[Samsung PRISM] Mid Review Report

Double exposure Videography

Team

1. College Professor(s):Dr. Rajeshwari B S
2. Students:
    1. Nidhi Prakash
    2. Swastika
    3. Neha K
3. Department: BMSCE - Information Science and Engineering

Date: 26th Feb 2019

Work-let Area – AI, ML | Double exposure Videography

SAMSUNG

Work-let expected duration – 6 months

2 **Members**

Problem Statement

- Double exposure Videography basically involves combining 2 video feeds to create effect on a portion of an video.
- Develop a AI model to apply Human segmentation on a video with HD resolution at 30fps. Apply Double exposure Videography on the identified Human region or background.

Sujay Udupa, Staff Engineer
sujay.udupa@samsung.com
9886243375

Akhil Nichenametla, Engineer
a.nichenamet@samsung.com
9703952630

Expectations

- Literature survey to identify the various ways to achieve Human segmentation on a video feed.
- Develop/Improve existing or new model for Human segmentation.
- Identify the Training Data Requirements and Create the required data set for this task
- Achieve >25fps on HD quality video

Training/ Pre-requisites

- Coursera trainings available for ML basics
- C++/Python Programming

Reference docs

- None

Kick Off <1st Month>

- Understanding Deep Learning concepts.
- Understanding Image Processing concepts
- Understand the problem in detail
- Literature Survey for Human segmentation

Milestone 1 <2nd Month>

- Identify the data requirements
- Preliminary Model architecture
- Describe the limitations of the solution space
- Consider non AI solution and make comparisons

Milestone 2 <4th Month>

- Final model architecture
- Training and validation of the model
- Report preliminary results

Closure <6th Month>

- Final model
- Runtime and memory performance optimizations
- Data augmentation if required

Proposed Approach /(Solution 1)

SAMSUNG PRISM

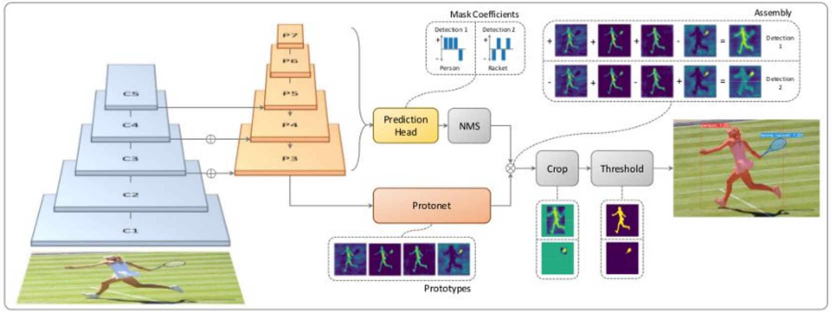
- **Concept Diagram** :
  (Clear detailed schematic / block diagram / flow chart depicting the proposed concept / solution)

YOLACT ++

Yoact + +: real time instance segmentation, from 29.8map/33.5fps to 34.1map/33.5fps

Yolact is the first real-time instance segmentation algorithm. Yolact + + optimizes yolact from three aspects of backbone network, branch and anchor, and improves 5map on the premise of maintaining real-time performance.

- Yollact + + is based on the mask interception of the whole graph, and zero padding is used for the insufficient size. However, mask scoring r-cnn uses the result of mask branches superimposed by ROI pooled features.
- Yolact + + does not use the full connection layer, which is the key to maintain the speed. It only increases the computing time by 1.2ms, while the module of mask scoring r-cnn needs 28ms.



Method	Backbone	FPS	Time	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
PA-Net [14]	R-50-FPN	4.7	212.8	36.6	58.0	39.3	16.3	38.1	53.1
RetinaMask [50]	R-101-FPN	6.0	166.7	34.7	55.4	36.9	14.3	36.7	50.5
FCIS [3]	R-101-C5	6.6	151.5	29.5	51.5	30.2	8.0	31.0	49.7
Mask R-CNN [2]	R-101-FPN	8.6	116.3	35.7	58.0	37.8	15.5	38.1	52.4
MS R-CNN [15]	R-101-FPN	8.6	116.3	**38.3**	58.8	41.5	17.8	40.4	54.4
YOLACT-550	R-101-FPN	**33.5**	**29.8**	29.8	48.5	31.2	9.9	31.3	47.7
YOLACT-400	R-101-FPN	45.3	22.1	24.9	42.0	25.4	5.0	25.3	45.0
YOLACT-550	R-50-FPN	45.0	22.2	28.2	46.6	29.2	9.2	29.3	44.8
YOLACT-550	D-53-FPN	40.7	24.6	28.7	46.8	30.0	9.5	29.6	45.5
YOLACT-700	R-101-FPN	23.4	42.7	31.2	50.6	32.8	12.1	33.3	47.1
YOLACT-550++	R-50-FPN	33.5	29.9	34.1	53.3	36.2	11.7	36.1	53.6
YOLACT-550++	R-101-FPN	27.3	36.7	34.6	53.8	36.9	11.9	36.8	55.1

