Semantic Scene Segmentation with Deep Convolutional Networks (DCNNs) using Atrous convolutions and Pyramid Spatial Pooling

Abstract

Deep Convolutional Neural networks (DCNNs) provide state-of-the-art performance on many visual tasks. Convolutional networks are used in object detection, classification. More recently, they have been shown to be useful in image generation (Generative Advesarial Networks), super resolution, key-point detection and semantic segmentation. Semantic segmentation is a particularly challenging task, which requires the model to make dense output predictions at the pixel-level. The complexity increases as network becomes deeper. DCNN increasingly learn more abstract features in deeper layers, but at the same time lose spacial context/information. This is no doubt useful for the classfication task, but in the case of semantic segmentation, leads to poor localization of pixels. Here, we analyze earlier methods in segmentation — Deep Conditional Random Fields (Deep CRFs) and Fully Convolutional Networks (FCN), and explore recent advancements. Particularly, we implement two methods which provide state-of-the-art performance on the semantic segmentation task — Atrous convolutions and Atrous Spatial Pyramid Pooling (ASPP), as inspired by the DeepLabV3 network.

1. Introduction

1.1. Semantic segmentation task

Semantic segmentation is the task of assiging a label to each individual pixel in an image. Another type of segmentation is the instance segmentation, where in contranst, we label each individual instance of an object in the image. As an example, instance segmentation distinguishes between different instances of a car, while semantic segmentation does not. In contemporary applications, we find that semantic segmentation has proven to be more useful compared as compared to instance segmentation. At the end, we touch upon Panoptic segmentation which combines semantic and instance segmentations.

Figure 1: Densely annotation COCO-stuff dataset

1.2. Comparison with Object classfication & Bouding box methods

The task of image classification is not new. Classification and boundary-detection problems are well-researched and have performant models. However, the labelling from these tasks are coarse, and do not adequately represent the scene well. The next step is to take the predictions from coarse inference to fine inference at the pixel-level.

1.3. Applications

Autonomous vehicles: For a vehicle to navigate through the roads, it has to understand the scene deeply. An autonomous vehicle has to track the road, other vehicles, traffic & stop signs, pedestrians and much more. Developing classificaion models for each task would be infeasible.

Moreover, we need a high-precision (finely annotated) map of the road to navigate effectively

Medical Imaging: Segmentation techniques are applied in differentiating between tumorous and healthy cells. Other applications include detecting fractures, cell boundaries, organ health.

Geographical and Agricultural: Land cover information is important for monitoring areas of deforestation, urbanization, solar panel placement. Segmenting satellite imagery is useful for mapping applications (Google maps). There are large-scale public datasets available for this task. for each pixel on a satellite image, land cover classification can be regarded as a multi-class semantic segmentation task. Road and building detection is also an important research topic for traffic management, city planning, and road monitoring

1.4. Challenges of using DCNNs

There are two main challenges while using Deep Convolutional Neural Networks (DCNNs) [3]. In traditional DCNNs, the repeated pooling operations (MaxPooling/AveragePooling) and striding, reduces the feature resolution. The network learns abstract representations and is invariant to image transformations such as scaling, translation. This is useful for the classificaion tasks, but fails while making dense predictions.

Secondly, objects in the trainig data exist at multiple scales. For example, a car could be near the viewer or much farther. Extracting features for each scale is tricky. [6] proposes using an Image pyramid to handle this. With this technique, multiple smaller networks operate on different scales of the input, the results are concatenated and subsequently, a 1x1 convolution is applied. Finally, a DCNN operates on this augmented feature space.

2. Background/Related Work

Research methods have shown that transfer learning [16], that is, using the learned representations by a model and building on top of it, provides an overall better performance. The next methods described implicity depend on the transfer learning approach via a backbone classifier.

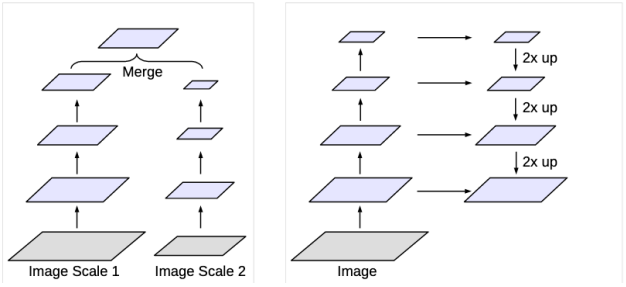


Figure 2: Left: Image Pyramid. Right: Encoder-Decoder architecture

Fully Convolutional Networks: [11], [14] proposes training convolutional networks end-to-end, with learning and inference performed on the whole image. In this paper, the authors use a pre-trained classfication network as a backbone (more on this later), to transfer the learned representations [13] for segmentation. Long et al, further adds a skip architecture, which combines semantic information from deep coarse layer with a shallower fine layer, similar to an encoder-decoder architecture.

