Deep3DCCS

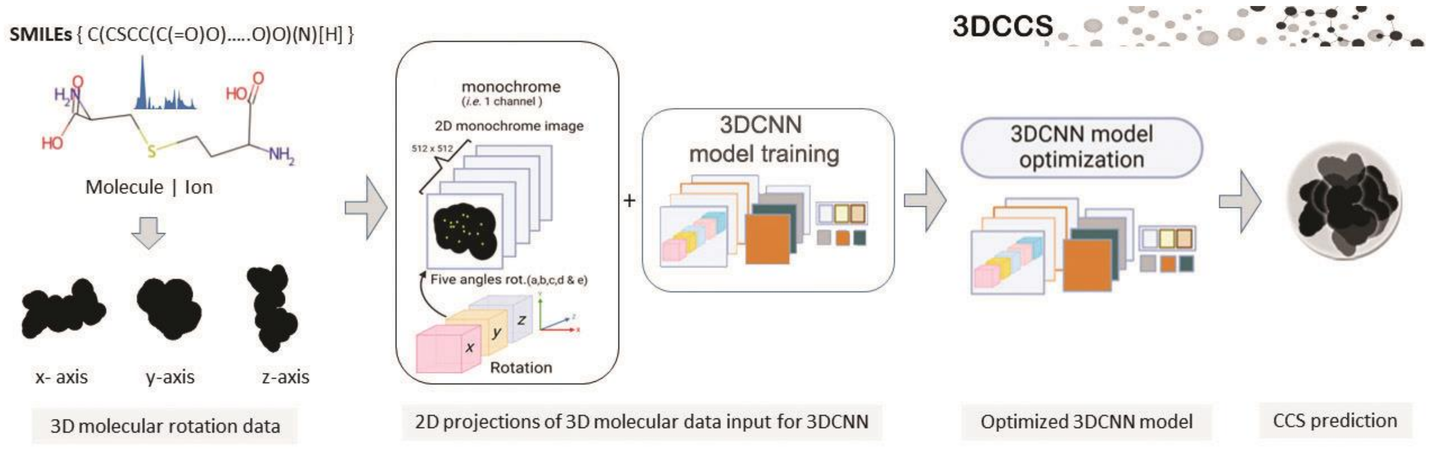
Deep Learning Approach for CCS Prediction from Multi-Angle Projections of Molecular Geometry

---

Overview

Deep3DCCS is a comprehensive deep learning framework for predicting **Collision Cross-Section (CCS)** values using three-dimensional molecular structure information derived from mass spectrometry data. The framework employs a specialized **3D Convolutional Neural Network (3DCNN)** operating on **multi-angle 2D projections** of molecular geometries. This design bridges molecular structure representation with ion mobility spectrometry, enabling accurate CCS prediction for applications in **metabolomics, lipidomics, and structural biology**.

Schematics of Deep3DCCS



Key Features

Molecular Structure Processing

- Converts **SMILES** into optimized **3D molecular structures**
- Utilizes **Gaussian View 6.0 / Gaussian 16**, **RDKit**, and computational chemistry methods
- Generates physically realistic geometries suitable for CCS modeling

Multi-View 2D Projections

- Generates rotational 2D projections around **x-, y-, and z-axes**
- Configurable rotational increments (1, 3, 5, or 10 angles per axis)
- Image resolutions from **8×8 to 192×192 pixels**
- Center-of-mass recentering and van der Waals surface visualization

3DCNN Regression Model

- Custom **3D CNN architecture** optimized for CCS regression
- Processes volumetric stacks of multi-angle projections
- Includes 3D convolution blocks, global average pooling, and dense regression layers

Automated Workflow (GUI)

- **PyQt5-based graphical interface**
- End-to-end pipeline from SMILES input to CCS prediction
- Modular workflow tabs:
    - Molecular Structure Optimization and 3D Structrue construction
    - Multi-view 2D Projection dataset Generation
    - Model Training /Parameter optimization/ Corss-validation
    - Inference

Comprehensive Evaluation

- Primary metric: **Relative Percentage Error (RPE)**
- Additional metrics: MAE, MAPE, Pearson r, R^2, standard error
- Automated visualizations:
    - Scatter plots
    - Residual density plots
    - QQ plots
    - Comparative bar charts

Installation

Prerequisites

- **Python ≥ 3.7** (tested with TensorFlow 2.10)
- Supported OS: **Windows 10/11**, **Ubuntu 20.04/22.04**
- _env (for windows system) will contain all virtual environment required for running Deep3DCCS. the portable version will be all _env embedded. Users are required to install CUDA v11.0 and cUDNN 8.5+ for proper running of GPU-version

Hardware Recommendations

- **GPU (recommended):** NVIDIA CUDA-compatible GPU (≥ 8 GB VRAM, 16 GB recommended for higher batch size)
- **System RAM:** ≥ 32 GB (can run smoothly in 16 GB system RAM)
- CPU-only execution is supported for small datasets trainining and any-size inferencing

Clone Repository

```
git clone https://github.com/vivekmathema/Deep3DCCS.git
cd Deep3DCCS
Deep3DCCS > run_deep3DCCS_gpu.bat (for windows)  |  Deep3DCCS > python3
deep3dcnn_main.py  (for linux)
```

Core Dependencies

```
pip install tensorflow==2.6.0
pip install keras==2.6.0
pip install scipy==1.4.1
pip install rdkit-pypi
pip install pyqt5 matplotlib numpy pandas scikit-learn opencv-python pillow
seaborn tqdm colorama termcolor
```

RDKit (Alternative via Conda)

```
conda install -c conda-forge rdkit
conda install -c conda-forge openbabel
```

Installation Verification

```
python -c "import tensorflow as tf; import rdkit; print('Installation
successful:', tf.__version__)"
```

A standalone **portable Windows (64-bit)** version will be released in the future upon acceptance of the manuscript.

