Mind the Gap: A Generative Approach to Interpretable Feature Selection and Extraction

Abraham Jose
ID :5068109, CAP6614
abraham@knights.ucf.edu

Abstract

DeepLab is the neural network model developed by google which used fully connected Conditional Random Field(CRF) to overcome poor localization of segmentation boundary when deep convolutions like Fully Convolution Networks(FCN) are used for segmentation. DeeplabV3 has the encoder-decoder structure, in which encoder encodes the multi-scale contextual information at multiple rates and multiple effective field-of-views using Atrous Spatial Pyramid Pooling. While the decoder network captures the object boundaries by gradually recovering spatial information, without DenseCRF post-processing. The DeeplabV3+ uses Xception model with depth-wise separable convolution to pooling and decoding modules, which results in faster and stronger encoder-decoder network. In this experiment the DeepLabV3+ has been trained on Panoptic COCO data-set with 96 classes(including 'ignore_label') which includes all the 'stuff' and 'things' in the MS COCO data-set.

1 Introduction

Deep learning in computer vision has helped us make great strides that we couldn't have done without it. The 2D feature extraction and feature response or activation when cascaded helped us extract higher level feature vectors for many of the computer vision tasks including Segmentation, Detection and Classification. For Semantic segmentation, the deep convolution neural network(DCNN) models are trained end-to-end, pixels-to-pixels on semantic segmentation. We have been using fully convolution neural network with encoder(samples the image to the segment groups) and a decoder(up samples the encoded matrix to recover the spatial boundary). However DCNN offers built-in invariance of DCNNs to local image transformations, which hamper its ability for precise localization of boundary, which is not desireable. Hence Atrous Spacial Pyramidal Pooling is employed for encoder network which allows for efficient dense computation of responses. For the decoder network, the model needs to capture and learn fine details such as the object boundary and precise localization of object pixels. This is obtained by using fully-connected Conditional Random Field(CRF), where we compares class scores by the encoder with low level information captured by the interaction of edges and pixels from the initial layers of neural network.

Atrous Convolution: The dilated convolution help us to create high density localization maps by enlarging the receptive field of the feature mask. Thus we can reduce the number of parameters required by using strides with Atrous convolution and thus reduce the computation requirement.

Spatial Pyramidal Pooling: The encoder uses feature vectors which can be used to probe the input data features at multiple scale and fields. In the last layer of encoder network, Spatial Pyramidal Pooling is employed with 4 levels of pooling(4 scale), which lets the image keep the aspect ratio by letting the model resize only the smaller side while retaining the aspect ratio. Thus Spatial Pyramidal Pooling gives us a constant sized flattened array irrespective of the input image size.

Dense Conditional Random Field: Dense CRF is inspired from fully Conditional Random Field(CRF) models which uses edges and low level features to infer the sparse structure of the objects. It uses mean field approximation to the CRF distribution of the edges and low level features from the 3-rd convolution layer of the encoder module to create the sparse structure or to localize the boundary of an object. By simplifying the decoder by up-sampling the encoded image by a factor of 4 and then concatenating with the low-level feature and then up-sampling again by 4 after applying a $3x3$ convolution it was possible to reduce computation overhead of CRF.

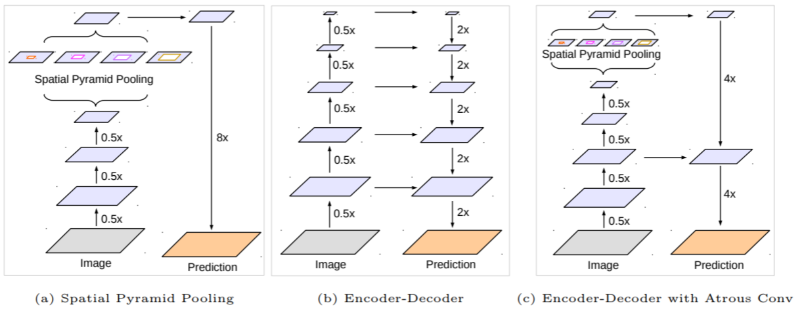


Figure 1: Spatial Pyramidal Pooling and Atrous Convolution in encoder-decoder DeepLabV3+

2 Pre-processing MS COCO for Semantic Data-set

Microsoft's Common Object in Context(COCO) data-set is a data-set from Microsoft that has open-sourced images and annotation for segmentation, detection and key-point detection. The Panoptic data-set from COCO has 133 base classes which includes 90 'things' classes and 43 'stuff' classes. The difference between 'stuff' and 'things' is that things are quantifiable like *chair,tv, person* etc.. while 'stuffs' are not, for example *sky, ceiling* etc.. Panoptic data-set from COCO has only instance segmentation which means that 'things' in panoptic are annotated separately for each object occurrence in the image.

