Project 5: Semantic Segmentation Deep Learning

CS 4476

Fall 2021

Brief

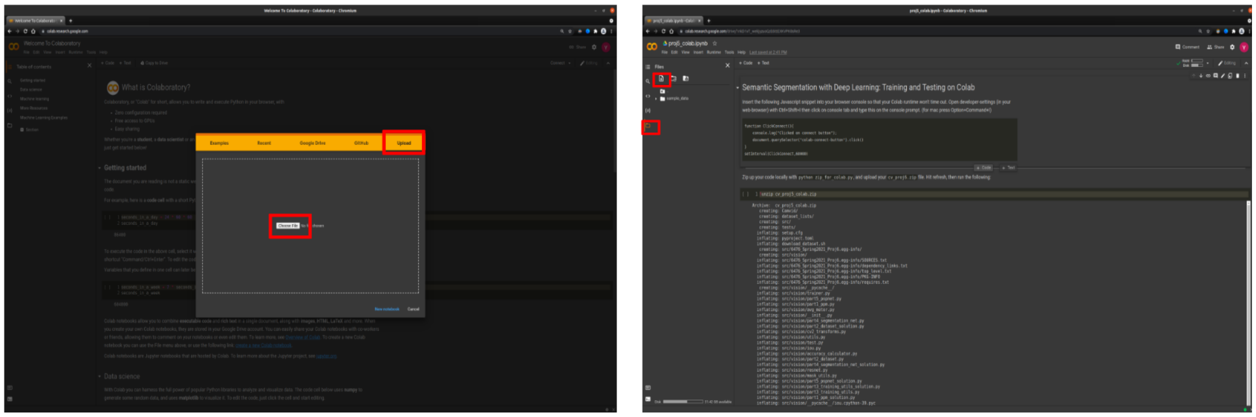
- Due: December 6, 2021 11:59PM
- Project materials including report template: Github
- Hand-in: through Gradescope
- Required files: `<your_gt_username>.zip`, `<your_gt_username>_proj5.pdf`

Overview

In this project, you will design and train deep convolutional networks for semantic segmentation.

Setup

1. Follow Project 5 Github README.
2. Run the notebook using `jupyter notebook proj5_local.ipynb` inside the repository folder.
3. After implementing all functions, ensure that all sanity checks are passing by running `pytest tests` inside the repository folder.
4. After passing the unit tests, run `python zip_for_colab.py` which will create `cv_proj5_colab.zip`
5. Upload `proj5_colab.ipynb` to Colab.
6. Upload `cv_proj5_colab.zip` to session storage.
7. Click `Run All` inside `Runtime` which will train your model.



Semantic Segmentation with Deep Learning: Training and Testing on Colab

Insert the following Javascript snippet into your browser console so that your Colab runtime won't time out. Open developer-settings (in your web-browser) with Ctrl+Shift+I then click on console tab and type this on the console prompt. (for mac press Option+Command+I)

```
function ClickConnect(){
  console.log("Clicked on connect button");
  document.querySelector("colab-connect-button").click()
}
setInterval(ClickConnect,60000)
```

Zip up your code locally with `python zip_for_colab.py`, and upload your `cv_proj6.zip` file. Hit refresh, then run the following:

```
!unzip cv_proj5_colab.zip
```

```
Archive:  cv_proj5_colab.zip
   creating: Camvid/
   creating: dataset_lists/
   creating: src/
   creating: tests/
  inflating: setup.cfg
  inflating: pyproject.toml
  inflating: download_dataset.sh
   creating: src/6476_Spring2021_Proj6.egg-info/
   creating: src/vision/
  inflating: src/6476_Spring2021_Proj6.egg-info/SOURCES.txt
  inflating: src/6476_Spring2021_Proj6.egg-info/dependency_links.txt
  inflating: src/6476_Spring2021_Proj6.egg-info/top_level.txt
  inflating: src/6476_Spring2021_Proj6.egg-info/PKG-INFO
  inflating: src/6476_Spring2021_Proj6.egg-info/requires.txt
   creating: src/vision/__pycache__/
  inflating: src/vision/trainer.py
  inflating: src/vision/part5_pspnet.py
  inflating: src/vision/part1_ppm.py
  inflating: src/vision/avg_meter.py
  inflating: src/vision/__init__.py
  inflating: src/vision/part4_segmentation_net.py
  inflating: src/vision/part2_dataset_solution.py
  inflating: src/vision/cv2_transforms.py
  inflating: src/vision/utils.py
  inflating: src/vision/test.py
  inflating: src/vision/iou.py
  inflating: src/vision/accuracy_calculator.py
  inflating: src/vision/part2_dataset.py
  inflating: src/vision/part4_segmentation_net_solution.py
  inflating: src/vision/resnet.py
  inflating: src/vision/mask_utils.py
  inflating: src/vision/part5_pspnet_solution.py
  inflating: src/vision/part3_training_utils_solution.py
  inflating: src/vision/part3_training_utils.py
  inflating: src/vision/part1_ppm_solution.py
  inflating: src/vision/__pycache__/iou.cpython-39.pyc
```