Usage

GUI Interface

Launch the main application:

```
python deep3dcnn_main.py
```

CUDA requirement for the GPU version

For systems running on windows, the tensorflow may reqruire CUDA/cuDNN runtime library files to utilize Nvidia GPU which shoudl be installed seperately. After users have installed default python environment manually (or run portable python environment "./_env/" to using supplied run_deep3DCCS_gpu.bat) follow the instructions in links below to install official CUDA/cuDNN library. A system windows restart is recommended after installation.

CUDA/cuDNN Core installation (for windows 64 bit OS running with NVIDIA GPU)

```
CUDA Toolkit 11.8:

link: https://developer.nvidia.com/cuda-11-8-0-download-
archive?target_os=Windows)

To install cuDNN 8.00 Library for CUDA 11.0 Goto NVIDIA's cuDNN downlaod
page:

link: https://developer.nvidia.com/rdp/cudnn-archive

and select download cuDNN for 11.x (login may be required)

link will be dynamically provided for: CUDA Download cuDNN v8.9.7 (December
5th, 2023), for CUDA 11.x

MSVC C and C++ Build Tools redistributable version for GPU/CUDA/cuDNN support

Depending upon the support packages, some windows system might require
Microsoft Visual C++ Redistributable  (~18 MB instalelr for x64 system)

link: https://learn.microsoft.com/en-us/cpp/windows/latest-supported-vc-
redist?view=msvc-170#latest-supported-redistributable-version
```

Portable GPU version release for Windows 64bit systems

A portable, GPU-enabled, and self-contained version of Deep3DCCS is available for download as a 7z compressed file.

```
link:
 https://drive.google.com/file/d/1cBSITajg9lfJ_R6ThiwLUPMmGccfzuT4/
view?usp=sharing
```

Users need to download the 7zip compressed file, decompress using (7zip or winRAR for win64) and run run_deep3DCCS_gpu.bat. All example datasets/models to train/validate are included

General Instruction for basic operation

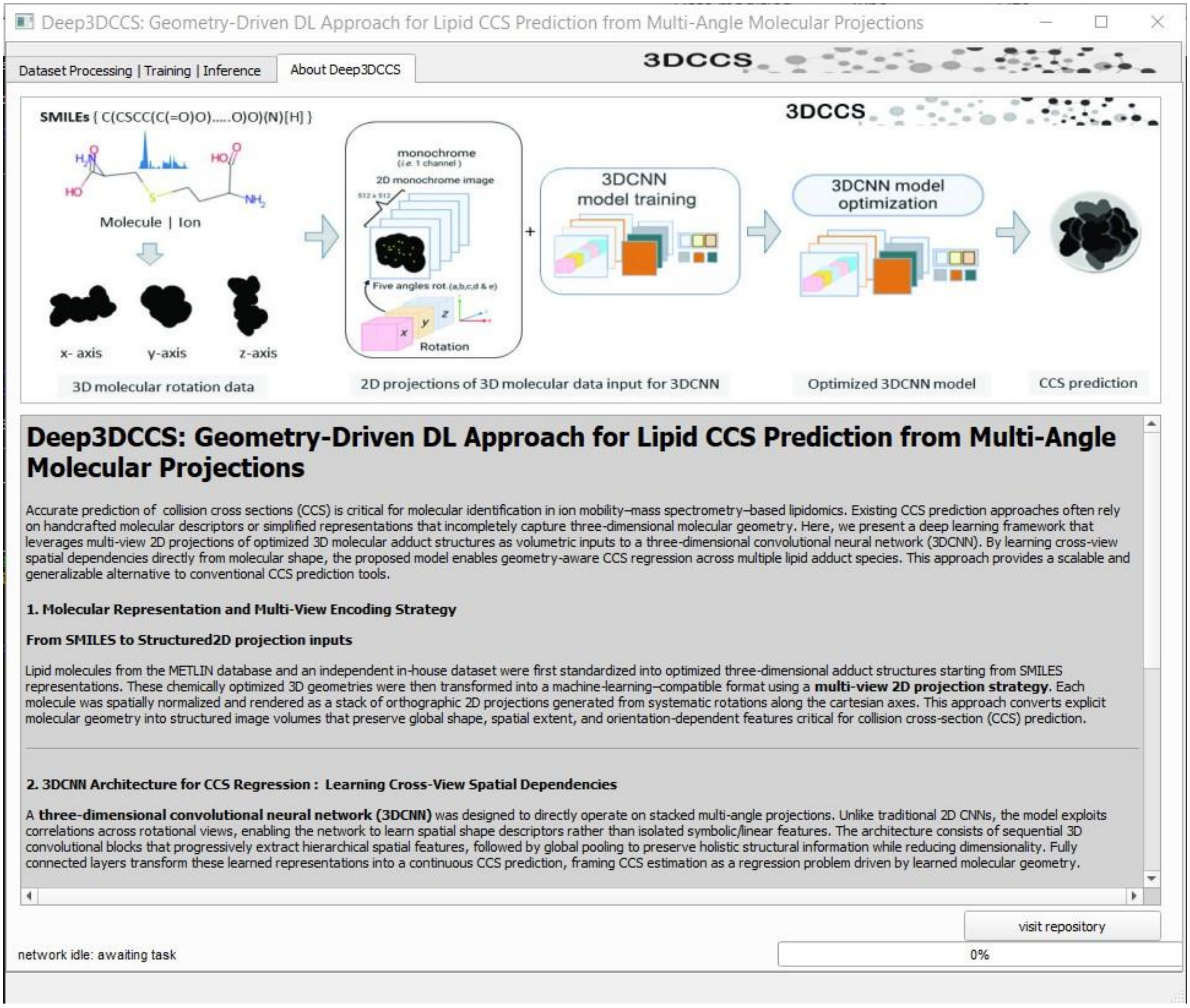
The Deep3DCCS GUI enables users to visually select and oeprate three key modules (seperated in three major tabs with in the main gui windows):