Image Pyramid: The same model, with shared weights, is applied to multi-scale inputs (figure). The features obtained from each are merged. The intuition is that the small-scale input provides context, while large scale input preserve the details of the image. The output of these layers are concatenated.

Encoder-Decoder: This architecture [4] consists of 2 parts. In the encoder, the spatial map is reduced as the network becomes deeper. The encoder tracks the overall context of the image. In the decoding step, the spatial dimensions are recovered. The upsampling is usually done using Deconvolutional layers (explained further).

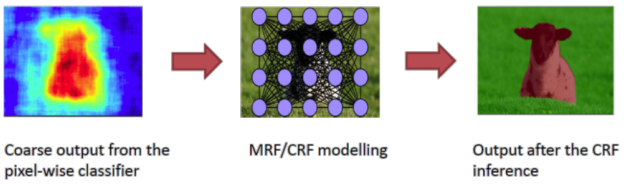


Figure 3: Illustration of Conditional Random Field

Conditional Random Fields (CRF): Conditional Random Fields are a type of discriminative model and are useful for predicting sequences. They use contextual information from previous labels. In the context of segmentation, the DCNNs are used as feature extractors while CRFs are used in post-processing to obtain finer inference. The proposed model does not use CRF.

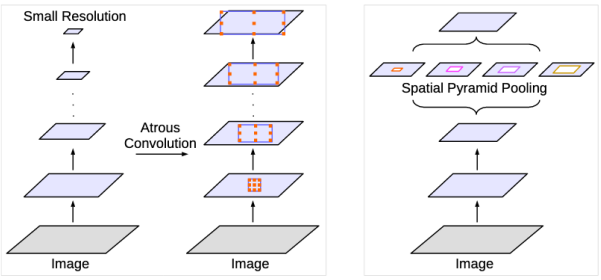


Figure 4: Left: Cascading Atrous convolutions. Right: Atrous Spatial Pyramid convolutions. The output are concatenated

Transposed Convolution: Transposed convolution [5], also known as de-convolution or dilation, is a method to up-sample an image. This technique has been used extensively for segmentation tasks. We use a feature extraction network like the ResNet and sub-sample the input (sometimes over 32 times). Subsequently, we use strided transposed convolutions to up-sample the image back to the original dimensions. One might ask, how is this different from other up-sampling techniques, such as bilinear interpolation. The main difference is, we can use our network to learn how to up-sample optimally. It does not use a predefined interpolation method and has learnable parameters. U-Net [15], uses a similar approach and further augments the upsampled layer by adding skip connections.

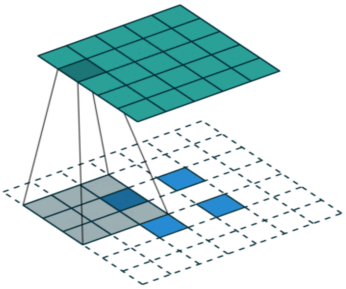


Figure 5: Stridied Transposed Convolution. Blue are input and green are output features image. The output dimensions are twice the input (Upsampling)

3. Approach

3.1. Pre-trained Backbone

Using a pre-trained backbone alleviates the need for engineered features, and produces a powerful representation that captures texture, shape and contextual information. We make use of the ResNet101 network, which has 101 layers pre-trained on the ImageNet data. This model can classify 1000 categories, some of which are superflous given our current dataset (90 categories). We remove the last fully connected layer (FC) of ResNet and replace them with 1x1 convolution layers. We also duplicate this last block several times as suggested by [12].

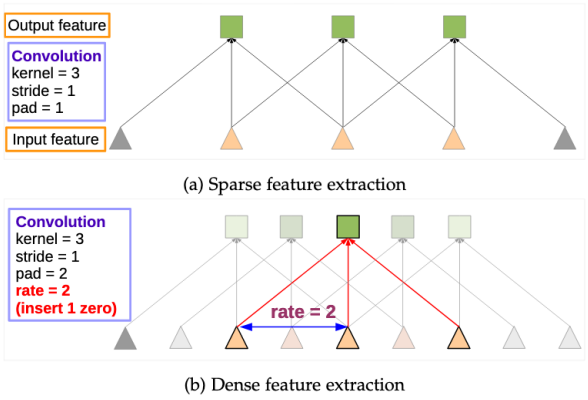


Figure 6: Top: Standard 1D convolution. Bottom: 1D atrous convolution. As we can see, it has a higher receptive field

3.2. Atrous Convolution

To combat the reduction in resolution of features by repeated pooling/striding operations, we make use of atrous convolutions [2]. á trous is frech for "with holes". This is precisely what the convolutions does. With this, we can control the resolution at which feature responses are computed. To understand this, we must first define the receptive field.