Figure 1: Figure describing setup 5, 6, 7

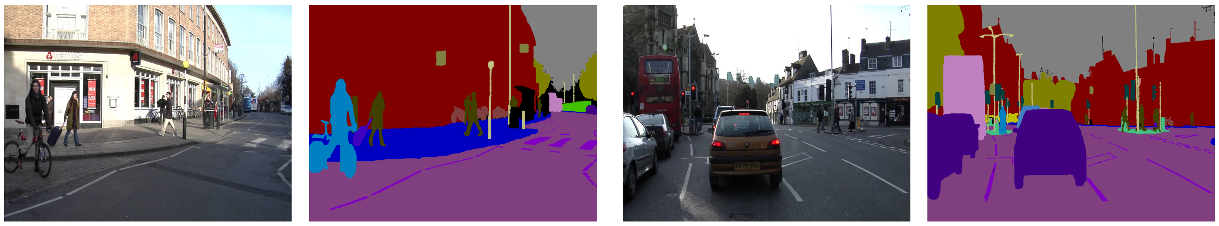
8. Generate the zip folder for the code portion of your submission once you've finished the project using `python zip_submission.py --gt_username <your_gt_username>`

Dataset

The dataset to be used in this assignment is the Camvid dataset, a small dataset of 701 images for self-driving perception. It was first introduced in 2008 by researchers at the University of Cambridge [1]. You can read more about it at the original dataset page or in the paper describing it. The images have a typical size of around 720 by 960 pixels. We'll downsample them for training though since even at 240 x 320 px, most of the scene detail is still recognizable.

Today there are much larger semantic segmentation datasets for self-driving, like Cityscapes, WildDashV2, Audi A2D2, but they are too large to work with for a homework assignment.

The original Camvid dataset has 32 ground truth semantic categories, but most evaluate on just an 11-class subset, so we'll do the same. These 11 classes are 'Building', 'Tree', 'Sky', 'Car', 'SignSymbol', 'Road', 'Pedestrian', 'Fence', 'Column_Pole', Sidewalk', 'Bicyclist'. A sample collection of the Camvid images can be found below:



(a) Image A, RGB (b) Image A, Ground Truth (c) Image B, RGB (d) Image B, Ground Truth

Figure 2: Example scenes from the Camvid dataset. The RGB image is shown on the left, and the corresponding ground truth "label map" is shown on the right.

1 Implementation

For this project, the majority of the details will be provided into two separate Jupyter notebooks. The first, `proj5_local.ipynb` includes unit tests to help guide you with local implementation. After finishing that, upload `proj5_colab.ipynb` to Colab. Next, zip up the files for Colab with our script `zip_for_colab.py`, and upload these to your Colab environment.

We will be implementing the PSPNet [3] architecture. You can read the original paper here. This network uses a ResNet [2] backbone, but uses *dilation* to increase the receptive field, and aggregates context over different portions of the image with a "Pyramid Pooling Module" (PPM).

