EE6550 Machine Learning HW5 Neural Network

A simple neural network trained with SGD for regression.

- Author: Yu-Chun (Howard) Lo
- Email: howard.lo@nlplab.cc

User Manual

Dev Environment

- Developed under Anaconda 4.3.0 (x86_64).
- Require Numpy for matrix operations.
- Tested on Python 3.6.0.

File Structure

- `/dataset`: **(Important)** The program reads datasets from this folder and performs training and testing on the specified training and test data files. (See the *Dataset Format* section below)
- `/logs`: **(Important)** Output reports are stored in this folder.
- `/hypothesis`: **(Important)** Output hypotheses are stored in this folder. (There is **timestamp** at the end of the file name, which corresponds to the report)
- `core.py`: Components to build our model.
- `utils.py`: Some utilities used by this program, such as loading dataset, normalize labels, etc.
- `regression_nn.py`: Our model implementation.
- `main.py`: **(Important)** The main program. User should train the model by running this program.

Dataset Format

- Currently, the program only supports reading `.csv` file.
- The true label (real value) of each item should locate at the first column.
- **(Important)** If you want to train your SVR with your own dataset, please be sure that you've followed the required format described above, and have placed your own training and test data files in the `/dataset` folder.

Getting Started

Train your model by running `python main.py` in terminal. Be sure that your terminal is under the same directory as `main.py`.

Note that we've set default values for required input arguments. Run `python main.py --help` to view input arguments information shown below.

```
usage: main.py [-h] [--header_filename HEADER_FILENAME]
               [--train_filename TRAIN_FILENAME]
               [--test_filename TEST_FILENAME] [--depth DEPTH]
```

```
                [--learning_rate LEARNING_RATE] [--reg_lambda REG_LAMBDA]
                [--K K] [--max_iteration MAX_ITERATION]

Nerual network (NN) for regression.

optional arguments:
  -h, --help            show this help message and exit
  --header_filename HEADER_FILENAME
                        Training dataset with header csv. This is used to
                        output hypothesis. (Default:
                        "energy_efficiency_cooling_load_training_header.csv")
  --train_filename TRAIN_FILENAME
                        Training dataset csv. (Default:
                        "energy_efficiency_cooling_load_training.csv")
  --test_filename TEST_FILENAME
                        Training dataset csv. (Default:
                        "energy_efficiency_cooling_load_testing.csv")
  --depth DEPTH         Number of weight matrices (Defualt: 2. (depth-1)
                        hidden layers)
  --learning_rate LEARNING_RATE
                        Step size of updating weights. (Default: 0.01)
  --reg_lambda REG_LAMBDA
                        Strength of L2 regularization for weights. (Default:
                        0.0)
  --K K                 Denotes for "K"-fold cross-validation for determine
                        the optimal hyper-parameters for NN. (Default: None)
  --max_iteration MAX_ITERATION
                        Max iteration for NN training algorithm to avoid not
                        converging. (Defualt: 30000)
```

For Grading Session

Here we show some guides for different test scenarios:

- Place the training data file(e.g. `xxx_training.csv`), testing data file(e.g. `xxx_testing.csv`) and training header file(e.g. `xxx_training_header.csv`) in the `/dataset` folder before running `main.py` with specified `--train_filename`, `--test_filename` and `--header_filename`.
- To specify model depth, for example, run `--depth=2` will generate 2-layer neural net with hidden size 20 (20 is our default hidden size)
- To specify learning rate, run `--learning_rate=0.01`. (Default: 0.0.1)
- To specify regularization term, run `--reg_lambda=0.1`. (Default: 0.0)
- To specify max iterations, run `--max_iteration=10000`. (Default: 30000)
- To perform K-fold cross-validation, for example, specify `--K=5`. (Default: None)
- All the required output information, such as class label mapping, cross-validation history, optimal hyper-parameters, etc., are stored at the `logs/` folder. Note that the log file name indicates what kernel type and number of K-fold you choosed, and when you run the program. This naming convension aims to help graders to choose which report to check after running the program.
- Note that the csv file name of our hypothesis in the `hypothesis/` folder is concatenated with the timestamp. This aims to let the graders know which hypothesis file to choose to test after running the program. (You may remove the timestamp for grading)

- Here is one of a example for running the following commands in the test scienarios:

```
(For 5-fold CV)
$ python main.py --header_filename="xxx_training_header.csv"
--train_filename="xxx_training.csv" --test_filename="xxx_testing.csv" --K=5
```

Report

All required output information are stored at `logs/`.

