THE HONG KONG UNIVERSITY OF SCIENCE & TECHNOLOGY
Department of Computer Science and Engineering
COMP4211: Machine Learning
Fall 2020 Project
Due time and date: 11:59 pm, Dec 22 (Tue), 2020.

In this project, you will perform turbulence detection based on multispectral images.

1 Project 1

We categorize the turbulence according to its intensity as "moderate" (MOD) or "severe" (SEV). For each reported turbulence occurrence, you are given the corresponding multispectral images (of size 224×224) with the following 4 bands: (i) band 8: which detects water vapor; (ii) band 12: ozone; (iii) bands 13: infrared; and (iv) band 14: another wavelength in the infrared spectrum. You are also given some multispectral images that do not contain turbulence (class NIL).

1.1 Tasks.

You are required to build a convolutional neural network (using python and pytorch) for the classification of multi-spectral images into the SEV/MOD/NIL classes. The data set proj1_data can be downloaded from the link: https://hkustconnect-my.sharepoint.com/:u:/g/personal/jliude_connect_ust_hk/EepKnZR7IGNKr9Gob1XrltkBj0n5_9JyMDBBe0ZMig-r0g?e=b0Dj9P. The images are numpy arrays stored in pickle format, and can be read with the provided MyDataset class. Optionally, you can perform additional pre-processing on the data.

1. First, build a baseline CNN model (baseline1) using

   - the cross-entropy loss; and
   - basic data augmentation strategies including random cropping, flipping, color jittering and rotation.

   Note that the CNNs covered in the tutorials are for color images (with the three R/G/B bands), while the multispectral images here have four bands.

2. Next, you have to build another baseline model (baseline2), with the same architecture as baseline1, but with the LDAM loss in [1]. Note that you are expected to read the paper [1] in order to understand what it is and how it works.

   - For a sample (x, y), the LDAM loss is defined as

   $$L_{\text{LDAM}}((x,y);f) = -\log \frac{e^{z_y - \Delta_y}}{e^{z_y - \Delta_y} + \sum_{j \neq y} e^{z_j}}, \quad \text{where} \quad \Delta_j = \frac{C}{n_j^{1/4}} \text{ for } j \in 1, \ldots, k.$$

   Here, f is the CNN model, k is the number of classes, C is a hyper-parameter to be tuned by using the validation set, z_j is the jth class output from the model $f(x)$, and n_j is the number of samples in the jth class.

3. Finally, by improving upon baseline1 or baseline2, you have to build a final model by implementing additional techniques of your choice. You may also simply turn baseline1 or baseline2 as your final model, but your grade will be partially based on the accuracy of this final CNN model on a test set (which is hidden from you).

For performance evaluation, we will use the precision, recall, and F1-value evaluated for each of the three classes (NIL/MOD/SEV) separately. These are defined as:

$$\begin{aligned} \text{Precision} &= \text{TP}/(\text{TP}+\text{FP}), \\ \text{Recall} &= \text{TP}/(\text{TP}+\text{FN}), \\ \text{F}_1 &= 2(\text{Precision} \cdot \text{Recall})/(\text{Precision} + \text{Recall}). \end{aligned}$$

Here, TP is the number of true positives (i.e., ground truth: positive; prediction: positive) for the class of interest, FP is the number of false positives (i.e., ground truth: negative; prediction: positive), and FN is the number of false negatives (i.e., ground truth: positive; prediction: negative). For example, when considering the SEV class, FN includes those samples whose ground-truths are SEV while the predictions are either NIL or MOD.

In your report, you have to provide detailed descriptions and codes on

- data preprocessing (if any);
- baseline1
  - including model specification, tuning procedure of hyperparameters, training and validation set performance (precision, recall, and F1) results on the three classes (NIL/MOD/SEV).
- baseline2
  - including tuning procedure of hyperparameters, training and validation set performance (precision, recall, and F1) results on the three classes.
- final model
  - including model specification, tuning procedure of hyperparameters, training and validation set performance (precision, recall, and F1) results on the three classes.
  - Note that if you have used techniques not covered in the lecture notes (e.g., other convolutional neural network constructs or machine learning models), you have to describe these techniques clearly.

References

[1] K. Cao, C. Wei, A. Gaidon, N. Arechiga, T. Ma, Learning imbalanced datasets with label-distribution-aware margin loss, Advances in Neural Information Processing Systems, 2019.