18-794: Introduction to Deep Learning and Pattern Recognition for Computer Vision

Assignment 3: Semantic Segmentation

Instructor: Marios Savvides

TAs: Zhantao Yang, Alex Liao, Pradhumna Guru Prasad, Hrishikesh Gokhale, Sarah Alharbi

Due Date: Tuesday, Nov. 7, 2025 11:59 pm

Total Points: 100

Submission: Submit your solutions, code and pdfs on Gradescope.

- **Collaboration policy:** All are encouraged to work together BUT you must do your own work (code and write up). If you work with someone, please include their name in your write-up and cite any code that has been discussed. If we find highly identical write-ups or code or lack of proper accreditation of collaborators, we will take action according to strict university policies. See the Academic Integrity Section detailed in the initial lecture for more information. Cases of exact same code submissions will be reported instantly.
- **Late Submission Policy:** You have a total of **5 late homework days** without penalty for the entire semester. You can use up to three days on one homework or one day on 4 different homeworks. You cannot use half-days or any other fractions. After you've used all your late days, your homework will be worth half credit if it is up to 24 hours late, and worth zero credit after that with NO exceptions.
- **Submitting your work:**
  - We will be using Gradescope (`https://gradescope.com/`) to submit the Problem Sets. Please use the provided template only. Submissions must be written in LaTeX. All submissions not adhering to the template will not be graded and receive a zero.
  - **Deliverables:** Please **complete all TODOs and missing codes**, and submit all the `.py` and `.ipynb` files. Add all relevant plots and text answers in the boxes provided in this file. To include plots you can simply modify the already provided latex code. Submit the compiled `.pdf` report as well.

NOTE: Partial points will be given for implementing parts of the homework even if you don't get the mentioned numbers as long as you include partial results in this pdf.

Problem 1: Prepare Data Pipeline (20pts)

Finish data pipeline code for training and evaluation.

1. (5pts) In *datasets/voc.py*, complete the *VOCSegmentation* class. Specifically, please finish the *__getitem()__* method and *decode_target* method.

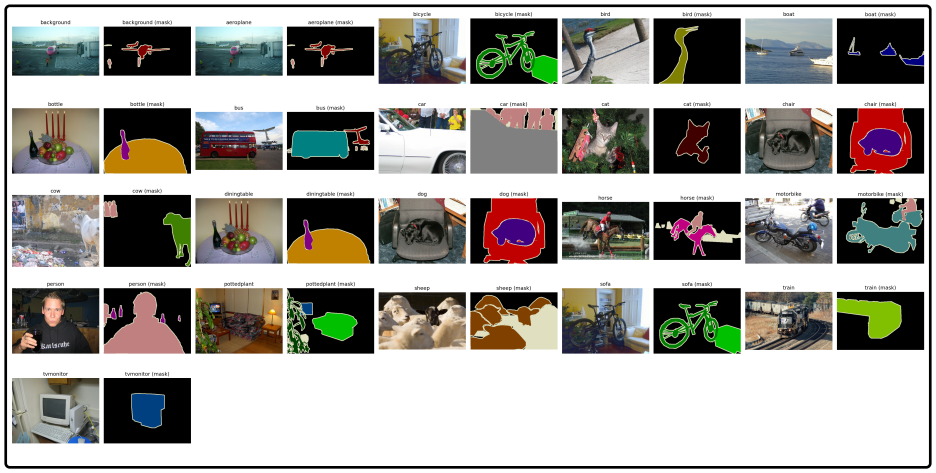
2. (5pts) How many categories are there in the dataset? In the training set, do you think this is a class-balanced set? And what can be the potential challenges while training a model on this dataset?

There are 21 categories in the dataset. This is not a class-balanced set. Class distribution for background is 73.36% while other 20 classes are around 1% per class. Potential challenges while training a model on this dataset could be model bias toward high occurance classes and bad performance on rare classes.

