COL8628/COL828 Assignment 2
Exploring Open-Vocabulary Object Detectors for Breast Cancer Detection using Mammograms

Instructions:

- **Deadline: 11:59PM, 20th November, 2025.**
- **Anti-Plagiarism Policy**: Any form of plagiarism will result in an F-grade.
- **Late submission policy**: **No late submissions allowed.**
- Cite your sources properly in the report.
- Use Python 3.9 and PyTorch for experiments.
- Submission: Via Moodle. This assignment is to be done **individually**.
- **Dataset**: Download link

Overview

This assignment introduces the concept and application of *Open-Vocabulary Object Detectors* (OVODs). These models are designed to detect objects at inference time that were not explicitly seen during training, using suitable textual descriptions provided at test time. To enhance their performance, learnable prompts have been investigated as an effective strategy.

One of the prominent OVODs is Grounding-DINO (G-DINO) [1], which has demonstrated excellent performance across a variety of downstream tasks. In this assignment, we will explore the use of learnable prompts for the task of breast cancer detection in mammograms using G-DINO. Furthermore, we will analyze the robustness of these prompts under varying levels of domain shift and also explore Semi-supervised prompt tuning.

You are provided with three mammography datasets, each accompanied by bounding-box annotations in the respective `csv` files. Each bounding box corresponds to a region of indicating the presence cancer within the image. The three datasets share the same label sets (i.e benign/malignant, where the malignant samples have bounding box annotations) but differ in their underlying image distributions, with domain shifts arising from variations in patient demographics, mammography imaging technology, or both. While building models for real-world deployments, understanding the impact of domain-shifts and training models in a label-efficient fashion is quite important, especially for medical use cases.

Task 1: Zero-Shot Evaluation

In the first task, you will set up G-DINO and evaluate its zero-shot performance on each of the three mammography datasets. For this, you will employ hand-crafted text prompts to detect

cancerous regions in the images. You are encouraged to experiment with a variety of text prompts and select the one that yields the best detection performance.

Additionally, you may tune hyperparameters such as the bounding-box threshold and text-threshold to study their impact on the model's performance. Document your rationale for choosing particular prompts and hyperparameters in your report.

Finally, report the Average Precision (`AP`) of the predicted bounding boxes on the test sets of each of the three datasets respectively. Compare their performances and discuss how they vary with different text prompts and hyperparameter settings.

Task 2: Robustness Analysis of Learned Prompts

Building on the previous assignment, you are now familiar with various prompt learning techniques such as Context Optimization (CoOp)[4] and Conditional Context Optimization (CoCoOp)[3]. In this task, you will apply these methods to the problem of object detection using G-DINO and analyze their robustness under different levels of domain shift.

2.1 Implement CoOp for G-DINO and train a set of prompts on Dataset A. Report the `AP` at different thresholds on Dataset A's test set. Then, evaluate the zero-shot performance of the learned prompts from Dataset A on the test sets of Datasets B and C. This will help assess the generalizability of prompts learned on Dataset A to other domains.

2.2 Implement CoCoOp for G-DINO and train a set of prompts on Dataset A. Report the `AP` at different thresholds on Dataset A's test set. Similarly, evaluate the zero-shot performance of the learned prompts from Dataset A on the test sets of Datasets B and C. As before, this setting assumes that the model has not seen any images from Datasets B and C during prompt tuning.

Repeat the above two experiments for Datasets B and C, each time treating the other two datasets as test sets. Compare the results to analyze robustness across domains. Finally, report which of the two prompt-learning methods (CoOp or CoCoOp) better handles domain shifts by producing prompts that generalize effectively without overfitting to the dataset they were tuned on.

Task 3: Semi-Supervised Prompt Tuning

So far, we have explored zero-shot detection and supervised prompt tuning across different datasets. In this task, we extend to the semi-supervised setting, where one dataset is fully labeled while another contains only a small labeled subset and the rest of the images remain unlabeled. Semi-supervised learning methods aim to exploit this unlabeled data together with the labeled portion to improve performance.

One widely used method in this area is *FixMatch* [2]. The key idea is to enforce consistency between weakly and strongly augmented versions of the same unlabeled image: the model generates predictions on a weakly augmented input, and high-confidence predictions are taken as pseudo-labels. These pseudo-labels are then used to compute a loss on the strongly augmented input. In this assignment, we adapt this idea to G-DINO in the **CoOp (Context Optimization)** setting. That is, **only the prompt vectors are trained**, while the G-DINO backbone and detection head remain frozen. You are encouraged to read the *FixMatch*[2] paper before proceeding to understand the experimental setup thoroughly.

Experiment

Assume Dataset A is fully labeled and Dataset B has only a small labeled subset (i.e., 10% of its training set) while the remaining images are unlabeled. Train CoOp prompts for G-DINO using both supervised and unsupervised components:

$$\mathcal{L} = \mathcal{L}_{\text{sup}} + \lambda_u \, \mathcal{L}_{\text{unsup}},$$

where $\mathcal{L}_{\text{sup}}$ is the standard detection loss computed on labeled images from Dataset A and the labeled subset of Dataset B, and $\mathcal{L}_{\text{unsup}}$ is the FixMatch-style loss on the unlabeled images from Dataset B. For the unsupervised part, predictions above a confidence threshold τ on weak augmentations are used as pseudo-labels for the strongly augmented version of the same image. Once trained, report the `AP` on Dataset B's test set.

