Package ‘alineR’

September 14, 2017

Type Package
Title Alignment of Phonetic Sequences Using the ‘ALINE’ Algorithm
Version 1.1.4
Date 2017-09-12
Maintainer Sean Downey <sean@codexdata.com>
LazyData true
LazyLoad true
Description Functions are provided to calculate the ‘ALINE’ Distance between words as per (Kondrak 2000) and (Downey, Hallmark, Cox, Norquest, & Lansing, 2008, <doi:10.1080/09296170802326681>). The score is based on phonetic features represented using the Unicode-compliant International Phonetic Alphabet (IPA). Parameterized features weights are used to determine the optimal alignment and functions are provided to estimate optimum values using a genetic algorithm and supervised learning. See (Downey, Sun, and Norquest 2017, <https://journal.r-project.org/archive/2017/RJ-2017-005/index.html>.
License GPL-3
NeedsCompilation yes
Repository CRAN
Date/Publication 2017-09-14 17:18:26 UTC
Author Sean Downey [aut, cre],
Guowei Sun [aut],
Greg Kondrak [cph] (Copyright holder of original ALINE algorithm C++ source code included in package.)

R topics documented:

aline-package ................................................................. 2
aline ................................................................. 3
ALINE.map ............................................................. 5
ALINE.segments ......................................................... 6
decode.ALINE ........................................................ 7
encode.ALINE ........................................................ 8
Description

Functions are provided to calculate the 'ALINE' Distance between cognate pairs. By default the Aline distance is returned (Downey et al. 2008). Several utility functions are provided including the ability to return the aligned character strings and the similarity score (Kondrak 1999), the ability to change feature weightings, and the ability to modify the IPA character mappings. The package includes functions for optimizing and plotting feature weights using expert alignment determinations and a genetic algorithm. We provide an R-interface to the aline C++ algorithm originally written by G. Kondrak (1999, 2000). The package authors would like to acknowledge Greg Kondrak (http://webdocs.cs.ualberta.ca/~kondrak/) for developing the original ALINE algorithm. The base code provided here has been substantially modified from the original version to provide integration with R and to enable various user-functions. This project was funded by the National Science Foundation Cultural Anthropology Program (Grant number SBS-1030031) and the University of Maryland College of Behavioral and Social Sciences.

To cite 'alineR' in a publication please use:


Details

Package: alineR
Type: package
Version: 1.1.4
Date: 2017-9-12
License: GPL-3

Author(s)

Sean S. Downey and Guowei Sun
Maintainer: Sean Downey (<downey.205@osu.edu>)
Developer/Programmer: Guowei Sun (<gwsun@umd.edu>)

References


---

aline

Calculate aline distances

Description

The main user function for returning Aline Distances. Also it provides options for additional outputs such as the raw alignments and individual distance measurements. Word lists are passed as two vectors (w1, w2) such that the nth element of each vector are compared.

Usage

aline(w1, w2, sim = FALSE, m1 = NULL, m2 = NULL, mark=FALSE, alignment = FALSE, ...)

Arguments

w1 A vector of IPA-encoded words.
w2 A second vector of IPA-encoded words to be aligned with w1.
sim By default calculates the aline distance (normalized between word pairs) as defined in Downey et al. 2008. If TRUE aline similarity scores from (Kondrak 2000) are returned.
m1 User defined IPA symbol. See map() for details.
m2 User defined ALINE symbol. See map() for details.
alignment If TRUE the function will return the aligned IPA word pairs.
mark If TRUE the result will mark the invalid characters with "@"
... Other parameters passed to raw.alignment().
Value

If `alignment=FALSE` the function returns a vector of scores such that the nth score is the aline distance between the nth elements of x and y.

If `alignment=TRUE` the function returns a data frame with each word pair represented in a column and with the following rows:

- `w1`: The original IPA-encoded word vector.
- `w2`: The original IPA-encoded word vector.
- `scores`: The similarity or distance score as defined by argument `sim`.
- `a1`: The alignment of the first word.
- `a2`: The alignment of the second word.

Note

This function will issue warnings and drop unknown characters if an input word contains unmapped IPA symbols. If this happens, the warning can be eliminated by appending an additional IPA-ASCII character mapping.

Author(s)

Sean Downey and Guowei Sun

References


See Also

`raw.alignment.map`

Examples

```r
x <- c(intToUtf8(c(361, 109, 108, 97, 116, 952)), intToUtf8(c(100, 105, 331, 331, 105, 114, 97)))
y <- c(intToUtf8(c(418, 109, 108, 97, 116, 952)), intToUtf8(c(100, 105, 110, 110, 105, 114, 97)))
# For CRAN requirement, to see x and y, type x,y in R console
x
y
aline(w1=x,w2=y)  # A warning is returned because of unknown character

# user substitution
aline(w1=x,w2=y, ml=intToUtf8(418), m2="o")
```
Description

An IPA-ASCII character map used for transforming IPA sequence into ASCII characters used by the C++ program. ALINE.map() is used internally by various functions. The original mapping schema is defined in (Kondrak 1999). The values provided here were derived from an Indonesian database so additional user-mappings for use with other language families can be enabled. Unicode integer values are stored in the dataframe. To view IPA see show.map().

