clstk Documentation

Nisarg Jhaveri

Jun 02, 2018

Getting Started

`clstk` is a free and open source tool-kit for cross-lingual summarization (CLS). The tool-kit is intended for the use of both, developers and researchers working on cross-lingual summarization. End-users wanting to use cross-lingual summarization in real-world applications can also benefit from the tool-kit.

The tool-kit currently contains implementation of three CLS methods along with bootstrap code and other modules required for CLS. We encourage developers to contribute more methods in the tool-kit.

CHAPTER 1

Installation

1.1 Clone repository

```
$ git clone https://github.com/nisargjhaveri/clstk
```

1.2 Python dependencies

The dependencies are listed in `requirements.txt`.

To install all the dependencies, run `pip` as followed.

```
$ pip install --upgrade -r requirements.txt
```

Also install nltk packages called `stopwords` and `punkt`.

```
$ python -m nltk.downloader stopwords punkt -d $NLTK_DATA
```

1.3 Setup CLUTO (optional)

http://glaros.dtc.umn.edu/gkhome/cluto/cluto/download

This is required if you want to use "linBilmes" summarizer.

Set an environment variable `CLUTO_BIN_PATH` with the path of directory containing `vcluster` binary file.

1.4 Setup ROUGE 1.5.5 (optional)

https://github.com/nisargjhaveri/ROUGE-1.5.5-unicode

This is required only if you plan to evaluate the summaries using ROUGE score.

Obtain and setup ROUGE 1.5.5 according to the instructions there.

Set an environment variable `ROUGE_HOME` with the path to ROUGE root directory, the one containing `ROUGE-1.5.5.pl` file.

1.5 Setup dependencies for TQE (optional)

https://github.com/nisargjhaveri/tqe

Install dependencies for `tqe` module according to the details provided in the link above.

1.6 Setup NeuralTextSimplification (optional)

https://github.com/senisioi/NeuralTextSimplification

Setup system from above URL and set `NTS_OPENNMT_PATH`, `NTS_MODEL_PATH` and `NTS_GPUS` variables accordingly.

CHAPTER 2

Datasets

2.1 DUC 2004 Gujarati Dataset

https://github.com/nisargjhaveri/duc2004-translated

This is a cross-lingual summarization evaluation dataset for English to Gujarati summarization. The dataset can be obtained from the link mentioned above.

You'll also need original DUC 2004 dataset as the above link does not contain source documents due to licensing reasons.

2.2 MultiLing Pilot 2011 Dataset

http://users.iit.demokritos.gr/~ggianna/TAC2011/MultiLing2011.html

This dataset contains parallel document sets in seven languages: English, Arabic, Czech, French, Greek, Hebrew and Hindi. Summaries for each document set is available in all languages, making the dataset usable for cross-lingual summarization evaluation.

The data needs to be cleaned and formatted for the use with `clstk`.

CHAPTER 3

Usage

3.1 Summarize

`sum.py` is used to summarize a document set. It allows selecting specific CLS method and parameters for that particular method.

```
$ python sum.py --help
usage: sum.py [-h] {linBilmes,coRank,simFusion} ...

Automatically summarize a set of documents

optional arguments:
  -h, --help            show this help message and exit

methods:
  Summarization method

  {linBilmes,coRank,simFusion}
```

The following command shows help for selected method.

```
$ python sum.py {method} --help
```

Following is the common pattern to run a CLS method on one document set.

```
$ python sum.py {method} [options] {source_directory}
```

All files stored in the directory `source_directory` are read and treated as a part of document set to summarize. The files are expected to be plain text files.

3.1.1 Required arguments

source_directory Directory containing a set of files to be summarized.

3.1.2 Common options

Here is a list of common optional arguments across all CLS methods.

