
bosonnlp.py Documentation

Release 0.11.1

BosonData

Nov 26, 2018

Contents

1	3
2	5
3 API	7
4 Exceptions	17
Python Module Index	19

BosonNLP HTTP API SDK

CHAPTER 1

bosonnlp [GitHub](#) [PyPI](#) *pip* :

```
$ pip install bosonnlp
```

bosonnlp MIT

CHAPTER 2

```
>>> from bosonnlp import BosonNLP
>>> nlp = BosonNLP('YOUR_API_TOKEN')
>>> nlp.sentiment('')
[[0.8758192096636473, 0.12418079033635264]]
```

BosonNLP BosonNLP HTTP API

```
class bosonnlp.BosonNLP(token, bosonnlp_url='https://api.bosonnlp.com', compress=True, session=None, timeout=60)
```

BosonNLP HTTP API

Parameters

- **token** (*string*) – API Token
- **bosonnlp_url** (*string*) – BosonNLP HTTP API URL *https://api.bosonnlp.com*
- **compress** (*bool*) – 10K True
- **timeout** (*int*) – HTTP 60

```
sentiment (contents, model='general')
```

BosonNLP

Parameters

- **contents** (*string or sequence of string*) –
- **model** (*string*) –

Returns

Raises *HTTPError* API

```
>>> import os
>>> nlp = BosonNLP(os.environ['BOSON_API_TOKEN'])
>>> nlp.sentiment(' ', model='food')
[[0.9991737012037423, 0.0008262987962577828]]
>>> nlp.sentiment([' ', ''], model='food')
[[0.9991737012037423, 0.0008262987962577828],
 [9.940036427291687e-08, 0.9999999005996357]]
```

```
convert_time (content, basetime=None)
```

BosonNLP

Parameters

- **content** (*string*) –
- **basetime** (*int or datetime.datetime*) – *datetime*

Raises *HTTPError* API

Returns

```
>>> import os
>>> nlp = BosonNLP(os.environ['BOSON_API_TOKEN'])
>>> _json_dumps(nlp.convert_time("2013"))
'{"timestamp": "2013-02-28 16:30:29", "type": "timestamp"}'
>>> import datetime
>>> _json_dumps(nlp.convert_time("83", datetime.datetime(2015, 9, 1)))
'{"timespan": ["2015-09-02 20:00:00", "2015-09-03 15:00:00"], "type":
↪ "timespan_0"}'
```

classify (*contents*)

BosonNLP

Parameters **contents** (*string or sequence of string*) –

Returns

Raises *HTTPError* API

```
>>> import os
>>> nlp = BosonNLP(os.environ['BOSON_API_TOKEN'])
>>> nlp.classify('')
[5]
>>> nlp.classify(['',
...              '',
...              'Facebook'])
[5, 4, 8]
```

suggest (*word, top_k=None*)

BosonNLP

Parameters

- **word** (*string*) –
- **top_k** (*int*) – 10 100

Returns

Raises *HTTPError* API

```
>>> import os
>>> nlp = BosonNLP(os.environ['BOSON_API_TOKEN'])
>>> nlp.suggest('', top_k=2)
[[1.0, '/ns'], [0.7493540460397998, '/ns']]
```

extract_keywords (*text, top_k=None, segmented=False*)

BosonNLP

Parameters

- **text** (*string*) –
- **top_k** (*int*) – 100
- **segmented** (*bool*) – *False* *text* *True*

Returns**Raises** `HTTPError` API

```
>>> import os
>>> nlp = BosonNLP(os.environ['BOSON_API_TOKEN'])
>>> nlp.extract_keywords(' ', top_k=2)
[[0.8391345017584958, ''], [0.3802418301341705, '']]
```

depparser (*contents*)

BosonNLP

Parameters **contents** (*string or sequence of string*)-**Returns****Raises** `HTTPError` API

```
>>> import os
>>> nlp = BosonNLP(os.environ['BOSON_API_TOKEN'])
>>> nlp.depparser(' ')
[{'head': [2, 2, -1],
  'role': ['TMP', 'SBJ', 'ROOT'],
  'tag': ['NT', 'NN', 'VA'],
  'word': ['', '', '']}]
>>> nlp.depparser([' ', ' '])
[{'head': [2, 2, -1],
  'role': ['TMP', 'SBJ', 'ROOT'],
  'tag': ['NT', 'NN', 'VA'],
  'word': ['', '', '']},
 {'head': [1, 2, -1],
  'role': ['DEC', 'NMOD', 'ROOT'],
  'tag': ['VA', 'DEC', 'NN'],
  'word': ['', '', '']}]
```

ner (*contents, sensitivity=None, segmented=False, space_mode='3'*)

