Real-time Domain Adaptation in Semantic Segmentation

TA: Antonio Tavera (antonio.tavera@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] to get familiar with the tasks. The student should be able to replicate some experiments proposed in [2]. As the next step, the student should implement an Adversarial Domain Adaptation algorithm, like in [6]. For the last part of the project, the student should implement a variation for the project, selecting from a set of possible ideas.

GOALS

1. Read [1][2][3][4][5] and get familiar with “Semantic Segmentation” Real Time networks”, “Domain Adaptation” and the datasets used;
2. Replicate the experiments detailed in the following;
3. Implement Domain Adaptation branch and perform the experiment detailed in the following;
4. Propose, implement and test a variation of the project.

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][2] to understand Semantic Segmentation and Real-time solution;
- [3] to get familiar with the several solutions to perform unsupervised domain adaptation in Semantic Segmentation, focusing principally on adversarial methods;
- [4] [5] to get familiar with the datasets that will be used in this project;
- [6] to get familiar with adversarial training technique.

2nd STEP) TESTING REAL-TIME SEMANTIC SEGMENTATION

Defining the baseline/upper bound for the domain adaptation phase

For this step you can assume for simplicity that your validation set is the same as the test set. Therefore:

- Dataset: CamVid [download] [4]
- Training Set: Train + Val folders
- Validation Set = Test Set: Test folder
- Training epochs: 50/100 epochs
- Backbone: ResNet-18/ResNet-101 (pre-trained on ImageNet)
- Semantic Classes: 11, the same of the paper and the ones indicated in the "classes_dict" file
- Metrics: Pixel Accuracy and Mean Intersection over Union (mIoU) [read this to understand the metrics]
- Code to take as starting point: [github]

Complete the table below using the parameters of [2] and the ones indicated in the first column:

Table 1) Experiment	**Accuracy (%)**	**mIoU (%)**	**Training Time (avg per-epoch)**
BiseNet (50 Epochs + ResNet-18)			
BiseNet (50 Epochs + ResNet-101)			
BiseNet (100 Epochs + ResNet-18)			
BiseNet (100 Epochs + ResNet-101)			

Take the best setting from the ones above, considering both mIoU and training time per-epoch, and repeat this experiment trying data augmentation mechanisms. To this aim, complete the *augmentation()* and *augmentation_pixel()* function in the "CamVid.py" dataset file. More in detail, implement horizontal flip for the *augmentation()* function and one or more pixel augmentation algorithms (like Gaussian Blur, Multiply ecc) for the *augmentation_pixel()* function. The decision of what kind of algorithm is left to

the student. Set probability to perform augmentation to 0.5. Report here the result of the experiment:

Table 2) Experiment	**Accuracy (%)**	**mIoU (%)**	**Training Time (avg per-epoch)**
Best Setting from Table 1			

Are the results changed?
The best results from the ones above will be the student upper bound (the maximum accuracy that the student wants/tries to reach) for the Domain Adaptation phase.

3rd STEP) IMPLEMENTING UNSUPERVISED ADVERSARIAL DOMAIN ADAPTATION

Perform adversarial training with labelled synthetic data (source) and unlabelled real-word data (target).

You can assume:

- Source Synthetic Labelled Dataset: IDDA [download] [5]
  - Implement loader class for the synthetic dataset
  - Remember to remap the IDDA semantic labels according to the 11 in common between CamVid, pay attention to the format of the IDDA semantic classes w.r.t to the CamVid ones. See the file in the zip folder for information about the semantic categories.

- Target Real-World Unlabelled Dataset: Camvid [download] [4]
  - The same as for step 2, notice that that during training semantic labels are not used
  - Target Training Set: train + val folders
  - Test Set: test folder
  - Semantic classes: 11

- Implement discriminator function, like in [6]
- Re-write train.py file to perform adversarial training between source and target domain
- Take the best setting of step 2 (number of epochs, ResNet backbone, data augmentation) and perform training. What is the maximum

accuracy reached when testing on test data? Report result here on the table:

Table 3) Experiment	**Accuracy (%)**	**mIoU (%)**	**Training Time (avg per-epoch)**
Adversarial Domain Adaptation			

4th STEP) IMPROVEMENTS

You can refer to this section to select and perform one variation for the project among the ones proposed:

a) **Different and lighter discriminator function**
   Apply the best practices exploited in real-time semantic segmentation (for example depthwise convolution) to your discriminator function, in order to make it lighter and faster. Test it and compare the result with step 3. Does training time per epoch change?

b) **Different domain adaptation technique**
   Referring to [3] and [8] you can implement a different domain adaptation technique. Test it and compare to step 3 results.

c) **Image-to-image translation to improve domain adaptation**
   You can implement a fast image-to-image translation algorithm like FDA[7] to improve the overall domain adaptation performances. You can implement another algorithm with respect to the one proposed. Test it and compare to step 3 results.

d) **Hyper-parameter optimization to improve results**
   You can start with the hyperparameters declared in the paper/code and optimize them further to get better results. Good hyperparameters to choose for optimization are: learning rate, batch size, number of frames, number of epochs, weight decay. You need also to optimize the weight of the loss of the adversarial domain adaptation task. Test it and compare to step 3 results.

e) **Other - Implement your idea**

AT THE END

- Deliver PyTorch scripts for all the required steps.
- Deliver this file with the tables compiled.
- Write a complete PDF report (with paper-style). The report should contain a brief introduction, a related work section, a methodological section for describing the algorithm that you're going to use, an experimental section with all the results and discussions, a final brief conclusion. Follow this **link** to open and create the template for the report.

EXAMPLE OF QUESTIONS YOU SHOULD BE ABLE TO ANSWER AT THE END OF THE PROJECT

- What is Semantic Segmentation?
- What is a domain shift?
- What is Domain Adaptation?
- What are the most common solutions to perform domain adaptation in Semantic Segmentation?
- What are the main reasons to use real-time Semantic Segmentation?
- How does adversarial learning technique work for domain adaptation?
- What are the main limitations of domain adaptation?

REFERENCES

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

[2] "BiSeNet: Bilateral Segmentation Network for Real-time Semantic Segmentation", Changqian Yu, Jingbo Wang, Chao Peng, Changxin Gao, Gang Yu, Nong Sang, **PDF**

[3] "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**

[4] **"**Semantic object classes in video: A high-definition ground truth database", Gabriel J. Brostow, Julien Fauqueur, Roberto Cipolla, **PDF**

[5] "IDDA: a large-scale multi-domain dataset for autonomous driving", Emanuele Alberti, Antonio Tavera, Carlo Masone and Barbara Caputo, **PDF**

[6] "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**

[7] **“**FDA: Fourier Domain Adaptation for Semantic Segmentation” , Yanchao Yang, Stefano Soatto, **PDF**
[8] You can find code for a lot of adversarial domain adaptation methods **here**.