TABLE 1: **MS COCO [10] Results** We compare to state-of-the-art methods for mask mAP and speed on COCO t and include several ablations of our base model, varying backbone network and image size. We denote the backbone archite network-depth-features, where R and D refer to ResNet [8] and DarkNet [1], respectively. Our base model, YOLAC ResNet-101, is 3.9x faster than the previous fastest approach with competitive mask mAP. Our YOLACT++-550 model with has the same speed while improving the performance of the base model by 4.3 mAP. Compared to Mask R-CNN, YOLACT 3.9x faster and falls behind by only 1.6 mAP.

Proposed Approach / Solution (Solution 2)

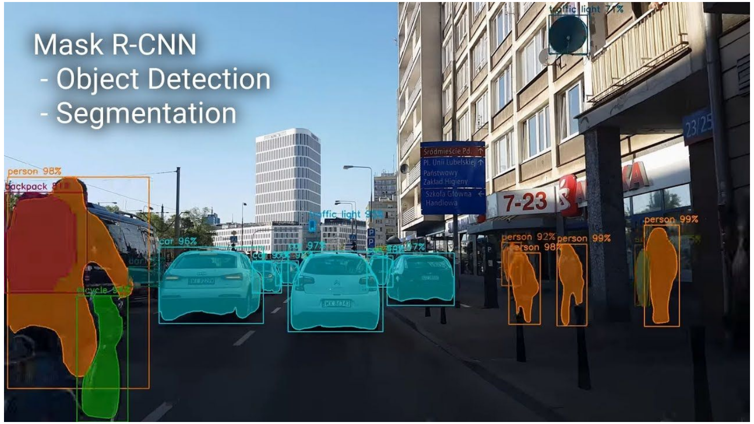
- **Concept Diagram** :
  (Clear detailed schematic / block diagram / flow chart depicting the proposed concept / solution)

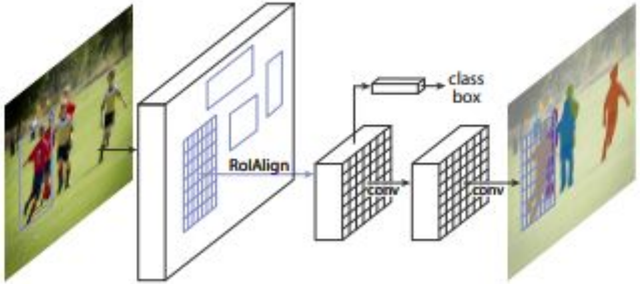
Mask R-CNN

Mask R-CNN is a two-stage **instance segmentation** in which each pixel is assigned to an individual object model that can be used to localize multiple objects in an image down to the pixel level. The first stage of the model extracts features (distinctive patterns) from an input image to generate region proposals that are likely to contain objects of interest. The second stage refines and filters those region proposals, predicts the class of every high-confidence object, and generates a pixel-level mask for each object.

Mask R-CNN architecture: Mask R-CNN is very similar to Faster R-CNN except there is another layer to predict segmented. The stage of region proposal generation is same in both the architecture the second stage which works in parallel predict class, generate bounding box as well as outputs a binary mask for each RoI.