Usage

data(ALINE.map)

Format

A data frame with 102 observations on the following 4 variables.

Aline  A column of ALINE symbols
uNval  Unicode encoding for the IPA character.
A.Val  Unicode value(s) for the ALINE character(s).

References


See Also

map

Examples

data(ALINE.map)
ALINE.segments

Similarity scores of aligned segments

Description

Return vector of similarity scores for each optimally aligned segment.

Usage

ALINE.segments(result, 
Syllabic = 5, Place = 40, Stop = 50, 
Voice = 10, Nasal = 10, Retroflex = 10, 
Lateral = 10, Aspirated = 5, Long = 1, 
High = 5, Back = 5, Round = 5, sk=10)

Arguments

result The value returned from raw.alignment() function, which is a list containing four 
elements.
Syllabic, Place, Stop, Voice, Nasal, Retroflex, Lateral, Aspirated, Long, High, Back, Round 
Feature weight used by the ALINE algorithm to determine the phonetic distance.

sk Skip penalty in determining the alignment

Details

This function returns the similarity scores for each pair of aligned segments from the optimal align- 
ment. The sum of these values is equal to the similarity score.

Value

vec A numeric vector. The length of the vector is equal to the number of aligned 
segments. The value of the ith element is the similarity score for that segment 
pair.

Author(s)

Guowei Sun

Examples

# align words 
result<-raw.alignment(c("watu","dat"))

# print the alignment followed by the sim score 
# for each pair of aligned segments 
cat(result[[3]],result[[4]],sep='\n') 
ALINE.segments(result)
**decode.ALINE**

**Decode ALINE ASCII output**

**Description**

Decode aligned ALINE ASCII output into the original IPA characters while indicating the optimal alignment with vertical bars ('|').

**Usage**

```
decode.ALINE(x, y, m1 = NULL, m2 = NULL)
```

**Arguments**

- `x`: A vector containing the original IPA word.
- `y`: A vector containing the aligned characters in ALINE notation.
- `m1`: A vector of IPA characters to encode. See map() for details.
- `m2`: A vector of ASCII ALINE encodings. See map() for detail.

**Value**

- `word`: The alignment of the word in the IPA notation.

**Warning**

The ALINE encoding scheme only accepts a single lower case character followed by one or more upper case characters. For example, "dD" can be accepted but "dd" can not.

**Note**

The original IPA word is required because of many-to-one relationships when mapping ALINE->IPA. For example, both intToUtf8(249) and intToUtf8(250) are mapped to ASCII 'u' (see map()) so the process cannot be reversed without the original IPA word. User-specified mappings should be consistent with encode.ALINE().

**Author(s)**

Sean Downey and Guowei Sun

**See Also**

`encode.ALINE`
**Examples**

```r
x <- intToUtf8(c(611, 117, 108, 108, 97))
y = "gS u l l a |
```

decode.ALINE(x, y)

# user-specified mapping. Should be consistent with encode.ALINE() function

```r
x <- intToUtf8(c(418, 109, 108, 97, 116, 952))
y = "o m l a t tS |
```

decode.ALINE(x, y, m1 = intToUtf8(418), m2 = "o")

---

**encode.ALINE**

*Encode IPA as ALINE ASCII notation*

---

**Description**

Translates a vector of IPA words into the ASCII encoding scheme used by aline via ALINE.map. Uses UTF-8 encodings.

**Usage**

```r
encode.ALINE(x, mark = FALSE, m1 = NULL, m2 = NULL)
```

**Arguments**

- **x**: A vector of IPA words to encode.
- **mark**: If FALSE unknown symbols are omitted; if TRUE invalid symbols are replaced with "@".
- **m1**: A vector of IPA characters to encode. See map() for detail.
- **m2**: A vector of ALINE encodings. See map() for detail

---

**Warning**

This function will return a warning if it encounters an IPA symbol not included in the ALINE map or defined by the user. It will then ignore that symbol and decode the rest. Use mark=TRUE to see what is being omitted and map(m1, m2) to provide a new mapping.

