TMOIA-GUI User Manual

Transformer-based multiomics integrative analysis

March 30, 2023

Version 1.0
Description Graphical user interface (GUI) for TMOIA.
License GPL-3
URL https://github.com/cma2015/TMOIA

1 Introduction

TMOIA-GUI is a graphical user interface (GUI) for TMOIA, a transformer-based multiomics integrative analysis approach for the phenotypic prediction and functional gene mining of complex traits. It is an open-source software that can run on Linux and Microsoft Windows. It is free for non-commercial users.

TMOIA-GUI contains two functional modules: TMOIA model construction and phenotypic prediction.

- ✧ For TMOIA model construction, it takes the multi-omics data and phenotypic data as inputs. Both classification and regression tasks can be performed.
- ✧ For phenotypic prediction, it takes trained model and omics feature data of new samples as inputs and returns the predicted phenotypic values.

2 Structure of TMOIA-GUI source code

```
TMOIA/GUI/src
├── gui_folder.spec
├── gui.py
├── main.py
├── model.cfg
├── tmoia.ui
├── tmoia.ico
├── tmoia.png
├── utils_gui.py
└── utils.py
```

The document contains the following 7 main components:

(1) **model.cfg**: Default configuration of model construction.

(2) **gui_folder.spec**: Specification file of package building for PyInstaller.

(3) **gui.py**: The python scripts of GUI actions.

(4) **main.py**: The python scripts of two main functions for model construction and phenotypic prediction.

(5) **tmoia.ui**: The GUI design document.

(6) **utils_gui.py**: Utilities for GUI actions.

(7) **utils.py**: Utilities for data loading, TMOIA model construction and prediction execution.

3 System requirements

Supported Operating Systems: Linux, Windows 10/11.

System architecture: x64.

Graphics: GPU with 12GB VRAM or larger.

4 Installation and start

For Windows

We provide an application package that **can run directly on Windows without any configuration. It is recommended for Windows users.**

(1) Download TMOIA-GUI self-extracting archive at releases page.

(2) Click to run **TMOIA-GUI_self-extracting.exe**. Select an appropriate path with user permissions to extract the program folder (e.g., DO NOT extract program files to **`C:\`** on Windows). The executable program and the affiliated files will appear in your selected directory.

(3) Enter the extracted folder. Click on the executable file **TMOIA-GUI.exe** to start.

For Linux

The TMOIA-GUI software can be manually configured on Linux. The following steps also works on Windows.

(1) Download source code. You need to extract the downloaded zip file.

(2) Install Anaconda or Miniconda from the Internet.

(3) Create a local conda environment using the file env_tmoia.yml. This process requires Internet connection. The conda environment called `tmoia` will be configured automatically:

```
conda env create -f env_tmoia.yml
```

(4) Enter the directory src where gui.py is located:

```
cd _your_path_to_/TMOIA/GUI/src
```

(5) Activate the conda environment `tmoia`:

