API documentation for keyword extraction

On this page we describe API of Corpus Architect which can be used for extracting keywords from a given text. The method uses reference corpus so it is necessary to provide also language of the text. There are several parameters which can control the process of extraction of keywords.

As response, JSON object is returned. It is dictionary which contains keys keywords, error, length and ref_corp. If a keyword list is returned, then value of keywords key is array of triplets (arrays) in format [word, frequency, keywordness_score], length contains number of tokens in the text and ref_corp contains id of corpus which was used for extracting keywords. If an error is encountered during processing, the array is empty and error key contains error message. Frequency stands for number of occurrences in the reference corpus. Keywordness score = \((\text{frequency in the text} + \text{<simple maths parameter>}) / (\text{frequency in the reference corpus} + \text{<simple maths parameter>})\). It roughly expresses relevance of the word in the text compared to a general text in the same language. The keywords list is sorted by keywordness in descending order.

! You are allowed to make only POST requests.

API parameters

Parameters are given to POST request using JSON dictionary where parameters are keys and their values as dictionary values.

Text (text)

One of three obligatory parameters. The text should be in UTF-8. Since only POST requests are supported, there is no limit for its length.

There is no default value for this parameter.

Passphrase (passphrase)

Now, authentication, is done via supplying a passphrase which must be assigned to a customer which wants to use this API. In this case, we don't need to add a new user account to the Sketch Engine.

There is no default value for this parameter.

Language (language)

You must also provide language of the text. Now, we support several major languages for which we have reference corpora. It is useful to know language of the text for tokenization of it, too. Allowed values are:

- english
- spanish
- german
- czech
- arabic
- chinese-simplified
• french
• hindi
• indonesian
• italian
• japanese
• persian
• portuguese
• russian

Default value = english.

**Simple math parameter** (**simple_maths_n**)

If you give a low number e.g. 1 you will get lower frequency keywords, whereas a higher N will get higher frequency keywords (for further details see SimpleMaths). Values are natural numbers, but usually 1, 10, 100 and 1000 are used.

Default value = 100.

**Attribute** (**attribute**)

Now, only word attribute is supported, otherwise, the input text would have to be lemmatized and PoS tagged which we can’t do for all supported languages. The only permitted value is word.

Default value = word.

**Exclude stop words from keyword list** (**exclude_stop_words**)

If you want to exclude stop words like *a, about, like, the, in, three, during, I, is, it, much, she, there* and other very frequent words, use value true.

Default value = true.

**Only alphanumeric characters** (**alphanumeric**)

If keywords in resulting list should contain only alphanumeric characters, use value true.

Default value = true.

**At least one alphabetic character** (**one_alphabetic**)

If keywords should consist from at least one alphabetic character, use value true.

Default value = true.
Minimal length of keywords (min_length)

With this parameter you may specify minimal length of keywords in the resulting list.

Default value = 1.

Minimal frequency of keywords (min_frequency)

You may limit keywords in list by their frequency in reference corpus.

Default value = 1.

Maximal number of keywords (max_keywords)

You may limit length of keyword list - only first N will be output.

Default value = 100.

Python example using simplejson and urllib2 modules

```python
#!/usr/bin/env python
#coding=utf-8

import urllib2, simplejson

data = simplejson.dumps({
    'text': '''Some long text here...''',
    'language': 'english',
    'passphrase': '...passphrase...',
    'simple_maths_n': 10,
    'attribute': 'word',
    'exclude_stop_words': True,
    'alphanumeric': True,
    'one_alphabetic': True,
    'min_length': 3,
    'max_keywords': 10,
    'min_frequency': 5
    })

req = urllib2.Request("https://beta.sketchengine.co.uk/get_keywords/", data)
opener = urllib2.build_opener()
f = opener.open(req)
obj = simplejson.load(f)
if obj.get('error') == '':
    print 'Length:', obj.get('length', 0)
    print 'Reference corpus:', obj.get('ref_corp', '')
    for k in obj.get('keywords', []):
        print ' %s	%d	%f' % tuple(k)
else:
    print 'Error encountered:', obj.get('error')
```
AJAX example (using jQuery library)

Since modern browsers don’t allow putting content from foreign domains (cross domain AJAX), you may spot a problem when calling the API from browser as follows:

XMLHttpRequest cannot load [URL]. Origin null is not allowed by Access-Control-Allow-Origin

For more info about this issue, read ?HTTP access control.

```javascript
function get_keywords() {
    $.ajax({
        url: 'https://beta.sketchengine.co.uk/get_keywords/',
        async: true,
        beforeSend: function () { $('#output').text('Loading...'); },
        type: 'POST',
        data: JSON.stringify({
            'text': $('#text').val(),
            'passphrase': '...passphrase...',
            'language': $('#language option:selected').val(),
            'simple_maths_n': $('#simple_maths_n').val(),
            'attribute': $('#attribute option:selected').val(),
            'exclude_stop_words': $('#exclude_stop_words').is(':checked'),
            'alphanumeric': $('#alphanumeric').is(':checked'),
            'one_alphabetic': $('#one_alphabetic').is(':checked'),
            'min_length': $('#min_length').val(),
            'max_frequency': $('#min_frequency').val(),
            'max_keywords': $('#max_keywords').val()
        }),
        success: function (data) {
            $('#output').text(data);
        },
        error: function (data, textStatus, errorThrown) {
            $('#output').text(textStatus);
        }
    });
}
```

Add this function inside `<script>` tag and you may use this HTML code with form to call the function above:

```html
<form>
    <textarea id="text"></textarea>
    <table>
        <tr>
            <td><label>Language:</label></td>
            <td><select id="language">
                <option value="english" selected>English</option>
                <option value="german">German</option>
                <option value="czech">Czech</option>
            </select></td>
        </tr>
        <tr>
            <td><label>Attribute:</label></td>
            <td><select id="attribute">
                <option value="word" selected>word</option>
                <option value="lemma">lemma</option>
                <option value="lempos">lempos</option>
            </select></td>
        </tr>
    </table>
</form>
```
First example (?article "Software" from Wikipedia)

Keywords extracted from plain-text of this wiki page.

