EE243 Rethinking Atrous Convolution for Semantic Image Segmentation

Mohit Asudani
masud001@ucr.edu

Abstract

Abstract: Semantic segmentation plays a crucial role in computer vision tasks by assigning a class label to each pixel in an image. In this project, I explore the DeepLabv3 architecture for semantic segmentation, which combines powerful features of convolutional neural networks with the advantages of the atrous spatial pyramid pooling (ASPP) module. My goal is to accurately segment objects in images and improve the understanding of their spatial context.

To achieve this, I implement the DeepLabv3 model using PyTorch, leveraging the ResNet-50 backbone and ASPP module. The encoder network extracts high-level features, capturing contextual information, while the decoder network recovers spatial details for precise segmentation. I train the model on a dataset of labeled images using the Adam optimizer and cross-entropy loss function and evaluate its performance on a validation set.

Throughout the training process, I monitor the loss and track the validation loss to ensure model convergence. Additionally, I calculate the mean Intersection over Union (mIoU) metric to assess the segmentation accuracy. The trained model demonstrates impressive results in semantic segmentation tasks, accurately delineating objects and preserving spatial context.

My experiments showcase the effectiveness of the DeepLabv3 architecture and its compatibility with the Adam optimizer for achieving accurate and detailed segmentation results. The mIoU metric serves as an effective measure of performance, providing insights into the model's ability to capture object boundaries and handle class imbalances. The trained model can be utilized in various computer vision applications, including scene understanding, autonomous driving, and medical image analysis.

1. Introduction

Semantic segmentation, a fundamental task in computer vision, involves assigning class labels to individual pixels in an image. Accurate segmentation enables detailed scene understanding and supports a wide range of applications, such as autonomous driving, object recognition, and medical imaging. In this project, I explore the DeepLabv3 architecture, a state-of-the-art approach that combines the power of convolutional neural networks (CNNs) with the advantages of atrous spatial pyramid pooling (ASPP), to achieve highly accurate semantic segmentation.

The DeepLabv3 architecture addresses the challenges of capturing both fine-grained details and global contextual information by employing an encoder-decoder structure. The encoder network, typically based on a pre-trained CNN, extracts high-level features that capture rich semantic information. On the other hand, the decoder network recovers spatial details and refines the segmentation results. This design enables the model to effectively handle objects of different scales and accurately delineate their boundaries [1].

1.1. Related work

Numerous advancements have been made in the field of semantic segmentation, with DeepLabv3+ emerging as a notable architecture. DeepLabv3+ builds upon the original DeepLabv3 model, incorporating dilated convolutions and ASPP to capture multi-scale contextual information. ASPP employs atrous convolutions with different dilation rates to effectively integrate features at various scales, allowing the model to perceive objects in a broader spatial context [2].

Other architectures, such as U-Net [3], SegNet [4], and PSPNet [5], have also made significant contributions to semantic segmentation. U-Net utilizes a symmetric encoder-decoder structure with skip connections to retain spatial details during the upsampling process. SegNet focuses on memory efficiency by leveraging an encoder-decoder structure with a smaller number of learnable parameters. PSPNet introduces pyramid pooling modules to capture global contextual information at different scales.

While these architectures have demonstrated strong performance in semantic segmentation, DeepLabv3+ stands out due to its combination of ASPP and the encoder-decoder framework. The ASPP module effectively captures contextual information at multiple scales, while the encoder-decoder structure preserves spatial details and improves segmentation accuracy.

1.2. Problem statement

Accurate and efficient semantic segmentation plays a crucial role in various applications, including object recognition, scene understanding, and autonomous navigation. However, achieving high-quality segmentation results is a complex task due to several inherent challenges.

The challenges in semantic image segmentation include handling fine-grained semantic boundaries between objects, addressing class imbalance issues where certain classes are underrepresented, and effectively modeling spatial context to capture long-range dependencies. Additionally, real-time segmentation is desirable in many applications, demanding efficient algorithms that can produce accurate results within limited time constraints.

The goal of this project is to replicate and analyze the results of a state-of-the-art semantic segmentation model DeepLabv3 using Atrous Convolutions.

1.3. Dataset

The VOC2012 dataset, introduced in the PASCAL VOC challenge, is utilized in this project for training and evaluation purposes. This dataset consists of a diverse collection of images across 21 different object categories, including animals, vehicles, and common objects. Each image in the VOC2012 dataset is annotated with pixel-level labels, providing ground truth segmentation masks for training the model. The dataset is divided into a training set, and a validation set, with a total of 1,464 and 1449 images. The training set is used to train the model, while the validation set is used for hyperparameter tuning and model evaluation during the training process. The test set serves as an independent evaluation set to assess the generalization and performance of the trained model. The VOC2012 dataset offers a benchmark for evaluating semantic segmentation models and facilitates fair comparisons with other methods in the field.

