CSE 164 Computer Vision
Final Project

UC Santa Cruz

Due: **June 8, 2022, 11:59 PM**

1 Semi-supervised Learning Challenge

In the code exercise, you've seen how to do semi-supervised learning by adding the entropy of unlabeled data to loss on the MNIST 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 semi-supervised learning techniques you've learnt in the class to train a classifier for a more difficult dataset – MedMNIST.

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

`https://www.kaggle.com/t/09298242c7034b5ebaa8f065a7a1b025`

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

1.1 Dataset

The full MedMNIST dataset Yang et al. [2021] consists of 10 pre-processed datasets from selected sources covering primary data modalities (e.g., X-ray, OCT, Ultrasound, CT), diverse classification tasks (binary/multi-class, ordinal regression and multi-label) and data scales (from 100 to 100,000). To save computation, we choose a subset of 6 different classes and around 60 labeled images for training and 6000 unlabled images for the final evaluation. All images in the training and the test sets are colored and have shape 64x64.

The dataset directory is structured as follows:

```
MedMNIST/
- labeled/
 - 0/
 - 1/
 - 2/
 - 3/
 - 4/
 - 5/
- unlabeled/
 - images
- test/
- sample.csv
- wnids.txt
```

The `wnids.txt` file contains all unique labels in string format. To ensure everyone has the same labels for final submission, **please use the code below to obtain a dictionary that maps from string labels to integer labels** before the data preprocessing step.

```
label_dict = {}
for i, line in enumerate(open("MedMNIST/wnids.txt", "r")):
    label_dict[line.rstrip("\n")] = i
```

Note that different implementations might leads to different category order. For instance, class 'a' may be mapped to class 0. To make sure your predictions are correctly evaluated, you have to ensure your mapping is consistent with the one in `wnids.txt`.

In the `labeled/` directory, there are 6 directories, each of them named by the string label. Inside, each of them contains 10 images with a particular label. The `unlabeled/` directory more than 40,000 unlabeled images. **You may only use the images in the `labeled/` directory and the `unlabeled/` directory for training**. The `test/` directory has all the test images with no labels. After finishing hyperparameter tuning, you will load all the test image and predict their labels using your final model. You need to submit a `.csv` file with two columns {`image_id, label`}. A sample submission `sample.csv` is available as a template for your final submission. You should leave the `image_id` column as is, but fill in the `label` column with your predictions.

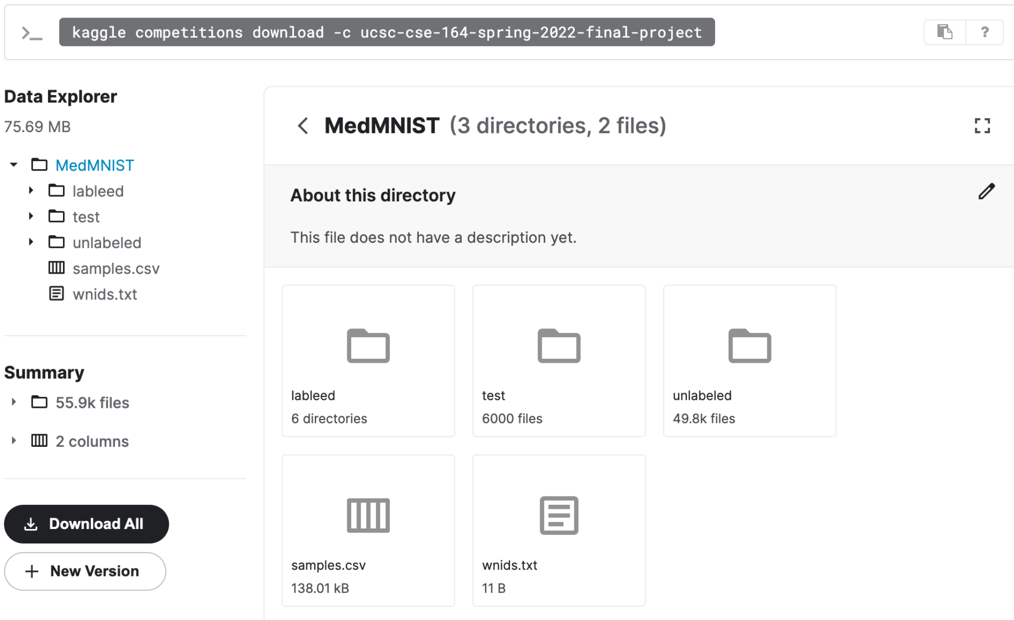

Figure 1: Click the "Download All" button to download the `.zip` file and use `unzip` command to get the dataset.

1.2 Implementation and training

We strongly suggest using the semi-supervised learning template code shown to you in the course as an initial baseline for further improvement. After that, you can make modifications on the existing code by adopting more advanced architectures, more sophisticated data augmentations, more effective semi-supervised learning techniques, tuning hyper-parameters and so on.

