CSE 144 Applied Machine Learning
Assignment 4

UC Santa Cruz

Due: **June 07, 2022, 11:59 PM**

1 Tiny ImageNet Challenge

In Assignment 4 your task is to train a classifier for a more difficult dataset – Tiny ImageNet.

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

https://www.kaggle.com/t/27dd29e8b1cd4987a9a27d6f44314eaf

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

1.1 Dataset

The full ImageNet consists of more than 1 million varying-dimension images with 1000 classes, which might be infeasible considering the amount of computation resources we have access to. The Tiny ImageNet dataset is a smaller version of the original ImageNet. Tiny ImageNet has images from 200 classes, each of them with 500 images. We further reduce the number of classes to 100 to save computation. All images in the training and the test sets are colored and have shape 64x64.

The dataset directory is structured as follows:

```
tiny-imagenet-100/


- test/
 - images/
- train/
 - n01443537
 - ...
- submission_sample.csv
- wnids.txt
- words.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("tiny-imagenet-100/wnids.txt", "r")):
    label_dict[line.rstrip("\n")] = i
```

In the `train/` directory, there are 100 directories, each of them named by the string label. Inside, each of them, the `image/` folder contains all the 500 images with a particular label. You can ignore files with `*_box.txt`. In the `test/` directory, the `image/` folder 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 `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.

To download the data from the Kaggle page shown below.

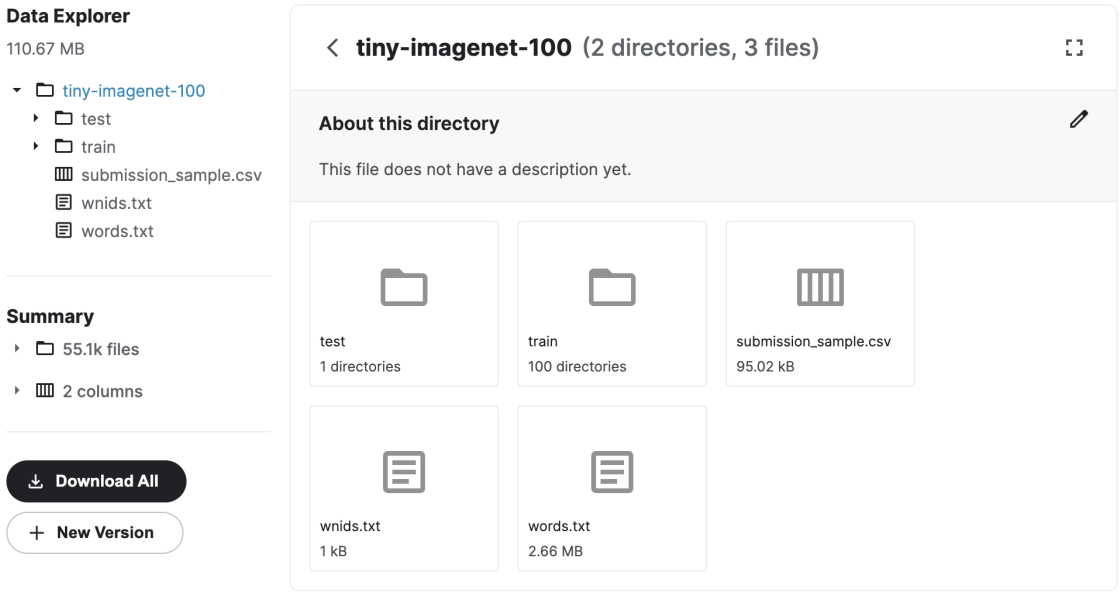

Figure 1: Click the "Download All" button to download the `.zip` file and use `unzip` command to get the dataset.

1.2 Implementation and training

We strongly suggest adapting the model you implemented for CIFAR-10 to this task to make sure your data is loaded and preprocessed correctly and using it as an initial baseline for further improvement. After that, you can make modifications on that existing model by changing the architecture, tuning hyperparameters, or adding data augmentations.

If your goal is to compete with other students, you can proceed to use larger pre-trained models from TensorFlow Hub. However, before diving into this, we strongly suggest going through Transfer learning and fine-tuning and Transfer learning with TensorFlow Hub to make sure you understand transfer learning and fine-tuning. A piece of advice: pre-trained models are usually several magnitudes larger than the models you trained for CIFAR-10, and some of them have hundreds of millions of parameters or even more. Therefore, fine-tuning introduces 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.

1.3 Reproducibility

Please make sure your final training, validation, 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 500 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.

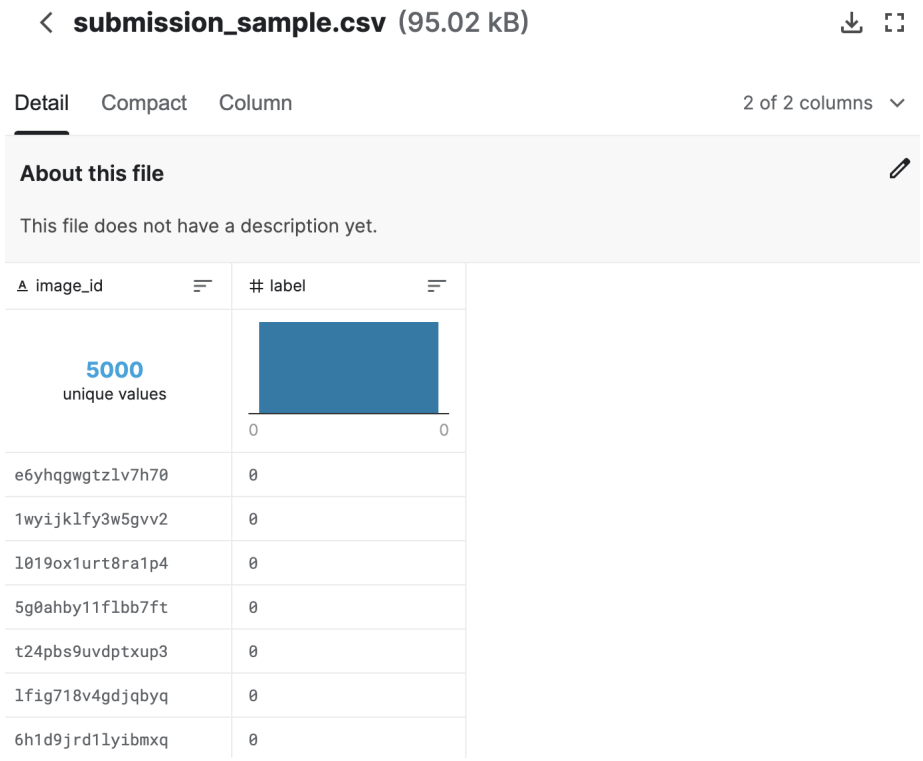

Figure 2: You must use the provided template file to submit your predictions or create a `.csv` file with exactly the same format. Otherwise, your submission will not be evaluated correctly.

Acknowledgment

The Tiny ImageNet we used in this project is a modified version of the original Tiny ImageNet provided by Fei-Fei Li, ANdrej Karpathy, and Justin Johnson for CS 231N competition at Stanford University. https://www.kaggle.com/c/tiny-imagenet

References

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