- Module 1: SMILEs Data pre-processing & 2D-projection database generation
- Module 2: 3DCCS model training
- Module 3: Inference using trained model
- All three steps can be summarized below.

Graphical user friendly inferface of Deep3DCCS (beta version)

The Deep3DCCS is a complete GUI-feature enagled software toolkit that can be run locally for both training and inference with full customization. Hovering mouse on most of the buttons in GUI interface will provide users short description of its corrsponding functions.

Welcome screen for Deep3DCCS gui inferface This is a basic welcome screen along with general information about the software. It also links to its github repository.


Deep3DCCS: Geometry-Driven DL Approach for Lipid CCS Prediction from Multi-Angle Molecular Projections
Dataset Processing | Training | Inference
About Deep3DCCS
3DCCS
SMILEs { C(CSCC(C(=O)O).....O)O)(N)[H] }
Molecule | Ion
x- axis
y-axis
z-axis
3D molecular rotation data
monochrome (i.e. 1 channel)
2D monochrome image
Five angles rot.(a,b,c,d & e)
Rotation
2D projections of 3D molecular data input for 3DCNN
3DCNN model training
3DCNN model optimization
Optimized 3DCNN model
CCS prediction
visit repository
network idle: awaiting task
0%

Deep3DCCS: Geometry-Driven DL Approach for Lipid CCS Prediction from Multi-Angle Molecular Projections

Accurate prediction of collision cross sections (CCS) is critical for molecular identification in ion mobility–mass spectrometry–based lipidomics. Existing CCS prediction approaches often rely on handcrafted molecular descriptors or simplified representations that incompletely capture three-dimensional molecular geometry. Here, we present a deep learning framework that leverages multi-view 2D projections of optimized 3D molecular adduct structures as volumetric inputs to a three-dimensional convolutional neural network (3DCNN). By learning cross-view spatial dependencies directly from molecular shape, the proposed model enables geometry-aware CCS regression across multiple lipid adduct species. This approach provides a scalable and generalizable alternative to conventional CCS prediction tools.

1. Molecular Representation and Multi-View Encoding Strategy

From SMILES to Structured2D projection inputs

Lipid molecules from the METLIN database and an independent in-house dataset were first standardized into optimized three-dimensional adduct structures starting from SMILES representations. These chemically optimized 3D geometries were then transformed into a machine-learning–compatible format using a **multi-view 2D projection strategy**. Each molecule was spatially normalized and rendered as a stack of orthographic 2D projections generated from systematic rotations along the cartesian axes. This approach converts explicit molecular geometry into structured image volumes that preserve global shape, spatial extent, and orientation-dependent features critical for collision cross-section (CCS) prediction.