The *receptive field* in Convolutional Neural Network is the region of the input space that affects a particular unit of the network. Broadly speaking, we can visualize a convolutional network as a pyramid, where the connections from the top to lower layer give us information about the which pixels in the input image affect the output features. The larger the receptive field, the more long term information the model can propagate forward.

Using atrous convolution, we increase the receptive field size, as defined by the rate (or dilation), without increasing the parameters. The atrous filter is rather simple. It involves upsampling the filter by inserting zeros' between adjacent filter weights.

$$y[i] = \sum_k x[i + r \cdot k]w[k] \tag{1}$$

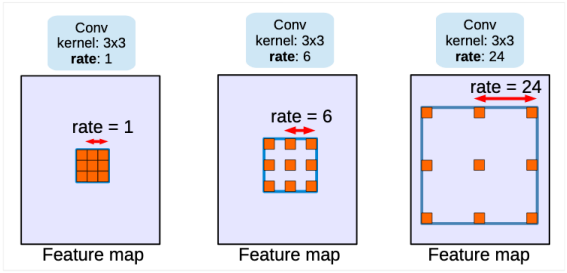


Figure 7: Atrous convolutions with varying rates

3.3. Atrous Spatial Pyramid pooling

To capture multi-scale feature representation [17] [7], we employ parallel atrous convolution layers (figure). This is inspired by the Pyramid Scene Parsing Net (PSPNet) [18].

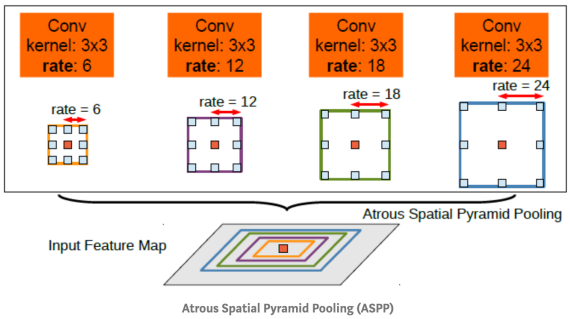


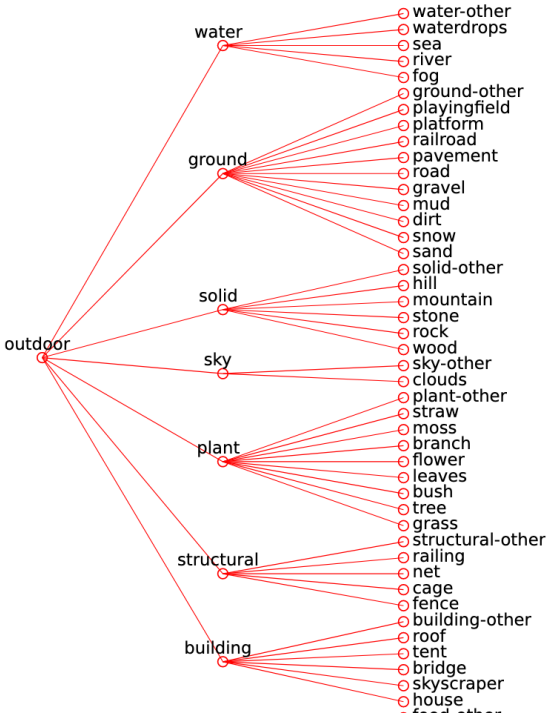
Figure 8: ASPP

4. Experiments

Framework: We use `PyTorch` 1.4 and `torchvision` modules. The `torchvision` module provides pretrained models for convenience. We make use of the ResNet101 as backbone. The model is trained on the an NVIDIA K80 GPU on the Google Cloud platform.

Datasets: For semantic segmentation task, we need a dataset with densely annotated images. The common datasets used are PASCAL VOC, CityScapes and COCO [10]. The COCO Stuff dataset [1] was released in 2018 to push the state of the art in semantic segmentation of stuff classes. Whereas the object detection task addresses thing classes (person, car, elephant), this task focuses on stuff classes (grass, wall, sky). "Things" are objects with a specific size and shape, that are often composed of parts. "Stuff" classes are background materials that are defined by homogeneous or repetitive patterns of fine-scale properties, but have no specific or distinctive spatial extent or shape. Why the focus on stuff? Stuff covers about 66% of the pixels in COCO. It allows us to explain important aspects of an image, including scene type; which thing classes are likely to be present and their location; as well as geometric properties of the scene. Some examples of the "Stuff" categories in COCO are,

```
banner blanket branch bridge bush
cabinet carpet ceiling-tile cloth
clothes clouds counter dirt
```



Preprocessing: The COCO dataset comprises of 164k densely annotated images of varying dimensions. It is divided into two tasks, the 'Stuff evaluation' and 'Panoptic evaluation'. Since, the panoptic dataset includes the instance segmentation, we exclude it from training.