- Backbone Network
- Region Proposal Network
- Mask Representation
- RoI Align



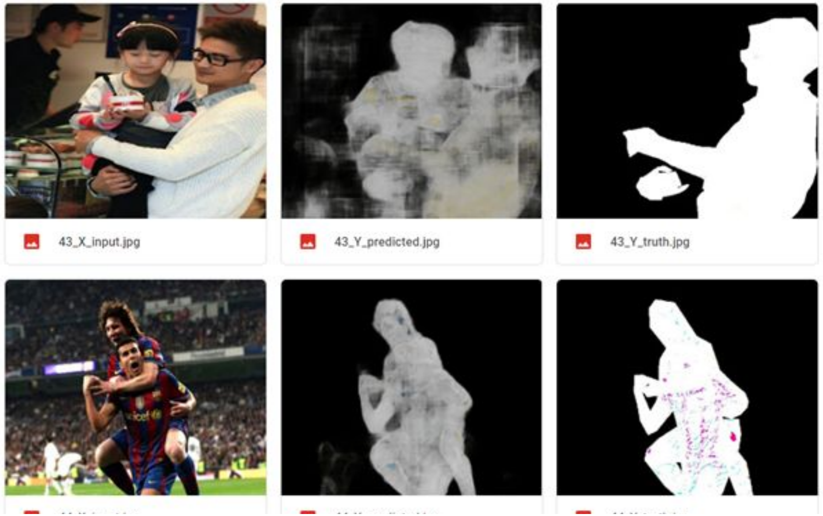


Other Frameworks

U-Net

U-net Segmentation can be used for semantic segmentation in which each pixel is assigned to an object category, we need to reconstruct the image from the feature vector created by CNN. So, here we convert the feature map into a vector and also reconstruct an image from this vector.

Images with black background: You may notice that in the 43 predicted image (43_Y_predicted.jpg), you can see that we have a mask (43_Y_truth.jpg) for the person at the right only. Well, after 44 epoch the Google Colab got crashed.



FCN

FCN realizes an end-to-end pixel-wise image semantic segmentation by fully convolutional layers. In contrast to traditional CNNs containing fully connection layers in the end, FCN maintains the spatial information of images well by using the convolutional layer. After pooling layers, the resolution of feature maps is reduced by a factor compared with the size of the original input image.

TABLE 1. Performance comparison between FCN and B-FCN.

Method	image size	IOU(%)	Segmentation Accuracy(%)	Time (ms)	GPU
FCN-32s	480×480	75.06	81.99	**27.0**	GTX1080
B-FCN-32s	480×480	79.18	87.52	34.9	GTX1080
FCN-16s	480×480	78.40	83.65	27.1	GTX1080
B-FCN-16s	480×480	82.99	89.99	35.0	GTX1080
FCN-8s	480×480	76.51	84.61	27.3	GTX1080
B-FCN-8s	480×480	**84.78**	**90.60**	35.7	GTX1080

The changes of loss value in training and testing processes, and accuracy in the test process of FCN-32s are plotted as curves in the figure. The horizontal axis represents the iteration times, where all training data are used in single iteration. The left vertical axis gives loss value and the right vertical axis depicts accuracy values. As can be seen from Fig. 7, in the third iteration the network obtains the highest test accuracy 94.1%, the lowest training and test loss value is 0.162, which are shown as black arrows in the figure.

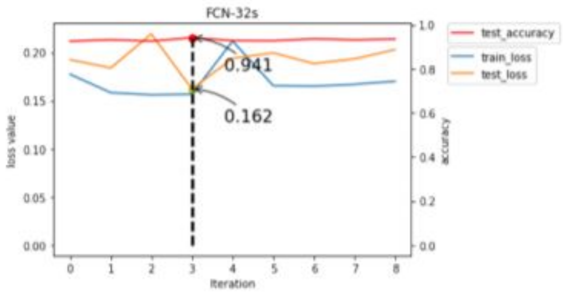


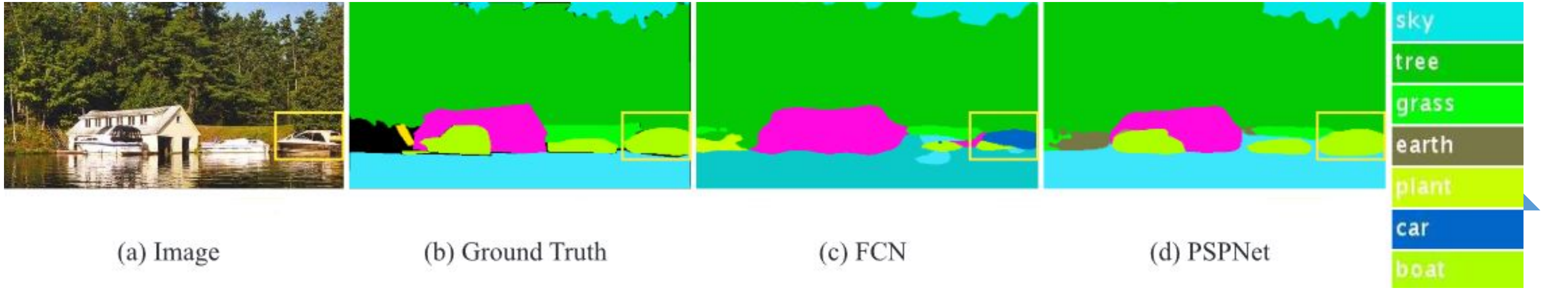
FIGURE 7. Training and testing loss, and accuracy curves of FCN-32s.