Comparison

Evaluate the trained CoOp prompts on Dataset B's test set and compare against the following:

1. CoOp prompts trained only on Dataset A and directly evaluated on Dataset B (zero-shot transfer).
2. CoOp prompts trained in a fully supervised manner on Dataset A and the complete labeled Dataset B (upper bound).

Hyperparameters

Use the following as starting points and explore their impact:

- Confidence threshold τ is the bounding box threshold, which can tuned based on your analysis from Task 1.
- Unsupervised weight $\lambda_u \in \{1.0, 2.0\}$ (default 1.0).
- Learning rate for CoOp prompts: 5×10^{-4} or 1×10^{-3}.
- **Hint:** For stable training, try using a warmup schedule: train with $\lambda_u = 0$ for a few epochs, then gradually increase to the target value.
- You are free to choose a randomized set of 10% samples from Dataset B as your labeled subset for this task.
- Augmentations: use medically plausible transformations. Weak augmentation may include flips and mild rescaling; strong augmentation may include intensity or contrast changes and small rotations.

Other Configurations

Repeat the same set of experiments by alternating the dataset roles: (i) train on fully labeled Dataset B with Dataset C as partially labeled, and (ii) train on fully labeled Dataset A with Dataset C as partially labeled. This will allow you to assess how semi-supervised CoOp prompt tuning performs across different dataset pairs and whether it consistently improves over the supervised baselines when testing across different domains.

Submission Format and Instructions

1. **Code submission.** Submit scripts for each task as follows:

   - **Task 1 (Zero-Shot Evaluation):** A single script `task_1.py`. This script should accept command line arguments for the dataset's test set path and the corresponding text prompt. The script evaluates the model on the specified test set and reports the required metrics.
   - **Task 2 (Supervised Prompt Learning with CoOp/CoCoOp):** Two scripts:
     (a) `train_task_2_x.py` — accepts training dataset path(s) and model save path. Trains CoOp or CoCoOp prompts on the labeled datasets. train_task_2_1.py corresponds to the CoOp script and train_task_2_2.py corresponds to the CoCoOp script.
     (b) `test_task_2_x.py` — accepts test dataset path(s) and model load path. Evaluates the trained prompts on one or more test sets and reports the required metrics. test_task_2_1.py corresponds to the CoOp script and test_task_2_2.py corresponds to the CoCoOp script.
   - **Task 3 (Semi-Supervised CoOp Prompt Tuning):** Two scripts:
     (a) `train_task_3.py` — accepts paths for the fully labeled dataset, partially labeled dataset, and model save path. Implements semi-supervised CoOp prompt training using the labeled and unlabeled data.
     (b) `test_task_3.py` — accepts test dataset paths and model load path. Evaluates the trained semi-supervised prompts and reports the required metrics.

2. **Environment.** Submit a `requirements.txt` file for setting up the Python environment.

3. **Documentation.** Provide a `README.md` with clear instructions on how to run each script. Use command line arguments for all paths and hyperparameters; do not hard-code them. Include example commands for training and testing each task.

4. **Model weights.** If the trained weights exceed the ZIP file limit of 25MB, upload them to a file-sharing service and include the link in your report. The trained weights shall **only** contain the learned prompts for each experiment and only one copy of the G-DINO weights used in your assignment. **Note:** The timestamp of the uploaded weights must be strictly before the submission deadline.

5. **Use of external repositories.** You may use publicly available repositories for G-DINO, CoOp, CoCoOp, FixMatch and related tools. G-DINO has multiple open-source implementations, one of the popular ones being from MMD (link). You are free to refer any of these implementations as per your choice. Any other external resource must be justified and cited in the report accordingly.

6. **Final submission.** Submit a single ZIP file on **Moodle** named `<EntryNumber>.zip`, containing:

   - all training and testing scripts (`.py` files),
   - `report.pdf`,
   - `requirements.txt`,
   - `README.md`.

   Do **not** include the datasets in your submission.

Evaluation Rubrics

- **Task 1: 15 marks**
  - Zero-shot evaluation with hand-crafted prompts and analysis across datasets.
- **Task 2: 40 marks**
  - Supervised CoOp prompt learning and robustness evaluation (train + test on various dataset pairs).
- **Task 3: 45 marks**
  - Semi-supervised CoOp prompt tuning on partially labeled datasets and comparison with supervised baselines.

References

[1] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Qing Jiang, Chunyuan Li, Jianwei Yang, Hang Su, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. In *European conference on computer vision*, pages 38–55. Springer, 2024.

[2] Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin A Raffel, Ekin Dogus Cubuk, Alexey Kurakin, and Chun-Liang Li. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *Advances in neural information processing systems*, 33:596–608, 2020.

[3] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Conditional prompt learning for vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

[4] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to prompt for vision-language models. *International Journal of Computer Vision*, 2022.