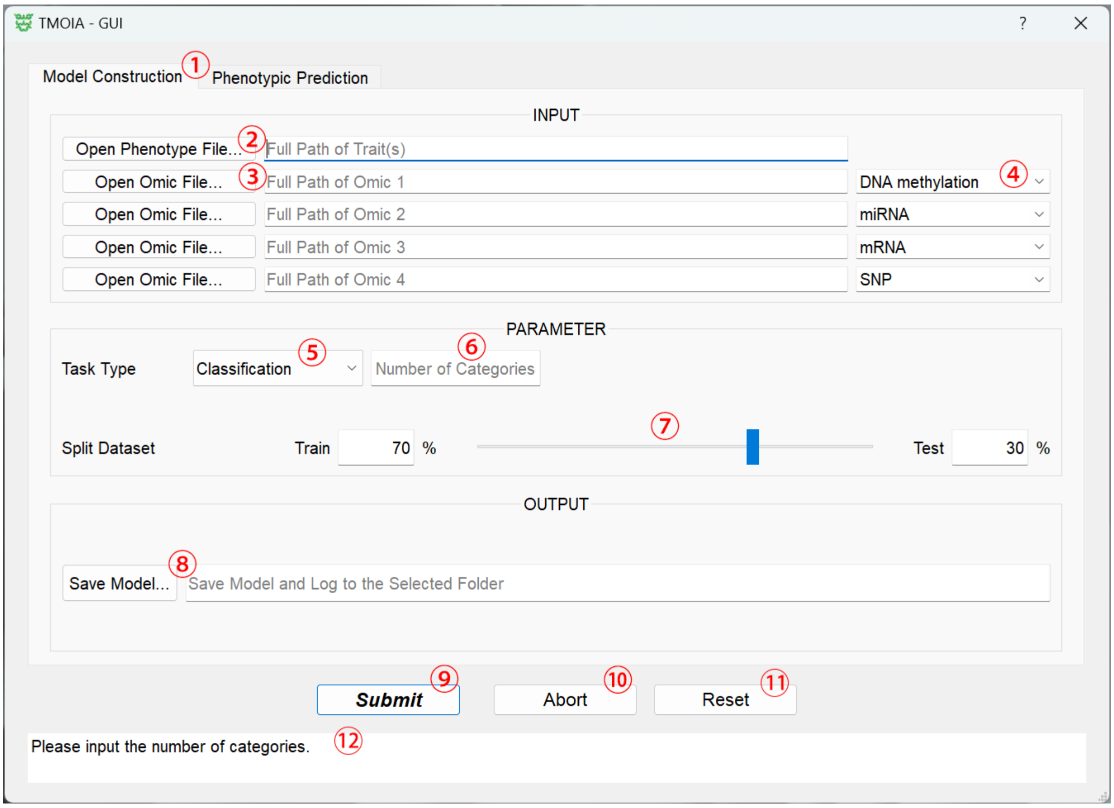
```
conda activate tmoia
```

(6) **Run GUI**:

```
python gui.py
```

5 GUI guide

5.1 GUI for model construction



DNA methylation
miRNA
mRNA
SNP

Task Type Classification

Split Dataset Train 70 % Test 30 %

Please input the number of categories.

①: Model construction mode or phenotypic prediction mode.

②: The path to a phenotype data file.

③: The path to a data file of an omic.

④: Define an omics title for the left inputted path.

⑤: Choose a task type for classification or regression.

⑥: The number of categories becomes available if the model works for classification.

⑦: The ratio of training set or validation set.

⑧: The path of saving your model.

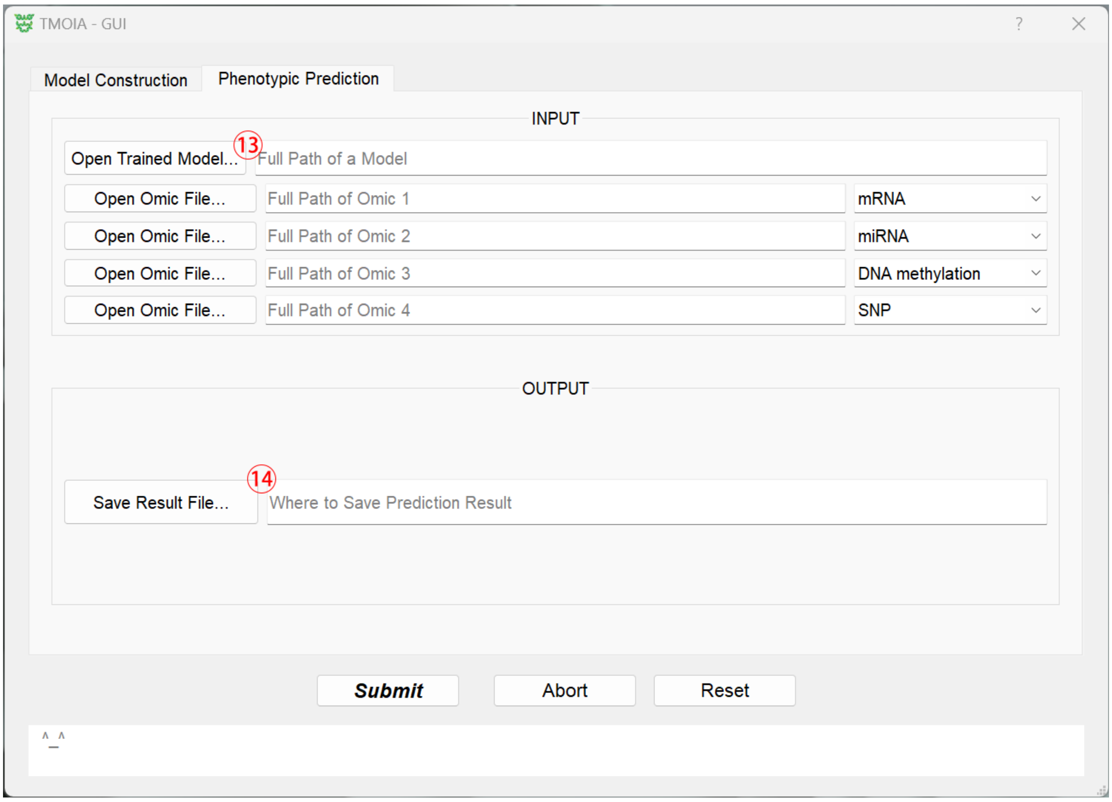
⑨: Click on **Submit** button to start training or prediction.

⑩: Click on Abort button to stop computation and exit the application.