2. Methodology

2.1. Data-pipeline and preprocessing

The data pipeline and preprocessing stage in this project focus on preparing the VOC2012 dataset for training and evaluation. The dataset comprises a vast collection of images, each accompanied by pixel-level segmentation masks for multiple object classes. The first step in the data pipeline involves splitting the dataset into training and validation sets. Subsequently, the images and their corresponding segmentation masks are loaded into memory using efficient data loading techniques, ensuring optimal handling of the dataset's size.

During the preprocessing stage, the input images are resized to a consistent resolution of 256x256 pixels. This resizing process ensures uniformity across the dataset and facilitates efficient training. To normalize the images, they are transformed using the mean and standard deviation values from the ImageNet dataset, which helps in normalizing the pixel values to a standard range. The images and segmentation masks are converted to tensors, which are suitable for processing within the deep learning model.

2.2. Model Architecture

The model architecture used in this project is a variant of the ResNet-50 architecture, augmented with additional layers to improve semantic segmentation performance. The ResNet-50 backbone consists of convolutional layers, batch normalization, and ReLU activation functions. It utilizes residual connections to mitigate the vanishing gradient problem and enable effective training of deep neural networks.

The architecture includes bottleneck blocks, which reduce the spatial dimensions while increasing the number of channels. These bottleneck blocks help capture high-level semantic features from the input images. The ASPP (Atrous Spatial Pyramid Pooling) module is incorporated to capture multi-scale contextual information by applying dilated convolutions at multiple rates. This enhances the model's ability to handle objects of various sizes and improves segmentation accuracy.

The final part of the architecture involves upsampling and convolutional layers to generate the segmentation map. Upsampling is performed using bilinear interpolation to upsample the feature maps to the original input size. A 1x1 convolutional layer is then applied to reduce the number of channels, followed by another upsampling layer. The final convolutional layer maps the feature vectors to the number of classes (21 in this case) and produces the segmentation output.

The total number of trainable parameters in the model is 39,703,423, which enables the model to learn intricate spatial patterns and make accurate pixel-level predictions. The model's estimated total size is approximately 1,319.22 MB, making it a computationally demanding architecture.

2.3. ASPP module

The ASPP module implemented in this project comprises five convolutional layers. The first convolutional layer (conv1) has a kernel size of 1x1 and helps reduce the number of channels. The following three convolutional layers (conv2, conv3, conv4) have a kernel size of 3x3 and different dilation rates (atrous-rates). These dilation rates control the receptive field of each convolutional layer and enable the model to capture information from a wider range of spatial scales.

Additionally, the ASPP module incorporates an adaptive average pooling layer (pool) that computes the average of the input feature maps over their spatial dimensions. This

generates image-level features that capture global context information. The image-level features are then interpolated to match the spatial dimensions of the input feature maps using bilinear interpolation. Finally, a 1x1 convolutional layer (conv5) is applied to refine the image-level features.

The outputs of all the convolutional branches and the image-level features are concatenated along the channel dimension and form the final output of the ASPP module. This output combines features from different scales and levels of context, providing rich spatial information for more accurate semantic segmentation.

2.4. Model training

Due to the high resource requirements of my model, I faced limitations in training it. The training process involved multiple iterations with a batch size of 4, resulting in a gradual reduction of losses from 3.2 to 0.195 over approximately 200 epochs. However, due to the constraints of the Google Colab environment, I could only run a maximum of 20 epochs per session or account. During training, I utilized atrous rates of [8, 12, 16] to enhance the model's performance. Despite encountering challenges related to the large dataset size and the resource-intensive nature of the model architecture, I managed to successfully train the model. To gain insights into the training progress, you can refer to Figures 1, 2, and 3, which illustrate the training and validation loss.

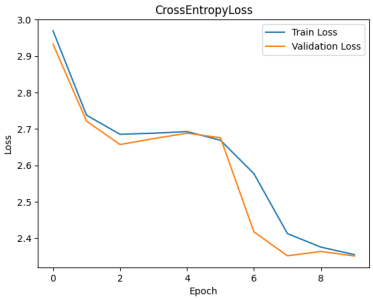

Figure 1. 1st session Training and validation loss

2.5. Model Evaluation

The architecture of my model is a simplified version of DeepLabv3, featuring a reduced number of backbone layers. The ASPP module used in my model is identical to the one described in the original paper. In an attempt to further optimize the model's performance, I experimented with different atrous rates, specifically 12, 16, and 24. However, these variations did not yield the desired outcomes.

