CkipClassic

Release v1.2.2

Mu Yang

May 07, 2021

OVERVIEW

CHAPTER

ONE

INTRODUCTION

A Linux Python wrapper for CKIP classic tools — CKIP Word Segmentation and CKIP Parser.

Attention: Please use CKIPNLP for structured data types and pipeline drivers.

Attention: For Python 2 users, please use PyCkip 0.4.2 instead.

1.1 Git

https://github.com/ckiplab/ckip-classic

1.2 PyPI

https://pypi.org/project/ckip-classic

1.3 Documentation

https://ckip-classic.readthedocs.io/

1.4 Contributers

- Mu Yang at CKIP (Author & Maintainer)
- Wei-Yun Ma at CKIP (Maintainer)

1.5 External Links

- Online Demo

1.6 Requirements

- Python 3.6+
- Cython 0.29+

Note that one should have CKIPWS/CKIPParser for this project:

- CKIP Word Segmentation Linux version 20190524+
  - Academic License
  - Commercial License
- CKIP Parser Linux version 20190725+
  - Academic License (Online Version)
  - Commercial License

CHAPTER

TWO

INSTALLATION

Attention:

- Offline version: CKIPWS (Academic/Commercial License) and CKIPParser (Commercial License).
- Online version: CKIPParser (Academic License).

2.1 Offline Version

Download CKIPWS and/or CKIPParser from above links. Denote `<ckipws-linux-root>` as the folder containing CKIPWS, and `<ckipparser-linux-root>` as the folder containing CKIPParser.

```
pip install --force-reinstall --upgrade ckip-classic \
   --install-option='--ws' \
   --install-option='--ws-dir=<ckipws-linux-root>' \
   --install-option='--parser' \
   --install-option='--parser-dir=<ckipparser-linux-root>'
```

Ignore ws/parser options if one doesn't have CKIPWS/CKIPParser.

Attention: Please use absolute paths.

2.2 Online Version

Register an account at *http://parser.iis.sinica.edu.tw/v1/reg.exe*

```
pip install --upgrade ckip-classic
```

2.3 Installation Options

Option	Detail	Default Value
`--[no-]ws`	Enable/disable CKIPWS.	False
`--[no-]parser`	Enable/disable CKIP-Parser.	False
`--ws-dir=<ws-dir>`	CKIPWS root directory.	
`--ws-lib-dir=<ws-lib-dir>`	CKIPWS libraries directory	`<ws-dir>/lib`
`--ws-share-dir=<ws-share-dir>`	CKIPWS share directory	`<ws-dir>`
`--parser-dir=<parser-dir>`	CKIPParser root directory.	
`--parser-lib-dir=<parser-lib-dir>`	CKIPParser libraries directory	`<parser-dir>/lib`
`--parser-share-dir=<parser-share-dir>`	CKIPParser share directory	`<parser-dir>`
`--data2-dir=<data2-dir>`	“Data2” directory	`<ws-share-dir>/Data2`
`--rule-dir=<rule-dir>`	“Rule” directory	`<parser-share-dir>/Rule`
`--rdb-dir=<rdb-dir>`	“RDB” directory	`<parser-share-dir>/RDB`

CHAPTER

THREE

USAGE

See https://ckip-classic.readthedocs.io/ for API details.

3.1 CKIPWS

CKIP Word Segmentation offline driver.

```
import ckip_classic.ws
print(ckip_classic.__name__, ckip_classic.__version__)

ws = ckip_classic.ws.CkipWs(logger=False)
print(ws(''))
for l in ws.apply_list(['', '']): print(l)

ws.apply_file(ifile='sample/sample.txt', ofile='output/sample.tag', uwfile='output/
↪sample.uw')
with open('output/sample.tag') as fin:
    print(fin.read())
with open('output/sample.uw') as fin:
    print(fin.read())
```

