Deep3DCCS

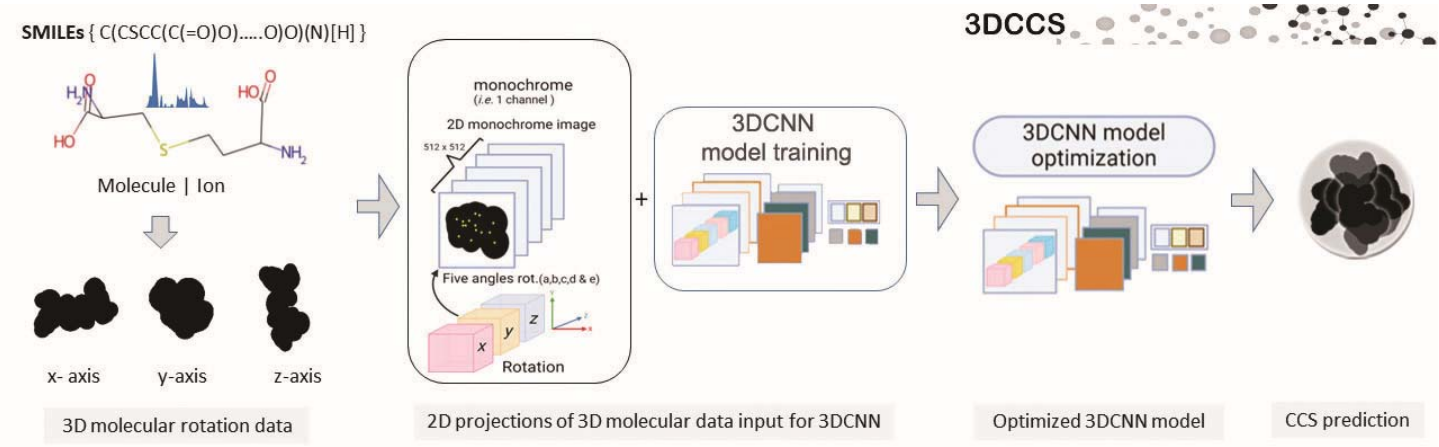
Deep Learning Approach for CCS Prediction from Multi-Angle Projections of Molecular Geometry

---

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

[Recommended] 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.

DOWNLOAD Deep3DCCS Windows 64bit Portable

https://drive.google.com/file/d/1MNw610K_th_XiDDnGqTO2lYfzHZs05V7/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

