Computer Vision 2024011303 Liu Hanzuo

HW4

Project Repository:
https://github.com/liuhanzuo/pytorch-segmentation

Introduction

In this homework, I mainly implement PSPNet and SCTNet for semantic segmentation tasks. The dataset used is the Cityscapes dataset, which contains 5000 high-resolution images with pixel-level annotations for 19 classes. I run my codes on three different nets: DeepLab, PSPNet, and SCTNet. The results show that the performance of SCTNet is better than that of DeepLab and PSPNet, which is consistent with the paper's conclusion.

Methodology

PSPNet PSPNet (Pyramid Scene Parsing Network) uses a pyramid pooling module to capture multi-scale context information. The pyramid pooling module divides the feature map into different regions and applies average pooling to each region, which allows the network to capture both local and global context information. The output of the pyramid pooling module is then concatenated with the original feature map and passed through a series of convolutional layers to produce the final segmentation map.

DeepLab DeepLab uses atrous convolution to capture multi-scale context information. Atrous convolution allows the network to control the resolution of the feature map and capture features at different scales. DeepLab also uses a conditional random field (CRF) to refine the segmentation results by considering the spatial relationships between pixels.

SCTNet SCTNet (Semantic Context Transformer Network) is a transformer-based network that captures long-range dependencies and global context information. It uses a self-attention mechanism to model the relationships between pixels in the feature map, allowing the network to capture both local and global context information. SCTNet also uses a multi-scale feature fusion module to combine features from different scales, which improves the segmentation performance.

Backbone I tested my result with the same backbone – resnet-152, which is the largest model in resnet series. I also tried Resnext series, but does not work well for mIOU. In

resnet series, we can find out that a deeper model results in a better performance, which meets the conclusion of scaling law.

Cross Validation I add a parser argument in trainig codes to control whether add a cross validation. If the argument is set to true, the training codes will split the dataset into 5 folds and train the model on each fold. The final mIOU will be the average of the mIOU of each fold. The cross validation can help to reduce overfitting and improve the generalization ability of the model.

Results

Note that the results are trained on an L20 GPU with 96G memory with training batch size of 32 and validation training size of 32. The training epoch is set to 100 and has a warming up of 10 epochs. The learning rate is set to 0.01 and decayed to 5e-5 with a cosine annealing scheduler. The training time is about 4 hours for each model. The results are shown in Table 2.

Table 1: Experimental Results on Cityscapes Validation Set

Model	**mIOU (%)**	**Pixel Accuracy (%)**
DeepLab	58.2	93.2
PSPNet	61.2	93.3
SCTNet	65.3	93.2

Also, some ablation studeis are also applied to the backbone model, the base model are all setted to DeepLab. We change the backbone to resnet-152, resnet-101, and resnet-50, and the results are shown in Table **??**. The results show that the performance of resnet-152 is better than that of resnet-101 and resnet-50, which is consistent with the conclusion of scaling law.

Table 2: Experimental Results on Cityscapes Validation Set

Backbone	**mIOU (%)**	**Pixel Accuracy (%)**
resnet152	58.2	93.2
resnet101	56.8	93.3
resnet50	56.5	93.4

Notes Due to the limitation of submission zip file size, I only upload the PSPNet+resnet152 model as the result, please view `README.md` for more details.

Acknowledgements

I would like to thank the authors of the pytorch-segmentation codebase for providing an excellent foundation and reference implementation, which greatly facilitated the development of this project.