MASS (Man for All SeasonS)

The Deep Learning RE classification model in Python

MASS, version 1.0, is a program that was developed with 3 main functions:

- Classify relation between entities pair in text documents
- Train new models with given corpora that follow the format
- Evaluate trained models with test dataset

We evaluated our model on 6 datasets, but in this example code, we included 2 typical datasets (one for generic data – semeval 2010, one for biomedical data – BC5).

1. Installation

This program was developed using Python version 3.5 and was tested on Linux and Windows system. We recommend using Anaconda 3 newest version for installing Python 3.5 as well as numpy, although you can install them by other means.

Other requirements:

1. numpy

   Included in Anaconda package
2. scipy

   Included in Anaconda package
3. h5py

   ```
   $ conda install -c anaconda h5py
   ```

4. Tensorflow

   ```
   $ pip install tensorflow     # Python 3.n; CPU support

   $ pip install tensorflow-gpu # Python 3.n; GPU support
   ```

   If you are install tensorflow with GPU support, please follow the instructions on the official document to install other required libraries for your platform. Official document can be found at https://www.tensorflow.org/install/

5. Keras

   ```
   $ pip install keras
   ```

6. sklearn

```
$ conda install -c anaconda scikit-learn
```

7. NLTK

```
$ conda install -c anaconda nltk
```

You must download wordnet corpus using **`nltk.download()`**

8. imbalanced-learn

```
$ conda install -c conda-forge imbalanced-learn
```

9. fastText

```
$ git clone https://github.com/facebookresearch/fastText.git

$ cd fastText

$ pip install .
```

To download the pre-trained fastText model, go to the download page at: https://fasttext.cc/docs/en/pretrained-vectors.html. You should download English **`bin+text`** zip file, extract the **`.zip`** file and then, put the **`wiki.en.bin`** file at **`data/w2v_model`**. Because of anonymous submission and review, we use generic fastText model for both BC5 and semeval 2010 datasets instead of using our trained fastText model on Biomedical text for BC5 dataset.

2. Usage

Build data

Before train and test the model, we must build some data:

- We pre-build trimmed embeddings before any run to boost up the speed of loading.
- We pre-build the dataset before feeding the data to the model. The datasets are then dumped to pickle file for later use.

To build dataset, run the following command:

```
$ python build_data.py
```

The result contains:

- Two (2) **`.npz`** file, located at **`data/w2v_model`**
- A pickle directory at **`data/pickle`**, containing four (4) pickle files.

Train the models

Use one of two testing file (**`test_bc5.py`** or **`test_semeval.py`**) to train the model and evaluate on two benchmark datasets.

Commands:

```
$ python test_bc5.py -h
usage: test_bc5.py [-h] [-i I] [-e E] [-p P] [-rus RUS] [-ros ROS]
                   [-rss RSS] [-msl MSL] [-cnn1 CNN1] [-cnn2 CNN2]
                   [-cnn3 CNN3] [-ft FT] [-wns WNS] [-char CHAR]
                   [-chari CHARI] [-pos POS] [-posi POSI]
                   [-rel REL] [-reli RELI] [-dir DIR] [-hd HD]
                   [-a A]

Multi-channel biLSTM-CNN for relation extraction

optional arguments:
  -h, --help    show this help message and exit
  -i I          Job identity
  -e E          Number of epochs
  -p P          Patience of early stop (0 for ignore early stop)
  -rus RUS      Random under sampling number
  -ros ROS      Random over sampling number
  -rss RSS      Random seed for re-sampler
  -msl MSL      Trimmed max sentence length
  -cnn1 CNN1    Number of CNN region size 1 filters
  -cnn2 CNN2    Number of CNN region size 2 filters
  -cnn3 CNN3    Number of CNN region size 3 filters
  -ft FT        Number of output fastText w2v embedding LSTM
dimension
  -wns WNS      Number of output Wordnet superset LSTM dimension
  -char CHAR    Number of output character embedding LSTM dimension
  -chari CHARI  Number of input character embedding LSTM dimension
  -pos POS      Number of output POS tag LSTM dimension
  -posi POSI    Number of input POS tag LSTM dimension
  -rel REL      Number of output dependency relation LSTM dimension
  -reli RELI    Number of input dependency relation LSTM dimension
  -dir DIR      Number of dependency direction embedding dimension
  -hd HD        Hidden layer configurations default '128,128'
  -a A          Alpha ratio default 0.55
```

All hyper-parameter is set default to the tuned values. You can change every setting of the model or try the default one.

Example 1: test with default setting on BC5 corpus

```
$ python test_bc5.py
```

Example 2: test with some configurations on semeval 2010 corpus

```
$ python test_semeval.py -ft 128 -cnn1 128 -cnn2 256 -hd 128,256
```

This command means:

- Change number of output fastText embedding LSTM dimension to 128
- Using 2 region sizes of CNN with 128 and 256 filter corresponding.
- Using 2 hidden layers before softmax layer with 128 and 256 nodes.

Evaluate the trained models

Semeval 2010

After train the model, the model is used to predict on the provided testing dataset. The prediction is written at **`data/output/`** with the name **`answer-$i`** with **`$i`** is the **`-i`** parameter on the running command (default is 0). The answer file can be used to evaluated with provided testing program.

Command to test the answer:

```
$ ./evaluate/semeval2010_task8_scorer-v1.2.pl <answer> <key>
```

The key file is stored at **`evaluate/KEY`** based on the golden testing file.

Example:

```
$ ./evaluate/semeval2010_task8_scorer-v1.2.pl data/output/answer-
example evaluate/KEY
```

The final result is printed at the last lines with format:

```
Micro-averaged result (excluding Other):
P = x/x =  xx.xx%     R = x/x =  xx.xx%     F1 =  xx.xx%
MACRO-averaged result (excluding Other):
P =  xx.xx%     R =  xx.xx%     F1 =  xx.xx%
<<< The official score is (9+1)-way evaluation with directionality
taken into account: macro-averaged F1 = xx.xx% >>>
```

Official score is of macro-averaged with (9+1)-way evaluation with directionality taken into account. With other ways of evaluation and other result, see the above lines.

BC5

For more clear insight evaluation, we re-implement the BC5 evaluation that strictly follows BC5 CDR rules.

The evaluated results is printed after training and using model.

The result is printed in the format:

```
Testing model over test set

result: abstract:  (p, r, f1) intra (p, r, f1)

result after apply chemical rule 1/2: abstract:  (p, r, f1)
```

The result contains:

- Results of baseline model with abstract level result (as same as the BC5 testing service) and intra level result (the result conduct on the intra-sentence relation, excluding cross-sentence relation).
- Results after apply chemical rule (as described in the paper) at abstract level (as same as the BC5 testing service).