3.2 CKIPParser

CKIP Parser offline driver.

```
import ckip_classic.parser
print(ckip_classic.__name__, ckip_classic.__version__)

ps = ckip_classic.parser.CkipParser(logger=False)
print(ps(''))
for l in ps.apply_list(['', '']): print(l)

ps.apply_file(ifile='sample/sample.txt', ofile='output/sample.tree')
with open('output/sample.tree') as fin:
    print(fin.read())
```

3.3 CKIPParserClient

CKIP Parser online client.

```
import ckip_classic.client
print(ckip_classic.__name__, ckip_classic.__version__)

ps = ckip_classic.client.CkipParserClient(username='USERNAME', password='PASSWORD')
print(ps('(Na)(T)(COMMACATEGORY)'))
for l in ps.apply_list(['(Na)(T)(COMMACATEGORY)', '(I)(D)(D)(D)(PERIODCATEGORY)']):␣
↪print(l)
```

CHAPTER

FOUR

FAQ

Danger: Due to C code implementation, both `CkipWs` and `CkipParser` can only be instance once.

Warning: CKIPParser fails if input text contains special characters such as `()+-:|`. One may replace these characters by

```
text = text
    .replace('(', '')
    .replace(')', '')
    .replace('+', '')
    .replace('-', '')
    .replace(':', '')
    .replace('|', '')
```

Tip: fatal error: Python.h: No such file or directory". What should I do?

Install Python development package

```
sudo apt-get install python3-dev
```

Tip: The CKIPWS throws "what(): locale::facet::_S_create_c_locale name not valid". What should I do?

Install locale data.

```
apt-get install locales-all
```

Tip: The CKIPParser throws "ImportError: libCKIPParser.so: cannot open shared object file: No such file or directory". What should I do?

Add below command to `~/.bashrc`:

```
export LD_LIBRARY_PATH=<ckipparser-linux-root>/lib:$LD_LIBRARY_PATH
```

CHAPTER

FIVE

LICENSE

CHAPTER

SIX

CKIP_CLASSIC PACKAGE

Subpackages

6.1 ckip_classic.client package

class ckip_classic.client.**CkipParserClient**(*, *username=None*, *password=None*)
Bases: object

The CKIP sentence parsing client.

Parameters

- **username** (*str*) – the username (default to the environment variable $CKIPPARSER_USERNAME).
- **password** (*str*) – the password (default to the environment variable $CKIPPARSER_PASSWORD).

Note: One may register an account at http://parser.iis.sinica.edu.tw/v1/reg.exe

---

apply(*text*)
Parse a sentence.

Parameters **text** (*str*) – the input sentence.

Returns *str* – the output sentence.

Hint: One may also call this method as __call__().

---

apply_list(*ilist*)
Parse a list of sentences.

Parameters **ilist** (*List[str]*) – the list of input sentences.

Returns *List[str]* – the list of output sentences.

Submodules

6.2 ckip_classic.ini module

ckip_classic.ini.**create_ws_lex**(**lex_list*)

Generate CKIP word segmentation lexicon file.

Parameters ***lex_list** (*Tuple[str, str]*) – the lexicon word and its POS-tag.

Returns

- **lex_file** (*str*) – the name of the lexicon file.
- **f_lex** (*TextIO*) – the file object.

Attention: Remember to close **f_lex** manually.

ckip_classic.ini.**create_ws_ini**(*, *data2_dir=None*, *lex_file=None*, *new_style_format=False*, *show_category=True*, *sentence_max_word_num=80*, ***options*)

Generate CKIP word segmentation config.

Parameters

- **data2_dir** (*str*) – the path to the folder "Data2/".
- **lex_file** (*str*) – the path to the user-defined lexicon file.
- **new_style_format** (*bool*) – split sentences by newline characters ("\n") rather than punctuations.
- **show_category** (*bool*) – show part-of-speech tags.
- **sentence_max_word_num** (*int*) – maximum number of words per sentence.