Deeplab v2-CRF, deeplab v3, deeplab v3-JFT

Deeplab developed by Google is an atrous convolution framework for semantic image segmentation. The earlier models, precisely Deeplab v1 and Deeplab v2 required a post processing step **CRF(conditional random field)** to segment the classes with more precise outlines. However in the latest version Deeplab-v3, this extra step is compensated by the improved framework. **DeepLabv3-JFT**: Employes ResNet-101 which has been **pretrained on both ImageNet and JFT-300M** dataset, **86.9%mIOU.**

PSPNet

PSPNet is yet another semantic segmentation model architecture which takes into account the global context of the image to predict the local level predictions hence gives better performance on benchmark datasets like PASCAL VOC 2012 and cityscapes. The model was needed because FCN based pixel classifiers were not able to capture the context of the whole image.



(a) Image (b) Ground Truth (c) FCN (d) PSPNet

Dataset(s) Analysis / Description

- **Dataset Capture / Preparation / Generation** :
  (Discuss the dataset generation process or if downloaded data provide details of what data & from where it was obtained etc… - 2 to 3 bullets only)

- **Dataset Understanding / Analysis** :
  (Provide 2 to 3 bullets about what is your understanding of the data / opinion about the data)

Baidu Person Segmentation Dataset: The persons to segment in the dataset contain different attached items, such as hats and bags. The images are captured from different sources, for example, advertisement, magazines, news, and street-shots. There are in total 5389 images and corresponding pixel-wise segmentation labels in the dataset.

OC human dataset: This dataset focus on heavily occluded human with comprehensive annotations including bounding-box, humans pose and instance mask. This dataset contains 13360 elaborately annotated human instances within 5081 images. With an average 0.573 MaxIoU of each person,

PASCAL VOC dataset: The PASCAL VOC 2012 dataset on semantic segmentation consists of 1464 labelled images for training, and 1449 for validation. There are 20 categories to be predicted, including aeroplane, bus, chair, sofa, etc. All images in the dataset are not larger than 500x500.

Cityscapes dataset: The Cityscapes dataset consists of 2975 street photos with fine annotation for training and 500 for validation. There are 19 classes of 7 categories in total. All images are in resolution of 2048x1536.

COCO dataset: COCO provides multi-object labeling, segmentation mask annotations, image captioning, key-point detection and panoptic segmentation annotations with a total of 81 categories, making it a very versatile and multi-purpose dataset. It has 330K images (>200K labeled), 1.5 million object instances 5 captions per image and 250,000 people with keypoints.

- **Dataset Pre-Processing / Related Challenges (if any)** :
  (List out the challenges you fore see in data handling wrt problem definition – 2 to 3 bullets only)

Data/Code Details

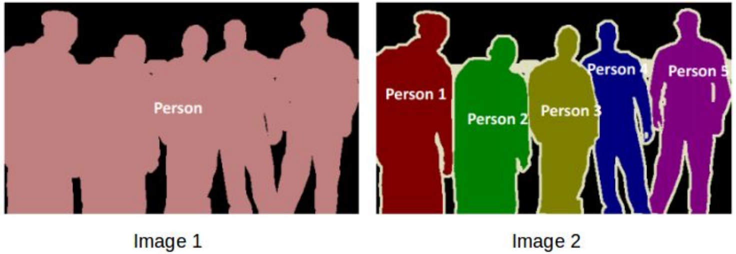
- **Data/Code details:**

Parameter	Details
KLOC [Lines of Code in Thousands]	[xx]
Model /Algorithm Details	[YOLACT++]
Details of Datasets uploaded [No of files – Images, Videos, etc.]	[COCO Dataset & PASCAL VOC]
List of Reports Uploaded [Name of All Documents uploaded	[xx]

Comparison

Given our projects requirements : a) >=30 fps video and 2) human segmentation, the suitable method to be deployed is YOLACT++ as it is an instance segmentation model which would differentiate in case of multiple people as opposed to semantic segmentation models like deeplabv3 and PSPNet which would label all humans as one and segment them as a whole. There would be a tradeoff between fps achievable and the accuracy achieved in segmentation with Mask RCNN as 30 fps with ~60% mIOU as compared to 12 fps with ~86% mIOU in the semantic segmentation models. This is an acceptable tradeoff.