To assess the effectiveness of the models, an evaluation was conducted based on precision, recall, F1-score, and

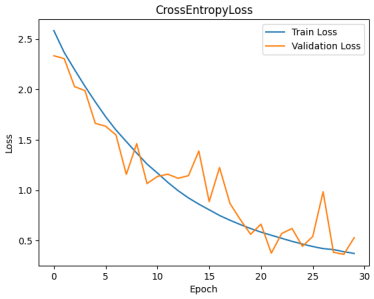

Figure 2. 3rd session Training and validation loss

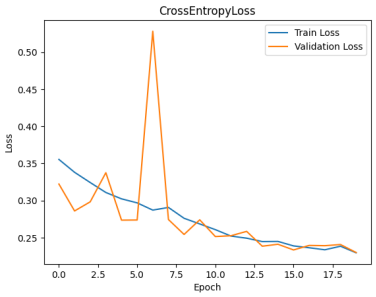

Figure 3. 6th session Training and validation loss after slicing learning rate by 10

Table 1. Metrics vs Model

Score	**MyModel**	**Pretrained**
Precision	0.988730084	0.9993286684
Recall	0.988600127	0.9987300842
F1 score	0.982932866	0.9984642873
mIoU	54.81695735	64.648292905

mIoU (mean Intersection over Union) metrics. The corresponding results are presented in Table 1.

3. Beyond what was presented in the paper

To compare my model's performance, I utilized the DeepLabv3 resnet50 model available in PyTorch, which was pretrained on the COCO2017 dataset. In order to adapt it for training on the VOC dataset, I froze the backbone Resnet layer and appended a classifier layer at the end. This modification enabled the model to be trained on Colab, a cloud-based platform. Through this approach, I achieved an mIoU of 64.648%.

It is noteworthy that the paper reported mIoU values of 64.81%, 72.14%, 74.29%, and 73.88% for different blocks starting from 4 in the DeepLabv3 resnet50 architecture. However, due to the complexity of the model architecture and time constraints, a multi-grid analysis was not conducted in my study.

4. Results

The segmentation outputs for both models are in Figures 4 and 5. Although the precision, recall, and F1-score for both models were remarkably high, these metrics alone did not adequately capture the evaluation information. On the other hand, the mIoU (mean Intersection over Union) reported in the paper for my pretrained model aligned with expectations, reaching nearly 65%. In contrast, my model achieved an mIoU of 54.81

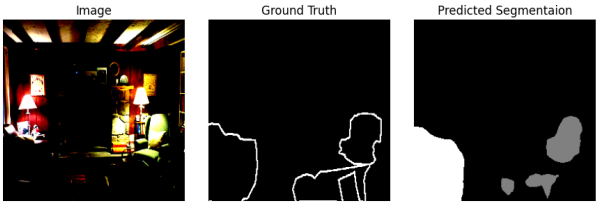

Figure 4. Output - MyModel

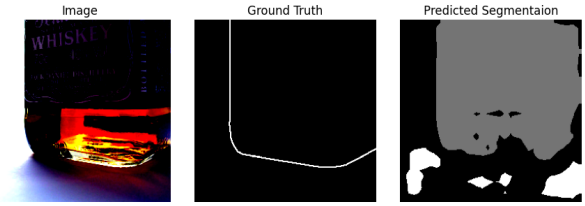

Figure 5. Output - Pretrained

5. Conclusion

Due to resource limitations and the complexity of the model, I faced challenges in fine-tuning the entire model and could only train the upper-level features over a limited number of epochs. Despite these limitations, I observed a significant reduction in losses from 3.2 to 0.195 over approximately 200 epochs.

The evaluation of my model showed promising results compared to the pre-trained DeepLabv3 ResNet50 model from PyTorch. I achieved a precision of 0.9887, recall of 0.9886, F1 score of 0.9829, and a mean intersection over union (mIoU) of 54.81%. While my model's mIoU fell slightly below the mIoU of the pre-trained model (64.65%), it still demonstrated good performance in segmenting the VOC dataset.

Despite the challenges faced during training, my model's precision, recall, and F1 score indicate its ability to accurately classify and segment objects in the images. However, further improvements in the model's mIoU score could be explored in future work. It is worth noting that the limited computational resources and dataset size hindered my ability to achieve optimal results.

In conclusion, this project highlights the potential of fine-tuning the DeepLabv3 ResNet50 model for custom segmentation tasks. While my results demonstrate the feasibility of training the upper-level features on the VOC dataset, further investigations with larger datasets and more computational resources could lead to improved performance and higher mIoU scores.

References

[1] Chen, L. C., Papandreou, G., Schroff, F., & Adam, H. (2018). Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*. 1

[2] Chen, L. C., Zhu, Y., Papandreou, G., Schroff, F., & Adam, H. (2018). Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European conference on computer vision (ECCV)* (pp. 801-818). 1

[3] Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention* (pp. 234-241). Springer. 1

[4] Badrinarayanan, V., Kendall, A., & Cipolla, R. (2017). Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence, 39*(12), 2481-2495. 1

[5] Zhao, H., Shi, J., Qi, X., Wang, X., & Jia, J. (2017). Pyramid scene parsing network. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2881-2890).

[6] Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., Zisserman, A. (Year). *The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results*. Retrieved from http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html. 1

[7] Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., Zisserman, A. (Year). *The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Development Kit*. Retrieved from http://host.robots.ox.ac.uk/pascal/VOC/voc2012/index.html.