⑪: Click on Reset button to restore UI.

⑫: The text browser shows useful tips.

5.2 GUI for phenotypic prediction



mRNA

miRNA

DNA methylation

SNP

⑬: The path to the model that needs to be loaded.

⑭: The path to the prediction result.

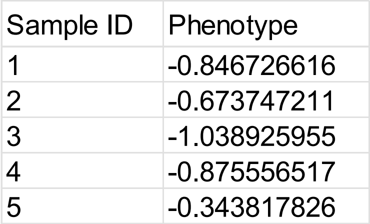
6 Input files

TMOIA-GUI can automatically split selected files into training set and validation set by the given ratio. You only need to select the data files to be used for modeling.

6.1 Input files for model construction

(1) A file of phenotype or labels in **csv** format. The table requires **columns for sample IDs and phenotypic values**. If you intend to execute classification task, the table should be filled with categories encoded by integers appropriately. Otherwise, z-score standardization is recommended for its values for regression task.

The example format:

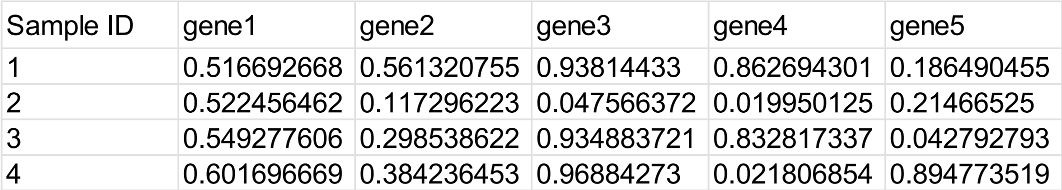


Sample ID	Phenotype
1	-0.846726616
2	-0.673747211
3	-1.038925955
4	-0.875556517
5	-0.343817826

Values of sample ID: numerals and strings are allowed.
Phenotypic values: integers and decimals are allowed.

(2) File of each omics data in **csv** format. These tables require **columns for sample IDs and the omics' features' values**. If the original feature values are continuous values, normalization or z-score standardization is recommended.

The example format:



Sample ID	gene1	gene2	gene3	gene4	gene5
1	0.516692668	0.561320755	0.93814433	0.862694301	0.186490455
2	0.522456462	0.117296223	0.047566372	0.019950125	0.21466525
3	0.549277606	0.298538622	0.934883721	0.832817337	0.042792793
4	0.601696669	0.384236453	0.96884273	0.021806854	0.894773519

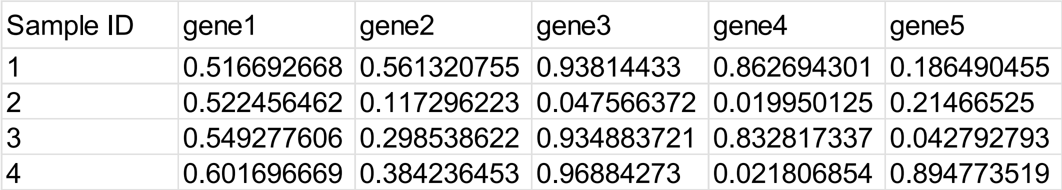
Values of sample ID: numerals and strings are allowed.
Values of each feature: integers and decimals are allowed.

6.2 Input files for phenotypic prediction

(1) A pretrained model file in **pth** format.

(2) File of each omics data in **csv** format. These tables require **columns for sample IDs and the omics' features' values**. If the original feature values are continuous values, normalization or z-score standardization is recommended.

The example format:



Sample ID	gene1	gene2	gene3	gene4	gene5
1	0.516692668	0.561320755	0.93814433	0.862694301	0.186490455
2	0.522456462	0.117296223	0.047566372	0.019950125	0.21466525
3	0.549277606	0.298538622	0.934883721	0.832817337	0.042792793
4	0.601696669	0.384236453	0.96884273	0.021806854	0.894773519

Values of sample ID: numerals and strings are allowed.
Values of each feature: integers and decimals are allowed.

7 Execute computation

7.1 Model construction

7.1.1 Configuration

The default configuration file **model.cfg** is located in TMOIA/GUI/src. Its parameters and definitions are listed below. You can customize the cfg file to suit your requirements.

```
"batch_size": 16,
"epoch_max": 550,
"patience": 30,
"learning_rate": 0.00001,
"Transformer_num_encoder": 4,
"Transformer_drop": 0.1,
"num_head": 1.
```

Definitions:

(1) **batch_size**: The number of training examples utilized in one iteration.

(2) **patience**: It works for early stopping strategy to avoid over-fitting. Early stopping is a technique used to prevent overfitting by stopping the training process if the model stops improving after a certain number of epochs. The patience parameter is the number of epochs to wait before stopping the training if no progress is made on the validation set.