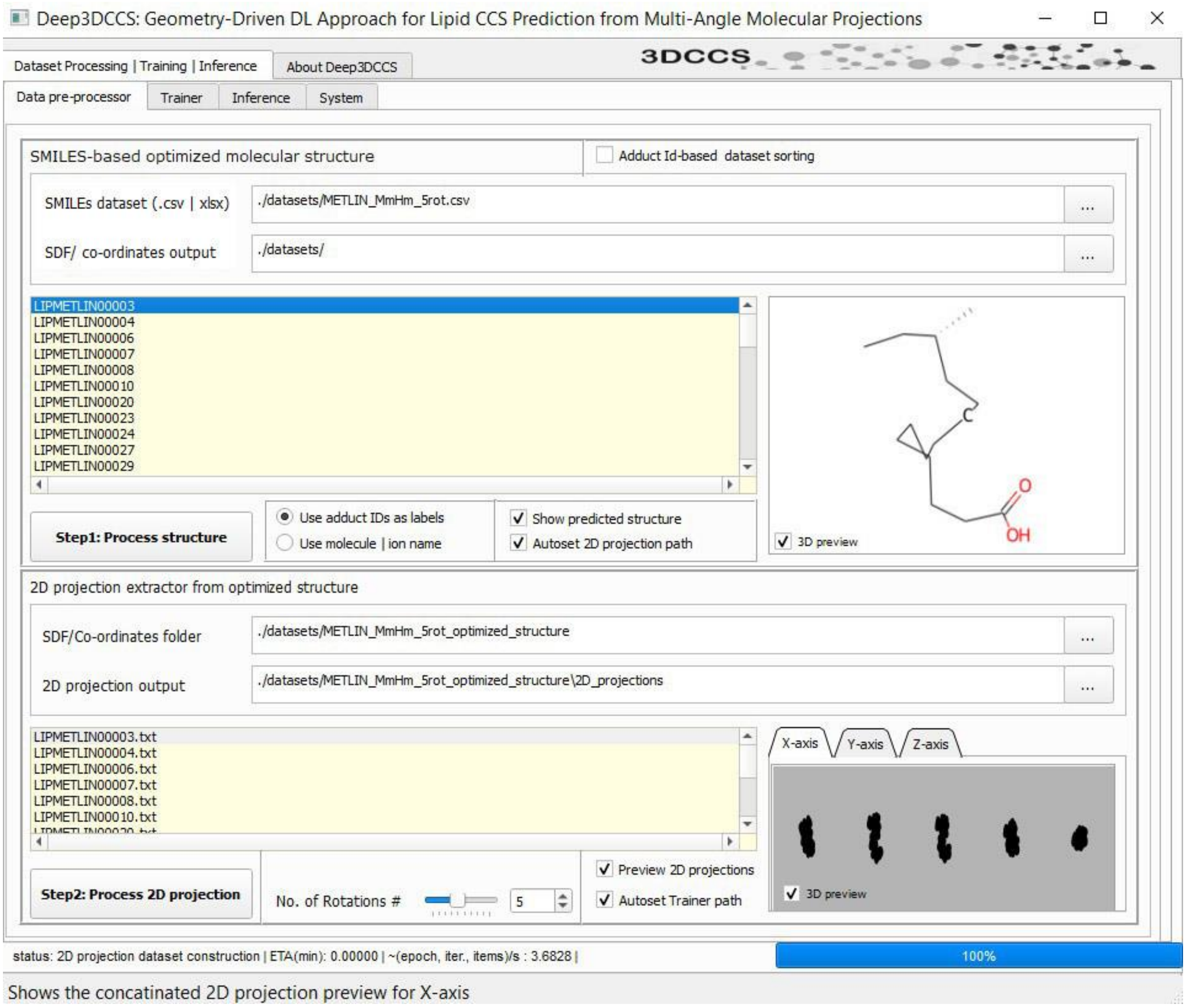
2. 3DCNN Architecture for CCS Regression : Learning Cross-View Spatial Dependencies

A **three-dimensional convolutional neural network (3DCNN)** was designed to directly operate on stacked multi-angle projections. Unlike traditional 2D CNNs, the model exploits correlations across rotational views, enabling the network to learn spatial shape descriptors rather than isolated symbolic/linear features. The architecture consists of sequential 3D convolutional blocks that progressively extract hierarchical spatial features, followed by global pooling to preserve holistic structural information while reducing dimensionality. Fully connected layers transform these learned representations into a continuous CCS prediction, framing CCS estimation as a regression problem driven by learned molecular geometry.

The Deep3DCCS data pre-processing module Module 1: Model pre-processing is a straight foward step in GUI window under the Tab "Data pre-processor". Users are required to upload the SMILEs datasets (.csv or xlsx files- Examples of formatted datasets are provided in "./datasets/" folder).The experimental CCS values is mandatory only for training SMILEs dataset pre-processing.

IMPORTANT NOTE: Molecule names may contain Latin or Latin-like encoded characters that can cause issues during preprocessing of .csv or .xlsx files. Therefore, the SMILES input file must always be saved using "latin-1" encoding. In Python, specify the encoding explicitly when writing the file (e.g., with open("SMILEs_datafile.csv", "w", encoding="latin-1") as f:) if you are creating a custom training or inference SMILES dataset.

- This is followed by choosing approperate options and pressing: "Step1:Process structure" button for 3D optimzied structrue datafiles generation from SMILEs.
- Subsequently, the next step is to produce 2DCCS compatible 2D-projection training/inference-compatible dataset by pressing "Step2: Process 2D projection" button.
- In each case, users must set/select correct number of rotations accordingly.
- Basic preview of the molecule structrue and 2D-projections can be seen in the preview window for each dataset processing step.



Deep3DCCS: Geometry-Driven DL Approach for Lipid CCS Prediction from Multi-Angle Molecular Projections

Dataset Processing | Training | Inference | About Deep3DCCS

Data pre-processor | Trainer | Inference | System

SMILES-based optimized molecular structure

☐ Adduct Id-based dataset sorting

SMILEs dataset (.csv | xlsx): ./datasets/METLIN_MmHm_5rot.csv

SDF/ co-ordinates output: ./datasets/

LIPMETLIN00003
LIPMETLIN00004
LIPMETLIN00006
LIPMETLIN00007
LIPMETLIN00008
LIPMETLIN00010
LIPMETLIN00020
LIPMETLIN00023
LIPMETLIN00024
LIPMETLIN00027
LIPMETLIN00029

Step1: Process structure

(●) Use adduct IDs as labels
( ) Use molecule | ion name

☑ Show predicted structure
☑ Autoset 2D projection path

☑ 3D preview

2D projection extractor from optimized structure

SDF/Co-ordinates folder: ./datasets/METLIN_MmHm_5rot_optimized_structure

2D projection output: ./datasets/METLIN_MmHm_5rot_optimized_structure\2D_projections

LIPMETLIN00003.txt
LIPMETLIN00004.txt
LIPMETLIN00006.txt
LIPMETLIN00007.txt
LIPMETLIN00008.txt
LIPMETLIN00010.txt