3. (5pts) Semantic Segmentation utilizes Accuracy and Mean IoU as evaluation metrics. Can you give an example to show why mIoU is a better metric than accuracy?

mIoU is a better metric than accuracy because of the class imbalance problem we mentioned in the previous question - it treats all classes equally and thus providing a more fair metric accessing the model's performance. For example, if an image is domoninated by background and only a small portion of the image is the actual object we care about, accuracy is not a good indicator in this context since the accuracy may be misleadingly high even the model fails on minority classes while mIoU may be lower and be a more fair evaluation metric.

4. (5pts) Pick one training image for each category. Plot the ground truth segmentation annotations (segmentation masks) side-by-side with the training images and show them in your report. Show your results as a large picture with 10 images per row. An example pair is giving for reference.



Problem 2: Build Segmentation Network (30 pts)

Please read the original DeepLabV3 and DeepLabV3+ paper and implement the networks.

1. (15pts) In network/_deeplab.py, complete ASPPConv, ASPPPooling and ASPP.

2. (15pts) Complete DeepLabHead & DeepLabHeadPlus. What are their differences? Please write your understanding in the report.

DeepLabHead differs from DeepLabHeadPlus in the way they use information from the backbone network. DeepLabHead takes the feature map from the backbone as the input and then use it to know what is in the image, while DeepLabHeadPlus introduces a fix to the problem DeepLabHead faces–sometimes the masks the DeepLabHead produce can be blurry with smooth edges for objects' boundaries–through concatenating the high-level features from ASPP modules with the original low-level feature map from the backbone, and thus producing better, and more sharp and precise objects boundaries.

Problem 3: Training and Evaluation (40 pts)

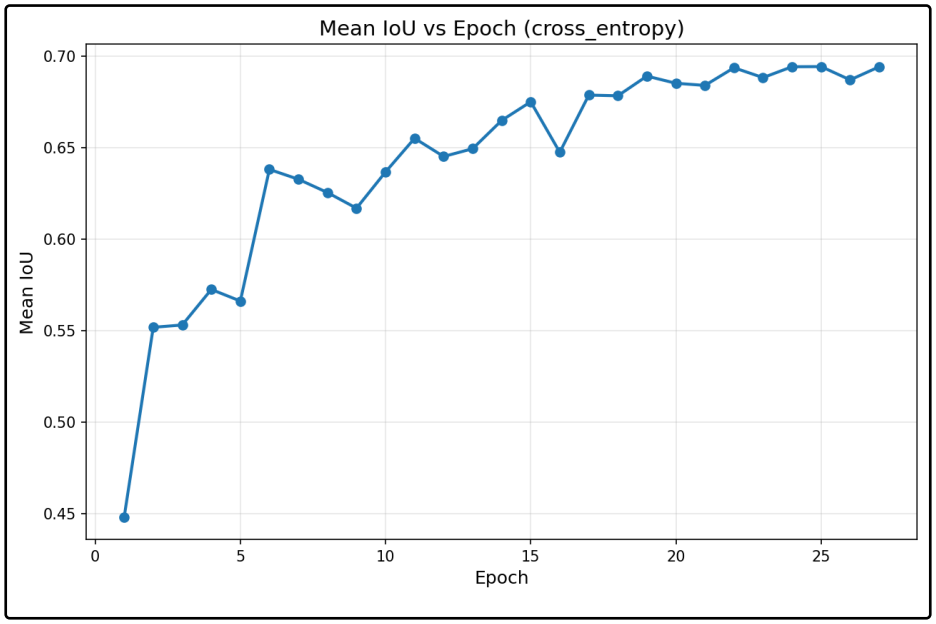
In "main.py" file, please complete the training loop and train the networks.

