Final Competition of Deep Learning 2022

Version 1.1

October 28, 2022

Introduction

As we're sadly approaching the end of this course, we thought it would be fun to fire up a competition. You will compete with your classmates on who can find the best self/semi-supervised learning algorithm. We will give you a dataset with a large amount of unlabeled data and a small amount of labeled data to train your model, and the final performance of your model will be evaluated on a hidden test set and posted on a public leaderboard.

We will be working in teams of 3-4 for this project. **Please fill the Team List as soon as possible**. If you have not chosen teams by the deadline below, we will randomly assign you to a team.

Please choose a teamname made out of only alphanumeric characters (no spaces, no special symbols) to expedite grading.

Overview

The dataset of color images of different sizes, that has the following structure:

- 512,000 unlabeled 224×224 images,
- 30,000 labeled (bounding boxes of 100 classes) training images with different sizes,
- 20,000 labeled (bounding boxes of 100 classes) validation images with different sizes

Overall, we'll keep for ourselves a hidden test set, with which we'll be testing your models. In order to improve performance when training your model with

few labeled samples, you'll need to make use of the unlabeled data. You're encouraged to utilize all kind of SSL techniques in order to beat your peers on the leaderboard, and, of course, learn something new.

Please follow this tutorial on Object Detection to learn about the metrics we will use.

Schedule

- `11/02 23:55`: team information submission deadline: Team List
- `11/18 23:55`: the first leaderboard submission deadline
- `12/02 23:55`: the second leaderboard submission deadline
- `12/09 23:55`: the final leaderboard submission deadline
- `12/14 16:55-18:55`: virtual poster session
- `12/20 23:55`: paper submission deadline

Leaderboard

There are three leaderboards during the whole competition. For all the leaderboards, you need to submit a trained model to us, and we will test its performance with the hidden testing set. Only the final leaderboard is used to grade your project. However, we highly recommend you to participate at least one leaderboard, so you can make sure you understand how to submit your model.

HPC

We will be providing HPC access for you to train your models. This is provided via Cloud Bursting. Because of the restrictions on HPC, we must use Singularity to run jobs. A guide is available here.

Each team with receive 400 hours of HPC GPU hours, and each individual receives 24 GPU hours. More can be provided on a case by case basis, but this should be enough to complete your projects. *Please do not use Greene for your projects.*

To login to Burst, please follow the instructions above to access the log in node. On the NYU network, generally `ssh <user-id>@burst.hpc.nyu.edu` would just work. Then, use this command to launch an interactive process on Burst:

```
log-burst$ srun --account=csci_ga_2572-2022fa -p interactive --pty /bin/bash
```

Burst and NYU HPC runs on separate filesystems, so you *must* use an interactive job to interact with the filesystem. Because of this, in order to transfer files between NYU HPC and Burst, you must use the `green-dtn` host - e.g. launch an interactive process above, and run this to download and upload files.

```
dtn-1$ scp greene-dtn:/scratch/<user-id>/file/from/nyu/hpc/filesystem
```

You may use GPU nodes via the following 3 partitions - `n1s8-v100-1`, `n1s16-v100-2`, and `n1c24m128-v100-4`. Note that you *need* to use `-gres gpu:1` in your SLURM resource request to use these partitions.

To use the team GPU hours, use `-account=csci_ga_2572_2022fa_01`, where your team number is at the end, e.g. `_01`, `_02`, `_03`, ... To use your individual 24 GPU hours, use `-account=csci_ga_2572-2022fa`. Notice there's a dash instead of underscore.

Singularity

In order to submit jobs on HPC, you must use singularity, the filesystem may suffer under pure anaconda. Singularity images allow users to install arbitrary software in a container specific to the project. We provide a host container and an overlay filesystem, you should not need to modify the host container, but you may copy the overlay filesystem and modify it as you desire. If the overlay is updated, we will make the announcement on Campuswire. The overlays are dated, and the last overlay updated will be the conda environment used in the project.

Once you have started a job in Burst, use this command to launch singularity and run jobs:

```
dtn$ singularity exec --nv --overlay /scratch/DL22FA/overlay-11-07.ext3:ro \
  -B /scratch/DL22FA/unlabeled-112.sqsh:/data:image-src=/ \
  -B /scratch/DL22FA/labeled.sqsh:/data:image-src=/ \
  -B /scratch -B /scratch_tmp \
```

```
  /scratch/DL22FA/cuda11.2.2-cudnn8-devel-ubuntu20.04.sif /bin/bash
bash-4.4$ source /ext3/env.sh
bash-4.4$ python /scratch/DL22FA/eval.py
```

A short explanation of the command is here:

- `-nv` means to use nvidia GPUs if available
- `-overlay` binds the overlay filesystem, a single file containing the anaconda environment. You may copy the overlay and use :rw to make changes to it.
- `-B` binds filesystem mounts - your home directory is bound by default, but we have to manually bind `/scratch` and `/scratch_tmp`, as well as binding the datasets (*do not unzip the dataset*)
- **We provide the datasets on Burst**. 112x112 images are given in `/scratch/DL22FA/unlabeled-112` and full sized images are in `/scratch/DL22FA/unlabeled-224`.
- Please use `/scratch_tmp` for your storage needs. This filesystem will be deleted at the end of the semester. Use the `scp` command given above to transfer files.

Submission

- Make sure your evaluation code runs in the conda environment provided in the shared google drive. You may train with any set of imports as long as evaluation works with the conda environment provided. It is subject to change (requested additions), please keep track of campuswire for changes.
- Zip your project as `team_#.zip`. Submit it to BrightSpace.
- We will run eval.py by copying it into the root of your submission with the path to the hidden test set, and checking the metrics. Check that this runs for you with the given conda environment. This means to create a `model.py` in the root directory with a defined `get_model` function in your project.

Rules

- You are not allowed to use any external images or pre-trained weight. You will fail the course if you violate this rule.
- You are not allowed to use the labeled validation set to train your models. You will fail the course if you violate this rule.
- Make sure you test your submission before sending to us. You will get penalty for submitting code unable to run. Submitting to the leaderboard will allow you to test if your code will reasonably run on our grading servers.