Package 'textcat'

December 4, 2020

Version 1.0-7

Title N-Gram Based Text Categorization

Description Text categorization based on n-grams.

Depends R (>= 3.2.0)

Imports tau (>= 0.0-11), slam

License GPL-2

LazyData yes

NeedsCompilation no

Author Kurt Hornik [aut, cre] (<https://orcid.org/0000-0003-4198-9911>),
Johannes Rauch [aut],
Christian Buchta [aut],
Ingo Feinerer [aut]

Maintainer Kurt Hornik <Kurt.Hornik@R-project.org>

Repository CRAN

Date/Publication 2020-12-04 11:24:41 UTC

R topics documented:

ECIMCI_profiles	*ECI/MCI N-Gram Profiles*

Description

N-gram profile db for 26 languages based on the European Corpus Initiative Multilingual Corpus I.

Usage

```
ECIMCI_profiles
```

Details

This profile db was built by Johannes Rauch, using the ECI/MCI corpus (http://www.elsnet.org/eci.html) and the default options employed by package **textcat**, with all text documents encoded in UTF-8.

The category ids used for the db are the respective IETF language tags (see language in package **NLP**), using the ISO 639-2 Part B language subtags and, for Serbian, the script employed (i.e., "scc-Cyrl" and "scc-Latn" for Serbian written in Cyrillic and Latin script, respectively; all other languages in the profile db are written in Latin script.)

References

S. Armstrong-Warwick, H. S. Thompson, D. McKelvie and D. Petitpierre (1994), Data in Your Language: The ECI Multilingual Corpus 1. In "Proceedings of the International Workshop on Sharable Natural Language Resources" (Nara, Japan), 97–106. https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.44.950

Examples

```
## Languages in the the ECI/MCI profile db:
names(ECIMCI_profiles)
## Key options used for the profile:
attr(ECIMCI_profiles, "options")[c("n", "size", "reduce", "useBytes")]
```

TC_profiles	*TextCat N-Gram Profiles*

Description

TextCat n-gram byte and character profile dbs for language identification.

Usage

```
TC_char_profiles
TC_byte_profiles
```

Details

TextCat (<https://www.let.rug.nl/vannoord/TextCat/>) is a Perl implementation of the Cavnar and Trenkle "N-Gram-Based Text Categorization" technique by Gertjan van Noord which was subsequently integrated into SpamAssassin. It provides byte n-gram profiles for 74 "languages" (more precisely, language/encoding combinations). The C library reimplementation libtextcat (<https://software.wise-guys.nl/libtextcat/>) adds one more non-empty profile.

TC_byte_profiles provides these byte profiles.

TC_char_profiles provides a subset of 56 character profiles obtained by converting the byte sequences to UTF-8 strings where possible.

The category ids are unchanged from the original, and give the full (English) name of the language, optionally combined the name of the encoding script. Note that 'scots' indicates Scots, the Germanic language variety historically spoken in Lowland Scotland and parts of Ulster, to be distinguished from Scottish Gaelic (named 'scots_gaelic' in the profiles), the Celtic language variety spoken in most of the western Highlands and in the Hebrides (see <https://en.wikipedia.org/wiki/Scots_language>).

Examples

```
## Languages in the TC byte profiles:
names(TC_byte_profiles)
## Languages only in the TC byte profiles:
setdiff(names(TC_byte_profiles), names(TC_char_profiles))
## Key options used for the profiles:
attr(TC_byte_profiles, "options")[c("n", "size", "reduce", "useBytes")]
attr(TC_char_profiles, "options")[c("n", "size", "reduce", "useBytes")]
```

---

textcat | *N-Gram Based Text Categorization*

Description

Categorize texts by computing their n-gram profiles, and finding the closest category n-gram profile.

Usage

```
textcat(x, p = textcat::TC_char_profiles, method = "CT", ...,
        options = list())
```

Arguments

x   a character vector of texts, or an R object which can be coerced to this using as.character, or a textcat profile db (see textcat_profile_db) created using the same method and options as p.

p   a textcat profile db. By default, the TextCat character profiles are used (see TC_char_profiles).

method	a character string specifying a built-in method, or a user-defined function for computing distances between n-gram profiles. See `textcat_xdist` for details.
...	options to be passed to the method for computing distances between profiles.
options	a list of such options.

