DNNGP Manual

DNNGP, a deep neural network-based method for genomic prediction using multi-omics data in plants

Authors: Kelin Wang, Muhammad Ali Abid, Awais Rasheed, Jose Crossa, Sarah Hearne, **Huihui Li***

Version 2.0

Encoding: UTF-8

February 4th, 2023

Cite: Wang K, Abid MA, Rasheed A, Crossa J, Hearne S, Li H. DNNGP, a deep neural network-based method for genomic prediction using multi-omics data in plants. Mol Plant. 2023 Jan 2;16:279-293.

Doi: 10.1016/j.molp.2022.11.004, PMID: 36366781

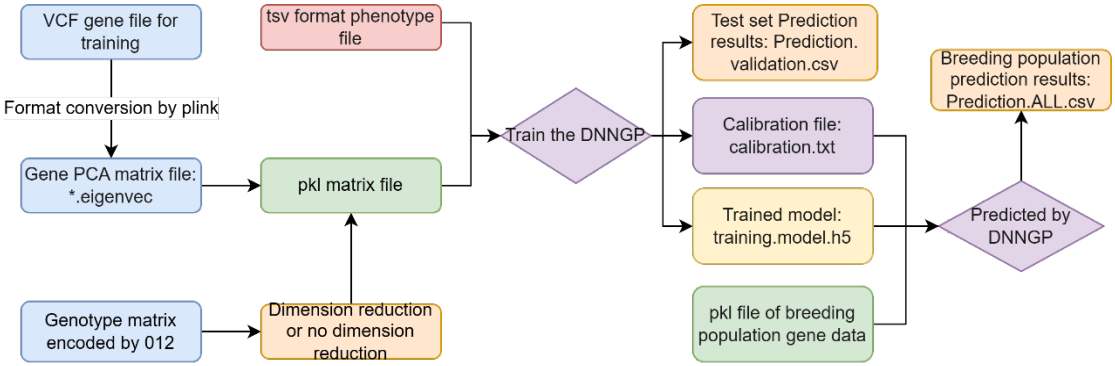
Contact:

Huihui Li: lihuihui@caas.cn

Table of Contents

1. DNNGP brief introduction

DNNGP is a genome-wide prediction model built based on deep learning theory, aiming to predict plant and animal phenotypes using genome-wide markers. In addition, DNNGP can be used for multi-omics data prediction of plants and animals. The model is mainly written using Python 3.6 and TensorFlow 1.15. The training and prediction process of the DNNGP was shown below:



1.1. DNNGP project address：https://github.com/AIBreeding/DNNGP

1.2. Document directory structure

DNNGP:

```
|   CN 使用说明.pdf
|   DNNGP-usermanual.pdf
|   requirements.txt
|
├─Input_files
|       tsv2pkl.py
|       wheat1.tsv
|       wheat599_pc95.tsv
|       wheat599_pc95.pkl
|
├─Output_files
|       calibration.txt
|       Prediction.ALL.csv
|       Prediction.validation.csv
|       training.model.h5
|
├─Scripts_for_generating_training_model
|       config_dnngp.py
|       dnngp.cp36-win_amd64.pyd
|       dnngp.cpython-36m-darwin.so
|       dnngp.cpython-36m-x86_64-linux-gnu.so
```

```
│       dnngp_runner.py
│       run.py
│
└─Scripts_for_prediction
        config_dnngp.py
        dnngp.cp36-win_amd64.pyd
        dnngp.cpython-36m-darwin.so
        dnngp.cpython-36m-x86_64-linux-gnu.so
        dnngp_runner.py
        run.py
```

The document contains the following five main sections：

(1) requirements.txt: Packages and their versions for environment building, and environment configuration.

(2) Input_files: This directory contains sample files for the input data.

(3) Scripts_for_generating training model: This directory contains the scripts needed to train the model, and the terminal displays the model prediction results for evaluating the training effect after the training is completed. Two files are also output: the trained model (training.model.h5) and the test set predictions (Prediction.validation.csv).

(4) Scripts_for_prediction: This directory contains the scripts needed for model prediction. Predictions are made for unknown phenotypes by reading the model trained in the previous step. Predict the breeding population phenotype and output the predicted value of ALL individuals (predication.all.csv).

(5) Output_files: This directory contains the output files of the DNNGP method after executing the input example files. The models are to be used in the following order:

① set up the runtime environment ② prepare the data ③ train the model ④ use the training model for prediction

2. Data preparation

Genotype data processing by plink2：

```
./plink --threads 30 --vcf *.vcf --pca --out pca10
```

--threads 30 Uses 30 threads

--vcf *.vcf Reads the vcf file

--pca 10 PC1-PC10 (configurable value ≤ number of samples ≤8000)

--out pca10 The name of the output file is pca10

If non-numeric chromosome numbers are present add the --allow-extra-chr parameter.

