**Instructions for Targeted\_extraction**

Note: Before running Targeted\_extraction\_main.py, please run Database\_construction.py first

1. Download and open PyCharm

2. Open Targeted\_extraction file and configure the required runtime environment

3. Download the model (Classifier99.pth), then unzip it into the save\_weights folder.

4. Run Targeted\_extraction\_main.py

5. Select blank group, administration group, and m/z list for Input

6. Select the corresponding mode

7. Click Run in the Output (usually it takes a while to wait)

8. Click Export in the Output and select the output path to get the list of results

**Model training for peak discrimination**

1. Download and open PyCharm

2. Open Targeted\_extraction file and configure the required runtime environment (GPU recommended)

3. Copy the data file (example.mzML) for model training and the table (example\_mz.csv) containing the corresponding m/z information to the current directory

4. Open train\_data set.py

4.1 Modify parameters

m/z tolerance when constructing EIC: delta\_ mz

eics, mz\_value\_1 = get\_EICs1(file, mz\_list, delta\_mz=0.005)

intensity threshold at peak apex during peak detection: intensity\_threshold

peaks, peak\_widths\_end, mz\_end, cnn\_data = peak\_detection(eics, mz\_value\_1, intensity\_threshold=100)

4.2 Modify the path to save the data-set used for model training

savemat('train\_data.mat', {'signal': data\_end1})

plt.savefig("photo/model\_{:06}.jpg".format(aa))

5. Run train\_data set.py

6. Manually label the data-set and add category information

7. Run split\_data1.py to automatically divide the above labeled data-set into a training set (train\_data\_train.mat), a validation set (train\_data\_val.mat) and a test set (train\_data\_test.mat).

8. Open train\_classfication.py

8.1 Modify training parameters

Number of peaks to be fed into the model at one time for training: batch\_size

batch\_size = 32

Learning rate of the optimizer: lr

optimizer = optim.Adam(net.parameters(), lr=0.001)

Number of rounds of model training: epochs

epochs = 100

8.2 Save the weights of the model

torch.save(net.state\_dict(), "./save\_weights/Classifier{}.pth".format(epoch))

9. Run train\_classfication.py

10. Run validation.py to implement the evaluation of the model by the test set.