In-order to use the panoptic data-set for this segmentation task, the instance segmentation has been converted to corresponding semantic. Further to reduce the number of data classes, we considered only super-classes in *'stuff'*. For example, the classes *river, sea, water-other* has super-class *water*. However, for some of the super-classes, the variation in the sub-classes is significant and they may hamper the training accuracy. For example *platform, snow* and *gravel* has different features which comes under the super-class 'ground' label. Refer Table 1 for the corresponding super-class and classes. The MS COCO data-set converted to semantic segmentation of 96 classes is in ADE20K semantic format for training. ie, the corresponding annotations are gray-scale images with pixel values in range of $[0, 95]$ where 0 is the *ignore_label* as mentioned early and rest of the values are class ids.

3 Experiments

To benchmark the experiments, train on the vanilla model of DeeplabV3+, the model has been downloaded from the research repository of deeplab research @ `https://github.com/tensorflow/models/tree/master/research/deeplab`. Also we are using the *xception* pre-trained on Image net data-set based encoder network and we will initialize the network with pre-trained weights. It helps us to converge decoder network at the earliest.
There has been 2 possible atrous rates that can be used for the training the model. We have done the experiments with these two rates at different batch sizes and training iterations.

3.1 Training

The model can be used to achieve two different output strides depending on the atrous convolution rates. ie, by using atrous rates $[6, 12, 18]$ and $[12, 24, 36]$ we can have encoder images that has been resized by 16 and 8 respectively. Following are the two experiments carried out:

Table 1: Table with super-class and classes mapping in COCO data-set.

Super-class	**Sub-classes**
ground	*gravel, platform, playingfield, railroad, road, sand, snow, pavement-merged, dirt-merged*
furniture-stuff	*counter, door-stuff, light, mirror-stuff, shelf, stairs, cabinet-merged, table-merged*
wall	*wall-brick, wall-stone, wall-tile, wall-wood, wall-other-merged*
textile	*banner, blanket, curtain, pillow, towel, rug-merged*
building	*bridge, house, roof, tent, building-other-merged*
floor	*floor-wood, floor-other-merged*
plant	*flower, tree-merged, grass-merged*
window	*window-blind, window-other*
water	*river, sea, water-other*
raw-material	*cardboard, paper-merged*
structural	*net, fence-merged*
solid	*mountain-merged, rock-merged*
food-stuff	*fruit, food-other-merged*

1. **Output stride = 16, batch size = 4 :** This is the bench marking case. We will use this result to compare with the improvements. Also, we trained the model close to 100K iterations for the results. The mIoU achieved was .26.
2. **Output stride = 8, batch size = 10 :** Changing the output stride, we should be able to get a better performing model in the given data-set. The model has been trained for 35K iterations and able to get a mIOU close to .17. It was not able to learn the details, however, it was able to get sharp image boundaries.
3. **Output stride = 16, batch size = 10 :** This benchmark is for comparing the results with same batch size. Upon 30K iterations, the model was able to perform close to .11 mIoU.

The training loss in particular was not very smooth. It tends to saturate after several iterations. Following graph[Figure 2] is the information on training loss for experiment 2.

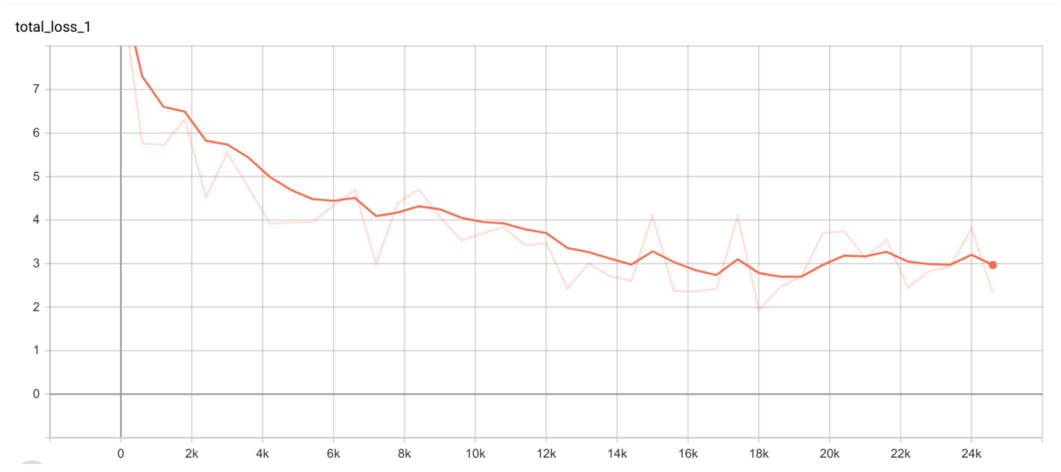
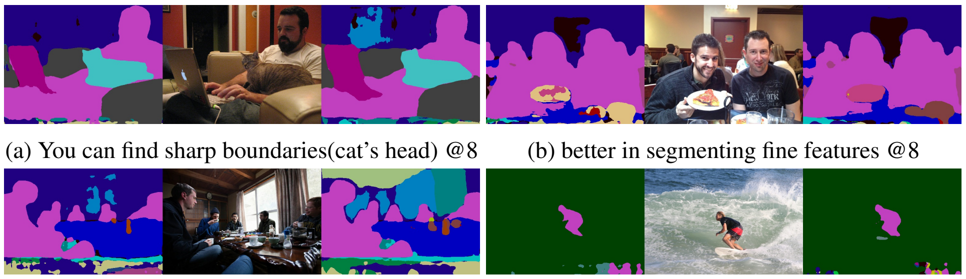