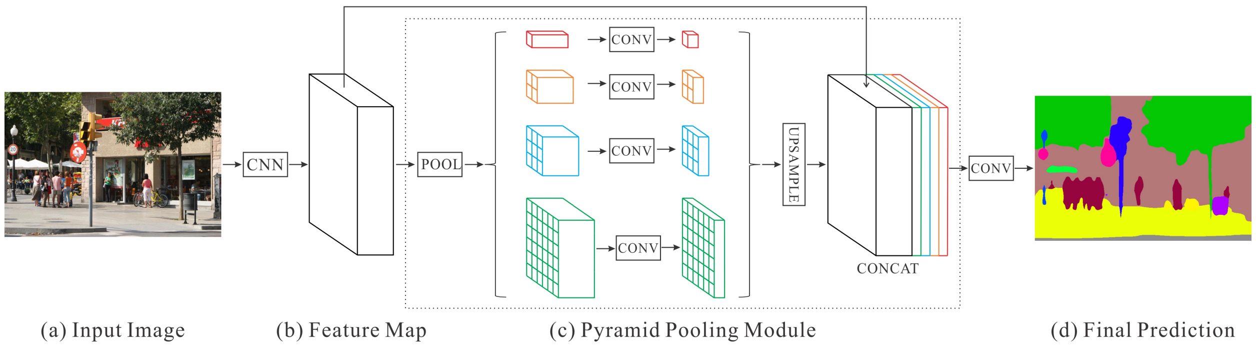


Figure 3: PSPNet architecture. The Pyramid Pooling Module (PPM) splits the $H \times W$ feature map into KxK grids. Here, 1×1, 2×2, 3×3, and 6×6 grids are formed, and features are average-pooled within each grid cell. Afterwards, the 1×1, 2×2, 3×3, and 6×6 grids are upsampled back to the original $H \times W$ feature map resolution, and are stacked together along the channel dimension.

You can read more about dilated convolution in the Dilated Residual Network here, which PSPNet takes some ideas from. Also, you can watch a helpful animation about dilated convolution here.

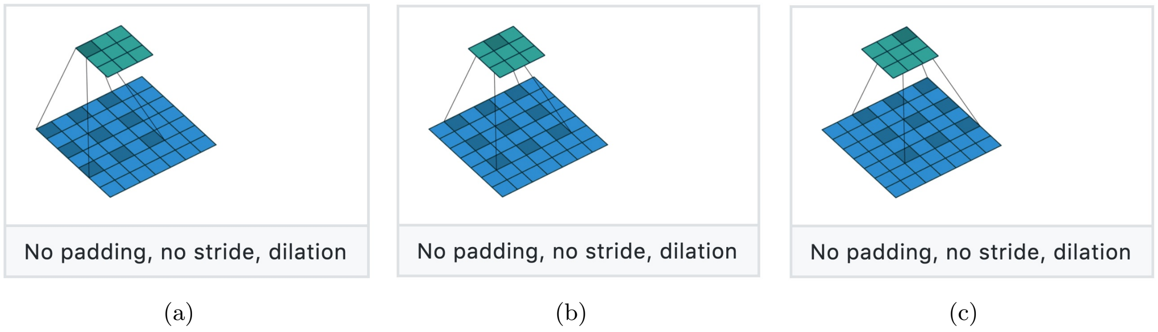

No padding, no stride, dilation
No padding, no stride, dilation
No padding, no stride, dilation
(a)
(b)
(c)

Figure 4: Dilation convolution. Figure source: https://github.com/vdumoulin/conv_arithmetic#dilated-convolution-animations

Suggested order of experimentation

1. Start with just ResNet-50, without any dilation or PPM, end with a 7×7 feature map, and add a 1×1 convolution as a classifier. Report the mean intersection over union (mIoU).
2. Now, add in data augmentation. Report the mIoU. (you should get around 48% mIoU in 50 epochs, or 56% mIoU in 100 epochs, or 58-60% in 200 epochs).
3. Now add in dilation. Report the mIoU.
4. Now add in PPM module. Report the mIoU.
5. Try adding in auxiliary loss. Report the mIoU (you should get around 65% mIoU over 100 epochs, or 67% in 200 epochs).

Extra Credit - Transfer Learning

Use the PSPNet trained on Camvid dataset as your pre-trained model and train it on Kitti road segmentation dataset. Report the mIoU that shows how well the model was able to transfer to Kitti dataset. (You will have to make an account with Kitti to download their data).

Rubric

- +8 pts: Part 1 Code
- +8 pts: Part 2 Code
- +8 pts: Part 3 Code
- +8 pts: Part 4 Code
- +40 pts: Part 5 Code (you'll need to reach 64% mIoU on your submitted grayscale PNG label maps to get full credit)
- +28 pts: Report
- -5*n pts: Lose 5 points for every time you do not follow the instructions for the hand-in format

Submission format

After running the colab, a new zip file with your model's outputs on the validation set called `grayscale_predictions.zip` will be created. Make sure that you put the zip file under the root directory of this project.

Refer to the Project 5 Github README.

Credits

Assignment developed by John Lambert and James Hays.

References

[1] Gabriel J. Brostow, Julien Fauqueur, and Roberto Cipolla. "Semantic Object Classes in Video: A High-definition Ground Truth Database". In: *Pattern Recogn. Lett.* 30.2 (2009).

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep Residual Learning for Image Recognition". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).* June 2016.

[3] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. "Pyramid Scene Parsing Network". In: *CVPR.* 2017.