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] to get familiar with the tasks. 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 techniques.

2nd STEP) TESTING REAL-TIME SEMANTIC SEGMENTATION

(link for downloading GTA5 and Cityscapes images: ***https://drive.google.com/drive/u/3/folders/1iE8wJT7tuDOVjEBZ7A3tOPZmNdroqG1m*****)**

a) Defining the 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: Cityscapes [4]
- Training Set: Train
- Validation Set = Test Set: Val folder
- Training epochs: 50
- Backbone: STDC (pre-trained on ImageNet) [2b]
- Semantic Classes: 19
- Metrics: Pixel Accuracy and Mean Intersection over Union (mIoU) [read this to understand the metrics]

Complete the table below:

Table 1) Experiment Cityscapes	**Accuracy (%)**	**mIoU (%)**	**Training Time (avg per-epoch)**
STDC 1 - 50 epochs	80.34%	53.90%	70.00s

b) Train on synthetic datasets.

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

- Dataset: GTA [5]
- Training Set: Train
- Validation Set = Test Set: Val folder
- Training epochs: 50
- Backbone: STDC (pre-trained on ImageNet) [2b]
- Semantic Classes: 19
- Metrics: Pixel Accuracy and Mean Intersection over Union (mIoU) [read this to understand the metrics]

Complete the table below:

Table 2) Experiment GTA	**Accuracy (%)**	**mIoU (%)**	**Training Time (avg per-epoch)**

STDC 1 - 50 epochs	79.50%	55.34%	142.46s

c) Evaluate the domain shift.

Test the model trained at point b) on the cityscapes val set. How the performance change? Why?

The performance decreased significantly because of the domain gap between the real world (target) and synthetic (training) datasets

Table 3) Experiment GTA → Cityscapes	**Accuracy (%)**	**mIoU (%)**	
STDC 1 - 50 epochs	44.34%	14.37%	

Try to perform some augmentation techniques during training of STDC [2b] on GTA [5]. Set the 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	67.82%	24.76%	197.36s

Are the results changed?

Yes, augmentation improved the performance of the model considerably because it increased diversity and variability of the training dataset

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: GTA [5]
  - Semantic classes: 19

- Target Real-World Unlabelled Dataset: Cityscapes [4]
  - The same as for step 2, notice that that during training semantic labels are not used
  - Target Training Set: train
  - Test Set: val folder
  - Semantic classes: 19

- 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 (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	73.03%	30.91%	141.92s

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**

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

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

[2b] "Rethinking bisenet for real-time semantic segmentation", Mingyuan Fan, Shenqi Lai, Junshi Huang, Xiaoming Wei, Zhenhua Chai, Junfeng Luo, and Xiaolin Wei, 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] "The cityscapes dataset for semantic urban scene understanding.**",** Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. PDF

[5] "Playing for data: Ground truth from computer games. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling" Stephan R. Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun, 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.

ADDITIONAL PAPERS

(to deepen the tasks if interested, but not necessary to accomplish the project)

Real-Time Semantic Segmentation

[9] ENet: A Deep Neural Network Architecture for Real-Time Semantic Segmentation ENet
[10] ICNet for Real-Time Semantic Segmentation on High-Resolution Images ICNet
[11] BiSeNet: Bilateral Segmentation Network for Real-time Semantic Segmentation ECCV2018 code
[12] ESPNetv2: A Light-weight, Power Efficient, and General Purpose Convolutional Neural Network cvpr2019code
[13] Fast-SCNN: Fast Semantic Segmentation Network arxiv2019 code blog
[14] LEDNet: A Lightweight Encoder-Decoder Network for Real-time Semantic Segmentation ICIP2019 code
[15] EADNet: Efficient Asymmetric Dilated Network for Semantic Segmentation arxiv2021 Fudan University , etc.
[16] AttaNet: Attention-Augmented Network for Fast and Accurate Scene Parsing AAAI2021
[17] Deep Dual-resolution Networks for Real-time and Accurate Semantic Segmentation of Road Scenes arxiv2021

Domain Adaptation in Semantic Segmentation

[18] 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]
[19] ADVENT: Adversarial Entropy Minimization for Domain Adaptation in Semantic Segmentation, Tuan-Hung Vu, Himalaya Jain, Maxime Bucher, Matthieu Cord, Patrick Pérez, [PDF]
[20] Semantic Image Synthesis with Spatially-Adaptive Normalization, Taesung Park, Ming-Yu Liu, Ting-Chun Wang, Jun-Yan Zhu, [PDF]
[21] CyCADA: Cycle-Consistent Adversarial Domain Adaptation, Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei A. Efros, Trevor Darrell, [PDF]