X-axis | Y-axis | Z-axis

☑ 3D preview

Step2: Process 2D projection

No. of Rotations #: 5

☑ Preview 2D projections
☑ Autoset Trainer path

status: 2D projection dataset construction | ETA(min): 0.00000 | ~(epoch, iter., items)/s : 3.6828 |

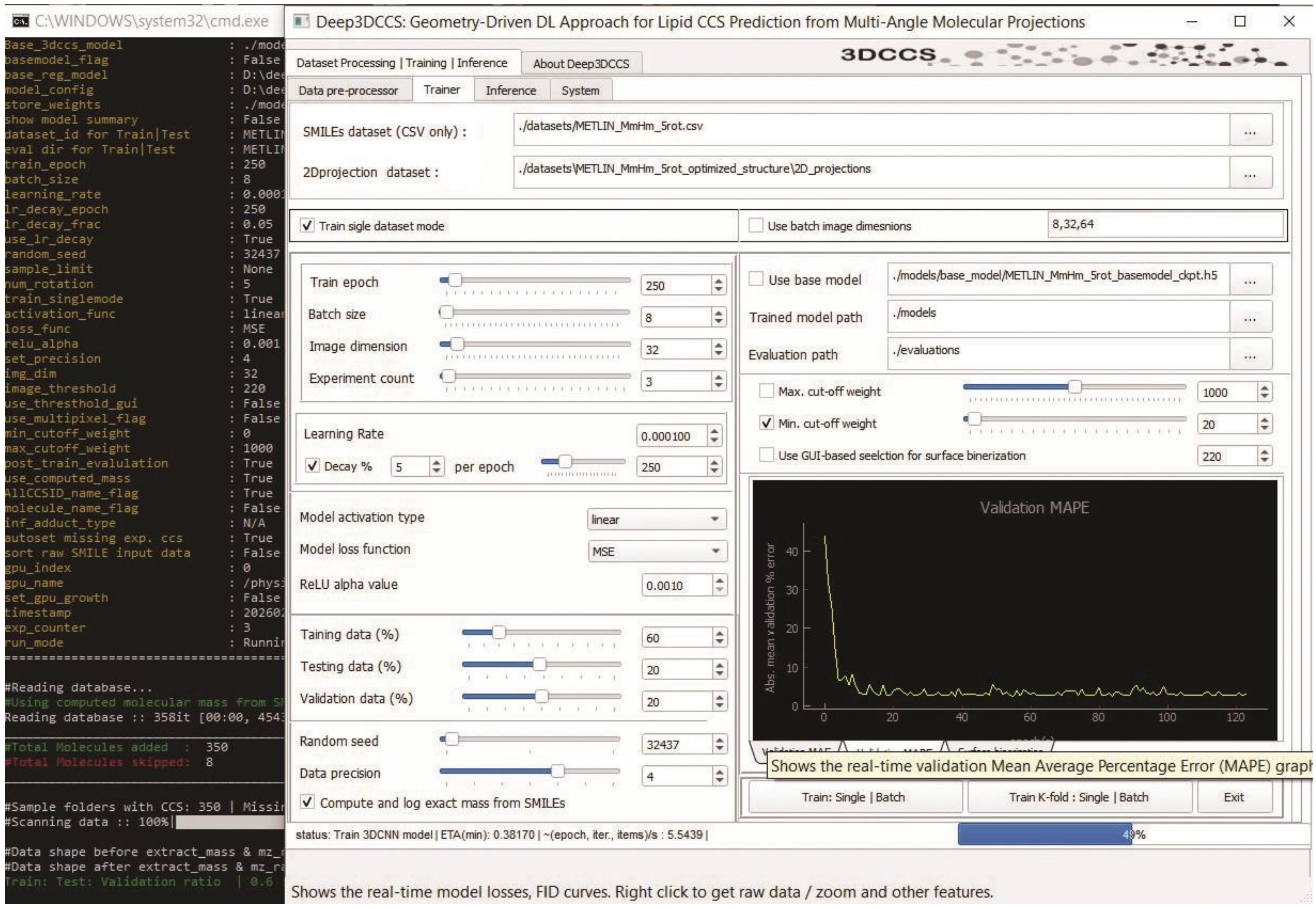
100%

Shows the concitinated 2D projection preview for X-axis

The Deep3DCCS data Model training module

- Module 2: This module can be accessed by selecting the "Trainer" tab in the main window.
- User will have to provide the SMILEs dataset (The same SMILEs dataset file which they used to pre-process 2D-projection dataset in "Data pre-processor" module. Only .CSV file is accepted at this point)

- This is followed by selecting 2D projection dataset folder path associated with the SMILEs dataset. The software by default sets path for the 2D projection datasets but users can change arrording to their experimental setup.
- Users can select storage path for the models and evaluations results.
- Users must setup the correct number of rotations and pixel resoultion for each training dataset that macthes with the training dataset.
- The users can train single or multiple dataset for different pixel resoultions and rotations based on their requirments.
- The option to train single dataset can be selected by tickmarking "Train single dataset mode" checkbox.
- The option to train multiple resoultion can be applied by tickmarking "Use batch image dimensions" checkbox.
- The batch or single mode training itself can be done for random experimentation mode by pressing [Train:Single | Batch] or K-fold [Train K-fold: single | Batch] buttons. (For further details, see under title "Optimization of Training Models")
-