./plink2 --allow-extra-chr --threads 30 --vcf *.vcf --pca 10 --out pca10

The result generates two files with the suffixes .eigenval and .eigenvec, eigenval shows the weight of each Principal component (PC), and the weight/weight sum of each PC is the degree of explanation of each PC. The .eigenval file is the PCA matrix we need to use.**Notes**

(1) The above commands are applicable to Powershell terminal under windows platform as well as Linux and Mac terminals. If you are using cmd terminal under windows platform, please replace . /plink2 for plink2.

(2) If you use PCA to convert genetic data, you need to first combine the data of breeding population and training population and then perform PCA analysis, and then separate the two after getting PCA matrix.

3. DNNGP running environment construction

(1) Download the DNNGP project file at https://github.com/AIBreeding/DNNGP.

(2) Running environment configuration:

Firstly, you need to install Anaconda (https://www.anaconda.com/) or Miniconda (https://docs.conda.io/en/latest/miniconda.html).

Then, execute the following command to configure the environment:

- Conda create-n dnngp python=3.6.5
- Conda activate dnngp
- cd dnngp
- conda install --yes --file requirements.txt
- pip install tensorflow-determinism==0.3.0

4. Input data file

After the environment is set up, you need to prepare each data file according to the sample data format. The example data files are located in the following directories:

../dnngp/input_file/

- wheat1.tsv: tab-delimited phenotype data file.
- tsv2pkl.py: Format conversion script from tsv to pkl file.
- wheat599_pc95.tsv: tab-delimited principal component matrix file.

- wheat599_pc95.pkl: model-readable principal component matrix file.

Example:

ID	env1
M1	1.67162948
M2	-0.25270276
M3	0.341815127
M4	0.785439489
M5	0.998317613
M6	2.336096876
M7	0.617410817

Example:

ID	PC1	PC2	PC3	….
M1	7.0408269	2.053877771	-6.161150675	….
M2	5.924749016	1.137903031	1.132296531	….
M3	5.953045926	1.082444715	1.139961515	….

5.Trian DNNGP model

This section requires the input of two files, namely the principal component matrix file prepared in the previous step and the phenotype data file.

Parameter Description:

--batch_size: the sample size called by the training model

--lr: initial learning rate

--epoch: number of iterations

--dropout1: first feature discard (to prevent overfitting)

--dropout2: second feature discard (to prevent overfitting)

--patience: no boosting threshold

--Seed: random seed

--SNP: principal component matrix file path

--pheno: phenotype data file path

--output: output directory

To access the Scripts_for_generating_training_model file directory:

```
cd Scripts_for_generating_training_model
```

DNNGP training example command:

```
python dnngp_runner.py --batch_size 28 --lr 0.001 --epoch 100 --dropout1 0.5 --dropout2 0.3 --patience 25 --Seed 123 --SNP "../input_files/wheat599_pc95.pkl"
```

--pheno "../input_files/wheat1.tsv" --output /Your_path/

Output files

Prediction.validation.csv： The prediction results of the DNNGP model for the test set (the serial number in the first column represents the name of the predicted value individual in the original dataset).

training.model.h5： The trained model files are used for the next step of phenotypic trait prediction for breeding populations.

calibration.txt： A companion file to the h5 model file containing the mean and standard deviation of the model training set for the next step in predicting phenotypic traits in breeding populations.

After the training is completed, the terminal displays the Pearson correlation coefficient between the predicted and true values as follows:

'Corr obs vs pred =', (0.582, 0.001)

The first number is the correlation coefficient (0.582) and the second number is the *p-value* (0.001).

6. Prediction of the data to be measured using the trained model

After getting the trained model file, we have to predict the phenotypic traits of the dataset to be tested (i.e. breeding population) in the prediction model folder. This part requires three input files, two of which are the model files generated in the previous training step, i.e., calibration.txt and training.model.h5, and the third is the breeding population principal component matrix file (*.pkl) in the same format as the data file used in the previous model training step.

To access the Scripts_for_prediction file directory:

cd Scripts_for_prediction

DNNGP prediction parameters:

--Model: path to the .h5 model file generated when training the model

--SNP: principal component matrixfile path of the dataset to be predicted

--cal: calibration file generated during training

--output: the directory where the prediction result files are generated

DNNGP prediction example command:

```
python dnngp_runner.py --Model "/Your_path/training.model.h5" --SNP "/Your_path
/wheat599_pc95.pkl" --cal "/Your_path/calibration.txt" --output /Your_path/
```

Prediction output file

Prediction.ALL.csv

This file is the predicted results of phenotypic traits for all individuals in the breeding population.

7. Special Notes

Scripts named run.py are available in both training and prediction folders for batch testing.

Run the example command:

```
python run.py
```

User Manual (online)：https://www.wolai.com/3pALrhmqueeLFnCYeF43Do