Trace, Machine Learning of Signal Images for Trace-Sensitive Mass Spectrometry: A Case Study from Single-Cell Metabolomics

Zhichao Liu,[†] Erika P. Portero,[‡] Yiren Jian,[†] Yunjie Zhao,[§] Rosemary M. Onjiko,[‡] Chen Zeng,[*,†] and Peter Nemes[*,‡]

[†]Department of Physics, The George Washington University, Washington, D.C. 20052, United States

[‡]Department of Chemistry and Biochemistry, University of Maryland, College Park, Maryland 20742, United States

[§]Institute of Biophysics and Department of Physics, Central China Normal University, Wuhan, Hubei 430079, China

***Correspondence to:** Peter Nemes (nemes@umd.edu); Chen Zeng (chenz@gwu.edu)

Table of Contents

Single-cell HRMS of Embryonic Cells

Reagents and Chemicals

LC-MS grade methanol, acetonitrile, formic acid, and water were purchased from Fisher Scientific (Hampton, NH). ACS grade sodium chloride, potassium chloride, and magnesium sulfate were from Fisher Scientific. Trizma hydrochloride, trizma base, and cysteine were from Sigma Aldrich (Saint Louis, MO).

Sample Collection and Metabolite Extraction

Embryos were obtained via gonadotropin-induced natural mating of adult *Xenopus laevis* following established protocols,[1] approved by the Institutional Animal Care and Use Committee (IACUC) of the University of Maryland, College Park (IACUC # R-DEC-17-57) or the George Washington University (IACUC # A311). The jelly coating from recently laid embryos was removed using a 2% cysteine solution as described elsewhere.[1] Dejellied embryos were transferred to a Petri dish containing 100% Steinberg's solution, prepared following previous protocols.[2] A fabricated microprobe[3] was used to aspirate ~10 nL of cellular content from identified dorsal-animal D11 cells in N = 5 different 16-cell *X. laevis* embryos, each collected from a different set of father and mother. Small polar metabolites were extracted from the aspirate in 4 µL of aqueous solution containing 40% (v/v) acetonitrile and 40% (v/v) methanol at 4 °C, vortex-mixed for ~1 min to facilitate lysis and metabolite extraction and centrifuged at 8,000 × *g* for 5 min at 4 °C to settle out cell debris. The cell debris was stored in the extraction solution at –80 °C until analysis.

CE-ESI-MS Analysis

Cell extracts were analyzed using a laboratory-built CE-ESI platform that was coupled to a quadrupole time-of-flight mass spectrometer (Impact HD, Bruker Daltonics, Billerica, MA). The same platform that we recently described in detail elsewhere was used here.[2, 3] In this study, 10 nL of cell extract were hydrodynamically injected into a fused silica capillary (40/105 µm inner/outer diameter, 1 m length) filled with 1% formic acid as the background electrolyte. Compounds were electrophoretically separated by applying +19,000–21,000 V to the background electrolyte at the inlet end of the capillary (yielding ~7.5 µA CE current). The outlet end of the CE capillary was fed into a coaxial sheath-flow electrospray ionization (ESI) interface, which supplied earth-grounded 50% (v/v) methanol in water (0.1% v/v formic acid) at 1 µL/min. Stereomicroscopic investigation of the electrified liquid meniscus at the tip of the emitter and mass spectrometric analysis of the generated ion current revealed the formation of stable Taylor-cone at approx. –1,700 V spray potential (applied to the mass spectrometer inlet plate orifice) and ~5 mm distance from the mass spectrometer orifice. The distance between the emitter tip and the mass spectrometer orifice was controlled to maintain the ESI source in the cone-jet electrospraying regime, which is most efficient for ion generation in the micro-flow regime.[4] Generated ions were analyzed at 2 Hz spectral acquisition rate between *m/z* 50–550 using the mass spectrometer under identical conditions to our previous study.[3]

Supplementary References

1. Sive, H. L.; Grainger, R. M.; Harland, R. M., *Early development of Xenopus laevis: a laboratory manual*. Cold Spring Harbor Laboratory Press: Cold Spring Harbor, N.Y., 2000; p ix, 338 p.
2. Onjiko, R. M.; Moody, S. A.; Nemes, P., *Proc. Natl. Acad. Sci. U. S. A.* **2015,** *112*, 6545-6550.
3. Onjiko, R. M.; Portero, E. P.; Moody, S. A.; Nemes, P., *Anal. Chem.* **2017,** *89*, 7069-7076.
4. Nemes, P.; Marginean, I.; Vertes, A., *Anal. Chem.* **2007,** *79*, 3105-3116.

