingham, L. V. Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *Int J Comput Vis DOI 10.1007/s*, 2009.

[4] K. He, G. Gkioxari, P. Dollr, and R. Girshick. Mask r-cnn. *CVPR*, 2017.

[5] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Very deep convolutional networks for large-scale image recognition. *CVPR*, 2015.

[6] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *NIPS*, 2017.

[7] X. Li, Y. Qi, and Z. Wang. Video object segmentation with re-identification. *CVPR*, 2017.

[8] X. Li, Y. Qi, Z. Wang, K. Chen, Z. Liu, J. Shi, P. Luo, C. C. Loy, and X. Tang. Video object segmentation with re-identification. *CVPR*, 2017.

[9] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollar. Microsoft coco: Common objects in context. *ECCV*, 2014.

[10] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. *CVPR*, 2016.

[11] F. Perazzi, A. Khoreva, R. Benenson, B. Schiele, and A.SorkineHornung. Learning video object segmentation from static images. *CVPR*, 2017.

[12] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. V. Gool, M. Gross, and A. Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. *CVPR*, 2016.

[13] P. O. Pinheiro, R. Collobert, and P. Dollar. Learning to segment object candidates. *CVPR*, 2016.

[14] J. Pont-Tuset and F. Perazzi. The 2017 davis challenge on video object segmentation. *CVPR*, 2017.

[15] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. *CVPR*, 2015.

[16] P. Voigtlaender and B. Leibe. Online adaptation of convolutional neural networks for video object segmentation. *BMVC*, 2017.

[17] C. C. L. Xiaoxiao Li. Video object segmentation with joint re-identification and attention-aware mask propagation. *CVPR*, 2018.