**Author(s)**

Sean Downey and Guowei Sun

**See Also**

`decode.ALINE`

**Examples**

```r
y <- c(intToUtf8(c(418, 109, 108, 97, 116, 952)), intToUtf8(c(100, 105, 110, 110, 105, 114, 97)))
y
```

code.ALINE(y, m1 = intToUtf8(418), m2 = "o")
**features.plot**

*Plot feature optimization result*

---

**Description**

Generates a 4x3 histogram panel plot using the optimization result from `optimize.features`.

**Usage**

```r
features.plot(R,
              first = FALSE,
              para = c(5, 40, 50, 10, 10, 10, 10, 5, 1, 5, 5, 5, 10),
              skip=FALSE,column=4,row=3)
```

**Arguments**

- `R`  
  Output from `optimize.features`
- `first`  
  If TRUE, plot only the first replicate. If FALSE, plot results from all independent replicates.
- `para`  
  The default feature weights to be plotted in the histogram.
- `skip`  
  If TRUE, include a 13th histogram for the SkipCost parameter.
- `column`  
  Number of parameter histogram plots in each row
- `row`  
  Number of parameter histogram plots in each column

**See Also**

- `generate.training, optimize.features`

**Examples**

```r
data<-data.frame(dog=c('dog','perro'),cat=c('cat','gato'),rat=c('rat','rata'))
M1<-generate.training(raw.data=data,search.size=100,table=FALSE)
M2<-optimize.features(set=M1,ranking=c(1,1,1),
                      num=20,step=5,replication=2,list=TRUE)
features.plot(M2)
```
generate.training  

Generate training dataset

Description

Generates an output file of training data to be used by a linguist to select the best alignments from a list of the unique set of possible alignments for each given pair of words.

Usage

```
generate.training(raw.data, search.size=1000, table=TRUE, 
                 file.out="candidate_alignments.csv")
```

Arguments

- **raw.data**: This is a 2*n matrix containing n ipa encoded cognate pairs.
- **search.size**: Number of time to randomize feature parameters while searching for unique alignments.
- **table**: `table=TRUE` will generate a csv file named by the user containing possible alignments in IPA encodings.
- **file.out**: Name of CSV file for output.

Value

A list containing two elements:

- **standard_ipa_symbol**: A data frame containing input cognate pairs and a list of possible alignments.
  UTF-8 IPA
- **ALINE_symbol**: Same as above, but using ALINE symbol for use in internal functions

Note

Expert determinations are used by the genetic algorithm to optimize feature weights. Feature parameters are randomly generated to find possible alignments, so setting `search.size` to larger values will ensure all possible alignments are found.

To generate the output file set `file.out` to some value and open the resulting file with a spreadsheet program. To ensure correct Unicode IPA formatting, make sure the file encoding is selected as UTF-8 when importing the generated csv file.

The function also returns an list containing two dataframes (IPA and Aline) that are used internally in the optimization process.

See Also

- `optimize.features`
Examples

# some cognates
data <- data.frame(dog=c('dog', 'perro'), cat=c('cat', 'gato'), rat=c('rat', 'rata'))

# write out a CSV file that can be opened in Excel and used for expert determinations
M <- generate.training(raw.data=data, search.size=100, file="open.with.excel.csv")

---

map  

User-defined Mappings

Description

Allows user-defined mappings from UTF-8 IPA to ASCII ALINE. User mappings are given precedence over defaults when duplicates exist. See notes for usage and allowable ASCII encodings.

Usage

map(m1, m2)

Arguments

m1  
a vector of IPA characters to encode.

m2  
a vector of ALINE encodings.

Value

map  
a dataframe with ALINE map that includes user-defined mappings.

Note

Valid ASCII ALINE encodings are defined in Kondrak 1999, pp. 19. Allowable lowercase letters are ["a"-"z"] and allowable uppercase modifiers are: _D_ental, palato-al_V_olar, retrofle_X_, _P_alatal, _S_pirant, _N_asal, _A_spirated, lo_H_ng, _F_ront, _C_entral. If an IPA character is mapped to an invalid ASCII code a warning is issued and the mapping is not accepted.

Author(s)

Sean Downey and Guowei Sun

References


See Also

ALINE.map
optimization.GA

Core optimization function for finding optimal weights.

Description

Cognate pairs and their determined alignment generated in ALINE format is used to find a set of optimal parameters in terms of number of correctly aligned pairs. A genetic algorithm is executed. It is called in the optimize.features function and is the core function for the optimization part.