Trace User Manual

Introduction

This document details the setup and execution of *Trace*, which uses deep learning (DL) to automate selection of signals from high-resolution mass spectrometry (HRMS) datasets. Input files are imported in open-access format (mzML), thus facilitating compatibility with broad types of mass spectrometers. The software package is provided as electronic supplementary document and can also be downloaded from: https://github.com/zhichaoliu2/Trace.

Trace performs a two-step analysis of each HRMS data file. During preprocessing and initial screening of the raw data, centroid MS data are surveyed to construct a series of one-dimensional (1D) extracted ion chromatograms (EICs), which are then screened by a continuous wavelet transform (CWT) to locate the center of each potential signal in the [mass/charge (*m/z),* time (*t*)] space. These results are exported into the "*Results_Initial-pks.txt"* file. Next, a two-dimensional (2D) image is rendered for each of these detected molecular features in the (*m/z, t*) space based on the corresponding HRMS files in profile mode, thus ensuring finer resolution in the m/z dimension. Results from this data processing step are exported into the "*Results_Images-pks.txt"* file. Finally, molecular features from "*Results_Initial-pks.txt"* are evaluated using DL. Features passing quality check are exported into the "*Results_Final-pks.txt*" fine, which tabulates the following peak parameters: accurate *m/z* values, separation time, peak intensity, peak area, signal-to-noise ratio, and peak membership information. These results can be imported into other software tools to facilitate metabolite identifications and quantification.

Our neural network model, once trained on a curated dataset, achieves consistent prediction of high accuracy and low false positive, independent of testing sets chosen from different initial screening criteria. Trace provides sufficient sensitivity and robustness to recognize trace-level signals in open-source HRMS datasets, even including single-cell studies (CE-ESI-HRMS demonstrated here).

Computer and Programming Needs

Hardware Requirements

- Operating system: All codes are written in Python (Version 2.7, also compatible with Python 3) and supported on Linux, Mac, and Windows systems.
- Memory: We recommend 32+ GB RAM for fast data processing and sufficient data storage capacity for raw and processed HRMS files.
- CPU: Clock speed and core number are driving factors of processing time. Multiprocessing is implemented automatically based on the number of available cores. Therefore, faster CPUs and multiple cores are recommended.
- GPU: A graphics processing unit (GPU) is recommended to speed up the initial training of the DL neural networks. While this training is provided under default settings for our CE-ESI-MS data, users are advised to perform independent training for their customized datasets.

Software Requirements

- *Trace* requires prior installation of Anaconda. To install Anaconda, please follow instructions at: https://www.anaconda.com/download/. Make sure to choose 'add conda to your PATH' during the installation.
- TensorFlow is required to conduct target shape evaluation (TensorFlow 1.0 or higher). To install TensorFlow, please follow instructions at: https://www.tensorflow.org/. Please ensure that Joblib is also installed for multiprocessing, e.g., by executing the 'pip install joblib' in the MS-DOS terminal after Anaconda is installed.

Installation Procedure

1. Prepare Input Files

- Export and convert the MS^1 spectra from each raw (primary) HRMS data file into the open-access mzML file format in both centroid and profile mode.
  Note that *Trace* calls on both the centroid and profile data to reduce data processing time. Note: Facilitating these steps, data converters are available from each mass spectrometer vendor or are freely downloadable from Proteowizard at: http://proteowizard.sourceforge.net.
- Organize profile files ("Input_profile.mzML") and centroid files ("Input_centroid.mzML") into the folder from which *Trace* is executed to simplify the execution of line commands.

