Package ‘LexFindR’

October 29, 2021

Title  Find Related Items and Lexical Dimensions in a Lexicon
Version  1.0.2
Date    2021-10-23

Description  Implements code to identify lexical competitors in a given list
of words. We include many of the standard competitor types used in spoken word
recognition research, such as functions to find cohorts, neighbors, and
rhymes, amongst many others. The package includes documentation for using a
variety of lexicon files, including those with form codes made up of multiple
letters (i.e., phoneme codes) and also basic orthographies. Importantly, the
code makes use of multiple CPU cores and vectorization when possible, making
it extremely fast and able to handle large lexicons. Additionally, the package
contains documentation for users to easily write new functions, allowing
researchers to examine other relationships within a lexicon.
Preprint: <https://psyarxiv.com/8dyru/>. Open ac-
token=9W1O9soCc9y0uSuwWSUYyfJAH0g46feNdc4O2WrhzyrdKcK8uzZx_
hDEtgbyzn3gvxdG5Cuj0j0c41VMFBqYCGTQmE2blN2Gw074LJ8ro1pEOAYDRFy6Lhf1nc719vD-zU7GDyKQXDAwPbrisvPBe3C3>.  
Citation: Li, Z., Crinnion, A.M. & Magnuson, J.S. (2021).

License GPL (>= 3)
Encoding UTF-8
LazyData true
RoxygenNote 7.1.1
Suggests tidyverse, knitr, rmarkdown, testthat, future.apply, tictoc
VignetteBuilder knitr
Depends R (>= 3.5.0)
URL https://github.com/maglab-uconn/LexFindR
BugReports https://github.com/maglab-uconn/LexFindR/issues
NeedsCompilation no
Author  ZhaoBin Li [aut, cre],
        Anne Marie Crinnion [aut],
        James S. Magnuson [aut, cph]
Maintainer  ZhaoBin Li <bzlplj@gmail.com>
Repository  CRAN
Date/Publication  2021-10-29 06:30:08 UTC

R topics documented:

get_cohorts  . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 2
get_cohortsP . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 3
get_embeds_in_target . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 4
get_embeds_in_targetP . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 5
get_fw . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 6
get_fwcp . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 6
get_homoforms . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 7
get_neighbors . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 8
get_neighborsP . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 9
get_nohorts . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 10
get_rhymes . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 11
get_target_embeds_in . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 12
get_target_embeds_inP . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 12
get_uniqpt . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 13
lemmalex . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 14
slex . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 15

Index  16

get_cohorts  Get cohort competitors

Description
Cohorts overlap in onset phoneme(s).

Usage
get_cohorts(
  target,
  lexicon,
  sep = " ",
  form = FALSE,
  count = FALSE,
  overlap = 2
)
get_cohortsP

Arguments

- **target**: Character string containing a target word
- **lexicon**: Character vector containing the lexical database
- **sep**: Separator in target and lexicon
- **form**: Whether to return words in lexicon
- **count**: Whether to return count of words
- **overlap** (*get_cohorts only*): Integer specifying the number of onset phonemes to overlap for matching with the target word

Value

The indexes of the competitors in the lexical database

Examples

```r
get_cohortsP("AA R K", c("AA R K", "AA R T", "B AA B"))
```

Description

Cohorts that are not neighbors

Usage

```r
get_cohortsP(
  target,
  lexicon,
  neighbors = "das",
  sep = " ",
  form = FALSE,
  count = FALSE
)
```

Arguments

- **target**: Character string containing a target word
- **lexicon**: Character vector containing the lexical database
- **neighbors** (*get_neighbors only*): Character vector specifying the type of neighbor to return. Return the delete, add, substitute neighbors of the target when 'd', 'a', and/or 's' is in neighbors respectively
- **sep**: Separator in target and lexicon
- **form**: Whether to return words in lexicon
- **count**: Whether to return count of words
get_embeds_in_target

Value

the indexes of the competitors in the lexical database

Examples

get_cohortsP("AA R K", c("AA R K", "AA R", "B AA B"), neighbors = "das")

get_embeds_in_target("AA R K", c("AA R K", "AA R", "B AA B"))
**get_embeds_in_targetP**  

*Get embeds-in-target PRIME*

---

**Description**

Items embedded in the target which are not cohorts or neighbors

**Usage**

```r
get_embeds_in_targetP(
  target,
  lexicon,
  neighbors = "das",
  sep = " ",
  form = FALSE,
  count = FALSE
)
```

**Arguments**

- **target**: Character string containing a target word
- **lexicon**: Character vector containing the lexical database
- **neighbors** *(get_neighbors only)*: Character vector specifying the type of neighbor to return. Return the delete, add, substitute neighbors of the target when ‘d’, ‘a’, and/or ‘s’ is in neighbors respectively
- **sep**: Separator in target and lexicon
- **form**: Whether to return words in lexicon
- **count**: Whether to return count of words