-h, --help show this help message and exit

-v, --verbose Show verbose information messages

--no-colors Don't show colors in verbose log

-s N, --size N Maximum size of the summary

-w, --words Caluated size as number of words instead of characters

--source-lang lang Two-letter language code of the source documents language. Defaults to *en*

-l lang, --target-lang lang Two-letter language code to generate cross-lingual summary. Defaults to source language.

3.2 Evaluate

Another script called `evaluate.py` is used to run and evaluate CLS methods over a CLS evaluation dataset.

Similar to `sum.py`, this script also needs the CLS method as first argument and other argument follows depending on the selected method.

```
$ python evaluate.py {method} [options] {source_path} {models_path} {summaries_path}
```

3.2.1 Required arguments

source_path Directory containing all the source files to be summarized. Each set of documents are expected to be in different directories inside this path.

models_path Directory containing all the model summaries. Each set of summaires are expected to be in different directory inside this path, having the same name as the corresponding directory in the source directory.

summaries_path Directory to store the generated summaries. The directory will be created if not already exists.

3.2.2 Common options

-h, --help show this help message and exit

--only-rouge Do not run summarizer. Only compule ROUGE score for existing summaries in summaries_path

-s N, --size N Maximum size of the summary

-w, --words Caluated size as number of words instead of characters

--source-lang lang Two-letter language code of the source documents language. Defaults to *en*

-l lang, --target-lang lang Two-letter language code to generate cross-lingual summary. Defaults to source language.

CHAPTER 4

Core

The core contains the bootstrap code for summarization needs. The core provides:

- A common standard structure for documents and summaries to ensure interoperability between different components.
- Utilities for loading document sets into the common structure.
- Common utilities on document sets, documents and sentences, for example sentence splitting, tokenization, etc.

4.1 `Sentence` class

class clstk.sentence.**Sentence**(*sentenceText*)
Bases: object

Class to represent a single sentence

__init__(*sentenceText*)
Set sentence text and translated text

Parameters **sentenceText** – sentence text

setText(*sentenceText*)
Set text for the sentence

Parameters **sentenceText** – sentence text

getText()
Get sentence text

Returns sentence text

setTranslation(*translation*)
Set translated text

Parameters **translation** – translated text

getTranslation()
Get translated text

The translated text defaults to sentence text

Returns translated text

setVector(*vector*)
Set sentence vector

Parameters **vector** – sentence vector

getVector()
Get sentence vector

Returns sentence vector

setTranslationVector(*vector*)
Set sentence vector for translated text

Parameters **vector** – sentence vector

getTranslationVector()
Get sentence vector for translated text

Returns sentence vector

setExtra(*key*, *value*)
Set extra key-value pair

Parameters

- **key** – key for the stored value
- **value** – value to store

getExtra(*key*, *default=None*)
Get extra value from key

Parameters

- **key** – key for the stored value
- **default** – default value if key not found

charCount()
Get character count for translated text

Returns Number of character in translated text

tokenCount()
Get token count for translated text

Returns Number of tokens in translated text

__weakref__
list of weak references to the object (if defined)

4.2 SentenceCollection class

class clstk.sentenceCollection.**SentenceCollection**
Bases: object

Class to store a colelction of sentences.

Also proivdes several common operations on the collection.

__init__()
Initialize the collection

setSourceLang(*lang*)
Set source language for the colelction

Parameters **lang** – two-letter code for source language

setTargetLang(*lang*)
Set target language for the colelction

Parameters **lang** – two-letter code for target language

addSentence(*sentence*)
Add a sentence to the colelction

Parameters **sentence** – sentence to be added

addSentences(*sentences*)
Add sentences to the colelction

Parameters **sentences** – list of sentence to be added

See also:

clstk.sentenceCollection.SentenceCollection.addSentence()

getSentences()
Get list of sentences in the collection

Returns list of sentences

getSentenceVectors()
Get list of sentence vectors for sentences in the collection

Returns `np.array` containing sentence vectors

getTranslationSentenceVectors()
Get list of sentence vectors for translations of sentences in the collection