Details

For each given text, its n-gram profile is computed using the options in the category profile db. Then, the distance between this profile and the category profiles is computed, and the text is categorized into the category of the closest profile (if this is not unique, NA is obtained).

Unless the profile db uses bytes rather than characters, the texts in x should be encoded in UTF-8.

References

W. B. Cavnar and J. M. Trenkle (1994), N-Gram-Based Text Categorization. In "Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval", 161–175. https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.53.9367

K. Hornik, P. Mair, J. Rauch, W. Geiger, C. Buchta and I. Feinerer (2013). The textcat Package for n-Gram Based Text Categorization in R. *Journal of Statistical Software*, **52**/6, 1–17. doi: 10.18637/jss.v052.i06.

Examples

```
textcat(c("This is an english sentence.",
          "Das ist ein deutscher satz."))
```

---

textcat_options   *Textcat Options*

Description

Get and set options used for n-gram based text categorization.

Usage

```
textcat_options(option, value)
```

Arguments

option	character string indicating the option to get or set (see **Details**). Can be abbreviated. If missing, all options are returned as a list.
value	Value to be set. If omitted, the current value of the given option is returned.

Details

Currently, the following options are available:

profile_method: A character string or function specifying a method for computing n-gram profiles (see [textcat_profile_db](#)).

Default: "textcat".

profile_options: A list of options to be passed to the method for computing profiles.

Default: none (empty list).

xdist_method: A character string or function specifying a method for computing distances between n-gram profiles (see [textcat_xdist](#)).

Default: "CT", giving the Cavnar-Trenkle out of place measure.

xdist_options: A list of options to be passes to the method for computing distances between profiles.

Default: none (empty list).

---

textcat_profile_db   *Textcat Profile Dbs*

Description

Create n-gram profile dbs for text categorization.

Usage

```
textcat_profile_db(x, id = NULL, method = NULL, ...,
                   options = list(), profiles = NULL)
```

Arguments

x	a character vector of text documents, or an R object of text documents extractable via as.character.
id	a character vector giving the categories of the texts to be recycled to the length of x, or NULL (default), indicating to treat each text document separately.
method	a character string specifying a built-in method, or a user-defined function for computing distances between n-gram profiles, or NULL (default), corresponding to using the method and options used for creating profiles if this is not NULL, or otherwise the current value of **textcat** option profile_method (see [textcat_options](#)).
...	options to be passed to the method for creating profiles.
options	a list of such options.
profiles	a textcat profile db object.

Details

The text documents are split according to the given categories, and n-gram profiles are computed using the specified method, with options either those used for creating profiles if this is not NULL, or by combining the options given in ... and options and merging with the default profile options specified by the **textcat** option profile_options using *exact* name matching. The method and options employed for building the db are stored in the db as attributes "method" and "options", respectively.

There is a c method for combining profile dbs provided that these have identical options. There are also a [method for subscripting and as.matrix and as.simple_triplet_matrix methods to "export" the profiles to a dense matrix or the sparse simple triplet matrix representation provided by package **slam**, respectively.

Currently, the only available built-in method is "textcnt", which has the following options:

n: A numeric vector giving the numbers of characters or bytes in the n-gram profiles.

    Default: 1 : 5.

split: The regular expression pattern to be used in word splitting.

    Default: "[[:space:][:punct:][:digit:]]+".

perl: A logical indicating whether to use Perl-compatible regular expressions in word splitting.

    Default: FALSE.

tolower: A logical indicating whether to transform texts to lower case (after word splitting).

    Default: TRUE.

reduce: A logical indicating whether a representation of n-grams more efficient than the one used by Cavnar and Trenkle should be employed.

    Default: TRUE.

useBytes: A logical indicating whether to use byte n-grams rather than character n-grams.

    Default: FALSE.

ignore: a character vector of n-grams to be ignored when computing n-gram profiles.

