# Package 'LexFindR'

January 20, 2025

**Title** Find Related Items and Lexical Dimensions in a Lexicon

**Version** 1.1.0

**Date** 2024-6-15

**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://osf.io/preprints/psyarxiv/8dyru/>. Open access: <doi:10.3758/s13428-021-01667-6>.
Citation: Li, Z., Crinnion, A.M. & Magnuson, J.S. (2021).
<doi:10.3758/s13428-021-01667-6>.

**License** GPL (>= 3)

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 7.3.1

**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 <li_zhaobin@icloud.com>

**Repository** CRAN

**Date/Publication** 2024-06-16 14:40:01 UTC

# Contents



---

`get_cohorts` *Get cohort competitors*

---

## Description

Cohorts overlap in onset phoneme(s).

## Usage

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

### 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

```
get_cohorts("AA R K", c("AA R K", "AA R T", "B AA B"))
```

---

## get_cohortsP *Get CohortsPrime*

---

### Description

Cohorts that are not neighbors

### Usage

```
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 |

## 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` *Get embedding competitors*

---

## Description

Embedding competitors are items embedded in target

## Usage

```
get_embeds_in_target(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_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

```
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

```
get_embeds_in_targetP("B AA R K IY", c("AA R K", "AA R", "AA R K IY", "B AA 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

```
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

```
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

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

### 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

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

---

## get_homoforms *Get homophones*

---

### Description

Homophones are items which sound similar to the target

### Usage

```
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

```
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

```
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

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

get_nohorts *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

| | |
|---|---|
| `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 |
| `mismatch` | (*get_rhymes* only) Integer specifying the number of onset phonemes to mismatch for matching with the target word |

## 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
)
```

### 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_target_embeds_inP("B AA R K", c("AA R K", "AA R", "B AA R K IY", "B AA R"))
```

---

## get_uniqpt *Get phonological uniqueness point*

---

### Description

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

### Usage

```
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

```
get_uniqpt("AA R K", c("AA R", "B AA B", "B AA R K"))
```

`lemmalex` *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

Reference: Brysbaert, M., & New, B. (2009). Moving beyond Kučera and Francis: A critical evaluation of current word frequency norms and the introduction of a new and improved word frequency measure for American English. Behavior research methods, 41(4), 977-990.

Kučera, H., & Francis, W. N. (1967). Computational analysis of present-day American English. Brown university press.

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

https://www.ugent.be/pp/experimentele-psychologie/en/research/documents/subtlexus

slex *slex ARPAbet*

## 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

TRACE slex lexicon with Frequencies: McClelland, J. L., & Elman, J. L. (1986). The TRACE model of speech perception. Cognitive psychology, 18(1), 1-86.

APRAbet transcription: Nenadić, F., & Tucker, B. V. (2020). Computational modelling of an auditory lexical decision experiment using jTRACE and TISK. Language, Cognition and Neuroscience, 1-29.

@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