C:\WINDOWS\system32\cmd.exe
Deep3DCCS: Geometry-Driven DL Approach for Lipid CCS Prediction from Multi-Angle Molecular Projections
Dataset Processing | Training | Inference
About Deep3DCCS
3DCCS
Data pre-processor
Trainer
Inference
System
Validation MAPE
Abs. mean validation % error
epoch(s)
0
10
20
30
40
0
20
40
60
80
100
120
Validation MAE
Validation MAPE
Surface binarization
Train: Single | Batch
Train K-fold : Single | Batch
Exit

```
Base_3dccs_model               : ./mode
basemodel_flag                 : False
base_reg_model                 : D:\dee
model_config                   : D:\dee
store_weights                  : ./mode
show model summary             : False
dataset_id for Train|Test      : METLIN
eval dir for Train|Test        : METLIN
train_epoch                    : 250
batch_size                     : 8
learning_rate                  : 0.0001
lr_decay_epoch                 : 250
lr_decay_frac                  : 0.05
use_lr_decay                   : True
random_seed                    : 32437
sample_limit                   : None
num_rotation                   : 5
train_singlemode               : True
activation_func                : linear
loss_func                      : MSE
relu_alpha                     : 0.001
set_precision                  : 4
img_dim                        : 32
image_threshold                : 220
use_thresthold_gui             : False
use_multipixel_flag            : False
min_cutoff_weight              : 0
max_cutoff_weight              : 1000
post_train_evalulation         : True
use_computed_mass              : True
AllCCSID_name_flag             : True
molecule_name_flag             : False
inf_adduct_type                : N/A
autoset missing exp. ccs       : True
sort raw SMILE input data      : False
gpu_index                      : 0
gpu_name                       : /physi
set_gpu_growth                 : False
timestamp                      : 20260
exp_counter                    : 3
run_mode                       : Runnin
========================================

#Reading database...
#Using computed molecular mass from SM
Reading database :: 358it [00:00, 4543

#Total Molecules added  :  350
#Total Molecules skipped:  8

#Sample folders with CCS: 350 | Missin
#Scanning data :: 100%|

#Data shape before extract_mass & mz_r
#Data shape after extract_mass & mz_ra
Train: Test: Validation ratio  | 0.6
```

SMILEs dataset (CSV only) : ./datasets/METLIN_MmHm_5rot.csv

2Dprojection dataset : ./datasets\METLIN_MmHm_5rot_optimized_structure\2D_projections

[✓] Train sigle dataset mode

[ ] Use batch image dimesnions: 8,32,64

Train epoch: 250
Batch size: 8
Image dimension: 32
Experiment count: 3

Learning Rate: 0.000100
[✓] Decay % 5 per epoch 250

Model activation type: linear
Model loss function: MSE
ReLU alpha value: 0.0010

Taining data (%): 60
Testing data (%): 20
Validation data (%): 20

Random seed: 32437
Data precision: 4
[✓] Compute and log exact mass from SMILEs

[ ] Use base model: ./models/base_model/METLIN_MmHm_5rot_basemodel_ckpt.h5
Trained model path: ./models
Evaluation path: ./evaluations

[ ] Max. cut-off weight: 1000
[✓] Min. cut-off weight: 20
[ ] Use GUI-based seelction for surface binerization: 220

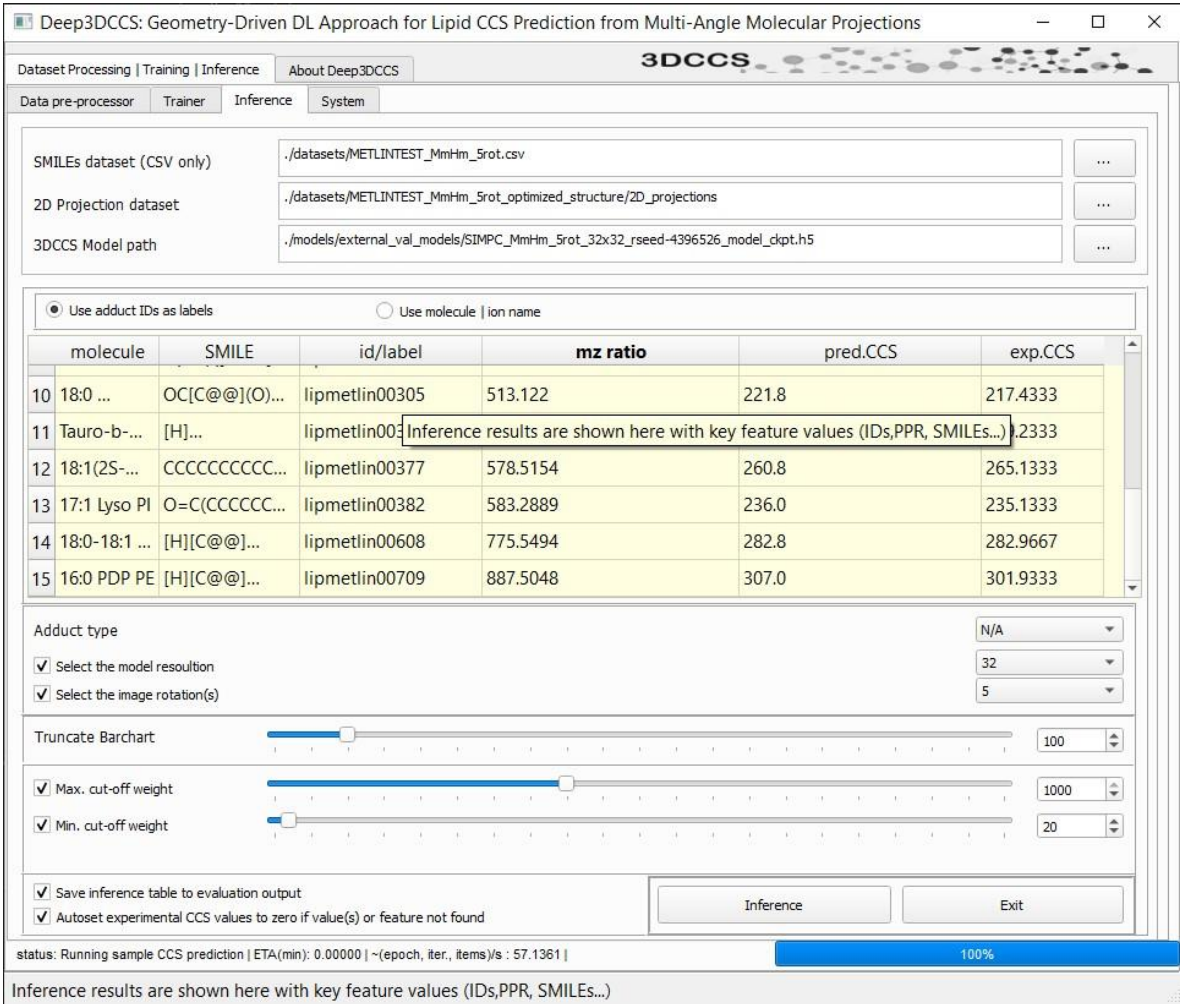
Shows the real-time validation Mean Average Percentage Error (MAPE) graph