1. (5pts) Build the optimizer. Since we will be using an ImageNet pretrained ResNet, we want to scale down the learning rate on the *__backbone__* component. Set up an optimizer such that the learning rate of the *__backbone__* is 0.1x the

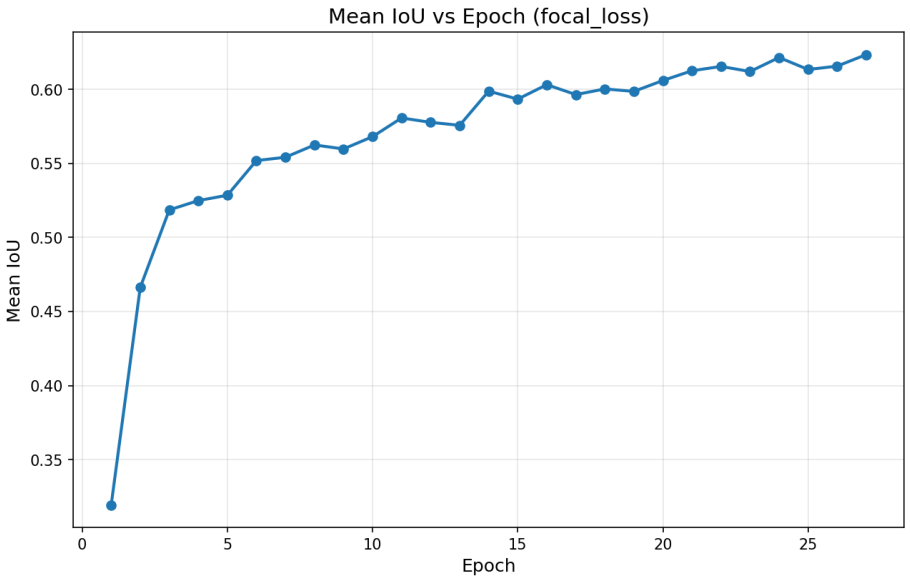
main learning rate.

2. (5pts) Build the learning rate scheduler. We will be using a step learning rate scheduler that reduce the learning rate by 0.1x every 1,000 iterations.

3. (15pts) Train the DeepLabV3+ with resnet50 backbone (imagenet pre-trained) on PascalVOC train split for 5k iteration, with *CrossEntropyLoss*. Plot the evaluation mIOU with an interval of 1 epoch in your report. (You should be able to train the model with a batch size of 8. In our test run, GPU memory usage is 6861MB.) We require your best mIoU to be at least 65 (the settings we provided and described should be able to reach around 65-67)