Thus the model we propose : Mask RCNN/YOLACT++



Semantic segmentation vs Instance segmentation

Parameter	Mask R-CNN	YOLACT++	pspNet	Deeplab v3/Xception
Speed	Slower as it is a 2-stage segmentation	Faster as it's only a single stage seg.	Fast	Faster
(mIOU)	~60		~82	~85
mAP	35.7	29.8		
Simplicity	Complicated model	Simpler	Simpler	Simpler
Fps achievable	30	33	8	12
Segmentation type	Instance	Instance	Semantic	Semantic
Prediction time	Comparatively slow	Fast (30mSec)	Fast	Fast (300 mSec)

We ran Mask RCNN image segmentation Demo provided by google on colab which used pretrained weights (COCO dataset) and the following image is the image segmented output.

Experimental Results / Simulations / Observations

- **Results** :
  (provide numerical data / bar charts / plots / images / videos / tabulated results etc. Use full slide or multiple slides up to max 3 slides to demonstrate the results)

Model : YOLACT
Pre-trained on : COCO dataset
Pre-trained weights : ResNet 50
Test video size : 24 MB
Output video size : 90 MB
Video Playback FPS: 10.68
Colab link :
https://colab.research.google.com/drive/1bHmxWaIIDGqN8-wDokVRc-2iQenZf37A?usp=sharing
Authors : Daniel Bolya, Chong Zhou, Fanyi Xiao, Yong Jae Lee!
Git : https://github.com/dbolya/yolact.git

Swastika©

- **Major Observations / Conclusions & Challenges :**
  (provide details about your findings, experimental opinion – Use separate slide if necessary)

Person prediction % : ~80 => Accurate model.
Challenge : Increase playback FPS

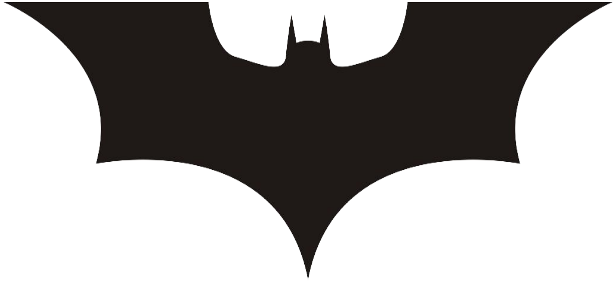
Further Plan to Complete Project

- **Final Probable Deliverables** :
  (Discuss in the form of bullets, what are the next steps to complete the solution, any road blocks / bottlenecks, any support needed from SRIB)

- We will be using Yolact ++ and Mask R-CNN for achieving human segmentation and optimize the algorithms for improving the fps.
- We will then train our model and then use Open CV to create a Double exposure video as we have done below.

- **IP Target / Plan** :
  (Any possibility of papers / patentable ideas / innovative aspects that can lead to patentable ideas)

This image has been created by using Opencv . Similarly, we will use opencv later to create a Double exposure video using 2 videos after human segmentation using mask RCNN/YOLACT++.



Link : https://stackoverflow.com/questions/55385613/how-can-i-cut-custom-shape-from-an-image-with-pil

Further Plan to Complete Project

- **Completion Plan**:

(High level plan to complete the project in next 8 weeks after review, in format below)

Week 1 to 2	• Step 1 = Research and learn more about implementing YOLACT++ • Step 2 = Find various methods to work on the fps of the videos.
Week 2 to 4	• Step 1 = Work with the Mask R-CNN model and compare the results with YOLACT++ • Step 2 = Learn about the dependence of fps on the processor used and find suitables substitutes.
Week 5 to 6	• Step 1= Train our model instead of using pre-trained weights and compare results. • Step 2= Dig deep into Open CV and other video manipulation libraries (pymedia, gst-python)
Week 7 to 8	• Step 1= Work on further required improvements. • Step 2

- **Challenges Anticipated:**

Thank you