Usage

GUI Interface

Launch the main application:

```
python deep3dcnn_main.py
```

General Instruction for basic operation

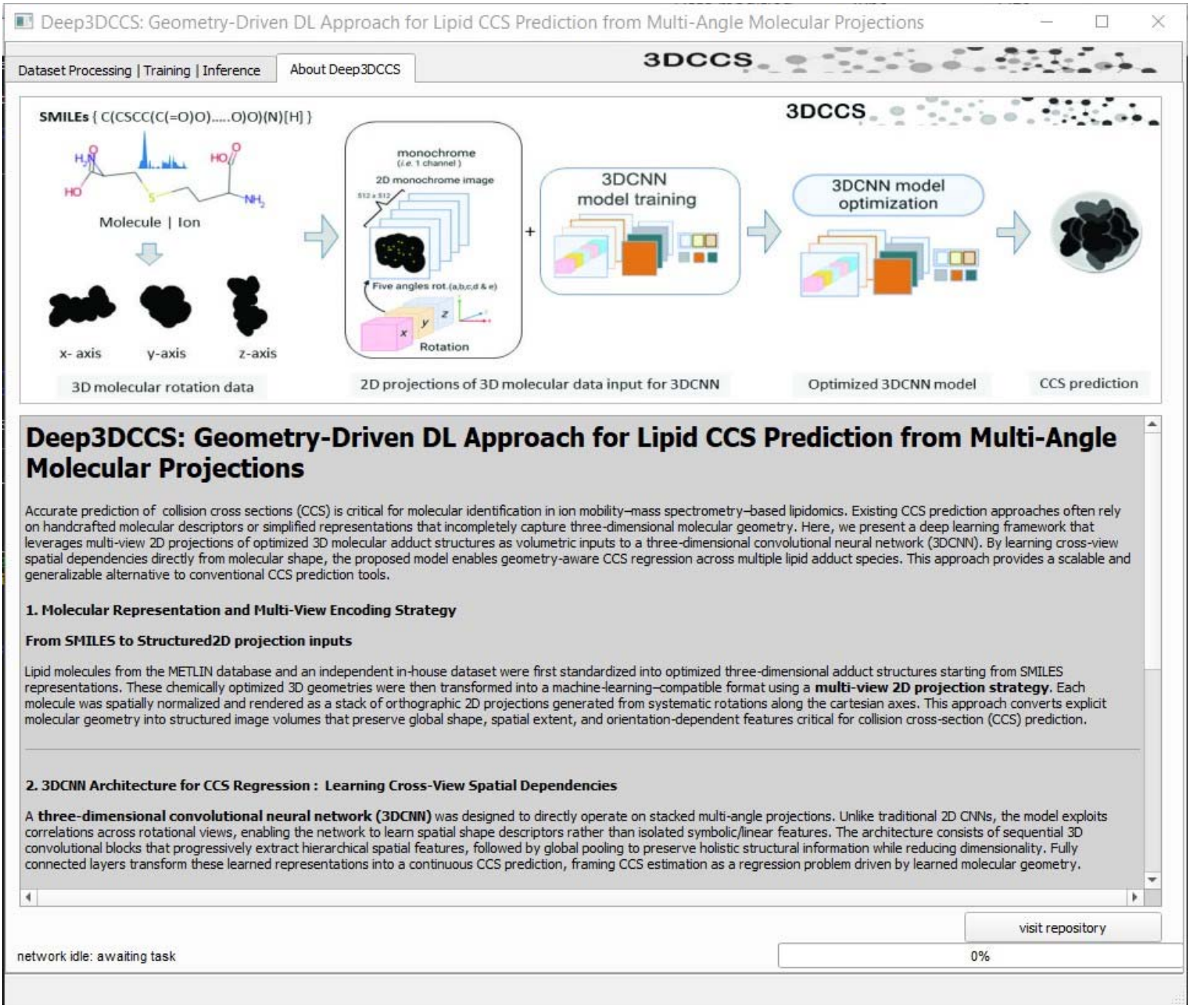
The Deep3DCCS GUI enables users to visually select and oeprate three key modules (seperated in three major tabs with in the main gui windows):

- 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)

