SRILM Python Binding Documentation

Release 1.0.0

Yi Su

September 04, 2016

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 >= 3.5.2
- SRI LM Toolkit >= 1.7.1
- liblbfgs >= 1.10 (for MaxEnt LM)
- Cython >= 0.20.1
- (optional) Sphinx >= 1.2.2

CHAPTER 2

INSTALL

To get started, first download SRI Language Modeling Toolkit.

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

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

Note: There is a minor bug in SRILM 1.7.1. You can optionally patch it by:

```
$ cd srilm-python
$ patch $SRILM/lm/src/MEModel.cc < srilm/MEModel.cc.patch
```

Note that you need to (re)build SRILM to activate it.

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>
```

Note: There is one test failure with OPTION=_c. I suspect it's a minor SRILM bug.

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 /home/yisu/Work/data/wsj/text.23-24: 3761 sentences, 78669 words, 0 OOVs
0 zeroprobs, logprob= -182850 ppl= 165.292 ppl1= 211.009
Ngram LM with Witten-Bell discount:
file /home/yisu/Work/data/wsj/text.23-24: 3761 sentences, 78669 words, 0 OOVs
0 zeroprobs, logprob= -183187 ppl= 166.851 ppl1= 213.095
Ngram LM with Kneser-Ney discount:
file /home/yisu/Work/data/wsj/text.23-24: 3761 sentences, 78669 words, 0 OOVs
0 zeroprobs, logprob= -179528 ppl= 150.64 ppl1= 191.454
Ngram LM with Chen-Goodman discount:
file /home/yisu/Work/data/wsj/text.23-24: 3761 sentences, 78669 words, 0 OOVs
0 zeroprobs, logprob= -178963 ppl= 148.283 ppl1= 188.316
Ngram LM with Jelinek-Mercer smoothing:
file /home/yisu/Work/data/wsj/text.23-24: 3761 sentences, 78669 words, 0 OOVs
0 zeroprobs, logprob= -184712 ppl= 174.115 ppl1= 222.826
MaxEnt LM:
file /home/yisu/Work/data/wsj/text.23-24: 3761 sentences, 78669 words, 0 OOVs
0 zeroprobs, logprob= -178757 ppl= 147.433 ppl1= 187.185
```

And for example.py:

```
$ ./example.py --order 3 --vocab wsj/dict --train wsj/text.00-20 --heldout wsj/text.21-22 --test wsj/
Ngram LM with Good-Turing discount: logprob = -182850.498553 denom = 82430.0 ppl = 165.291999209
Ngram LM with Witten-Bell discount: logprob = -183186.586563 denom = 82430.0 ppl = 166.851104561
Ngram LM with Kneser-Ney discount: logprob = -179527.687043 denom = 82430.0 ppl = 150.64028419
Ngram LM with Chen-Goodman discount: logprob = -178963.100995 denom = 82430.0 ppl = 148.283165135
```

```
Ngram LM with Jelinek-Mercer smoothing: logprob = -184712.194621 denom = 82430.0 ppl = 174.115329327
MaxEnt LM: logprob = -178740.10768 denom = 82430.0 ppl = 147.362371816
```

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