apertium-python Documentation

Andi Qu

Nov 24, 2018

Contents

1 Contents 1

2 Apertium 3

3 Analysis 5

4 Generation 7

5 Translation 9

6 Indices and tables 11

Python Module Index 13

CHAPTER 1

Contents

1.1 Usage

1.1.1 Analysis

Performing Morphological Analysis

Method 1: One can create `Analyzer` objects on which the `analyze()` method can be run.

```
In [1]: import apertium
In [2]: a = apertium.Analyzer('en')
In [3]: a.analyze('cats')
Out[3]: [cats/cat<n><pl>, ./.<sent>]
```

Method 2: Alternatively, the library provides an option to directly run the `analyze()` method.

```
In [1]: import apertium
In [2]: apertium.analyze('en', 'cats')
Out[2]: cats/cat<n><pl>
```

1.1.2 Generation

Performing Morphological Generation

Method 1: Just like the `Analyzer`, One can create `Generator` objects on which the `generate()` method can be run::

```
In [1]: import apertium
In [2]: g = apertium.Generator('en')
In [3]: g.generate('^cat<n><pl>$')
Out[3]: 'cats'
```

Method 2: Running `generate()` directly::

```
In [1]: import apertium
In [2]: apertium.generate('en', '^cat<n><pl>$')
Out[2]: 'cats'
```

1.1.3 Installing more modes from other language data

One can also install modes by providing the path to the lang-data using this simple function::

```
In [1]: import apertium
In [2]: apertium.append_pair_path('..')
```

1.1.4 Translation

Performing Translations::

```
In [1]: import apertium
In [2]: t = apertium.Translator('eng', 'spa')
In [3]: t.translate('cats')
Out[3]: 'Gatos'
```

1.2 About

1.2.1 Introduction

- The code-base is in development for the GSoC '18 project called **Apertium API in Python**.
- The Apertium core modules are written in C++.
- This project is an attempt to make the Apertium modules available in python, which because of it's simplicity is more appealing to users.

1.2.2 About the Existing Code Base

- The existing code base has the subprocess implementation of the basic functions of Apertium.
- A branch called `windows` has the implementation for the `windows` support and will soon be available on master. Detailed instructions can be found here

1.2.3 Contribute

- Issue Tracker: https://www.github.com/apertium/apertium-python/issues
- Source Code: https://www.github.com/apertium/apertium-python

CHAPTER 2

Apertium

exception apertium.**ModeNotInstalled**

apertium.**append_pair_path**(*pair_path*)

Parameters **pair_path** (*str*) –

CHAPTER 3

Analysis

class apertium.analysis.**Analyzer**(*lang*)

analyzer_cmds
Dict[str, List[List[str]]]

lang
str

analyze(*in_text*, *formatting='txt'*)
Runs apertium to analyze the input

Parameters

- **in_text** (*str*) –
- **formatting** (*str*) –

Returns List[LexicalUnit]

apertium.analysis.**analyze**(*lang*, *in_text*, *formatting='txt'*)

Parameters

- **lang** (*str*) –
- **in_text** (*str*) –
- **formatting** (*str*) –

Returns List[LexicalUnit]

CHAPTER 4

Generation

class apertium.generation.**Generator**(*lang*)

generation_cmds
Dict[str, List[List[str]]]

lang
str

generate(*in_text*, *formatting='none'*)

Parameters

- **in_text** (*str*) –
- **formatting** (*str*) –

Returns Union[str, List[str]]

apertium.generation.**generate**(*lang*, *in_text*, *formatting='none'*)

Parameters

- **lang** (*str*) –
- **in_text** (*str*) –
- **formatting** (*str*) –

Returns Union[str, List[str]]

CHAPTER 5

Translation

class apertium.translation.**Translator**(*l1*, *l2*)

translation_cmds
Dict[Tuple[str, str], List[List[str]]]

l1
str

l2
str

translate(*text*, *mark_unknown=False*, *format=None*, *deformat='txt'*, *reformat='txt'*)

Parameters

- **text** (*str*) –
- **mark_unknown** (*bool*) –
- **format** (*Optional[str]*) –
- **deformat** (*str*) –
- **reformat** (*str*) –

Returns str

apertium.translation.**translate**(*l1*, *l2*, *text*, *mark_unknown=False*, *format=None*, *deformat='txt'*, *reformat='txt'*)

Parameters

- **text** (*str*) –
- **mark_unknown** (*bool*) –
- **format** (*Optional[str]*) –
- **deformat** (*str*) –

- **reformat** (*str*) –

Returns str

CHAPTER 6

Indices and tables

- genindex
- modindex
- search

Python Module Index

a

apertium, 3
apertium.analysis, 5
apertium.generation, 7
apertium.translation, 9

Index

A

analyze() (apertium.analysis.Analyzer method), 5
analyze() (in module apertium.analysis), 5
Analyzer (class in apertium.analysis), 5
analyzer_cmds (apertium.analysis.Analyzer attribute), 5
apertium (module), 3
apertium.analysis (module), 5
apertium.generation (module), 7
apertium.translation (module), 9
append_pair_path() (in module apertium), 3

G

generate() (apertium.generation.Generator method), 7
generate() (in module apertium.generation), 7
generation_cmds (apertium.generation.Generator attribute), 7
Generator (class in apertium.generation), 7

L

l1 (apertium.translation.Translator attribute), 9
l2 (apertium.translation.Translator attribute), 9
lang (apertium.analysis.Analyzer attribute), 5
lang (apertium.generation.Generator attribute), 7

M

ModeNotInstalled, 3

T

translate() (apertium.translation.Translator method), 9
translate() (in module apertium.translation), 9
translation_cmds (apertium.translation.Translator attribute), 9
Translator (class in apertium.translation), 9