(3) **epoch_max**: The maximum number of epochs. An epoch is one complete pass through the entire training dataset. It works with patience, if the patience does not stop the training process and the epoch reach the maximum number, the process stops. The best model will be saved, which gains the best score on validation set.

(4) **learning_rate**: The step size at each iteration while moving toward a minimum of a loss function.

(5) **Transformer_num_encoder**: The encoder number of feature extraction and feature integration. Theoretically, the more encoders there are, the more variable feature information can be extracted.

(6) **Transformer_drop**: The dropout probability of position-wise feed-forward networks in encoder module. An appropriate dropout probability will improve the performance of modeling different datasets.

(7) `num_head`: The number of heads is to split the input features, and is used for parallel computing with multi-head attention. The number of heads selected must be divisible by the input features.

7.1.2 Submit

(1) Select proper training data files to fill blanks in GUI.

(2) Click on ***Submit*** button to start training.

7.2 Phenotypic prediction

The model used for prediction should be a TMOIA model trained by the software.

(1) Select a model file and proper omics data files to fill blanks in GUI. TMOIA can automatically read configuration, features used in modeling and templates of output from `cfg` and `csv` files accompanying your selected model file.

(2) Click on ***Submit*** button to start prediction.

8 Output files

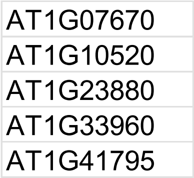
8.1 Output files of model construction

(1) <your_model>**.pth**: The output model.

(2) <your_model>**.pth.cfg**: The configuration of modeling. Its items are the same as items in default configuration.

(3) <your_model>**.pth_template_result.csv**: The template of prediction result. Its format is the same as the inputted phenotype file's.

(4) <your_model>**.pth_**<one_omics_name>**.csv**: The list of each omics' features used in model construction. For example:



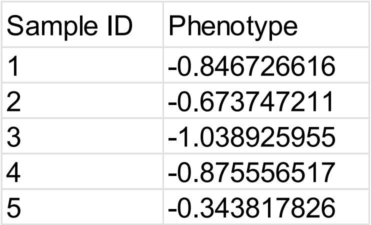
AT1G07670
AT1G10520
AT1G23880
AT1G33960
AT1G41795

Column name is not required. The values are feature names.

8.2 Output file of phenotypic prediction

(1) **<your_prediction_result>.csv**: The prediction results are phenotypic values or encoded categories.

The format depends on **<your_model>.pth_template_label.csv**, for example:



Sample ID	Phenotype
1	-0.846726616
2	-0.673747211
3	-1.038925955
4	-0.875556517
5	-0.343817826

Values of sample ID: numerals or strings.
Phenotypic values: integers or decimals.

9 Running verification with demo data

Two demo datasets are provided at **demo_data.zip**. Please unzip the archive after download. The two demo datasets are generated from The Cancer Genome Atlas Program (TCGA) and 1001 Arabidopsis Genomes that were used in our research respectively. Please select a dataset's files in GUI to verify installation and configuration.

The zip file structure is given below.

```
demo_data
├── classification
│   └── KIPAN_omics_3_feat_300_sample_172
│       ├── labels_172.csv
│       ├── meth_100.csv
│       ├── mirna_100.csv
│       └── mrna_100.csv
└── regression
    └── FT_omics_3_feat_250_sample_90
        ├── meth_150.csv
        ├── mrna_50.csv
        ├── pheno_90.csv
        └── snp_50.csv
```

(1) **classification_kipan**: multi-omics datasets of **kidney cancer type classification**.

(2) **KIPAN_omics_3_feat_300_sample_172**: the KIPAN dataset has 3 omics and total 300 features for 172 samples.

(3) labels_172.csv: cancer type labels of 172 samples of the KIPAN dataset.

(4) meth_100.csv: 100 DNA methylation features of 172 KIPAN samples.

(5) mirna_100.csv: 100 miRNA expression features of 172 KIPAN samples.

(6) mrna_100.csv: 100 mRNA expression features of 172 KIPAN samples.

(7) **regression_ath**: multi-omics datasets of **Arabidopsis flowering time regression**.

(8) **FT_omics_3_feat_250_sample_90**: the Arabidopsis flowering time dataset has 3 omics and total 250 features for 90 samples.

(9) pheno_90.csv: z-score treated flowering time phenotypes of 90 Arabidopsis samples.

(10) meth_150.csv: 150 DNA methylation features of 90 Arabidopsis samples. It contains 50 mCHG features, 50 mCHH features and 50 mCG features.

(11) mrna_50.csv: 50 mRNA expression features of 90 Arabidopsis samples.

(12) snp_50.csv: 50 single nucleotide polymorphism (SNP) features of 90 Arabidopsis samples.