    Default: "_" (corresponding to a word boundary).

size: The maximal number of n-grams used for a profile.

    Default: 1000L.

This method uses textcnt in package **tau** for computing n-gram profiles, with n, split, perl and useBytes corresponding to the respective textcnt arguments, and option reduce setting argument marker as needed. N-grams listed in option ignore are removed, and only the most frequent remaining ones retained, with the maximal number given by option size.

Unless the profile db uses bytes rather than characters (i.e., option useBytes is TRUE), text documents in x containing non-ASCII characters must declare their encoding (see Encoding), and will be re-encoded to UTF-8.

Note that option n specifies *all* numbers of characters or bytes to be used in the profiles, and not just the maximal number: e.g., taking n = 3 will create profiles only containing tri-grams.

Examples

```
## Obtain the texts of the standard licenses shipped with R.
files <- dir(file.path(R.home("share"), "licenses"), "^[A-Z]",
             full.names = TRUE)
texts <- sapply(files,
                function(f) paste(readLines(f), collapse = "\n"))
names(texts) <- basename(files)
## Build a profile db using the same method and options as for building
## the ECIMCI character profiles.
profiles <- textcat_profile_db(texts, profiles = ECIMCI_profiles)
## Inspect the 10 most frequent n-grams in each profile.
lapply(profiles, head, 10L)
## Combine into one frequency table.
tab <- as.matrix(profiles)
tab[, 1 : 10]
## Determine languages.
textcat(profiles, ECIMCI_profiles)
```

---

textcat_xdist	*Cross-Distances Between N-Gram Profiles*

Description

Compute cross-distances between collections of n-gram profiles.

Usage

```
textcat_xdist(x, p = NULL, method = "CT", ..., options = list())
```

Arguments

x	a textcat profile db (see `textcat_profile_db`), or an R object of text documents extractable via `as.character`.
p	NULL (default), or as for x. The default is equivalent to taking p as x (but more efficient).
method	a character string specifying a built-in method, or a user-defined function for computing distances between n-gram profiles, or NULL (corresponding to the current value of **textcat** option xdist_method (see `textcat_options`). See **Details** for available built-in methods.
...	options to be passed to the method for computing distances.
options	a list of such options.

Details

If x (or p) is not a profile db, the n-gram profiles of the individual text documents extracted from it are computed using the profile method and options in p if this is a profile db, and using the current **textcat** profile method and options otherwise.

Currently, the following distance methods for n-gram profiles are available.

"CT": the out-of-place measure of Cavnar and Trenkle.

"ranks": a variant of the Cavnar/Trenkle measure based on the aggregated absolute difference of the ranks of the combined n-grams in the two profiles.

"ALPD": the sum of the absolute differences in n-gram log frequencies.

"KLI": the Kullback-Leibler I-divergence $I(p,q) = \sum_i p_i \log(p_i/q_i)$ of the n-gram frequency distributions p and q of the two profiles.

"KLJ": the Kullback-Leibler J-divergence $J(p,q) = \sum_i (p_i - q_i) \log(p_i/q_i)$, the symmetrized variant $I(p,q) + I(q,p)$ of the I-divergences.

"JS": the Jensen-Shannon divergence between the n-gram frequency distributions.

"cosine" the cosine dissimilarity between the profiles, i.e., one minus the inner product of the frequency vectors normalized to Euclidean length one (and filled with zeros for entries missing in one of the vectors).

"Dice" the Dice dissimilarity, i.e., the fraction of n-grams present in one of the profiles only.

For the measures based on distances of frequency distributions, n-grams of the two profiles are combined, and missing n-grams are given a small positive absolute frequency which can be controlled by option eps, and defaults to 1e-6.

Options given in ... and options are combined, and merged with the default xdist options specified by the **textcat** option xdist_options using *exact* name matching.

Examples

```
## Compute cross-distances between the TextCat byte profiles using the
## CT out-of-place measure.
d <- textcat_xdist(TC_byte_profiles)
## Visualize results of hierarchical cluster analysis on the distances.
plot(hclust(as.dist(d)), cex = 0.7)
```

Index