Snippet (See more in `logs/`):

```
--------------------------------------------------------------------------------
-----------------
[*] Cross validation history:
 -  Parameter: {'depth': 4, 'reg_lambda': 0.0, 'max_iteration': 30000} | Cross
validation error: [ 4.08458964]
 -  Parameter: {'depth': 2, 'reg_lambda': 0.0, 'max_iteration': 30000} | Cross
validation error: [ 5.14624725]
 -  Parameter: {'depth': 3, 'reg_lambda': 0.0, 'max_iteration': 30000} | Cross
validation error: [ 5.33495332]
 -  Parameter: {'depth': 3, 'reg_lambda': 0.001, 'max_iteration': 30000} | Cross
validation error: [ 5.40552295]
 -  Parameter: {'depth': 2, 'reg_lambda': 0.001, 'max_iteration': 30000} | Cross
validation error: [ 5.64238829]
 -  Parameter: {'depth': 5, 'reg_lambda': 0.001, 'max_iteration': 30000} | Cross
validation error: [ 5.97097392]
 -  Parameter: {'depth': 4, 'reg_lambda': 0.001, 'max_iteration': 30000} | Cross
validation error: [ 6.02975134]
 -  Parameter: {'depth': 6, 'reg_lambda': 0.0, 'max_iteration': 30000} | Cross
validation error: [ 6.19784738]
 -  Parameter: {'depth': 2, 'reg_lambda': 0.01, 'max_iteration': 30000} | Cross
validation error: [ 6.27289822]
 -  Parameter: {'depth': 2, 'reg_lambda': 0.005, 'max_iteration': 30000} | Cross
validation error: [ 6.37716251]
 -  Parameter: {'depth': 5, 'reg_lambda': 0.0, 'max_iteration': 30000} | Cross
validation error: [ 6.58103282]
 -  Parameter: {'depth': 3, 'reg_lambda': 0.005, 'max_iteration': 30000} | Cross
validation error: [ 7.07741607]
 -  Parameter: {'depth': 3, 'reg_lambda': 0.01, 'max_iteration': 30000} | Cross
validation error: [ 7.7029069]
 -  Parameter: {'depth': 4, 'reg_lambda': 0.005, 'max_iteration': 30000} | Cross
validation error: [ 7.83644334]
 -  Parameter: {'depth': 4, 'reg_lambda': 0.01, 'max_iteration': 30000} | Cross
validation error: [ 8.09004805]
 -  Parameter: {'depth': 6, 'reg_lambda': 0.001, 'max_iteration': 30000} | Cross
validation error: [ 8.62358754]
 -  Parameter: {'depth': 5, 'reg_lambda': 0.01, 'max_iteration': 30000} | Cross
validation error: [ 8.63376377]
 -  Parameter: {'depth': 5, 'reg_lambda': 0.005, 'max_iteration': 30000} | Cross
validation error: [ 9.27886413]
 -  Parameter: {'depth': 6, 'reg_lambda': 0.005, 'max_iteration': 30000} | Cross
validation error: [ 10.59537494]
 -  Parameter: {'depth': 6, 'reg_lambda': 0.01, 'max_iteration': 30000} | Cross
```

```
validation error: [ 10.81596514]
--------------------------------------------------------------------------------
----------------
[*] Best parameter: {'depth': 4, 'reg_lambda': 0.0, 'max_iteration': 30000}
[*] Best cross validation error: [ 4.08458964]
[*] Start to train on full training data and evaluate on test data ...
--------------------------------------------------------------------------------
----------------
[*] Datetime: 14:11:18
[*] Train file path: "dataset/energy_efficiency_cooling_load_training.csv"
[*] Test file path: "dataset/energy_efficiency_cooling_load_testing.csv"
[*] Best parameter: {'depth': 4, 'reg_lambda': 0.0, 'max_iteration': 30000}
[*] Network shape: [(9, 20), (21, 20), (21, 20), (21, 1)]
[*] Performance: Test error: [ 7.28077631]
--------------------------------------------------------------------------------
----------------
[*] Saving SGD hypothesis to "hypothesis/SGD_hypothesis_header-[14:11:18].csv" ...
[*] Output SGD hypothesis to "hypothesis/SGD_hypothesis_header-[14:11:18].csv"
success.
--------------------------------------------------------------------------------
----------------
```

Our hypothesis is located at `hypothesis/SGD_hypothesis_header-[14:11:18].csv`. The saving format is as the same as what professor described at *ilms*.

Note that the corresponding hypothesis csv file is indicated by the `HH:MM:SS` timestamp, which is the time that the program finished training.

Discussion

- I've run dozens of hidden layer size and finally turned out that 20 is good enough.
- If regularization term is specified, test error might increase, but the overfitting problem might be improved (training error is not much larger than validation error)