Returns `np.array` containing sentence vectors

generateSentenceVectors()
Generate sentence vectors

generateTranslationSentenceVectors()
Generate sentence vectors for translations

translate(*sourceLang*, *targetLang*, *replaceOriginal=False*)
Translate sentences

Parameters

- **sourceLang** – two-letter code for source language
- **targetLang** – two-letter code for target language
- **replaceOriginal** – Replace source text with translation if `True`. Used for early-translation

simplify(*sourceLang*, *replaceOriginal=False*)
Simplify sentences

Parameters

- **sourceLang** – two-letter code for language
- **replaceOriginal** – Replace source sentences with simplified sentences. Used for early-simplify.

__weakref__
list of weak references to the object (if defined)

4.3 Corpus class

class clstk.corpus.**Corpus**(*dirname*)
Bases: *clstk.sentenceCollection.SentenceCollection*

Class for source documents. Contains utilities for loading document set.

__init__(*dirname*)
Initialize the class

Parameters **dirname** – Directory from where source documents are to be loaded

load(*params*, *translate=False*, *replaceWithTranslation=False*, *simplify=False*, *replaceWithSimplified=False*)
Load source docuement set

Parameters

- **params** – dict containing different params including sourceLang and targetLang.
- **translate** – Whether to translate sentences to target language
- **replaceWithTranslation** – Whether to replace source sentences with translation
- **simplify** – Whether to simplify sentences
- **replaceWithSimplified** – Whether to replace source sentences with simplified sentences

4.4 Summary class

class clstk.summary.**Summary**
Bases: *clstk.sentenceCollection.SentenceCollection*

charCount()
Get total number of character in all the sentences

tokenCount()
Get total number of tokens in all the sentences

getSummary()
Get printable summary generated from source text

getTargetSummary()
Get printable summary generated from translated text

CHAPTER 5

Utils

Different utilities

5.1 nlp utils

clstk.utils.nlp.**getSentenceSplitter**()
Get sentence splitter function

Returns A function which takes a string and return list of sentence as strings.

clstk.utils.nlp.**getTokenizer**(*lang*)
Get tokenizer for a given language

Parameters **lang** – language

Returns tokenizer, which takes a sentence as string and returns list of tokens

clstk.utils.nlp.**getDetokenizer**(*lang*)
Get detokenizer for a given language

Parameters **lang** – language

Returns detokenizer, which takes list of tokens and returns a sentence as string

clstk.utils.nlp.**getStemmer**()
Get stemmer. For now returns Porter Stemmer

Returns stemmer, which takes a token and returns its stem

clstk.utils.nlp.**getStopwords**(*lang*)
Get list of stopwords for a given language

Parameters **lang** – language

Returns list of stopwords including common puncuations

5.2 `ProgressBar` class

class clstk.utils.progress.**ProgressBar**(*totalCount*)
Bases: object

Class to manage and show pretty progress-bar in the console

__init__(*totalCount*)
Initialize the progressbar

Parameters **totalCount** – Total items to be processed

done(*doneCount*)
Move progressbar ahead

Parameters **doneCount** – Out of totalCount, this many have been processed

complete()
Complete progress

__weakref__
list of weak references to the object (if defined)

CHAPTER 6

Evaluate

Module containing ROUGE implementations.

6.1 RougeScore

Python implementation of ROUGE score.

Taken and adopted from:

- https://github.com/miso-belica/sumy/blob/master/sumy/evaluation/rouge.py
- https://github.com/google/seq2seq/blob/master/seq2seq/metrics/rouge.py

class clstk.evaluation.rougeScore.**RougeScore**(*tokenizer=None*, *stemmer=None*)
Bases: object

Implementation of ROUGE score.