BosonNLP

Parameters

- **contents** (*string or sequence of string*)-
- **sensitivity** (*int 3*)- 1 5
- **segmented** (*boolean False*)-
- **space_mode** (*int, 0-3 3*)-

Returns**Raises** `HTTPError` API

```
>>> import os
>>> nlp = BosonNLP(os.environ['BOSON_API_TOKEN'])
>>> nlp.ner(' ', sensitivity=2)
[{'entity': [[0, 2, 'product_name'],
             [2, 3, 'job_title'],
             [3, 4, 'person_name']],
  'tag': ['ns', 'n', 'n', 'nr'],
  'word': ['', '', '', '']}]
```

```
>>> nlp.ner([' ', 'XP'])
[{'entity': [[0, 2, 'product_name'],
            [2, 3, 'job_title'],
            [3, 4, 'person_name']],
  'tag': ['ns', 'n', 'n', 'nr'],
  'word': ['', '', '', '']},
 {'entity': [[0, 2, 'product_name'],
            [3, 4, 'time']],
  'tag': ['nz', 'nx', 'nl', 't', 'ad', 'v'],
  'word': ['', 'XP', '', '', '', '']}
```

tag (*contents*, *space_mode=0*, *oov_level=3*, *t2s=0*, *special_char_conv=0*)
BosonNLP

Parameters

- **contents** (*string* or *sequence of string*)–
- **space_mode** (*int*, 0–3)–
- **oov_level** (*int*, 0–4)–
- **t2s** (*int*, 0–1)–
- **special_char_conv** (*int*, 0–1)– Tab

Returns

Raises `HTTPError` API

<http://docs.bosonnlp.com/tag.html>

```
>>> import os
>>> nlp = BosonNLP(os.environ['BOSON_API_TOKEN'])
```

```
>>> result = nlp.tag(' ')
>>> _json_dumps(result)
'[{"tag": ["ns", "n", "n", "nr"], "word": ["", "", "", ""]}']
```

```
>>> format_tag_result = lambda tagged: ' '.join('%s/%s' % x for x in_
↳ zip(tagged['word'], tagged['tag']))
>>> result = nlp.tag(" ")
>>> format_tag_result(result[0])
'/ns /n /n /nr'
```

```
>>> result = nlp.tag(" ", space_mode=2)
>>> format_tag_result(result[0])
'/ns /n /n /w /nr'
```

```
>>> result = nlp.tag(['57', '""'], oov_level=0)
>>> format_tag_result(result[0])
'/ns /v /n /n /vi /n /v /v 57/m /q'
```

```
>>> format_tag_result(result[1])
'"/wyz /n /a "/wyy /d /pbei /v'
```

summary (*title*, *content*, *word_limit=0.3*, *not_exceed=False*)
BosonNLP

Parameters

- **title** (*unicode*) –
- **content** (*unicode*) –
- **word_limit** (*float or int*) – float 0.0-1.0 int

Note: 1

- **not_exceed** (*bool False*) –

Returns

Raises *HTTPError* API

<http://docs.bosonnlp.com/summary.html>

```
>>> import os
>>> nlp = BosonNLP(os.environ['BOSON_API_TOKEN'])
```

```
>>> content = (
    '1022'
    'TVCTO'
    ''
    ''
    ''
    'TVTVOS')
>>> title = 'TVCTO'
```

```
>>> nlp.summary(title, content, 0.1)
1022TVCTO
```

cluster (*contents, task_id=None, alpha=None, beta=None, timeout=1800*)

BosonNLP

Parameters

- **contents** (*sequence of string or sequence of (_id, text) or sequence of {'_id': _id, 'text': text}*) – (_id, text) {'_id': _id, 'text': text} _id _id
- **task_id** (*string*) – task_id
- **alpha** (*float*) – 0.8 cluster
- **beta** (*float*) – 0.45 cluster
- **timeout** (*float*) – 1800 30

Returns

Raises *HTTPError* - API

TaskError -

TimeoutError - *timeout*

```
>>> import os
>>> nlp = BosonNLP(os.environ['BOSON_API_TOKEN'])
>>> nlp.cluster(['', '', '', '',
...            '', '', ''])
[{'_id': 0, 'list': [0, 1], 'num': 2}]
```

create_cluster_task (*contents=None, task_id=None*)