Usage

optimization.GA(A1, data, num, step = 5, plot = TRUE)

Arguments

A1 A 2*n matrix containing the correct alignment of the input data
A 2*n matrix containing the pairs of words to be aligned
num The size of initial population in the genetic algorithm
step number of iterations for the genetic algorithm
plot plot the convergence process of the algorithm

Value

R a list, containing
performance The number of correctly aligned pairs
optimized_parameters a matrix containing all the optimal parameters after the optimization

See Also

optimize.features

Examples

data <- as.matrix(data.frame(dog=paste('dog', 'perro'), cat=paste('cat', 'gato'), rat=paste('rat', 'rata')))
M <- generate.training(raw.data=data, search.size=100, table=FALSE)
alignment <- rbind(M[[2]][4,], M[[2]][5,])
optimization.GA(A1=alignment, data=data, num=5, step=3, plot=FALSE)
optimize.features

Supervised learning with a genetic algorithm

Description

Runs a genetic algorithm to find optimal parameter settings based on expert alignment determinations.

Usage

```r
optimize.features(set, ranking, num = 200, step = 45, replication = 5,
list = FALSE)
```

Arguments

- **set**: the output from `set.generation` function, which is a two element list containing the original word pairs and possible alignments.
- **ranking**: a vector specifying the correct alignment in the candidate alignments generated.
- **num**: number of populations in the genetic algorithm.
- **step**: number of iterations in the genetic algorithm.
- **replication**: number of independent genetic algorithm optimizations.
- **list**: Whether or not to return the entire result of the genetic algorithm which contains a big list of possible parameters and corresponding performance in each independent replication.

Value

If `list=FALSE`, the function returns a single vector representing the optimal parameter values.

If `list=TRUE`, the function returns a list where each top-level element corresponds to the number of replications. Within each replicate, two elements are returned:

- **performance**: Performance values for each population.
- **optimized_parameters**: Feature values at each step in the optimization process.

Examples

```r
# This simplified example illustrates the supervised learning workflow
# some cognate data
data<-data.frame(dog=c('dog','perro'),cat=c('cat','gato'),rat=c('rat','rata'))

# generate training data for linguist (not written)
M1<-generate.training(raw.data=data, search.size=100)

# optimize features using expert determinations: 1,1,1
optimize.features(set=M1, ranking=c(1,1,1),
num=20, step=5, replication=2, list=FALSE)
```
Description

The R/C++ interface functions to ALINE. It is called by aline(), which is the preferred way to access it in most cases. The default features weights are those defined in Kondrak (2000).

Usage

raw.alignment(s,
Syllabic = 5, Place = 40, Stop = 50,
Voice = 10, Nasal = 10, Retroflex = 10,
Lateral = 10, Aspirated = 5, Long = 1,
High = 5, Back = 5, Round = 5, sk=10)

Arguments

s  A pair of ASCII-encoded words as defined by ALINE.map().
Syllabic, Place, Stop, Voice, Nasal, Retroflex, Lateral, Aspirated, Long, High, Back, Round Feature weights used to determine the optimal alignment.

sk  The skip penalty used to determine the optimal alignment.

Value

A list containing the following elements:

'word pairs'  The original word pair in ALINE ASCII encoding.
'similarity score'  The similarity score returned by ALINE.
alignment1  The alignment of the first word presented in ALINE symbols.
alignment2  The alignment of the second word presented in ALINE symbols.

References


Examples

s<-c("digNgNira","dinnira")
raw.alignment(s)
show.map

---

**Description**

Generating a dataframe containing the IPA and ASCII representation of the integer values stored in map.

**Usage**

`show.map()`

**Details**

CRAN policy specifies IPA characters cannot be stored in the dataframe ALINE.map. This function enables users to see the actual IPA characters and how they are mapped to ALINE encodings.

**Value**

A dataframe containing the following columns:

<table>
<thead>
<tr>
<th>IPA</th>
<th>IPA characters</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALINE</td>
<td>ALINE characters</td>
</tr>
<tr>
<td>uNval</td>
<td>Integer values for the IPA characters</td>
</tr>
<tr>
<td>aNval</td>
<td>Integer values for the ALINE characters</td>
</tr>
</tbody>
</table>

**Examples**

`show.map()`
Index

*Topic Candiate Alignments to Choose From
   generate.training, 10
*Topic Genetic Algorithm
   optimize.features, 13
*Topic Optimal Parameter
   optimize.features, 13
*Topic Similarity Score
   ALINE.segments, 6
*Topic \textasciitilde{kwd1}
   features.plot, 9
   optimization.GA, 12
*Topic \textasciitilde{kwd2}
   features.plot, 9
   optimization.GA, 12
*Topic alignment
   ALINE.segments, 6
*Topic csv Table
   generate.training, 10
*Topic datasets
   ALINE.map, 5
*Topic package
   aline-package, 2

aline, 3
aline-package, 2
ALINE.map, 5, 11
ALINE.segments, 6

decode.ALINE, 7, 8
encode.ALINE, 7, 8
features.plot, 9
generate.training, 9, 10
map, 4, 5, 11

optimization.GA, 12
optimize.features, 9, 10, 12, 13