__init__(*tokenizer=None*, *stemmer=None*)
x.__init__(...) initializes x; see help(type(x)) for signature

rouge_n(*summary*, *model_summaries*, *n=2*)
Computes ROUGE-N of two text collections of sentences.

rouge_l_sentence_level(*evaluated_sentences*, *reference_sentences*)
Computes ROUGE-L (sentence level) of two text collections of sentences.

rouge_l_summary_level(*evaluated_sentences*, *reference_sentences*)
Computes ROUGE-L (summary level) of two text collections of sentences.

rouge(*hyp_refs_pairs*, *print_all=False*)
Calculates and prints average rouge scores for a list of hypotheses and references

Parameters

- **hyp_refs_pairs** – List containing pairs of path to summary and list of paths to reference summaries

- **print_all** – Print every evaluation along with averages

__weakref__
list of weak references to the object (if defined)

6.2 ExternalRougeScore

Integration with external ROUGE tool-kit.

We recommend the use of https://github.com/nisargjhaveri/ROUGE-1.5.5-unicode

`ROUGE_HOME` variable needs to be set to run this.

class clstk.evaluation.externalRougeScore.**ExternalRougeScore**
Bases: object

Integration with external ROUGE tool-kit.

rouge(*summaryRefsList*)
Runs external ROUGE-1.5.5 and prints results

Parameters **summaryRefsList** – List containing pairs of path to summary and list of paths to reference summaries

__weakref__
list of weak references to the object (if defined)

CHAPTER 7

Translate

Implementation of different translators.

In general you should not need to use these directly.

See also:

clstk.sentenceCollection.SentenceCollection.translate() and *clstk.corpus.Corpus.load()*

7.1 googleTranslate

Translate using Google Translate.

To use this, environmental variable `GOOGLE_APPLICATION_CREDENTIALS` needs to be set with file continaining your key for Google Cloud account.

See https://cloud.google.com/translate/docs/reference/libraries

`clstk.translate.googleTranslate.`**`translate`**(*text*, *sourceLang*, *targetLang*)

Translate text

Parameters

- **`text`** – Text, each line contains one sentence
- **`sourceLang`** – Two-letter code for source language
- **`targetLang`** – Two-letter code for target language

Returns translated text and list of translated sentences

Return type (translation, sentences)

7.2 googleTranslateWeb

DO NOT use this for commercial purpuses

clstk.translate.googleTranslateWeb.**translate**(*text*, *sourceLang*, *targetLang*, *sentencePerLine=True*)

Translate text

Parameters

- **text** – Text, each line contains one sentence
- **sourceLang** – Two-letter code for source language
- **targetLang** – Two-letter code for target language

Returns translated text and list of translated sentences

Return type (translation, sentences)

CHAPTER 8

Simplify

Simplify sentences using different methods.

In general you should not need to use these directly.

See also:

clstk.sentenceCollection.SentenceCollection.simplify() and *clstk.corpus.Corpus.load()*

8.1 neuralTextSimplification

Neural Text Simplification.

You need to set `NTS_OPENNMT_PATH`, `NTS_MODEL_PATH` and `NTS_GPUS` environmental variables to use this.

`clstk.simplify.neuralTextSimplification.`**`simplify`**(*sentences*, *lang*)

Simplify sentences using NTS

Parameters

- **`sentences`** – List of sentence
- **`lang`** – Language of sentences

Returns List of simplified sentences

CHAPTER 9

Translation Quality Estimation

Estimate translation quality

9.1 qualityEstimation

Translation Quality Estimation

Setup dependencies for TQE to use this. https://github.com/nisargjhaveri/tqe

You also need to train model using the said tqe system.

`clstk.qualityEstimation.qualityEstimation.`**`estimate`**(*sentenceCollection*, *modelPath*)

Estimate translation quality for each sentence in collection. It sets an extra value with key `qeScore` on each sentence.

Parameters **`sentenceCollection`** (*clstk.sentenceCollection.SentenceCollection*) – SentenceCollection to estimate quality

See also:

clstk.sentence.Sentence.getExtra()

Python Module Index

C

Index

Symbols

A

C

D

E

G

L

P

R

S

T