First, we center crop the image to have a fixed dimensions of 224 x 224 (using *torchvision*'s `transforms.CenterCrop()`). Further, we convert the *PIL* Images into tensors and normalize them by subtracting the means and dividing by the standard deviation of the ImageNet dataset, $mean = [0.485, 0.456, 0.406], std = [0.229, 0.224, 0.225]$.

This helps our model converge faster and ensuring a uniform distribution of the color space. As a sidenote, the same normalization is applied during inference time.

ASPP Block:

Here we show a part of the implementation [8] of the Atrous Spatial Pyramid Pooling layer. Notice the different dilation rates in the `ASPP` module. Note that in the forward pass of the `ASPP` module, the individual blocks are concatenated.

```
class ASPPBlock(nn.Module):
  def __init__(self, rate, m):
    self.aspp = nn.Conv2d(C, depth,
        kernel_size=3, stride=1,
        dialtion=rate, bias=False)
    self.aspp_bn = nn.BatchNorm2d(depth,
        momentum=m)
    self.relu = nn.ReLU(inplace=True)

class ASPP(nn.Module):
  def __init__(self, C, depth,
      num_classes, m=0.0003):
    super(ASPP, self).__init__()

    self.global_pooling =
        nn.AdaptiveAvgPool2d(1)
    self.relu = nn.ReLU(inplace=True)

    self.aspp1 = ASPPBlock(rate=1, m=m)
    self.aspp2 = ASPPBlock(rate=6, m=m)
    self.aspp3 = ASPPBlock(rate=12, m=m)
    self.aspp4 = ASPPBlock(rate=18, m=m)

    self.aspp5 = nn.Conv2d(C, depth,
        kernel_size=1, stride=1,
        bias=False)
    self.aspp5_bn = nn.BatchNorm2d(depth,
        m=momentum)
```

Metrics:

The most commonly used metric for segmentation is the *Intersection Over Union (IoU)*. It is computed as follows,

$$IoU = TP/(TP + FP + FN) \tag{2}$$

where TP=true positives, FP=false positives, and FN=false negatives. The IoU is calculated per-image. Hence, for a given dataset, we define the following metrics,

Metric	Description
Mean IoU (mIoU)	Average over IoU over each class
Frequency Weighted IoU (fIoU)	weights every class proportional to the number of pixels for that class
Mean Accuracy (mAcc)	denotes the fraction of correct pixels per class averaged over classes
Pixel Accuracy (pAcc)	denotes the fraction of correct pixels

It is important to note that, during computation of the metric we ignore the pixels (0 – black pixel), to avoid an unreasonably high mIoU.

4.1. Training

We start by defining a custom `torch.Dataset` and a `DataLoader`.

Batch Size: We experimented with batch sizes 8, 16 and 32. Increasing the batch size over 32, led to worse accuracy on the validation set. For all further experiments, we fixed the batch size to 16.

Batch Size	8	16	32
mIoU	56.02	61.12	28.0

Pretrained backbone: We tried different backbones for feature extraction, ResNet50, ResNet18 and ResNet101. Resnet101 provided higher mean IoU on the validation set. However, the inference time was longer.

Optimizers: We explored various optimizers such as Stochastic Gradient Descent (SGD), Adam, RMSProp.

Hyperparameters: We use the same hyper-parameters as in the DeepLabV3 model. We set the learning rate to $\alpha = 1e-4$ and $momentum = 0.9997$

Augmentation: Random ColorJittering of the data did not yield any improvements.