Length: 3204 Reference corpus: entenent

<table>
<thead>
<tr>
<th>word</th>
<th>frequency</th>
<th>keywordness_score</th>
</tr>
</thead>
<tbody>
<tr>
<td>software</td>
<td>157</td>
<td>225.188706</td>
</tr>
<tr>
<td>hardware</td>
<td>20</td>
<td>50.452344</td>
</tr>
<tr>
<td>microsoft</td>
<td>15</td>
<td>47.498688</td>
</tr>
<tr>
<td>programming</td>
<td>11</td>
<td>28.355465</td>
</tr>
<tr>
<td>application</td>
<td>21</td>
<td>27.617109</td>
</tr>
<tr>
<td>operating</td>
<td>13</td>
<td>27.261984</td>
</tr>
<tr>
<td>applications</td>
<td>13</td>
<td>22.428778</td>
</tr>
<tr>
<td>programs</td>
<td>16</td>
<td>21.760401</td>
</tr>
<tr>
<td>apis</td>
<td>6</td>
<td>19.720981</td>
</tr>
</tbody>
</table>
Second example: Carol's Alice in Wonderland

The API extracts these keywords from the well-known fiction, got from The Gutenberg Project:

Length: 37592 Reference corpus: ententen

<table>
<thead>
<tr>
<th>word</th>
<th>frequency</th>
<th>keywordness_score</th>
</tr>
</thead>
<tbody>
<tr>
<td>alice</td>
<td>403</td>
<td>108.070754</td>
</tr>
<tr>
<td>queen</td>
<td>75</td>
<td>19.313309</td>
</tr>
<tr>
<td>turtle</td>
<td>58</td>
<td>16.052738</td>
</tr>
<tr>
<td>hatter</td>
<td>56</td>
<td>15.879250</td>
</tr>
<tr>
<td>gryphon</td>
<td>55</td>
<td>15.626327</td>
</tr>
<tr>
<td>mock</td>
<td>56</td>
<td>15.195047</td>
</tr>
<tr>
<td>herself</td>
<td>83</td>
<td>15.059740</td>
</tr>
<tr>
<td>rabbit</td>
<td>47</td>
<td>12.957416</td>
</tr>
<tr>
<td>duchess</td>
<td>42</td>
<td>12.123300</td>
</tr>
<tr>
<td>king</td>
<td>63</td>
<td>11.674443</td>
</tr>
<tr>
<td>dormouse</td>
<td>40</td>
<td>11.633408</td>
</tr>
<tr>
<td>gutenberg</td>
<td>33</td>
<td>9.776340</td>
</tr>
<tr>
<td>mouse</td>
<td>39</td>
<td>9.682684</td>
</tr>
<tr>
<td>said</td>
<td>458</td>
<td>9.388244</td>
</tr>
<tr>
<td>tone</td>
<td>40</td>
<td>9.314341</td>
</tr>
<tr>
<td>hare</td>
<td>31</td>
<td>9.130406</td>
</tr>
<tr>
<td>cat</td>
<td>37</td>
<td>9.110194</td>
</tr>
<tr>
<td>march</td>
<td>34</td>
<td>8.817291</td>
</tr>
<tr>
<td>caterpillar</td>
<td>28</td>
<td>8.383761</td>
</tr>
<tr>
<td>project</td>
<td>87</td>
<td>7.818975</td>
</tr>
<tr>
<td>voice</td>
<td>47</td>
<td>6.950982</td>
</tr>
<tr>
<td>began</td>
<td>58</td>
<td>6.877441</td>
</tr>
<tr>
<td>went</td>
<td>83</td>
<td>6.655782</td>
</tr>
<tr>
<td>little</td>
<td>127</td>
<td>6.639094</td>
</tr>
<tr>
<td>round</td>
<td>41</td>
<td>6.589788</td>
</tr>
<tr>
<td>dear</td>
<td>29</td>
<td>6.501213</td>
</tr>
<tr>
<td>replied</td>
<td>29</td>
<td>6.320019</td>
</tr>
<tr>
<td>looked</td>
<td>45</td>
<td>5.883069</td>
</tr>
<tr>
<td>foundation</td>
<td>25</td>
<td>5.784868</td>
</tr>
<tr>
<td>thought</td>
<td>74</td>
<td>5.480465</td>
</tr>
<tr>
<td>soup</td>
<td>18</td>
<td>5.404122</td>
</tr>
<tr>
<td>quite</td>
<td>55</td>
<td>5.359216</td>
</tr>
</tbody>
</table>
Support

In case of problems with or misfunctioning of the API, please, contact us at support@sketchengine.co.uk.