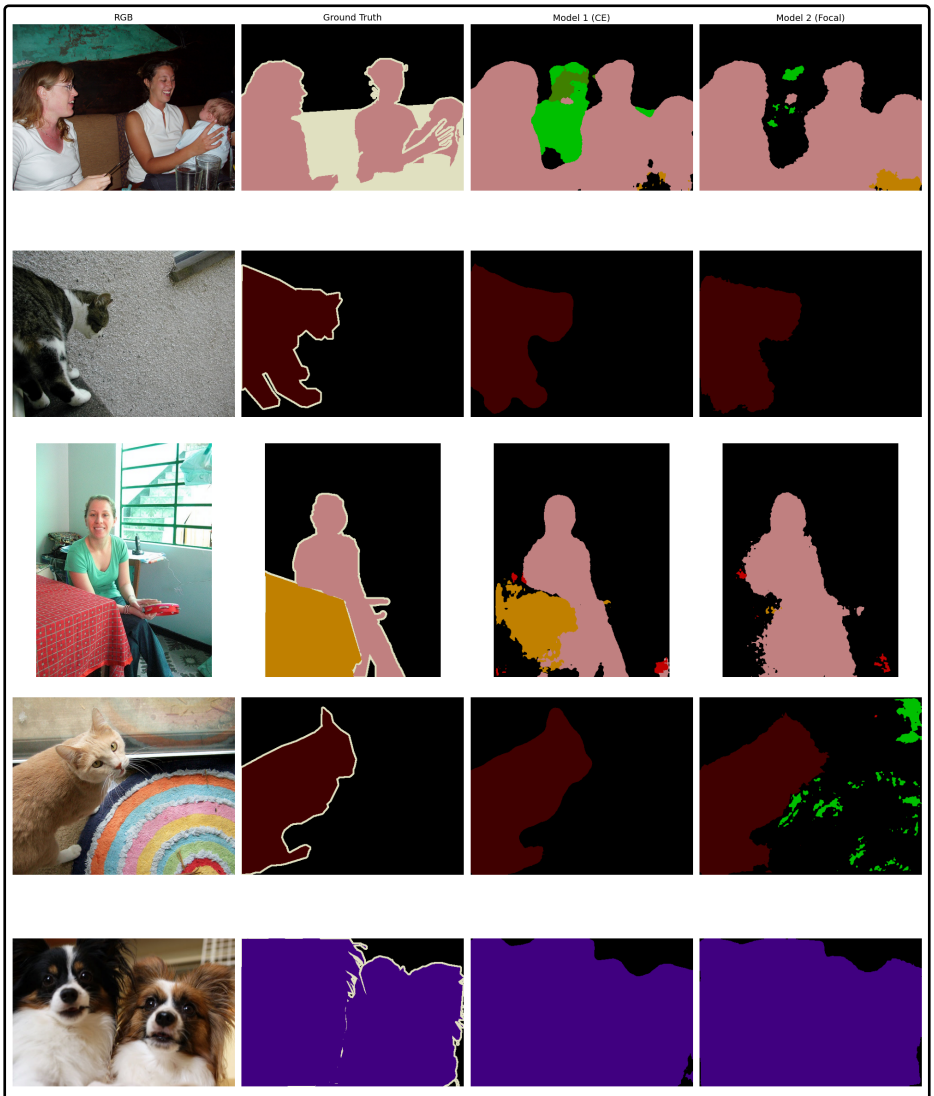


4. (10pts) Based on what you have learned in the dataset, train the same network using the same settings, but in this time, with a different loss function. Write your loss function in *utils/loss.py*. **What loss function do you propose to use and why?** Plot the evaluation mIOU with an interval of 1 epoch in your report. Note that you are allowed to use torch.nn functions/modules, but you should add something beyond what is already implemented by torch.nn (Hint: What do you learn from the Detection assignment when it comes to class-imbalanced training?). You may not observe performance improvements or you may even see slight degradation, if this is the case, please explain why you think this happens.



I propose to use Focal Loss because it can better handle the issue of class imbalance. It down-weights those easy example such as background and thus focusing more on those difficult examples. I observed slight degradation on performance. This may happen due to the reason that 1) the cross entropy loss with ignore_index can already handle the boundaries well, and 2) in our case, the dataset isn't in an extreme imbalanced scenario in which focal loss is mainly designed and suited for.

5. (5pts) For the trained models, report the best performances in terms of mIoU and accuracy for both models, and pick 5 images in problem 1, show the ground-truth vs both models' predictions side-by-side. There should be 4 columns, | RGB Image | Ground Truth Annotation | Model1 Prediction | Model2 Prediction |



- For Model1:
Overall Acc: 0.915231,
Mean Acc: 0.829170,
FreqW Acc: 0.853919,
Mean IoU: 0.694047,
Best mIoU: 0.6941.
- For Model2:
Overall Acc: 0.877949,
Mean Acc: 0.829967,
FreqW Acc: 0.799578,
Mean IoU: 0.623538,
Best mIOU: 0.6235376878813629.

Problem 4: Segment-Anything-Model (10pts)

Segment-Anything-Modelis recently proposed by Meta AI Research that produces complete high-quality object masks. Please follow the [installation] instruction, and download the pretrained model checkpoint. You can also modify the provided [Google Colab script] as well to save some AWS credits.

1. Run the model on the same 5 images in Problem 2.5 and append the SAM results as the 5th column. | RGB Image | Ground Truth Annotation | Model1 Prediction | Model2 Prediction | SAM Prediction |. What do you think are the differences between semantic segmentation and SAM?

Differences between semantic segmentation and SAM include: 1) the semantic segmentation models assign specific class labels to each pixel while SAM detects individual object boundaries and separate objects into masks instead of assigning specific class labels to each pixel; 2) the semantic segmentation models have a fixed number of classes while SAM can have infinite and undefined number of classes; 3) SAM does not require extensive retraining when working on a new task.