Returns

- **ini_file** (*str*) – the name of the config file.
- **f_ini** (*TextIO*) – the file object.

Attention: Remember to close **f_ini** manually.

ckip_classic.ini.**create_parser_ini**(*, *ws_ini_file*, *rule_dir=None*, *rdb_dir=None*, *do_ws=True*, *do_parse=True*, *do_role=True*, *sentence_delim=','*, ***options*)

Generate CKIP parser config.

Parameters

- **rule_dir** (*str*) – the path to "Rule/".
- **rdb_dir** (*str*) – the path to "RDB/".
- **do_ws** (*bool*) – do word segmentation.
- **do_parse** (*bool*) – do parsing.
- **do_role** (*bool*) – do role.
- **sentence_delim** (*str*) – the sentence delimiters.

Returns

- **ini_file** (*str*) – the name of the config file.
- **f_ini** (*TextIO*) – the file object.

Attention: Remember to close **f_ini** manually.

6.3 ckip_classic.parser module

class ckip_classic.parser.**CkipParser**(*, *logger=False*, *ini_file=None*, *ws_ini_file=None*, *lex_list=None*, ***kwargs*)

Bases: `object`

The CKIP sentence parsing driver.

Parameters

- **logger** (*bool*) – enable logger.
- **lex_list** (*Iterable*) – passed to *ckip_classic.ini.create_ws_lex()*, overridden **lex_file** for *ckip_classic.ini.create_ws_ini()*.
- **ini_file** (*str*) – the path to the INI file.
- **ws_ini_file** (*str*) – the path to the INI file for CKIPWS.

Other Parameters

- ** – the configs for CKIPParser, passed to *ckip_classic.ini.create_parser_ini()*, ignored if **ini_file** is set.
- ** – the configs for CKIPWS, passed to *ckip_classic.ini.create_ws_ini()*, ignored if **ws_ini_file** is set.

Danger: Never instance more than one object of this class!

apply(*text*)

Parse a sentence.

Parameters **text** (*str*) – the input sentence.

Returns *str* – the output sentence.

Hint: One may also call this method as `__call__()`.

apply_list(*ilist*)

Parse a list of sentences.

Parameters **ilist** (*List[str]*) – the list of input sentences.

Returns *List[str]* – the list of output sentences.

apply_file(*ifile*, *ofile*)

Parse a file.

Parameters

- **ifile** (*str*) – the input file.
- **ofile** (*str*) – the output file (will be overwritten).

6.4 ckip_classic.ws module

class ckip_classic.ws.**CkipWs**(*, *logger=False*, *ini_file=None*, *lex_list=None*, ***kwargs*)

Bases: object

The CKIP word segmentation driver.

Parameters

- **logger** (*bool*) – enable logger.
- **lex_list** (*Iterable*) – passed to *ckip_classic.ini.create_ws_lex()* overridden **lex_file** for *ckip_classic.ini.create_ws_ini()*.
- **ini_file** (*str*) – the path to the INI file.

Other Parameters ** – the configs for CKIPWS, passed to *ckip_classic.ini.create_ws_ini()*, ignored if **ini_file** is set.

Danger: Never instance more than one object of this class!

apply(*text*)

Parse a sentence.

Parameters **text** (*str*) – the input sentence.

Returns *str* – the output sentence.

Hint: One may also call this method as __call__().

apply_list(*ilist*)

Parse a list of sentences.

Parameters **ilist** (*List[str]*) – the list of input sentences.

Returns *List[str]* – the list of output sentences.

apply_file(*ifile*, *ofile*, *uwfile=''*)

Segment a file.

Parameters

- **ifile** (*str*) – the input file.
- **ofile** (*str*) – the output file (will be overwritten).
- **uwfile** (*str*) – the unknown word file (will be overwritten).

CHAPTER

SEVEN

INDEX

CHAPTER

EIGHT

MODULE INDEX

PYTHON MODULE INDEX

C

INDEX

A

C

M