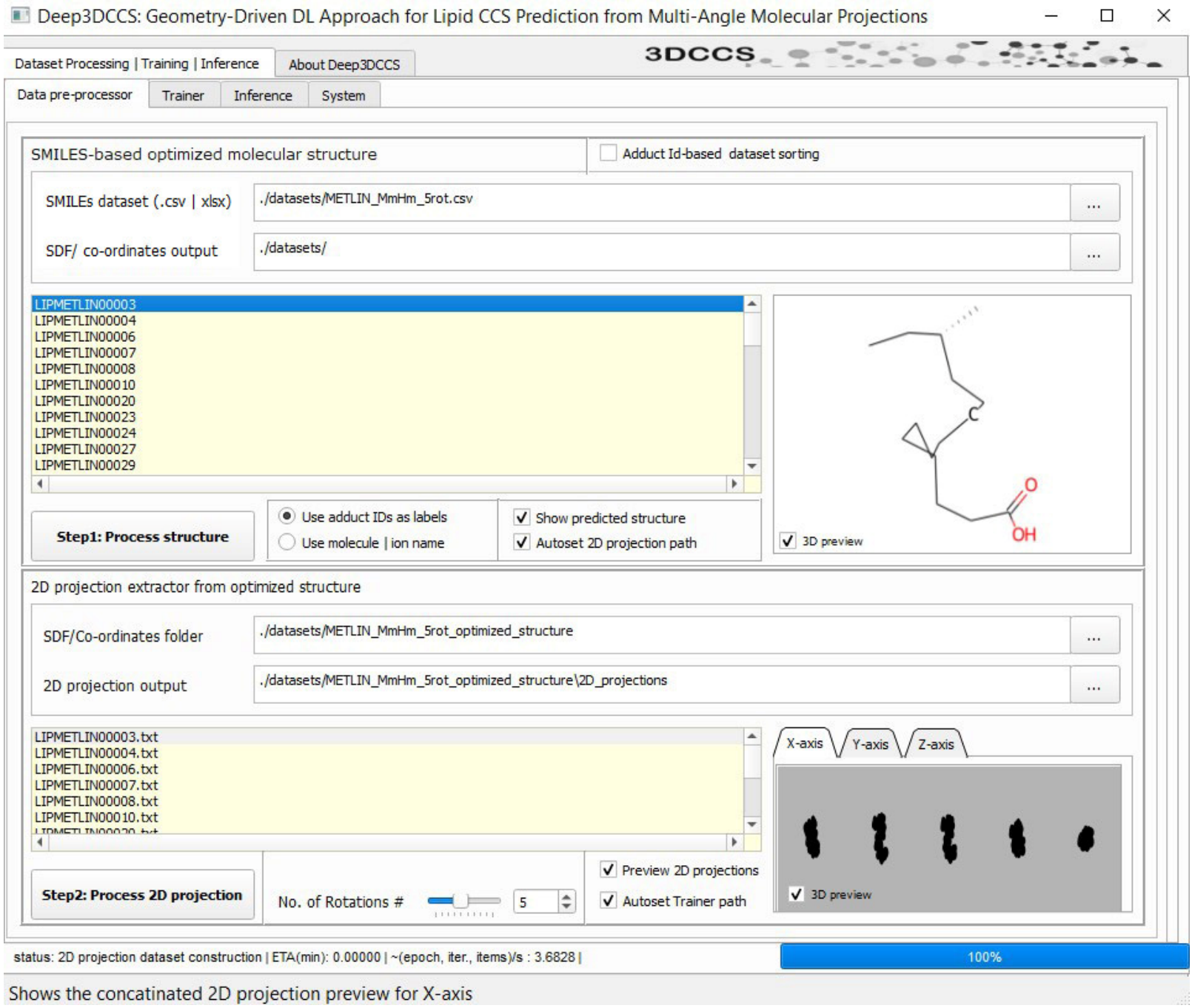
The Deep3DCCS is a complete GUI-feature enagled software toolkit that can be run locally for both training and inference with full customization. Hovering mouse on most of the buttons in GUI interface will provide users short description of its corrsponding functions.

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.