2. Software Usage

1. Open the TRACE.py file (using an integrated development environment, IDE, such as PyCharm, or other editor capable of editing .py files) in the folder that contains the main program. Locate "Big_RAM" in the file and set BIG_RAM to: "0" if the available RAM is $<$ 32 GB and "1" if the available RAM is $\geq$ 32 GB.
2. Open terminal (e.g., Command Prompt "cmd" in Windows), and navigate the active directory to the folder of *Trace*.
3. Execute *Trace* by running the line command: ***python TRACE.py*** (hit ENTER). Three text files will be generated in the "Results" folder, which are the following:
   - "Initial-pks.txt": This file contains initial scanning results.
   - "Images-pks.txt": This file contains 2D images of the potential signals by the initial scanning.
   - "Final-pks.txt": This file contains the final peak list.

3. Parameters and Customization

General Parameters

“Input_centroid.mzML”	Name of the HRMS file containing MS^1-only spectra in centroid mode.
‘Input_profile.mzML’	Name of the HRMS file containing MS^1-only spectra in profile mode.
BIG_RAM	Specifies the available RAM on the computer. Set to “0” if RAM < 32 GB (default setting). Set to “1” if RAM ≥ 32 GB.
mz_min	The minimum *m/z* value to be evaluated. Default setting in our study: 25.0.
mz_max	The maximum *m/z* value to be evaluated. Default setting in our study: 550.0.
mz_r	The *m/z* bin for signal detection and evaluation (window is ± this value). Default setting in our study: ± 0.005 Da.
window_mz	m/z window used to plot the signal image on each side of the m/z value. This value is set depending on m/z sampling rate and the average width of each m/z peak. Default setting in our study: ± 0.024 Da.
window_rt	Spectral window used to plot the signal image on each side of the separating peak. This value is set depending on the average peak width in a separation experiment. Default setting in our study: ± 30 spectra.
ms_freq	Spectral data acquisition rate. Default setting in our study: 2 (spectra/second).
K_means	Option K-means cluster analysis of signal images. We did not utilize this option in this study.

Parameters of CWT*

widths	Wavelet widths for CWT initial scanning Default calculated as: range(1, 10* ms_freq, 1) + [20* ms_freq]
min_len_eic	Minimum length of a EIC to be scanned by CWT. Default setting: 6 data points.
gap_thresh	Threshold for ridge detection. Default setting: 1.
min_length	Minimum length a ridge line needs. Default setting calculated as: len(width)*0.2 (i.e., 1/5-th the number of widths.)
min_snr	The signal-to-noise (SNR) threshold for detecting a signal using CWT. Default setting: 4.
window_size	Size of window used to calculate noise. Default setting: 30
perc	Percentile within the window_size used to calculate noise floor. Default setting: 90.

*Note: Additional information on CWT is available from reference (P. Du, R. Sudha, M. B. Prystowsky, and R. H. Angeletti, Data reduction of isotope-resolved LC-MS spectra, Bioinformatics 2007, 23, 1394-1400).

Parameters of Deep Learning-based Image Evaluation

W_conv1	First convolutional layer of CNN. Default setting: [4, 4, 1, 32] (4*4 convolutional filter; 1 channel; 32 filters).
h_pool1	Pool layer after the first convolutional layer. Default setting: max_pool (ksize = [2,2], padding='SAME').
W_conv2	Similar to W_conv1. Default setting: [4, 4, 32, 64]
h_pool2	Similar to h_pool1. Default setting: max_pool (ksize = [2,2], padding='SAME').
W_fc1	Fully connected layer after h_pool2. Default setting: 256 (number of neurons in this layer).
keep_prob	The portion of information to be kept during training the machine. Default setting: 0.5.
learning_rate	The rate of adjusting the weights during the training process. Default setting: 0.0001.

Output Files

Initial_pks.txt	Results of initial scanning based on centroid data.
Images_pks.txt	Images of the initial signals (60*12 pixels for each image by default). Each line stands for a flatted signal image, i.e., 720 numbers in each line in default case.
Final_pks.txt	Final signal list, (m/z, retention time, intensity, peak area, SNR) for each signal.

