SRILM Python Binding

Release 3.0.0

Mar 30, 2020

Contents

This project aims to bring the power of the SRILM Toolkit to Python 3.

Instead of faithfully wrapping SRILM C++ classes, we create a new set of APIs to give them a Pythonic look-and-feel while preserving the raw power of SRILM Toolkit as much as possible. In the process, we also try to 'smooth away' some of the idiosyncrasies of the SRILM APIs.

CHAPTER 1

DEPENDENCIES

- Python 3 >= 3.7.3
- SRI LM Toolkit >= 1.7.3
- liblbfgs >= 1.10 (for MaxEnt LM)
- Cython >= 0.29.16
- (optional) Sphinx >= 2.4.4

CHAPTER 2

INSTALL

To get started, first download SRI Language Modeling Toolkit.

Install SRILM Toolkit:

```
$ mkdir srilm-1.7.3
$ tar xf srilm-1.7.3.tar.gz -C srilm-1.7.3
$ cd srilm-1.7.3
$ export SRILM=$PWD
$ make HAVE_LIBLBFGS=1 MAKE_PIC=yes World
$ make cleanest
```

Then check out this project and put it *under* the root directory of SRILM:

```
$ cd $SRILM
$ git clone https://github.com/nuance1979/srilm-python
```

Build SRILM Toolkit with 'HAVE_LIBLBFGS=1' to make sure MaxEnt LM is usable.

Now you can build this project by:

```
$ cd srilm-python
$ make
```

If you specified build options in your SRILM build, then use the same option again:

```
$ cd srilm-python
$ make OPTION=<your_srilm_build_option>
```

You might need to specify your library and/or include pathes by editing either setup.py or Makefile. Note that there are '–include-dirs' and '–library-dirs' options for 'python setup.py build_ext'. See usage by:

```
$ python3 ./setup.py build_ext --help
```

CHAPTER 3

EXAMPLES

If successful, you can take a look at the example script:

```
$ ./example.py --help
```

Or try it interactively by:

```
$ python3
...
>>> import srilm
```

I also included a shell script calling SRILM command line tools corresponding to the example.py script:

```
$ ./example.sh
```

As a sanity check, here are the output of example.sh with the WSJ portion of Penn Treebank with the 'industry standard' split and preprocessing:

```
$ ./example.sh 3 wsj/dict wsj/text.00-20 wsj/text.21-22 wsj/text.23-24 2>/dev/null
Ngram LM with Good-Turing discount:
file wsj/text.23-24: 3761 sentences, 78669 words, 0 OOVs
0 zeroprobs, logprob= -182850.5 ppl= 165.292 ppl1= 211.0094
Ngram LM with Witten-Bell discount:
file wsj/text.23-24: 3761 sentences, 78669 words, 0 OOVs
0 zeroprobs, logprob= -183186.6 ppl= 166.8511 ppl1= 213.0954
Ngram LM with Kneser-Ney discount:
file wsj/text.23-24: 3761 sentences, 78669 words, 0 OOVs
0 zeroprobs, logprob= -179527.7 ppl= 150.6403 ppl1= 191.4538
Ngram LM with Chen-Goodman discount:
file wsj/text.23-24: 3761 sentences, 78669 words, 0 OOVs
0 zeroprobs, logprob= -178963.1 ppl= 148.2832 ppl1= 188.316
Ngram LM with Jelinek-Mercer smoothing:
file wsj/text.23-24: 3761 sentences, 78669 words, 0 OOVs
0 zeroprobs, logprob= -184712.2 ppl= 174.1153 ppl1= 222.8264
MaxEnt LM:
```

(continues on next page)

(continued from previous page)

```
file wsj/text.23-24: 3761 sentences, 78669 words, 0 OOVs
0 zeroprobs, logprob= -178745 ppl= 147.3824 ppl1= 187.1175
```

And for example.py:

```
$ ./example.py --order 3 --vocab wsj/dict --train wsj/text.00-20 --heldout wsj/text.
↪21-22 --test wsj/text.23-24 2>/dev/null
Ngram LM with Good-Turing discount: logprob = -182850.49858691066 denom = 82430.0 ppl␣
↪= 165.29199936652992
Ngram LM with Witten-Bell discount: logprob = -183186.5865671382 denom = 82430.0 ppl␣
↪= 166.85110458088772
Ngram LM with Kneser-Ney discount: logprob = -179527.68699812848 denom = 82430.0 ppl␣
↪= 150.64028400082367
Ngram LM with Chen-Goodman discount: logprob = -178963.10104117272 denom = 82430.0␣
↪ppl = 148.28316532800372
Ngram LM with Jelinek-Mercer smoothing: logprob = -184712.19462050498 denom = 82430.0␣
↪ppl = 174.11532932716725
MaxEnt LM: logprob = -178744.9764584251 denom = 82430.0 ppl = 147.3824149925418
```

CHAPTER 4

DOCUMENTATION

You can read it here or make it from scratch by:

```
$ make docs
```

CHAPTER 5

UNIT TESTS

You can run unit tests by:

```
$ make test
```

CHAPTER 6

API

You can get usage info the Python way, e.g.,:

```
$ python3
...
>>> import srilm
>>> help(srilm.vocab.Vocab)
```

CHAPTER 7

Indices and tables

- genindex
- modindex
- search