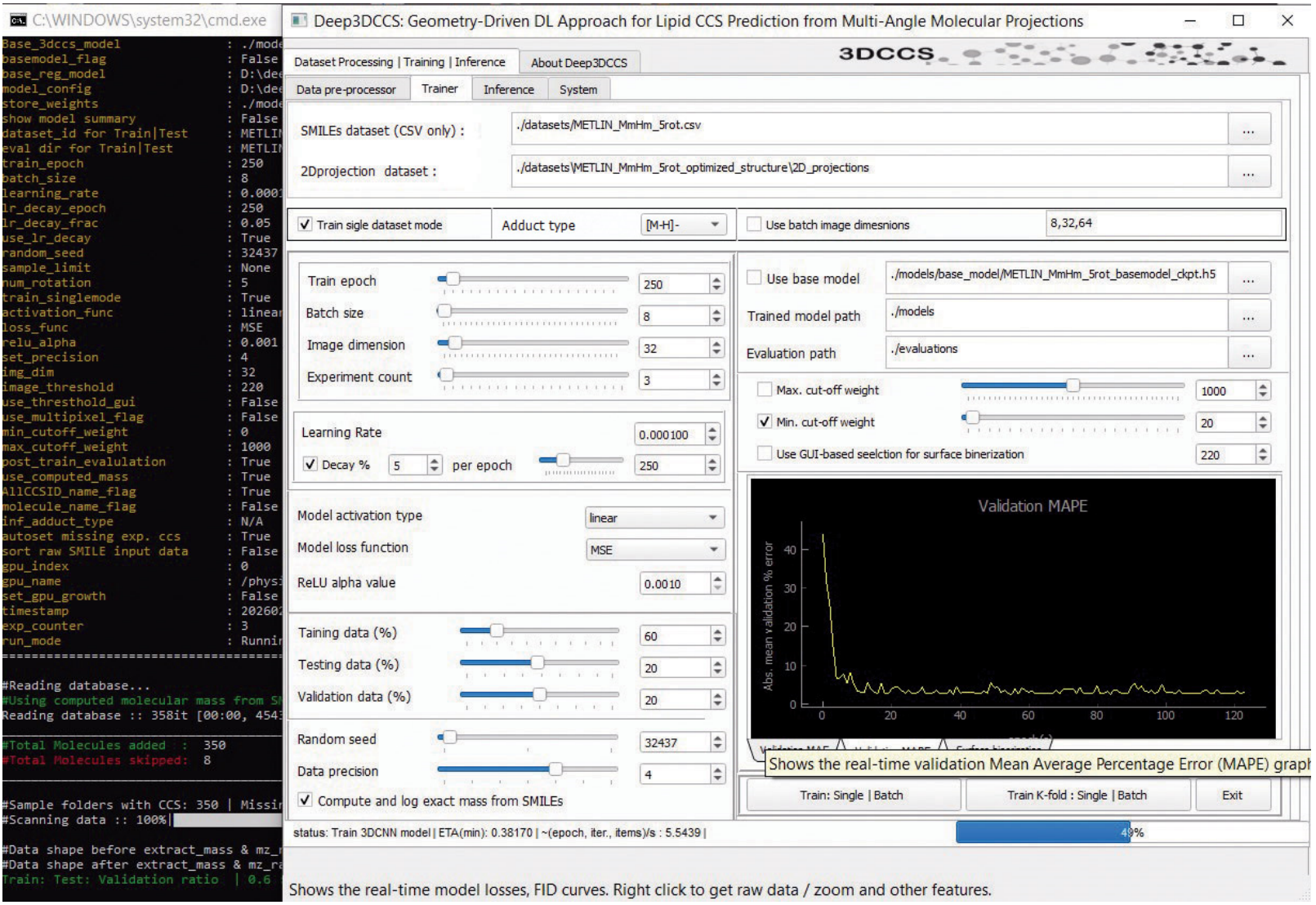
The Deep3DCCS data pre-processing module Module 1: Model pre-processing is a straight foward step in GUI window under the Tab "Data pre-processor". Users are required to upload the SMILEs datasets (.csv or xlsx files- Examples of formatted datasets are provided in "./datasets/" folder).The experimental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