Training the Deep Learning Model

Although we are providing the trained DL model from this study, users are advised to perform independent training for customized datasets, particularly if different types of experimental conditions or technologies were used to acquire the data. Besides the python code provided (*Training_Model.py*), users need to prepare their own training data, specifically both positive (true) and negative (false) signal sample images (*imgs-train.txt*) and their labels (*label-train.txt*). In the image file, each line should stand for a flatted signal image (rows connect to a single row in order). The label file should be in one column indicating whether (1) or not (0) the signal image stands for a true signal in the image file of corresponding row. For example, if N (>1000 recommended for better model performance) samples are collected and labeled, then the data size should be (N*720) for image file and N*1 for label file. To run the program, execute the *Training_Model.py* code by entering "python Training_Model.py" in the terminal window. The TRACE.py file highlights user-defined parameters for potential optimization:

```
import os
import math
import numpy as np
from scan_cwt_1 import scan
from scan_cwt_1_mp import scan_mp
from getImage_2 import get_image
from predict_3 import predict
import multiprocessing as mp

num_cores = mp.cpu_count()  ## Count the cores of the PC.
print ('Number of cores detected in this PC:', num_cores)

NUM_C = np.max(num_cores-2, 1)  ## MP use (all-2) threads by default.

Big_RAM = 0  ## 0 or 1: if the RAM of PC is big enough (>= 32 GB)

K_means = None  ## Or some interger (2~10 recommended); for k-means
clustering of signal images

window_mz = 6  # the m/z range is 6 points (on both sides)
window_rt = 30  # The time range is 30 points (on both sides)

RESULTS_PATH = "./Results"
if not os.path.isdir(RESULTS_PATH):  ## Will create a folder for results.
os.makedirs(RESULTS_PATH)

## First step: Preprocessing and initial scanning.
pks_initial =  scan_mp( 'Input_centroid.mzML', NUM_C = NUM_C )  ##

## Second step: Signal image evaluation.
images = get_image( 'Input_profile.mzML', pks_initial, Big_RAM , window_mz,
window_rt)

pks_final = predict(images, pks_initial, K_means = K_means)

print ('Done! Final results in ./Results folder.')
```

Supplementary Figures

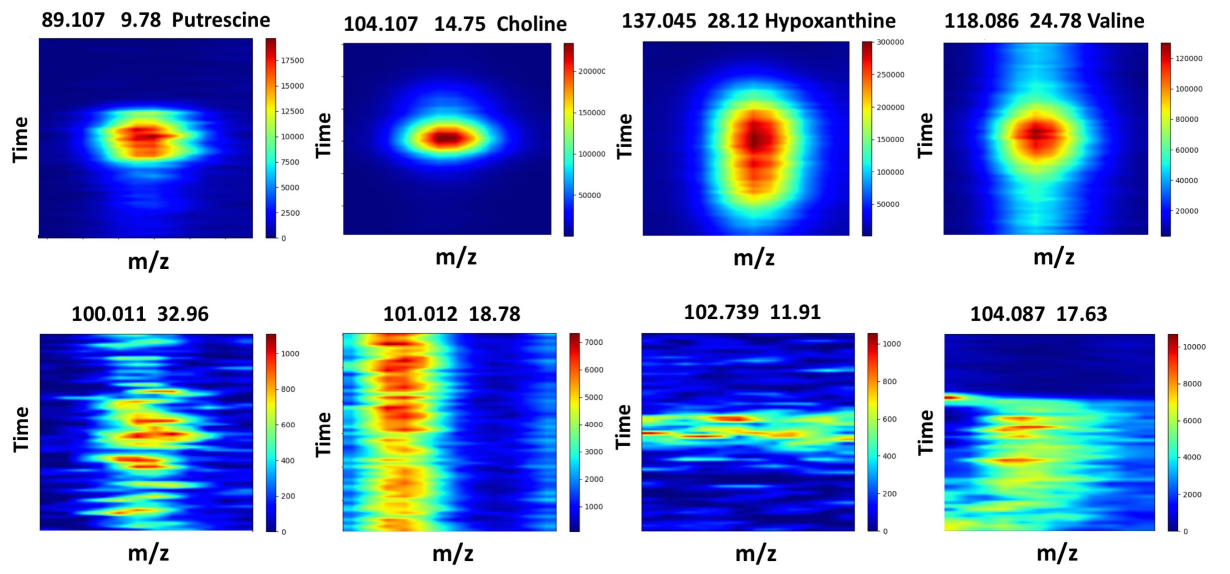

Figure S1. Examples of high-fidelity **(top row)** and low-fidelity **(bottom row)** signal images from our training dataset from single-cell CE-ESI-HRMS experiments. Each figure is annotated with the accurate mass and migration time information as well as compound name if identified.