ClusterTask

Parameters

- **contents** (None or sequence of string or sequence of (*_id*, *text*) or sequence of {'_id': *_id*, 'text': *text*}) – (*_id*, *text*) {'_id': *_id*, 'text': *text*} *_id* *_id* None
- **task_id** (*string*) – None *task_id*

Raises

HTTPError - API *contents* None *push*

Returns *ClusterTask*

comments (*contents, task_id=None, alpha=None, beta=None, timeout=1800*)

BosonNLP

Parameters

- **contents** (sequence of string or sequence of (*_id*, *text*) or sequence of {'_id': *_id*, 'text': *text*}) – (*_id*, *text*) {'_id': *_id*, 'text': *text*} *_id* *_id*
- **task_id** (*string*) – None *task_id*
- **alpha** (*float*) – 0.8 cluster
- **beta** (*float*) – 0.45 cluster
- **timeout** (*float*) – 1800 30

Returns

Raises *HTTPError* - API

TaskError -

TimeoutError - *timeout*

```
>>> import os
>>> nlp = BosonNLP(os.environ['BOSON_API_TOKEN'])
>>> nlp.comments(['', '', '', '',
...             '', '', ''] * 2)
[{'_id': 0, 'list': [['', 3], ['', 10]],
  'num': 2, 'opinion': ''},
 {'_id': 1, 'list': [['', 4], ['', 11]],
  'num': 2, 'opinion': ''},
 {'_id': 2, 'list': [['', 5], ['', 12]],
  'num': 2, 'opinion': ''},
 {'_id': 3, 'list': [['', 6], ['', 13]],
  'num': 2, 'opinion': ''}]
```

create_comments_task (*contents=None, task_id=None*)

CommentsTask

Parameters

- **contents** (None or sequence of string or sequence of (*_id*, *text*) or sequence of {'_id': *_id*, 'text': *text*}) – (*_id*, *text*) {'_id': *_id*, 'text': *text*} *_id* *_id* None
- **task_id** (*string*) – None *task_id*

Raises

HTTPError - API *contents* None *push*

Returns *CommentsTask*

class bosonnlp.**ClusterTask** (*nlp*, *contents=None*, *task_id=None*)
create_cluster_task

```
>>> import os
>>> nlp = BosonNLP(os.environ['BOSON_API_TOKEN'])
>>> cluster = nlp.create_cluster_task()
```

push

```
>>> cluster.push(['', '', '', ''])
>>> cluster.push([''])
>>> cluster.push(['', ''])
```

analysis

```
>>> cluster.analysis()
```

wait_until_complete

```
>>> cluster.wait_until_complete(2 * 60) # 2
```

status

```
>>> cluster.status()
'done'
```

result

```
>>> cluster.result()
[{'_id': '9e90c56e-f1bb-4605-b995-304af733207a',
  'list': ['9e90c56e-f1bb-4605-b995-304af733207a',
           'a3feff6b-6d1b-4f46-a2d8-0eea25c7f17f'],
  'num': 2}]
```

clear

```
>>> cluster.clear()
True
```

Parameters *nlp* – *BosonNLP* *cluster*

analysis (*alpha=None*, *beta=None*)

Parameters

- **alpha** (*float*) – 0.8 cluster
- **beta** (*float*) – 0.45 cluster

Raises *HTTPError* - API

clear ()

Returns

Raises *HTTPError* - API

push (*contents*)**Parameters**

contents (sequence of string or sequence of (*_id*, *text*) or sequence of {'_id': *_id*, 'text': *text*}) - (*_id*, *text*) {'_id': *_id*, 'text': *text*} *_id* *_id*

Raises *HTTPError* - API

result ()**Returns**

Raises *HTTPError* - API

status ()**Returns**

received	
running	
done	

Raises *HTTPError* - API

TaskNotFoundError -

TaskError -

wait_until_complete (*timeout=None*)**Parameters**

timeout (*float*) - None

Raises *HTTPError* - API

TaskNotFoundError -

TaskError -

TimeoutError - *timeout*

class bosonnlp.**CommentsTask** (*nlp*, *contents=None*, *task_id=None*)

create_comments_task

```
>>> import os
>>> nlp = BosonNLP(os.environ['BOSON_API_TOKEN'])
>>> comments = nlp.create_comments_task()
```

push

```
>>> comments.push(['', '', '', ''] * 2)
>>> comments.push([''] * 2)
>>> comments.push(['', ''] * 2)
```