status: Train 3DCNN model | ETA(min): 0.38170 | ~(epoch, iter., items)/s : 5.5439 | 49%

Shows the real-time model losses, FID curves. Right click to get raw data / zoom and other features.

The Deep3DCCS data Inference/ccs prediction using trained module

- Module 3: This is the final inference/prediction module.
- Users will have to provide the optimized/trained model for particular rotation and adduct user "3DCCS model path".
- User will have to set the 2D-projection dataset path asscoiated with the SMILEs input datafile under "2D projection dataset" path. The software by default sets path for the 2D projection datasets but users can change arrording to their experimental setup.
- The experimental CCS valus are not mandatory for unknown compounds but is required for conducting external validation.
- Users can input the sample data provided along with the code or their own inference dataset for model validation.
- All SMILEs samples for inference/external validation should be processed in same way as for training dataset
- Although any pixel resoultion is accepted during inferencing for a model trained with particular rotation, it is recommended to use same pixel rotation that the model was originally trained on.
- Once the path/parameters setup is ready, press "Inference" button to compute the CCS values.
- Finally, inference results along with SMILESs , molecule IDs, name, predicted ccs, experimental ccs (if available), RPE will be displayed in the result table box and stored in the "./evaluation" folder along with other by default. -If user has provided SMILEs data with experimental ccs for external validation, the several comapartive charts will be generated in the evaluation folder along with ccs prediction results.


Deep3DCCS: Geometry-Driven DL Approach for Lipid CCS Prediction from Multi-Angle Molecular Projections
Dataset Processing | Training | Inference
About Deep3DCCS
3DCCS
Data pre-processor
Trainer
Inference
System
Inference
Exit

SMILEs dataset (CSV only): ./datasets/METLINTEST_MmHm_5rot.csv

2D Projection dataset: ./datasets/METLINTEST_MmHm_5rot_optimized_structure/2D_projections

3DCCS Model path: ./models/external_val_models/SIMPC_MmHm_5rot_32x32_rseed-4396526_model_ckpt.h5

[✓ Use adduct IDs as labels] Use molecule | ion name

	molecule	SMILE	id/label	mz ratio	pred.CCS	exp.CCS
10	18:0 ...	OC[C@@](O)...	lipmetlin00305	513.122	221.8	217.4333
11	Tauro-b-...	[H]...	lipmetlin003 Inference results are shown here with key feature values (IDs,PPR, SMILEs...)			.2333
12	18:1(2S-...	CCCCCCCCCC...	lipmetlin00377	578.5154	260.8	265.1333
13	17:1 Lyso PI	O=C(CCCCCC...	lipmetlin00382	583.2889	236.0	235.1333
14	18:0-18:1 ...	[H][C@@]...	lipmetlin00608	775.5494	282.8	282.9667
15	16:0 PDP PE	[H][C@@]...	lipmetlin00709	887.5048	307.0	301.9333

Adduct type: N/A

✓ Select the model resoultion: 32

✓ Select the image rotation(s): 5

Truncate Barchart: 100

✓ Max. cut-off weight: 1000

✓ Min. cut-off weight: 20

✓ Save inference table to evaluation output

✓ Autoset experimental CCS values to zero if value(s) or feature not found

status: Running sample CCS prediction | ETA(min): 0.00000 | ~(epoch, iter., items)/s : 57.1361 | 100%

Inference results are shown here with key feature values (IDs,PPR, SMILEs...)