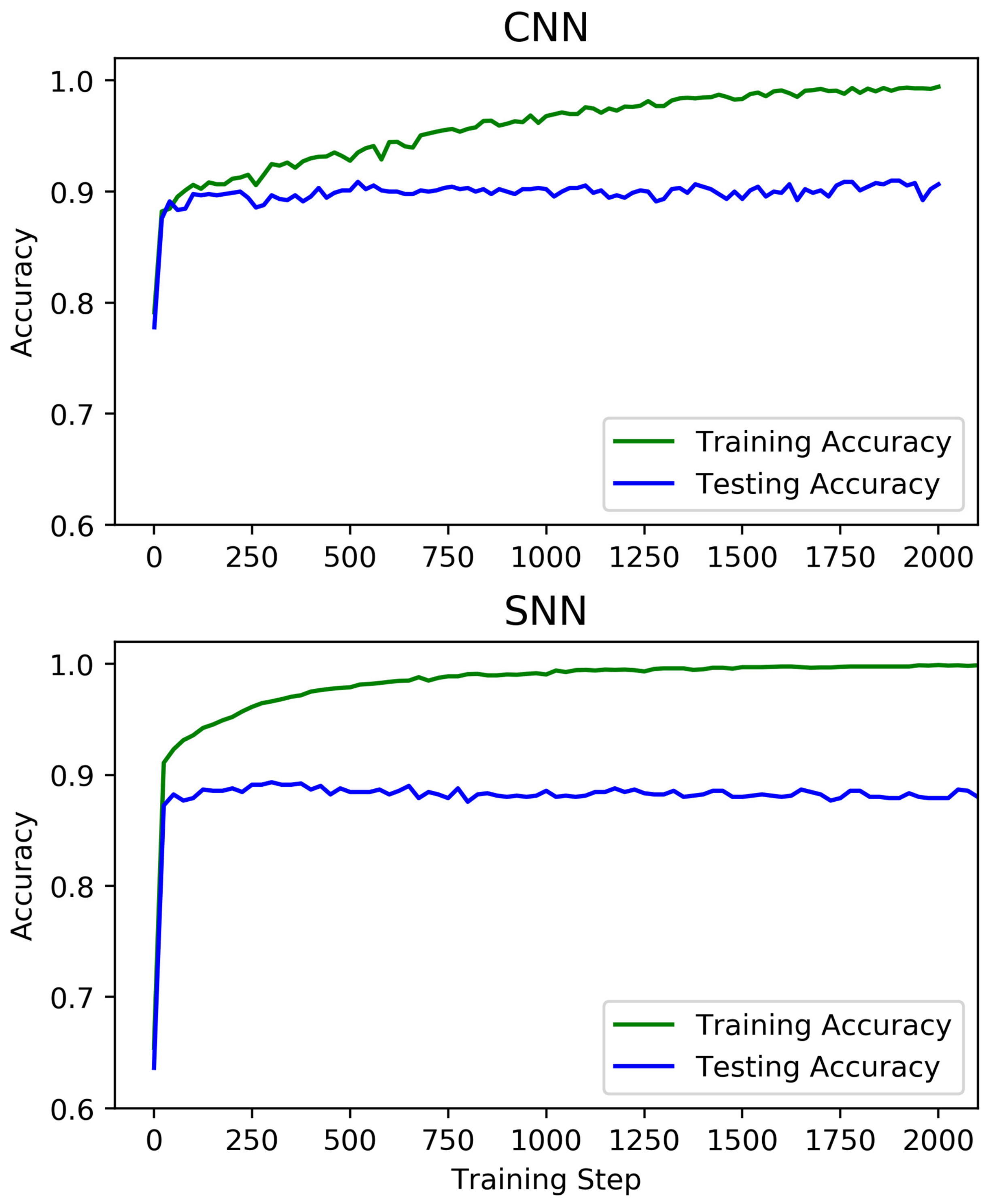

Figure S2. Training and testing accuracy of two models: **(Upper)** CNN model and **(Lower)** SNN model. The training accuracy can achieve up to 99.9% for both models, while the test accuracy plateaued at around 91% and 90%, respectively.

