International Institute of Information Technology, Hyderabad
Spring 2024 CS7.505: Computer Vision
Assignment 4: Segmentation and CLIP
8 April 2024

Instructions:

- The goal of this assignment is to get familiar with dense prediction models and CLIP.
- You should upload the assignment as a jupyter notebook with appropriate cells (markdown and code) containing: (1) code that you wrote, (2) keep relevant outputs, and (3) your report and observations (markdown cells). The file should be uploaded in the courses portal.
- We recommend Python.
- You may want to use Google Colab for both questions in the assignment.
- Include the assignment number, your name, and roll number in the first cell of the notebook submission.
- **Make sure that the assignment that you submit is your own work. Any breach of this rule could result in serious actions including an F grade in the course.**
- The experiments and writing it all together can take time. Start your work early and do not wait till the deadline.

Submission: Any time before *22nd Apr 2024, 23:59 IST*

1 Assignment

This assignment provides hands-on experience on the Unet architecture for segmentation. In the second question, we compare ImageNet vs. CLIP pretraining. Please use relevant libraries, do not implement from scratch. You are expected to solve both questions.

Q1: Segmentation [4 points]

1. [1 point] **Preparation.** We will use an existing illustrative code for segmentation using a Unet in Pytorch. Please make a copy of the following notebook as a google colab project and ensure you are able to run it: `https://www.kaggle.com/code/dhvananrangrej/image-segmentation-with-unet-pytorch/notebook`.

   The first cell should contain the necessary code to download the segmentation dataset (a subset of Cityscapes). Carefully study the code, especially focusing on the data and model.

   Do not change the model definition, dataset class, or evaluation metric (currently being computed over 10 images or 1 batch of the validation set). You may need to fix one bug `iter(data_loader).next()`, but this does not influence the other parts.

   Hint: You may train the model for 1 epoch as it takes quite some time to complete all 10 epochs.

2. [2 points] **Importance of skip connections.** The above model is a standard Unet architecture. The key novelty in the Unet are the skip connections between the encoder and decoder. But how important are they?

   In this experiment, modify the model to disconnect the skip connections. As compared to the original model, does the new model perform well? Report the results, both qualitatively and quantitatively (IoU over the same 10 images as before).

   Hint 1: When removing the skip connections, you will need to halve the number of input channels too.

   Hint 2: For a fair comparison to the above model, please train the new model without skip connections for the same number of epochs (1 if you followed the above point!).

3. [1 point] **Metric.** The metric implemented in the notebook calculates IoU directly (`logical_and` and `logical_or`) and does not seem correct.

   First, explain what is the issue with the metric in the current form.

   Next, re-write the metric function to instead compute an IoU for each class. Report the mean over all classes as the score for each image, and the mean over all images as the score for the validation set (or 10 images therein).

   Is the mIoU score different between the model with and without skip connections? Explain why.

Q2: Contrastive Language-Image Pretraining [6 points]

1. [1 point] **Setup models.** Load the ResNet-50 (RN50) model, initialized in two different ways:
   (a) ImageNet pretraining (`torchvision.models` can be used, specifically look at `IMAGENET1K_V1`); and
   (b) OpenAI's CLIP (see `https://github.com/openai/CLIP`).

   Do the visual encoders have the same architecture? If not, please describe and explain the differences.

   Hint: When you load the CLIP model, you will get both the vision and text encoders - be sure to differentiate between them as necessary.

2. [1 point] **Setup data.** Understand the ImageNet challenge dataset (1000 labels of ILSVRC).

   (i) What label hierarchy is used in ImageNet? (ii) What does a synset mean? (iii) Could grouping objects based on synsets lead to problems for visual recognition? (iv) State 3 types of visual differences we can expect to see in images with objects corresponding to the same synset.

3. [1 point] **Setup zero-shot CLIP.** Similar to the ImageNet pretrained RN50, set up CLIP to generate probability scores for the 1000 ImageNet categories.

   Test it with a few example images to check that it identifies the correct object category.

   Hint: You may treat the cosine similarities as "logits".

4. [1.5 points] **CLIP vs ImageNet pretraining.** Pick 10 classes from ImageNet (not all from the same branch, e.g., not all dogs). For each class:

   (i) Find 2 images that work well with CLIP, but not with ImageNet pretrained RN50. Reason about why this may be the case. From where did you get these images?

   (ii) Find 1 image that works well with ImageNet pretraining but not CLIP. Reason about why this may be the case. From where did you get these images?

   Note: For the purpose of this question, we will say that the model "works well" if it generates the correct category label within the top-5 highest scoring labels.

   Hint: Reading the CLIP paper (`https://arxiv.org/abs/2103.00020`) can help you solve this question quickly as it shows examples where ImageNet does not work, but CLIP works.

5. [1.5 points] **FP16.** While deep learning has primarily relied on "single" floating precision (32 bits or 4 bytes per parameter), 16-bit floating point numbers are useful for saving on memory. In this question, we will only consider CLIP's image encoder.

   (i) Convert the RN50 CLIP image encoder model to fp16. Calculate the (wall-clock) time required to encode an image. Estimate the difference between the fp32 (original) model and the fp16 model. Note, it is common to report the mean over 100 runs and indicate both the mean and standard deviation.

   (ii) For 5 images (1 per class), recalculate the probabilities using the fp16 model. Are there significant differences between the fp32 and fp16 outputs? Why?

   (iii) Using `nvidia-smi` or a profiler, note and explain the differences in memory usage for a forward pass between fp32 and fp16 models. If you do not use the profiler, please include some screenshots for the assignment.

   Hint: `https://pytorch.org/blog/understanding-gpu-memory-1/` Categorized Memory Usage may help.

2 Submission

We recommend that you submit a report as a single jupyter notebook with relevant cell outputs as mentioned at the top. In case you are not using Python, please share code (with instructions on how to execute) and a separate pdf report.

Submit the file in the courses / moodle portal before the deadline: **22nd Apr 2024 23:59 IST**. The moodle portal may show a different date due to the grace period, do not get confused.

The report/notebook should contain:

- A description of the problem, algorithms, results and comparison of methods based on the experiments you performed.
- Challenges you faced and learnings from the experiments.

Remember, you are expected to write the complete code for the assignment yourselves. DO NOT COPY ANY PART FROM ANY SOURCE not limited to your friends, seniors or the internet.