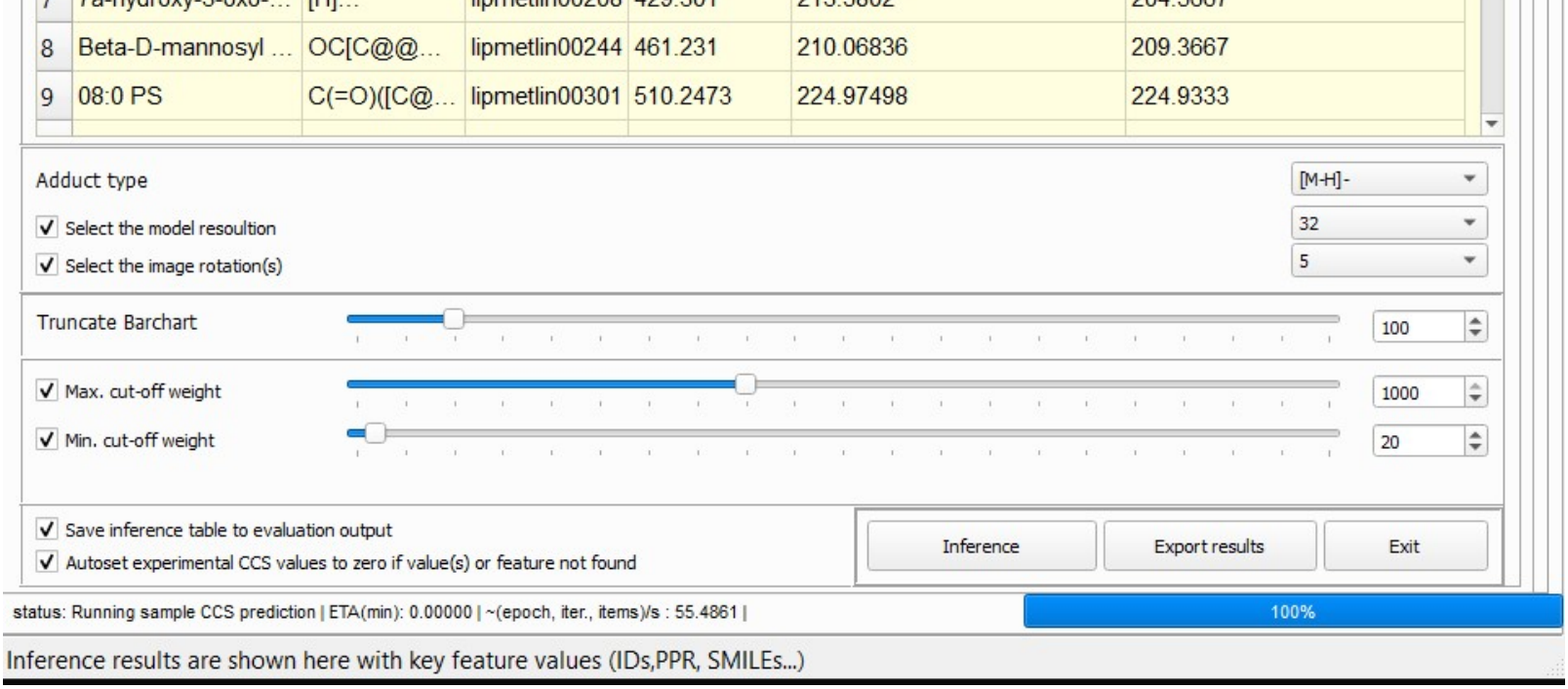
- This is followed by choosing approperate options and pressing: "Step1:Process structure" button for 3D optimzied structrue datafiles generation from SMILEs.
- Subsequently, the next step is to produce 2DCCS compatible 2D-projection training/inference-compatible dataset by pressing "Step2: Process 2D 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.

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 resolution 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")
-

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.

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.

Valid SMILEs Input datafile teample (csv file)

A valid templat csv file suitable for both Train and inference/prediction is providded: ./datasets/template.csv Note that the column 'exp_ccs' for experimental ccs values are not required for the prediction Note:mz_ratio data is (optional). However, if the user supplies mz_ratio values, it will be included as a part of information metadata. We highly recommend keeping the mz-ratio data for proper documentation purpose in mass spectrometry

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

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

CUDA Toolkit 11.8/12.x:

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

To install cuDNN 8.00+ Library for CUDA 11.x/12.x Goto NVIDIA's cuDNN downlaod page:

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

and select download cuDNN for 11.x (or 12.x if installing v12.x) (login may be required)

Download 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)

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

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.

Prerequisites

- **Python ≥ 3.7** (tested with TensorFlow-GPU 2.6.0)
- Supported & recommended OS: **Windows 10/11**, **Users can try on Linux Ubuntu if they can setup similar python environment with Tensorflow and CUDA support**
- _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-v12.6 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)

Clone Repository

```
git clone https://github.com/vivekmathema/Deep3DCCS.git
cd Deep3DCCS
Deep3DCCS > run_deep3DCCS_gpu.bat (for windows 64bit GPU/portable version)  |  (linux users are advised to install python environment and then use Deep3DCCS > python3 deep3dcnn_main.py )
```

Core Dependencies

```
pip install tensorflow-gpu==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
```

The requirments can be installed uisng

```
pip install -r requirements-gpu.txt
```

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

This project is licensed under the **MIT License**. See `LICENSE` for details.

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**.