Figure 2: Loss: for output stride=8 at batch size =10, model saturates at around 16K iterations

3.2 Inference

From the above experiments it is clear that through higher atrous convolution network, we are able to gradually increase the performance to produce better boundary decisions compared to the low atrous rate network. However, while we are using atrous rate $[12, 24, 36]$, we are increasing the parameters by $2x$ times in each atrous layer leaving us with a bigger canvas with $image\_size/8$ to decode information from through CRF, which requires very intense computation to train. Through the empirical data, comparing the training performance by the deeplab team, Liang-Chieh et al. 2018, we can infer that the model performance is poor as the the number of training classes increases and we will reach it's saturation point. $Figure[3]$ gives better insights about training.



(a) You can find sharp boundaries(cat's head) @8 (b) better in segmenting fine features @8

(c) bigger segments are correctly labeled @16 (d) surfing board is segmented as water @8

Figure 3: The three sets of images are prediction(twice atrous rates $[]12, 24, 38]$), actual image, prediction(using atrous rates $[6, 12, 18]$) for the purpose of comparison.

In $Figure3, (a)$, we can clearly see that the semantic results from the bigger network has more sharper segmentation results, which is attributed to up sampling in the bigger canvas. $Figure3, (b)$ The network could understands the fine details including $food - stuff$ in the plate precisely for the bigger atrous rate model. However, in $Figure(c)$ we can see that the network was not able to distinguish between the ceiling, wall and window pane. Throughout the results, we can find the same pattern where the $[12, 24, 38]$ network is troubled with the bigger semantic classes and not being able to differentiate between the classes, which can be attributed to lesser iterations and saturation of training loss. In $Figure3(d)$ it is clearly understandable that the bigger model mistook surfing board for water, compared to the smaller network which predicts it correctly. Perhaps, the bigger network might need more learning time.

4 Conclusion

DeepLabV3+ is the state of the art technique that can be employed for semantic segmentation of images which uses multiple techniques including Atrous convolution, Spatial Pyramidal Pooling and Dense CRF decoding. As the number of classes for semantic segmentation increases, the problem becomes more challenging compared to the results the team achieved from PASCAL VOC 2012 and Cityscape data-sets, where the number of classes are limited. We will also need well defined semantic dataset to carry out experiments.

References

[1] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, Hartwig Adam.: "Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation" In *CVPR* (2018)

[2] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille.: " Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs" (2016)

[2] M.Holschneider,R.Kronland-Martinet,J.Morlet,andP. Tchamitchian, "A real-time algorithm for signal analysis with the help of the wavelet transform". In: : Time-Frequency Methods and Phase Space (1989)

[3] Philipp Krähenbühl & Vladlen Koltun.: "Efficient Inference in Fully Connected CRFs with Gaussian Edge Potentials." In:*Advances in Neural Information Processing Systems* (2011)

[4] Caesar, H., Uijlings, J., Ferrari, V.: "COCO-Stuff: Thing and stuff classes in context." In: *CVPR.* (2018)

[5] He, K., Zhang, X., Ren, S., Sun, J.: "Spatial pyramid pooling in deep convolutional networks for visual recognition." In: *ECCV.* (2014)

[6] Jonathan LongEvan ShelhamerTrevor Darrell.: "Fully Convolutional Networks for Semantic Segmentation." In:*CVPR* (2015).