The GUI provides:

- **Molecular Optimization:** SMILES → optimized 3D structures
- **2D Projection Generation:** Multi-angle image stacks
- **Model Training:** Architecture and hyperparameter configuration
- **Inference:** CCS prediction on unseen molecules

Command-Line Tools

- **TO DO: in future version** , currently to make the users familiar with all steps and avoid any confusion in parameters setting, only GUI version is available.

Experimental Workflow

Sample pre-processing (for both Training and inference)

Pre-processing Step #1: Select the CSV or XLS files contanining sample IDs, SMILES, Adduct information to create standard 3D image dataset **Pre-processing Step #2:** Select the subsequent 3D image dataset to creat final 2D training dataset angle projection dataset.

- This 2D projection dataset along with the SMILEs dataset file (CSV only) file is used for model training and inference/external validation.

Optimization of Training Models

** There is no golden rule to idetify best model training/optimiation parameters in deep learnining/machine learnining. We suggest conducting two step experimentation to: (i) First identlty a potentially suitable parameters using random sampling for model training (ii) Using 4-fold cross-validation with the observed most accurate parameters configuration for model optimization. Finally use thwe optimized model for inference. Additionaly, the optimized model parameteres can be used to train on entire train/test dataset and futher validated on external validation dataset to access the real-world generalization of the optimized model.

Experiment 1: Resolution & Rotation Optimization

- Resolution range: **8×8 to 192×192**
- Rotations per axis: **1–10**
- 40 replicates per configuration
- Training: 250 epochs, batch size 8, Adam (lr=1e-4)

Experiment 2: 4-Fold Cross-Validation

- Optimal parameters from Experiment 1
- Stratified 4-fold split by adduct
- 15 replicates with different random seeds

NOTE: Although a fixed random seed is used to ensure reproducibility, the inherent complexity of the 3D CNN architecture can lead to minor variations between training runs. Nevertheless, the overall performance trend remains consistent, and differences in inference results are not significant.

Rotation-specific datasets are generated by changing the "Xrot" label in the filename and selecting the same rotation value (e.g., 5rot) in the software. The RAW CSV/XLS files remain identical; only the software's processing differs.

A single RAW dataset per adduct can generate all rotation variants. Separate datasets are provided here only for demonstration. Based on the selected 2D projection configuration, the software automatically generates the corresponding 3D dataset and 2D projections.

Optional: An identical Microsoft XLS file is included for better annotation and customization of the input data.

Model Architecture

- Input: *(N_rotations* × *H* × *W* × *Channels)*
- 4 × 3D convolution blocks (32 → 64 → 128 → 256 filters)
- Batch normalization + anisotropic pooling
- GlobalAveragePooling3D
- Dense layers: 128 → 64 (LeakyReLU, Dropout 0.3)
- Output: Single linear neuron (CCS regression)
- Optimizer: **Adam (lr=0.0001)**

Repository Structure

```
Deep3DCCS/
├── deep3dcnn_main.py.py
├── _core.py
├── utility_modules.py
├── helper_tools.py
├── ui_interface.ui
├── run_deep3DCCS_GPU.bat
├── run_deep3DCCS_CPU.bat
├── datasets/
├── models/
├── evaluations/
├── configs/
├── assets/
├── requirements-cpu.txt
└── requirements-gpu.txt
```

NOTE: For inference or prediction on molecules with unknown experimental CCS values, the option "Autoset experimental CCS values to zero if value(s) or feature not found" must be enabled. This forces all experimental CCS values to zero. This option is applicable to CSV SMILES input datafiles that lack the exp_ccs column entirely or contain missing values (e.g., N/A or NULL)for some molecules. In both cases, the scoring metrics reported below are not meaningful, as no comparison with known experimental CCS values is possible.

Evaluation Metrics

- **Relative Percentage Error (RPE)**
- Mean Absolute Percentage Error (MAPE)
- Pearson Correlation (r)
- Coefficient of Determination (R^2)

Interpretation Guide:

- RPE < 3%: Excellent
- 3–5%: Good
- 5–10%: Moderate
    - 10%: Requires refinement or the model is not optimzied to deal with such structure due to limited dataset and/or model architecture.

Benchmarking

Deep3DCCS is benchmarked against:

- **CCSBASE**
- **ALLCCS**

Evaluation includes RPE distribution, paired statistical tests, computational efficiency, and external dataset validation.

Output Files

- **3D Structures:** SDF + Cartesian coordinate files
- **2D Projections:** PNG stacks with JSON metadata
- **Models:** HDF5 (.h5) + checkpoints
- **Results:** CSV, JSON summaries, publication-quality plots

---

Citation

If you use Deep3DCCS, please cite the paper :

```
@article{deep3dccs2024,
  title={Deep3DCCS: Deep Learning Approach for CCS Prediction from Multi-
Angle Projections of Molecular Geometry},
  author={Siriraj Metabolomics \& Phenomics Center},
  journal={Journal of Cheminformatics},
  year={2026},
  publisher={Springer}
}
```

License

Contact

Siriraj Metabolomics & Phenomics Center (SiMPC)
Faculty of Medicine Siriraj Hospital
Mahidol University, Bangkok 10700, Thailand

Funding & Acknowledgments

Supported by:

- National Higher Education Science Research and Innovation Policy Council (NXPO)
- PMU-B (Program Management Unit for Human Resource & Institutional Development)

Computational resources provided by the **SiMPC High-Performance Computing Facility**.

Note: Deep3DCCS is an active research project. Please check the GitHub repository for updates, report issues, and refer to documentation for advanced usage.