Real-time Domain Adaptation in Semantic Segmentation

TA: Claudia Cuttano (claudia.cuttano@polito.it)

OVERVIEW

The main objective of this project is to become familiar with the task of Domain Adaptation applied to the Real-time Semantic Segmentation networks. The student should understand the general approaches to perform Domain Adaptation in Semantic Segmentation and the main reason to apply them to real-time networks. Before starting, the student should read [1] [2] [3] [4] [5] to get familiar with the tasks. As the next step, the student should train the real-time segmentation network [4] on the source dataset [6]-urban to define the upper bound. Then, he/she should train the network [4] on the source dataset [6]-urban and evaluate the drop of the performance when directly testing the trained model on the target images [6]-rural. For the last part of the project, the student should implement two domain adaptation techniques [7][8] to mitigate the performance drop.

USEFUL LINKS

Dataset: https://zenodo.org/records/5706578
Note: download Train.zip to extract the training images for the Urban/Rural splits. Download Val.zip to extract the validation images for the Rural split.
Model: https://github.com/XuJiacong/PIDNet

1st STEP) RELATED WORKS

Reading paper to get familiar with the task

Before starting it is mandatory to take time to familiarize yourself with the tasks of Semantic Segmentation, Domain Adaptation and Real-time Semantic Segmentation. It is compulsory to understand what are the main problems and the main solutions to tackle them in literature. More in detail, read:

- [1] to understand Semantic Segmentation;
- [2][3][4] to understand classic and real-time solutions for Semantic Segmentation;

- [5] to get familiar with the several solutions to perform unsupervised domain adaptation in Semantic Segmentation;
- [5] [6] to get familiar with the datasets that will be used in this project;

2nd STEP) TESTING SEMANTIC SEGMENTATION NETWORKS

a) Classic semantic segmentation network.

For this step, you have to train a classic segmentation network (DeepLabV2 [2]) on the LoveDA-urban dataset.

- *Dataset*: LoveDA-urban [6]
- *Training epochs*: 20
- *Backbone*: R101 (pre-trained on ImageNet) [2]
- *Metrics*: Mean Intersection over Union (mIoU) [read this to understand the metrics], latency, FLOPs, number of parameters.

Table 1) LoveDA	**mIoU (%)**	**Latency**	**FLOPs**	**Params**
DeepLabV2 - 20 epochs	0.3761	0.125 s	185.015 GFLOPs	43.016 M

b) Real-time semantic segmentation network.

For this step, you have to train a real-time segmentation network (PIDNet [4]) on the LoveDA-urban dataset.

- *Dataset*: LoveDA-urban dataset [6]
- *Training epochs*: 20
- *Backbone*: PIDNet-S (pre-trained on ImageNet) [3]
- *Metrics*: mIoU, latency, FLOPs, number of parameters.

Table 2) Real-time LoveDA	**mIoU (%)**	**Latency**	**FLOPs**	**Params**
PIDNet - 20 epochs	0.3455	0.011 s	5.933 GFLOPs	7.624 M

3rd STEP) DOMAIN SHIFT

From now on, we will employ PIDNet as our segmentation to ease the resource requirements of the next experiments. Consider as upper bound the results obtained in Table 2, i.e. the segmentation networks trained on the labeled source images (LoveDA-urban).

a) Evaluating the domain shift problem in Semantic Segmentation

In semantic segmentation collecting manually annotated images is expensive. To this end, in this step we employ the images from LoveDA-urban [6] (source domain) to train our real-time segmentation network, which is then evaluated on the target images from LoveDA-rural [6] (target domain).

- *Training Set*: LoveDA-urban [6]
- *Validation Set*: LoveDA-rural [6]
- *Training epochs*: 20
- *Metrics*: mIoU

Table 3) Domain Shift Urban → Rural	**mIoU (%)**	**Background**	**road**	**building**	**water**	**Barren**	**Forest**	**Agriculture**
PIDNet - 20 epochs	0.2551	0.5016	0.3358	0.259	0.2633	0.0647	0.1212	0.2390

How the performance change with respect to Table 2? Why?

b) Data augmentations to reduce the domain shift

A naive solution to improve the generalization capability of the segmentation network consists in the usage of data augmentations during training. Through them, we i) virtually expand the dataset size and ii) modify the visual appearance of source images in order to make them more similar to the target ones.

Specifically, we repeat the previous experiment, introducing data augmentations at training time (e.g. horizontal flip, Gaussian Blur, Multiply, ecc.). The decision of what kind of algorithm is left to the student. Set the probability to perform augmentation to 0.5.