analysis

```
>>> comments.analysis()
```

wait_until_complete

```
>>> comments.wait_until_complete(2 * 60) # 2
```

status

```
>>> comments.status()
'done'
```

result

```
>>> comments.result()
[{'_id': 0,
  'list': [['', '19c248e3-605b-4785-8785-ccd2d1b034cc'],
           ['', '576d1d08-ff02-4bc5-9edf-fbc7a6915e44']],
  'num': 2,
  'opinion': ''},
 {'_id': 2,
  'list': [['', 'a75e24bd-8597-4865-8254-2e9cab229770'],
           ['', '47db4d92-6328-45cd-98f8-099691e82c07']],
  'num': 2,
  'opinion': ''},
 {'_id': 4,
  'list': [['', 'da0cd13f-4f13-4476-a541-6214df3b4dd9'],
           ['', '89aecf45-4b78-4522-9ed2-ea76ed552f24']],
  'num': 2,
  'opinion': ''},
 {'_id': 5,
  'list': [['', 'a5c1f6a9-b6a6-4877-b073-0b59bc67fa48'],
           ['', '9d38ecab-5860-44e9-9e3c-9ad4d4ed3547']],
  'num': 2,
  'opinion': ''}]
```

clear

```
>>> comments.clear()
True
```

Parameters *nlp* – *BosonNLP* *comments*

analysis (*alpha=None*, *beta=None*)

Parameters

- **alpha** (*float*) – 0.8 cluster
- **beta** (*float*) – 0.45 cluster

Raises *HTTPError* - API

clear()

Returns

Raises *HTTPError* - API

push (*contents*)

Parameters

contents (sequence of string or sequence of (*_id*, *text*) or sequence of {'_id': *_id*, 'text': *text*}) – (*_id*, *text*) {'_id': *_id*, 'text': *text*} *_id* *_id*

Raises *HTTPError* - API

result()

Returns

Raises *HTTPError* - API

`status()`

Returns

received	
running	
done	

Raises *HTTPError* - API

TaskNotFoundError -

TaskError -

wait_until_complete(*timeout=None*)

timeout(*float*) - None

Parameters

Raises *HTTPError* - API

TaskNotFoundError -

TaskError -

TimeoutError - *timeout*

CHAPTER 4

Exceptions

```
exception bosonnlp.HTTPError (*args, **kwargs)
    An HTTP error occurred.

exception bosonnlp.TaskNotFoundError (*args, **kwargs)

exception bosonnlp.TaskError (*args, **kwargs)

exception bosonnlp.TimeoutError
```


b

bosonnlp, [1](#)

A

`analysis()` (*bosonnlp.ClusterTask method*), 13
`analysis()` (*bosonnlp.CommentsTask method*), 15

B

`BosonNLP` (class in *bosonnlp*), 7
`bosonnlp` (module), 1

C

`classify()` (*bosonnlp.BosonNLP method*), 8
`clear()` (*bosonnlp.ClusterTask method*), 13
`clear()` (*bosonnlp.CommentsTask method*), 15
`cluster()` (*bosonnlp.BosonNLP method*), 11
`ClusterTask` (class in *bosonnlp*), 13
`comments()` (*bosonnlp.BosonNLP method*), 12
`CommentsTask` (class in *bosonnlp*), 14
`convert_time()` (*bosonnlp.BosonNLP method*), 7
`create_cluster_task()` (*bosonnlp.BosonNLP method*), 11
`create_comments_task()` (*bosonnlp.BosonNLP method*), 12

D

`depparser()` (*bosonnlp.BosonNLP method*), 9

E

`extract_keywords()` (*bosonnlp.BosonNLP method*), 8

H

`HTTPError`, 17

N

`ner()` (*bosonnlp.BosonNLP method*), 9

P

`push()` (*bosonnlp.ClusterTask method*), 13
`push()` (*bosonnlp.CommentsTask method*), 15

R

`result()` (*bosonnlp.ClusterTask method*), 14
`result()` (*bosonnlp.CommentsTask method*), 15

S

`sentiment()` (*bosonnlp.BosonNLP method*), 7
`status()` (*bosonnlp.ClusterTask method*), 14
`status()` (*bosonnlp.CommentsTask method*), 15
`suggest()` (*bosonnlp.BosonNLP method*), 8
`summary()` (*bosonnlp.BosonNLP method*), 10

T

`tag()` (*bosonnlp.BosonNLP method*), 10
`TaskError`, 17
`TaskNotFoundError`, 17
`TimeoutError`, 17

W

`wait_until_complete()` (*bosonnlp.ClusterTask method*), 14
`wait_until_complete()` (*bosonnlp.CommentsTask method*), 16