If your goal is to compete with other students, you may choose to use larger models. See TensorFlow Hub for some examples. Keep in mind that images in MedMNIST are only of size 64x64, but many advanced models are designed for images of size 224x224, which means you need to manually tune their architecture on MedMNIST. A piece of advice: Modern models are usually several magnitudes larger than the model you've seen in the the semi-supervised learning template code. Using those models will introduce a significant amount of computation overhead. For example, a ResNet [He et al., 2015] with 50 layers may take hours or even a day to train using a weak GPU (because you need to use a small batch size due to the limited GPU memory). 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. Besides, MedMNIST is a relatively small and easy dataset, so small models may perform just as well as those large models which are prone to overfitting.

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 600 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 you code on Canvas and instructions on how to reproduce your experiments**. In your submission to Canvas, please include 1) instructions of how to run all your code and information about setups, 2) `.py` or `.ipynb` files containing all the code to reproduce the results (including the predictions), 3) a link to your final model file (preferably in Google Drive). Do not upload your model directly to Canvas.

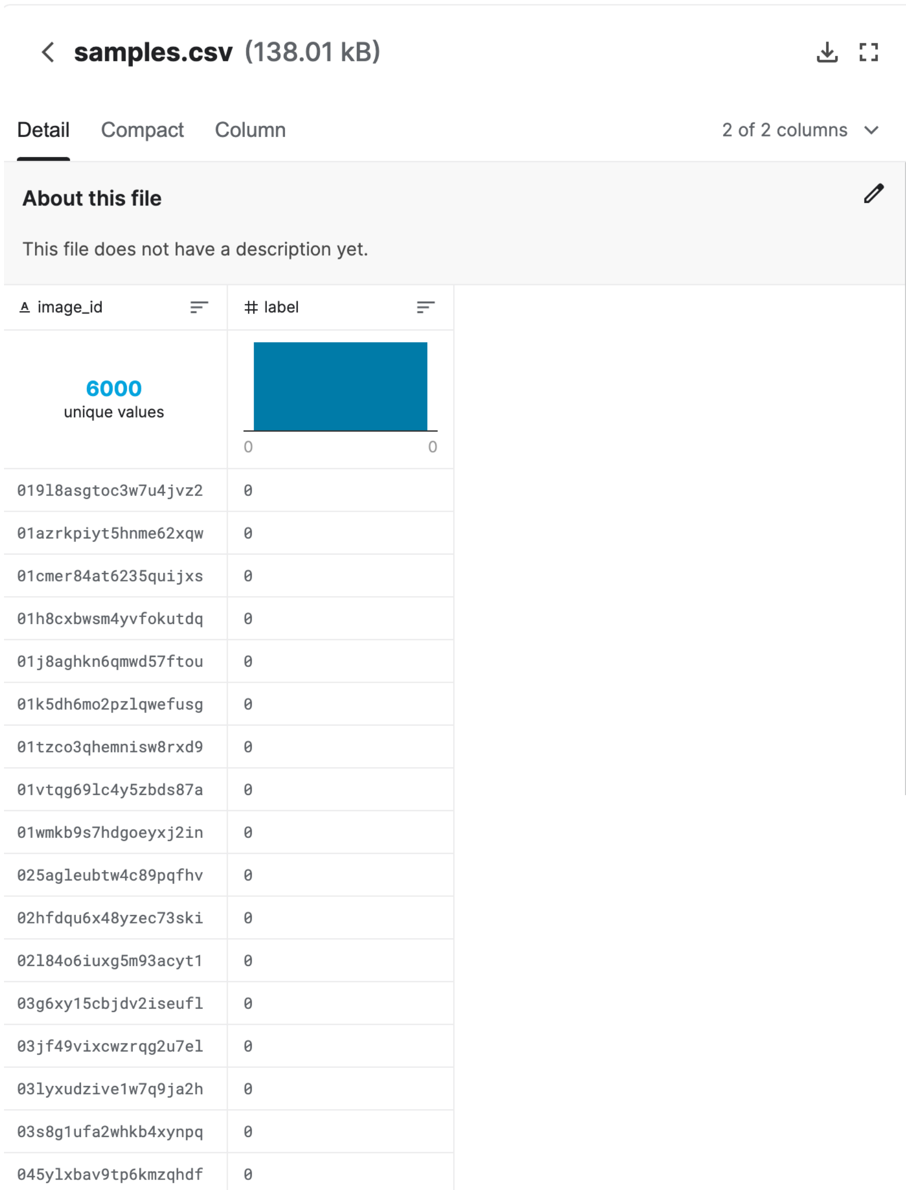

Figure 2: You must use the provided template file to submit your predictions or create a `.csv` file with exactly the same format. Otherwise, your submission will not be evaluated correctly.

References

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. corr abs/1512.03385 (2015), 2015.

J. Yang, R. Shi, and B. Ni. Medmnist classification decathlon: A lightweight automl benchmark for medical image analysis. In *IEEE 18th International Symposium on Biomedical Imaging (ISBI)*, pages 191–195, 2021.