Table 4) Augmentations Urban → Rural	**mIoU (%)**	**Background**	**Building**	**Road**	**Water**	**Barren**	**Forest**	**Agriculture**
PIDNet - 20 epochs - Horizontal Flip	0.2515	0.4307	0.3055	0.2612	0.3187	0.0199	0.0848	0.3395
PIDNet - 20 epochs - Gaussian blur	0.2137	0.4125	0.2897	0.2080	0.2805	0.0189	0.0396	0.2474
PIDNet - 20 epochs - Elastic transform	0.2718	0.5206	0.3529	0.2618	0.2966	0.0558	0.0631	0.3518
PIDNet - 20 epochs - HueSaturationValue	0.2729	0.5269	0.2096	0.2373	0.4528	0.0368	0.1347	0.3120
PIDNet - 20 epochs - ColorJitter	0.2706	0.5136	0.3262	0.2778	0.2803	0.0510	0.1210	0.3242
PIDNet - 20 epochs - Spatter	0.2375	0.4888	0.3758	0.2176	0.2719	0.0450	0.0798	0.1832

PIDNet - 20 epochs - Flip + Gaussian blur	**0.2805**	0.5226	0.3686	0.2507	0.3873	0.0392	0.1028	0.2921
PIDNet - 20 epochs - Elastic transform + HueSaturationValue	0.2743	0.4226	0.2731	0.2958	0.4197	0.0252	0.1759	0.3082
PIDNet - 20 epochs - HorizontalFlip + Gussian Blur + ColorJitter	**0.2834**	0.5085	0.3261	0.3073	0.3559	0.0749	0.0459	0.3655

4^{th} STEP) DOMAIN ADAPTATION

To effectively tackle the problem of domain shift, various domain adaptation techniques has been proposed. Domain adaptation solutions are mainly divided into two approaches:

- **Adversarial approaches.** These methods involve a game between two parts of a model. One part, the feature extractor, tries to learn features that are indistinguishable between the source and target domains. The other part, the discriminator, tries to tell whether those features came from the source or target domain. This push-and-pull leads to features that are both discriminative (useful for the task) and domain-invariant (work well on both domains).
- **Image-to-image translation approaches.** The focus lies in translating images from the source domain to resemble the style of the target domain. The idea is that if we can make images from the different domains look similar, the model's performance will transfer more easily.

4a) Adversarial approach

You can assume:

- Source Labelled Dataset: LoveDA-urban [6]
- Target Unlabelled Dataset: LoveDA-rural [6]
- Implement discriminator function, like in [7]
- Take the best setting of step 3b (data augmentation) and perform training.

Table 5) Domain Shift Urban → Rural	**mIoU (%)**	**Background**	**Building**	**Road**	**Water**	**Barren**	**Forest**	**Agriculture**
PIDNet - 20 epochs	0.2263	0.2778	0.1676	0.1373	0.4411	0.0496	0.1445	0.3664

4b) Image-to-image approach

You can assume:

- Source Labelled Dataset: LoveDA-urban [6]
- Target Unlabelled Dataset: LoveDA-rural [6]
- Implement image-to-image adaptations: DACS [8]
- Take the best setting of step 3b (data augmentation) and perform training.

Table 6) Domain Shift Urban → Rural	**mIoU (%)**	**Background**	**building**	**road**	**water**	**Barren**	**Forest**	**Agriculture**
PIDNet - 20 epochs	0.2586	0.2460	0.1724	0.2533	0.4809	0.0515	0.1990	0.4072

5th STEP) EXTENSION

The final step of the project involves exploring additional techniques or modifications to further improve the performance of the domain adaptation task. Here some examples:

- Apply Style Transfer Preprocessing: Preprocess source domain images with a style-transfer model to match the target domain's appearance.
- Explore alternative real-time networks (e.g. STDC **PDF**, PEM **PDF**, …).
- Explore Alternative Segmentation Losses: Investigate how using different losses impacts performance and domain adaptation (simpler alternative).
- Hyperparameter Tuning: Explore different learning rates, batch sizes, and data augmentation probabilities to optimize performance (simpler alternative).
- … use your imagination!

REFERENCES

[1] "A Brief Survey on Semantic Segmentation with Deep Learning", Shijie Hao, Yuan Zhou, Yanrong Guo, **PDF**

[2] "DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFS", Liang-Chieh Chen, George Papandreou, Kevin Murphy, Alan L. Yuille **PDF**

[3] “BiSeNet: Bilateral Segmentation Network for Real-time Semantic Segmentation”, Changqian Yu, Jingbo Wang, Chao Peng, Changxin Gao, Gang Yu, Nong Sang, **PDF**
[4] PIDNet: “A Real-time Semantic Segmentation Network Inspired by PID Controllers”, **PDF**
[5] “A Review of Single-Source Deep Unsupervised Visual Domain Adaptation”, Sicheng Zhao, Xiangyu Yue, Shanghang Zhang, Bo Li, Han Zhao, Bichen Wu, Ravi Krishna, Joseph E. Gonzalez, Alberto L. Sangiovanni-Vincentelli, Sanjit A. Seshia, Kurt Keutzer, **PDF**
[6] “LoveDA: A Remote Sensing Land-Cover Dataset for Domain Adaptive Semantic Segmentation”, **PDF**
[7] “Learning to Adapt Structured Output Space for Semantic Segmentation”, Yi-Hsuan Tsai, Wei-Chih Hung, Samuel Schulter, Kihyuk Sohn, Ming-Hsuan Yang, Manmohan Chandraker, **PDF**
[8] “DACS: Domain Adaptation via Cross-domain Mixed Sampling”, Wilhelm Tranheden, Viktor Olsson, Juliano Pinto, Lennart Svensson, **PDF**