4.2. Result visualization

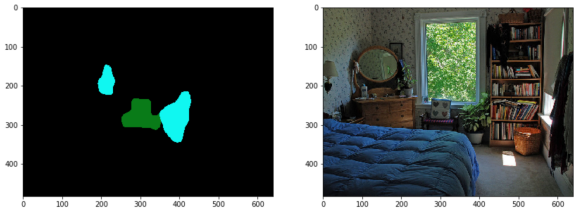


Figure 10: The model segments the two house plants and the chair

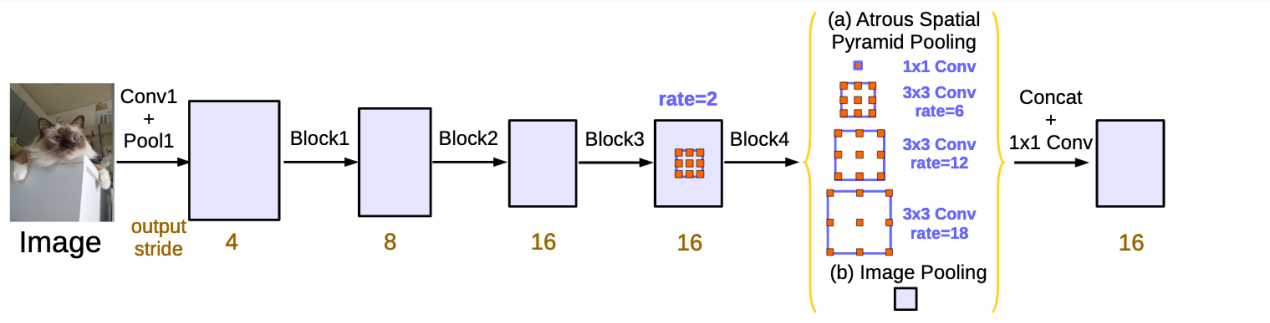


Figure 9: The Architecture of the propsed model

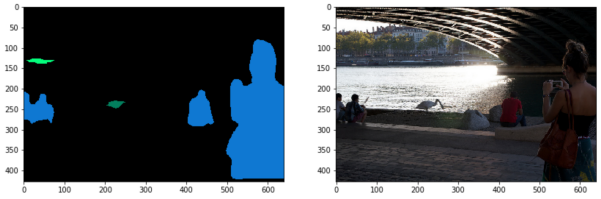


Figure 11: The model correctly recognizes the people in the images

5. Conclusion

We show that Atrous convolutions are better at localization of pixels than Fully Convolutional networks. They are better at capturing long-range context of the image, wihout increasing the number of parameters. Additionally, we demonstrate the usefulness of transfer learning by way of a pre-trained feature extraction network (ResNet).

6. Future

Recently, a new type of segmentation task has emerged, known as the Panoptic segmentation. It combines the instance-aware and sematic segmentation classes into one [9], providing a unified view of image segmentation. This would be the next advancement in improving real-world vision applications, especially in autonomous vehicles. The primary metric to assess the performance of this task is the Panoptic Quality (PQ), measuring the ratio of the sum of IoU ratios of true positive values to the sum of all the values —

$$PQ = \frac{\sum_{p,g \in TP} IoU(p,g)}{|TP| + \frac{1}{2}|FP| + \frac{1}{2}|FN|} \quad (3)$$

We hope to apply this model on the COCO-Panoptic dataset.

Also, the inference time for a image is well over 2 seconds, which for real applications is impractical. We hope to explore ways to reduce this mean inference time.

References

[1] Holger Caesar, Jasper R. R. Uijlings, and Vittorio Ferrari. Coco-stuff: Thing and stuff classes in context. *CoRR*, abs/1612.03716, 2016.

[2] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4):834–848, Apr 2018.

[3] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *CoRR*, abs/1706.05587, 2017.

[4] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. *CoRR*, abs/1802.02611, 2018.

[5] Vincent Dumoulin and Francesco Visin. A guide to convolution arithmetic for deep learning, 2016.

[6] C. Farabet, C. Couprie, L. Najman, and Y. LeCun. Learning hierarchical features for scene labeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1915–1929, Aug 2013.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *Lecture Notes in Computer Science*, page 346–361, 2014.

[8] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift, 2015.

[9] Alexander Kirillov, Kaiming He, Ross Girshick, Carsten Rother, and Piotr Dollár. Panoptic segmentation, 2018.

[10] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. *Lecture Notes in Computer Science*, page 740–755, 2014.

[11] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2015.

[12] 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.

[13] Maxime Oquab, Leon Bottou, Ivan Laptev, and Josef Sivic. Learning and transferring mid-level image representations using convolutional neural networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

[14] Tobias Pohlen, Alexander Hermans, Markus Mathias, and Bastian Leibe. Full-resolution residual networks for semantic segmentation in street scenes. *CoRR*, abs/1611.08323, 2016.

[15] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *CoRR*, abs/1505.04597, 2015.

[16] Lisa Torrey and Jude Shavlik. Transfer learning. In *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*, pages 242–264. IGI Global, 2010.

[17] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions, 2015.

[18] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul 2017.