Package ‘phonetisr’

February 26, 2025

Title A Naive IPA Tokeniser

Version 0.1.0

Date 2025-02-23

Description It provides users with functions to parse International Phonetic Alphabet (IPA) transcriptions into individual phones (tokenisation) based on default IPA symbols and optional user specified multi-character phones. The tokenised transcriptions can be used for obtaining counts of phones or for searching for words matching phonetic patterns.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.3.2

Imports cli, dplyr, lifecycle, magrittr, stringi, stringr, tibble, Unicode

Depends R (>= 2.10)

Suggests rmarkdown, knitr, tidyverse

VignetteBuilder knitr

URL `https://github.com/stefanocoretta/phonetisr`, `https://stefanocoretta.github.io/phonetisr/`

NeedsCompilation no

Author Stefano Coretta [aut, cre] (<`https://orcid.org/0000-0001-9627-5532`>)

Maintainer Stefano Coretta <`stefano.coretta@gmail.com`>

Repository CRAN

Date/Publication 2025-02-26 13:10:02 UTC

Contents

`featurise` *Add features to list of phones*

Description

This function counts occurrences of phones and includes basic phonetic features.

Usage

```
featurise(phlist)
```

Arguments

`phlist`	A list of phones or the output of `phonetise()`.

Value

A tibble.

Examples

```
ipa <- c("ada", "buba", "kiki", "sa\u0283a")
ip_ph <- phonetise(ipa)
featurise(ip_ph)
```

`get_no_ipa` *Get non-IPA characters.*

Description

Given a vector of characters, it returns those which are not part of the IPA.

Usage

```
get_no_ipa(chars)
```

Arguments

`chars`	A vector of characters.

Value

A vector.

Examples

```
get_no_ipa(c("a", "\0283", ">"))
```

`ipa_symbols` *List of IPA symbols*

Description

List of IPA symbols

Usage

```
ipa_symbols
```

Format

A data frame with 143 rows and 12 variables:

IPA IPA symbol.

unicode Unicode code.

uni_name Unicode name.

ipa_name IPA name.

phon_type The phonetic type of the symbol.

type General character type (`consonant`, `vowel`, `diacritic`).

height_ipa Vowel openness.

height Vowel height.

backness Vowel backness.

rounding Vowel rounding.

voicing Consonant voicing.

place Consonant place of articulation.

manner Consonant manner of articulation.

lateral Is the consonant lateral?

sonorant Is the phone sonorant?

kl_swadesh *Klingon Swadesh list*

Description

The Swadesh list in Klingon.

Usage

```
kl_swadesh
```

Format

A data frame with 195 rows and 4 variables:

id Swadesh list item number.

gloss English gloss.

translit Klingon transliteration.

ipa IPA transcription.

phonetise *Tokenise IPA strings*

Description

`phonetise()` tokenises strings of IPA symbols (like phonetic transcriptions of words) into individual "phones". The output is a list.

Usage

```
phonetise(
  strings,
  multi = NULL,
  regex = NULL,
  split = TRUE,
  sep = " ",
  sanitise = TRUE,
  ignore_stress = TRUE,
  ignore_tone = TRUE,
  diacritics = FALSE,
  affricates = FALSE,
  v_sequences = FALSE,
  prenasalised = FALSE,
  all_multi = FALSE,
  sanitize = sanitise
)

phonetize(
  strings,
  multi = NULL,
  regex = NULL,
  split = TRUE,
  sep = " ",
  sanitise = TRUE,
  ignore_stress = TRUE,
  ignore_tone = TRUE,
  diacritics = FALSE,
  affricates = FALSE,
  v_sequences = FALSE,
  prenasalised = FALSE,
  all_multi = FALSE,
  sanitize = sanitise
)
```

Arguments

`strings`	A character vector with a list of words in IPA.
`multi`	A character vector of one or more multi-character phones as strings.
`regex`	A string with a regular expression to match several multi-character phones.
`split`	If set to `TRUE` (the default), the tokenised strings are split into phones (i.e. the output is a vector with one element per phone). If set to `FALSE`, the string is not split and the phones are separated with the character defined in `sep`.
`sep`	A character to be used as the separator of the phones if `split = FALSE` (default is , space).
`sanitise`	Whether to remove all non-IPA characters (`TRUE` by default).
`ignore_stress`	If `TRUE` (the default), stress marks are not parsed.
`ignore_tone`	If `TRUE` (the default), tone marks and letters are not parsed.
`diacritics`	If set to `TRUE`, parses all valid diacritics as part of the previous character (`FALSE` by default).
`affricates`	If set to `TRUE`, parses homorganic stop + fricative as affricates.
`v_sequences`	If set to `TRUE`, collapses vowel sequences (`FALSE` by default).
`prenasalised`	If set to `TRUE`, parses prenasalised consonants as such (`FALSE` by default).
`all_multi`	If set to `TRUE`, `diacritics`, `affricates`, `v_sequences` and `prenasalised` are all set to `TRUE`.
`sanitize`	Alias of `sanitise`.

Value

A list of phonetised strings.

Examples

```
# using unicode escapes for CRAN policy
ipa <- c("p\u02B0a\u0303k\u02B0", "t\u02B0um\u0325", "\u025Bk\u02B0\u026F")
ph <- c("p\u02B0", "t\u02B0", "k\u02B0", "a\u0303", "m\u0325")

phonetise(ipa, multi = ph)

ph_2 <- ph[4:5]

# Match any character followed by <\u02B0> with ".\u02B0".
phonetise(ipa, multi = ph_2, regex = ".\u02B0")

# Same result.
phonetise(ipa, regex = ".(\u0303|\u0325|\u02B0)")

# Don't split strings and use "." as separator
phonetise(ipa, multi = ph, split = FALSE, sep = ".")
```

`ph_search` *Search phones*

Description

Given a vector of phonetised strings, find phones.

Usage

```
ph_search(phlist, phonex)
```

Arguments

`phlist`	The output of `phonetise()`.
`phonex`	A phonetic expression. Supported shorthands are `C` for consonant, `V` for vowel, and # for word boundary.

Value

A list.

Examples

```
ipa <- c("p\u02B0a\u0303k\u02B0", "t\u02B0um\u0325", "\u025Bk\u02B0\u026F", "pun")
ph <- c("p\u02B0", "t\u02B0", "k\u02B0", "a\u0303", "m\u0325")
ipa_ph <- phonetise(ipa, multi = ph)
ph_search(ipa_ph, "#CV")

# partial matches are also returned
ph_search(ipa_ph, "p")
```

```
# use regular expressions
ph_search(ipa_ph, "p\u02B0?V")
```

Index