15-780: Graduate Artificial Intelligence
Homework 3

Due date: `2/25/2025 11:59pm`

Instructions

- Please provide detailed and well-reasoned responses to the questions below.
- Answers should be clear, concise, and well-supported with examples or explanations.
- Submit your responses and code on the course's gradescope page by the due date.

Deliverables

- Your responses to the question as a PDF document to **Homework 3 (written)**.
- Your code as a zip file to **Homework 3 (code)**.

Questions

Data and code template for this homework can be found at hw3.zip.

1 Linear Classifier

In this problem, we investigate image classification with supervised machine learning. As models are increasingly deployed in diverse real-world scenarios, it becomes crucial to understand how they behave under label noise and domain shift. The code template is provided in `hw3_1.ipynb` (you are *not* required to use this code template).

1.1 Linear probing on CLIP features

We will use linear probing on image features encoded by the CLIP image encoder. We have provided a set of pre-extracted features from a CLIP image encoder in `cup_train.npy`, `mug_train.npy`, `cup_validation.npy`, and `mug_validation.npy`. Each file contains the encoded features of real images in a particular class. Your goal is to implement a linear classifier on top of these features to predict the image labels.

Let $D_{\text{train}} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$ be your training data, where:

$$\mathbf{x}_i \in \mathbb{R}^d \quad \text{(feature vector from CLIP)} \quad \text{and} \quad y_i \in \{1, 2, \ldots, K\} \quad \text{(class label)}.$$

A linear classifier seeks to learn a weight matrix $\mathbf{W} \in \mathbb{R}^{d \times K}$ and bias vector $\mathbf{b} \in \mathbb{R}^K$ such that the predicted logits for sample i are

$$\hat{\mathbf{z}}_i = \mathbf{W}^\top \mathbf{x}_i + \mathbf{b}.$$

Questions:

1. Implement the linear classifier (e.g., you could simply use `torch.nn.Linear`) and train it using SGD and Adam. For each optimizer, plot the training and validation accuracies over the training trajectory for at least 10 checkpoints. The training accuracy should have converged at the end of training.
2. Plot the training loss over time for both SGD and Adam. Did you observe a difference between the optimizers?

1.2 Performance under label noise

In this section, you will evaluate the robustness of your linear classifier to label noise. To simulate label noise, modify your training data as follows:

1. Randomly select 20% of the training examples.
2. For each selected example, flip its label to the opposite label.

Questions:

1. Plot the accuracies with label noise over the training trajectory (with the Adam optimizer). How does the introduction of 20% label noise affect the training accuracy and convergence of your linear classifier?

1.3 Performance under domain shift

Real-world data often differ from the data on which models are trained. In this part, you will investigate how well a linear probe trained on real images (without label noise) generalizes to another domain. For this purpose, we provided `cup_validation_painting.npy` and `mug_validation_painting.npy`. Each file contains the encoded features of paintings in a particular class. Your goal is to evaluate the checkpoints in Q1.1 on these paintings.

Questions:

1. Plot the accuracies on paintings over the training trajectory (with the Adam optimizer). How does the accuracy on real images compare to the accuracy on paintings?

1.4 Effect of Regularization

Now we move on to understand how/if adding regularization to your linear probe changes the performance under label noise and domain shift.

The regularized objective for your linear probe can be written as:

$$\min_{\mathbf{W},\mathbf{b}} \mathbb{E}_{(\mathbf{x}_i,y_i)\sim D_{\text{train}}} \ell(\mathbf{W},\mathbf{b};\mathbf{x}_i,y_i) \;+\; \lambda R(\mathbf{W}),$$

where $\lambda \geq 0$ is the regularization coefficient and we will use ℓ_2 Regularization for $R(\cdot)$:

$$R(\mathbf{W}) = \sum_{j,k} \mathbf{W}_{j,k}^2.$$

Questions:

1. Implement an ℓ_2-regularized version of the linear probe. Try λ over a range of values (e.g., 1e-4, 5e-4, 1e-3, 5e-3)). Repeat the experiments in 1.2 and 1.3. Plot the accuracies under label noise and domain shift over the training trajectory (with the Adam optimizer).

2. Does ℓ_2-regularized linear regression improve the performance under label noise or domain shift? Provide your interpretation on why it works / doesn't work.

2 MLP and Residual Connections

In this problem, we play with the MLP architecture and aim to understand the importance of residual connections. The code template is provided in `hw3_2.ipynb`.

Questions:

1. Implement both the vanilla MLP architecture and the residual MLP architecture. Run the training code with the given hyper-parameters and show the resulting plots.

2. Repeat the pipeline with the SGD optimizer and learning rate $\{0.1, 0.01, 0.001\}$. Show the resulting plots.

3. Comment on the relation between the empirical results and the class materials.

   (a) What was the benefit if any of adding the residual connections?

   (b) Track the norms of the gradients of the parameters of various layers across training for both vanilla and residual MLP. What do you observe?