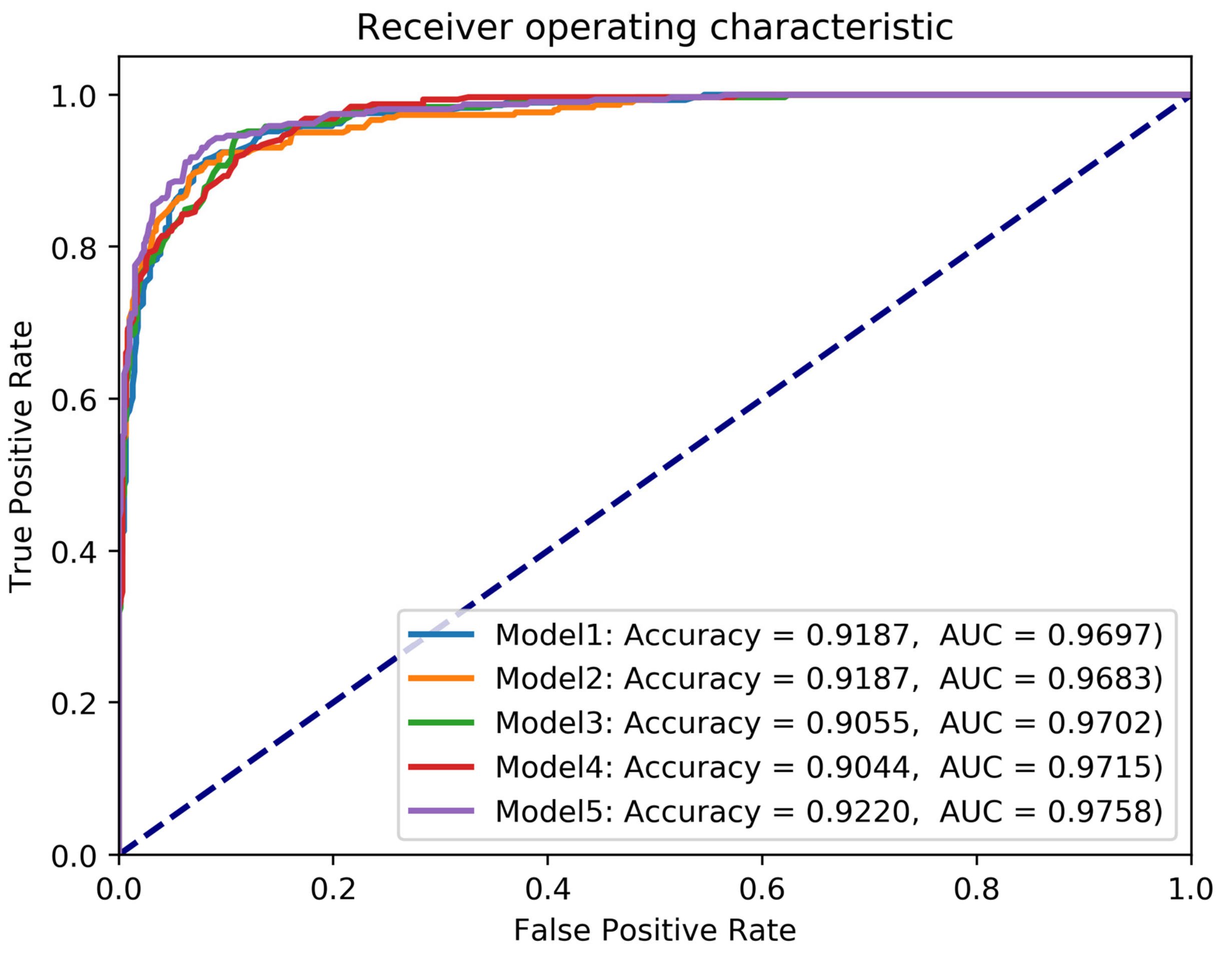

Figure S3. Receiver operating characteristic (ROC) curve of CNN models. The model was trained and tested 5 times with the training set. 80% of labeled image samples were used for training and the remaining 20% of samples were used to calculate the ROC curve.