**Value**

the indexes of the competitors in the lexical database

**Examples**

```r
```
**get fw**

*Get the log Frequency Weight (FW) of a competitor set*

**Description**

Get the log Frequency Weight (FW) of a competitor set

**Usage**

```r
get_fw(competitors_freq, pad = 0)
```

**Arguments**

- `competitors_freq`
  Numeric vector containing the frequencies of competitors (including itself)
- `pad`
  Value to add to frequencies before taking log; if your minimum frequency is 0, consider adding a value between 1 and 2; if your minimum frequency is between 0 and 1, consider adding 1

**Value**

FW

**Examples**

```r
get_fw(c(10, 50), pad = 1)
```

---

**get fwcp**

*Get the log Frequency Weighted Competitor Probability (FWCP)*

**Description**

Get the log Frequency Weighted Competitor Probability (FWCP)

**Usage**

```r
get_fwcp(target_freq, competitors_freq, pad = 0, add_target = FALSE)
```
get_homoforms

Arguments

- `target_freq`: Frequency of target word
- `competitors_freq`: Numeric vector containing the frequencies of competitors (including itself)
- `pad`: Value to add to frequencies before taking log; if your minimum frequency is 0, consider adding a value between 1 and 2; if your minimum frequency is between 0 and 1, consider adding 1
- `add_target`: Boolean; set to TRUE if you want the target frequency added to the denominator; only do this if the target is not already included in the competitor set (e.g., if the target is in the lexicon, it will be captured as its own neighbor, its own cohort, etc.)

Value

log FWCP

Examples

```r
get_fwcp(100, c(10, 50), pad = 1)
```

get_homoforms

Get homophones

Description

Homophones are items which sound similar to the target

Usage

```r
get_homoforms(target, lexicon, sep = " ", form = FALSE, count = FALSE)
```

Arguments

- `target`: Character string containing a target word
- `lexicon`: Character vector containing the lexical database
- `sep`: Separator in target and lexicon
- `form`: Whether to return words in lexicon
- `count`: Whether to return count of words

Value

the indexes of the competitors in the lexical database

Examples

```r
get_homoforms("AA R K", c("AA R K", "AA R", "B AA B"))
```
get_neighbors

Get phonological neighbors

Description

Phonological neighbors are items which can be converted to the target by one add, delete and substitute operation.

Usage

get_neighbors(
  target,
  lexicon,
  neighbors = "das",
  sep = " ",
  form = FALSE,
  count = FALSE
)

Arguments

target Character string containing a target word
lexicon Character vector containing the lexical database
neighbors (get_neighbors only) Character vector specifying the type of neighbor to return. Return the delete, add, substitute neighbors of the target when 'd', 'a', and/or 's' is in neighbors respectively
sep Separator in target and lexicon
form Whether to return words in lexicon
count Whether to return count of words

Value

the indexes of the competitors in the lexical database

Examples

get_neighbors("AA R K", c("AA R K", "AA R", "B AA B"), "d")
get_neighbors("AA R K", c("AA R K", "AA R", "B AA B"), "da")
get_neighbors("AA R K", c("AA R K", "AA R", "B AA B"), "das")
**get_neighborsP**

**Get NeighborssPrime**

---

### Description

Neighbors which are not cohorts or rhymes

### Usage

```r
get_neighborsP(
  target,
  lexicon,
  neighbors = "das",
  sep = " ",
  form = FALSE,
  count = FALSE
)
```

### Arguments

- `target`: Character string containing a target word
- `lexicon`: Character vector containing the lexical database
- `neighbors`: *(get_neighbors only)* Character vector specifying the type of neighbor to return. Return the delete, add, substitute neighbors of the target when 'd', 'a', and/or 's' is in neighbors respectively
- `sep`: Separator in target and lexicon
- `form`: Whether to return words in lexicon
- `count`: Whether to return count of words

### Value

the indexes of the competitors in the lexical database

### Examples

```r
get_neighborsP("AA R K", c("AA R K", "AA R", "B AA B"), neighbors = "das")
```
get_nohorts

Description

Items which are both cohorts and neighbors

Usage

get_nohorts(
  target,
  lexicon,
  neighbors = "das",
  sep = " ",
  form = FALSE,
  count = FALSE
)

Arguments

target
  Character string containing a target word
lexicon
  Character vector containing the lexical database
neighbors
  (get_neighbors only) Character vector specifying the type of neighbor to return. Return the delete, add, substitute neighbors of the target when 'd', 'a', and/or 's' is in neighbors respectively
sep
  Separator in target and lexicon
form
  Whether to return words in lexicon
count
  Whether to return count of words

