1D-CNN- Instruction

1. **What this code does**
   This Python script trains and evaluates a 1D Convolutional Neural Network (1D-CNN) model to predict cysteine (Cys) concentration from SERS spectra. The inputs are the .npy spectra files produced by DataPreperation.py (saved into TrainingData/ and TestingData/). The script:

• Loads all .npy spectra from TrainingData/ and TestingData/
• Assigns a "true" Cys label to each spectrum based on the cultivar name at the start of the filename
• Trains a 1D-CNN regression model (PyTorch) using the training set
• Saves the best model weights as best_model.pth (based on lowest test loss)
• Evaluates the model on the test set and prints performance metrics (MAE, RMSE, R^2, etc.)
• Saves multiple output plots and CSV summary tables (loss curve, predicted vs true plots, per-culti var summaries)

---

2. **What the script expects from the user**

Input files
• The script expects .npy files inside these folders:

- ../TrainingData
- ../TestingData
  • These .npy files must be the output from your preparation pipeline (each file should contain spectra intensity arrays).

Important: filename format (cultivar prefix)
• The code determines the cultivar by checking which cultivar name the filename starts with. For example:

- AAC_Liscard_...something....npy → cultivar = AAC_Liscard
- CDC_Athabasca_...something....npy → cultivar = CDC_Athabasca
  • If a file does not start with one of the cultivar names listed in cultivar_cys_map, the script will skip it and print a warning:
  "Skipping unknown file (no cultivar prefix match)".

Important: array length must match input_length
• Each spectrum must have exactly 1496 points (set by input_length = 1496).
• If a file has a different length, the script will stop with an error.

---

3. **Parameters you may need to change**

Data paths
• These are the folders the script loads from (relative to where you run the script):

- train_data_dir = "../TrainingData"

- test_data_dir = "../TestingData"
  If your folders are located somewhere else, update these two lines.

Cultivar names and true Cys values
• The “true” Cys labels are hard-coded in:

- cultivar_cys_map = {...}
  If you add/remove cultivars or update true Cys values, update this dictionary.
  • Optional measurement standard deviations are stored in cultivar_sd_map and are only used if you later choose to display HPLC SD error bars.

Training settings
• You can change:

- num_epochs = 100
- batch_size = 32
- lr = 1e-4
  (Other variables like patience and factor are defined but not used in the current version of the script.)

---

4. What model it trains (brief description)
   • The model is a regression 1D-CNN called Cys1DCNN.
   • Input shape expected by the model is (batch, 1, 1496) where:

- 1 is the single “channel” (one spectrum)
- 1496 is the number of Raman-shift points
  • The network uses 4 convolution blocks (Conv1d + BatchNorm + ReLU + MaxPool) then fully connected layers to output a single number (predicted Cys).

---

5. **What files/folders it creates (outputs)**
   In the directory where you run this script, it creates/saves:

Model checkpoint
• best_model.pth (the best model weights based on lowest test loss)

CSV outputs
• test_predictions.csv (row-by-row true vs predicted Cys, with cultivar column)
• mean_vs_true_by_cultivar.csv (per-cultivar mean predicted, errors, SD/SE, counts)

Plots
• Model_Loss.png (training vs testing loss curve across epochs)
• calibration_plot_testing_set.png (scatter plot: predicted vs true on test set)
• Predicted_vs_True_Separate_Errors.png (per-spectrum error bars by cultivar)
• Predicted_vs_True_Separate_Errors_MEAN.png (mean predicted vs true per cultivar with error bars)
• bar_mean_true_vs_pred.png (bar plot of true vs mean predicted per cultivar)
• If the test set contains only one cultivar, it also saves:

- prediction_distribution_singlecultivar.png (histogram of predictions + confidence interval)

---

6. How to run it on a local computer

Step A: Install requirements
You need Python 3 plus:
• numpy, pandas, matplotlib
• scikit-learn
• torch (PyTorch)
• scipy (only needed for the single-cultivar confidence interval block)

Install example:

pip install numpy pandas matplotlib scikit-learn torch scipy

Step B: Make sure your folder structure matches the paths
Example layout (recommended):

```
ProjectFolder/
  TrainingData/          (contains .npy files from DataPreperation.py)
  TestingData/           (contains .npy files from DataPreperation.py)
  CNN_Code/
    train_1dcnn.py       (this script)
```

Because the script uses ../TrainingData and ../TestingData, you would run it from inside CNN_Code/.

Step C: Run the code