CSE 144 Applied Machine Learning
Final Project

UC Santa Cruz

Due: Mar 21, 2025, 11:59 PM

1 Transfer Learning Challenge

In the previous assignment, you've seen how to train a Simple CNN on the CIFAR-10 dataset, a simple dataset which does not require sophisticated data preprocessing or model design choice to achieve a decent accuracy. In this section, your task is to apply the transfer learning techniques you've learnt in the class to train a classifier for a more difficult dataset.

Please use the URL below to access the Kaggle page of this competition.

https://www.kaggle.com/competitions/ucsc-cse-144-Winter-2025-final-project/

The Kaggle page is for evaluation and submission only, please refer to this instructions for details.

1.1 Dataset

To save computation, we sample from different datasets, forming a total of 100 classes. For each class, there are 10 sampled training images, l0 eval images. All images in the training and the eval sets are colored, but they might come of different shape. Thus, it is strongly advised to resize all images to the same size(e.g. 224x224) before training.

The dataset directory is structured as follows:

```
train/
- 0/
  - 0.jpg
  - 1.jpg
  ...
  - 9.jpg
- 1/
....
- 99/
...
test/
- 0.jpg
- 1.jpg
...
- 999.jpg
- sample_submission.csv
```

In the `train/` directory, there are 100 directories, each of them named by the string label. Inside, each of them contains ∼10 images with a particular label. The `test/` directory contains 1000 unlabeled images.

You may only use the images in the `train/` directory for training. After finishing hyperparameter tuning, you will load all the test image and predict their labels using your final model. You need to submit a `submission.csv` file with two columns {`ID`, `Label`}. A sample submission `sample_submission.csv` is available as a template for your final submission. You should leave the `ID` column as is, but fill in the `Label` column with your predictions.

The training samples are already grouped by categories. Note that different implementations might lead to different category orders. For instance, class '0' may be mapped to class 1. To make sure your predictions are correctly evaluated, **you must use label 0 for class '0', label 1 for class '1', and similar for other classes**. If the label orders are scrambled, the best result you are going to get is no better than random guessing.

1.2 Implementation and training

We strongly suggest reusing the code from the previous assignment. After that, you can make modifications on the existing code by adopting more advanced architectures, more sophisticated data augmentations, tuning hyper-parameters and so on. **Different from the assignment, you are required to load strong pre-trained model weight and fine-tune it on our dataset**. This enables fast convergence and decent accuracy with limited data and compute. See TorchVision Model for some trained model weights.

A piece of advice: Modern models are usually several magnitudes larger than the model you've seen in the previous assignment. Using those models will introduce a significant amount of computation overhead. Before using such models, try playing with them by training on a tiny proportion of the training data to see if your model can fit into the GPU memory, and how long it takes to train your model. Also, since our dataset is a relatively easy dataset with only 1000 samples, large models may overfit.

1.3 Reproducibility

Please make sure your final training and test accuracies are reproducible. In other words, with your submitted code and documents, someone else should be able to reproduce most of the results you get with similar means and standard deviations. Good practices of reproducibility include using a manual seed, repeating experiments and reporting the averaged training and validation accuracies/losses, providing detailed instructions on how to run your code to produce the results, etc. **If your final accuracy cannot be reproduced by the code submitted, it will have an impact on your score of this competition.**

1.4 Submission

As described in Section 1.1, you need to submit a `.csv` file with image IDs and predicted labels on Kaggle. The public leaderboard will give you a rough estimated score. Note that **you should not use this score to infer your final accuracy**, because the public leaderboard is based on only ∼10% samples from the final test set. Please only use public leaderboard to make sure your submitted file is in the correct format.

Additionally, **you are required to submit your code/model on Canvas and instructions on how to reproduce your experiments**. In your submission to Canvas, please include 1) A report including instructions on how to run your code, experimental setup etc; 2) `.py` or `.ipynb` files containing all the code to reproduce the results (including the predictions), 3) a link to your final model weight in Google Drive. Do not upload your model directly to Canvas.