Value

the indexes of the competitors in the lexical database

Examples

get_nohorts("AA R K", c("AA R K", "AA R", "B AA B"), neighbors = "das")
get_rhymes

Get rhyme competitors

Description

Rhymes overlap in all except onset phoneme(s)

Usage

get_rhymes(
  target,
  lexicon,
  sep = " ",
  form = FALSE,
  count = FALSE,
  mismatch = 1
)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>target</td>
<td>Character string containing a target word</td>
</tr>
<tr>
<td>lexicon</td>
<td>Character vector containing the lexical database</td>
</tr>
<tr>
<td>sep</td>
<td>Separator in target and lexicon</td>
</tr>
<tr>
<td>form</td>
<td>Whether to return words in lexicon</td>
</tr>
<tr>
<td>count</td>
<td>Whether to return count of words</td>
</tr>
<tr>
<td>mismatch</td>
<td>(get_rhymes only) Integer specifying the number of onset phonemes to mismatch for matching with the target word</td>
</tr>
</tbody>
</table>

Value

the indexes of the competitors in the lexical database

Examples

get_rhymes("AA R K", c("AA R K", "B AA R K", "B AA B"))
get_target_embeds_in  Get embedded competitors

Description
Embedded competitors are items which the target embedded in.

Usage
get_target_embeds_in(target, lexicon, sep = " ", form = FALSE, count = FALSE)

Arguments
- target: Character string containing a target word
- lexicon: Character vector containing the lexical database
- sep: Separator in target and lexicon
- form: Whether to return words in lexicon
- count: Whether to return count of words

Value
the indexes of the competitors in the lexical database

Examples
get_target_embeds_in("AA R K", c("AA R K", "B AA R K", "B AA B"))

get_target_embeds_inP  Get target-embeds-in PRIME

Description
Items the target embeds into which are not cohorts or neighbors

Usage
get_target_embeds_inP(
    target,  
    lexicon,  
    neighbors = "das",  
    sep = " ",  
    form = FALSE,  
    count = FALSE
)

)
**get_uniqpt**

**Arguments**

- **target**: Character string containing a target word
- **lexicon**: Character vector containing the lexical database
- **neighbors**: *(get_neighbors only)* Character vector specifying the type of neighbor to return. Return the delete, add, substitute neighbors of the target when 'd', 'a', and/or 's' is in neighbors respectively
- **sep**: Separator in target and lexicon
- **form**: Whether to return words in lexicon
- **count**: Whether to return count of words

**Value**

the indexes of the competitors in the lexical database

**Examples**

```r
```

---

**get_uniqpt**

Get phonological uniqueness point

**Description**

Phonological uniqueness point is the index at which the target becomes unique in the lexicon

**Usage**

```r
get_uniqpt(target, lexicon, sep = " ")
```

**Arguments**

- **target**: Character string containing a target word
- **lexicon**: Character vector containing the lexical database
- **sep**: Separator in target and lexicon

**Value**

Target is not unique: length + 1, else index where target becomes unique in lexicon

**Examples**

```r
get_uniqpt("AA R K", c("AA R", "B AA B", "B AA R K"))
```
Lemmalex dictionary

Description

Lemmalex is primarily based on the SUBTLEXus subtitle corpus (based on American subtitles with 51 million items in total) reduced to lemma using a copyrighted database (Francis and Kučera, 1982). The pronunciation is given by CMU Pronouncing Dictionary.

Usage

lemmalex

Format

An object of class tbl_df (inherits from tbl, data.frame) with 17750 rows and 3 columns.

Details


CMU Pronouncing Dictionary: http://www.speech.cs.cmu.edu/cgi-bin/cmudict

@format A table with 20,293 rows and 3 variables:

Item  SUBTLEXus dictionary reduced to lemmas
Frequency  Number of times the item appeared in the SUBTLEXus corpus
Pronunciation  ARPAbet transcription according to CMU ...

Source

Description

TRACE slex lexicon translated by Nenadić and Tucker into ARPAbet pronunciation

Usage

slex

Format

An object of class data.table (inherits from data.frame) with 212 rows and 3 columns.

Details


@format A table with 212 rows and 2 variables:

Item  TRACE slex transcription
Pronunciation  ARPAbet transcription ...

Source

https://era.library.ualberta.ca/items/61319cc6-436a-428c-b960-545bdc9bd5d3
Index

* datasets
  lemmalex, 14
  slex, 15

get_cohorts, 2
get_cohortsP, 3
get_embeds_in_target, 4
get_embeds_in_targetP, 5
get_fw, 6
get_fwcp, 6
get_homoforms, 7
get_neighbors, 8
get_neighborsP, 9
get_nohorts, 10
get_rhymes, 11
get_target_embeds_in, 12
get_target_embeds_inP, 12
get_unidept, 13

lemmalex, 14

slex, 15