Package 'AcousticNDLCodeR'

January 20, 2025

Type Package

Title Coding Sound Files for Use with NDL

Version 1.0.2

Date 2018-07-06

Maintainer Denis Arnold <arnold@ids-mannheim.de>

Depends R (>= 3.0.0)

Description Make acoustic cues to use with the R packages 'ndl' or 'ndl2'. The package implements functions used
in the PLoS ONE paper:
Denis Arnold, Fabian Tomaschek, Konstantin Sering, Florence Lopez, and R. Harald Baayen (2017).
Words from spontaneous conversational speech can be recognized with human-like accuracy by an error-driven learning algorithm that discriminates between meanings straight from smart acoustic features, bypassing the phoneme as recognition unit. PLoS ONE 12(4):e0174623 <doi:10.1371/journal.pone.0174623>
More details can be found in the paper and the supplement.
'ndl' is available on CRAN. 'ndl2' is available by request from <konstantin.sering@uni-tuebingen.de>.

Imports tuneR, zoo, seewave, parallel

License GPL (>= 2)

LazyData TRUE

RoxygenNote 6.0.1

NeedsCompilation no

Author Denis Arnold [aut, dtc, cre],
Elnaz Shafaei Bajestan [ctb]

Repository CRAN

Date/Publication 2018-07-06 12:00:03 UTC

Contents

`AcousticNDLCodeR` *AcousticNDLCodeR-Package*

Description

Package to make acoustic cues to use with ndl or ndl2.

Details

The packages main function is `makeCues`. `readTextGridFast`, `readTextGridRobust`, `readESPSAnnotation` and `readWavesurfer` are helper functions that read the corresponding annotation files and return a data.frame. `CorpusCoder` codes a whole corpus given a vector with the path to and names of wave files and a vector for the annotation files. `word_classification_data` provides data from Arnold et al 2017 https://doi.org/10.1371/journal.pone.0174623

Author(s)

Denis Arnold

References

Reference to to paper in accepted form.

Examples

```
## Not run:
# assuming the corpus contains wave files and praat textgrids

setwd(~/Data/MyCorpus) # assuming everything is in one place

#assuming you have one wav for each annotation

Waves=list.files(pattern="*.wav",recursive=T)
Annotations=list.files(pattern="*.TextGrids",recursive=T) # see above

# Lets assume the annotation is in UTF-8 and you want everything from a tier called words
# Lets assume tha you want to dismiss everything in <|>
# Lets assume that have 4 cores available
```

```
# Lets assume that you want the defaut settings for the parameters

Data=CorpusCoderCorpusCoder(Waves, Annotations, AnnotationType = "TextGrid",
TierName = "words", Dismiss = "<|>", Encoding, Fast = F, Cores = 4,
IntensitySteps = 5, Smooth = 800)


## End(Not run)
```

CODE *Helper function for makeCues*

Description

Helper function for makeCues

Usage

```
CODE(SPEC, num)
```

Arguments

SPEC	Spectrum representation made in makeCues()
num	Number of the part

Value

A string containing the coding. Each band is seperated by "_".

Author(s)

Denis Arnold

CorpusCoder *Codes a corpus for use with NDL with vector of wavefile names and a vector of TextGrid names provided*

Description

Codes a corpus for use with NDL with vector of wavefile names and a vector of TextGrid names provided

Usage

```
CorpusCoder(Waves, Annotations, AnnotationType = c("TextGrid", "ESPS"),
  TierName = NULL, Dismiss = NULL, Encoding, Fast = F, Cores = 1,
  IntensitySteps, Smooth)
```

Arguments

Waves	Vector with names (and full path to if not in wd) of the wave files.
Annotations	Vector with names (and full path to if not in wd) of the TextGrid files.
AnnotationType	Type of annotation files. Suported formats are praat TextGrids (set to "TextGrid") and ESPS/Wavesurfer (set to "ESPS") files.
TierName	Name of the tier in the TextGrid to be used.
Dismiss	Regular expression for Outcomes that should be removed. Uses grep. E.g. "<|>" would remove <noise>,<xxx>, etc. Default is NULL.
Encoding	Encoding of the annotation file. It is assumed, that all annotation files have the same encoding.
Fast	Switches between a fast and a robust TextGrid parser. For Fast no "\n" or "\t" may be in the transcription. Default is FALSE.
Cores	Number of cores that the function may use. Default is 1.
IntensitySteps	Number of steps that the intensity gets compressed to. Default is 5
Smooth	A parameter for using the kernel smooth function provied by the package zoo.

Value

A data.frame with $Cues and $Outcomes for use with ndl or ndl2.

Author(s)

Denis Arnold

Examples

```
 ## Not run:
 # assuming the corpus contains wave files and praat textgrids

   setwd(~/Data/MyCorpus) # assuming everything is in one place

   #assuming you have one wav for each annotation

   Waves=list.files(pattern="*.wav",recursive=T)
   Annotations=list.files(pattern="*.TextGrids",recursive=T) # see above

# Lets assume the annotation is in UTF-8 and you want everything from a tier called words
   # Lets assume tha you want to dismiss everything in <|>
   # Lets assume that have 4 cores available
   # Lets assume that you want the defaut settings for the parameters

   Data=CorpusCoderCorpusCoder(Waves, Annotations, AnnotationType = "TextGrid",
   TierName = "words", Dismiss = "<|>", Encoding, Fast = F, Cores = 4,
   IntensitySteps = 5, Smooth = 800)


## End(Not run)
```

getBoundary *Helper function for makeCues that splits the signal based on the envelope of the signal*

Description

Helper function for makeCues that splits the signal based on the envelope of the signal

Usage

```
getBoundary(Wave, smooth = 800)
```

Arguments

Wave	A Wave object (see tuneR)
smooth	A parameter for using the kernel smooth function provied by the package zoo.

Value

A vector with the sample numbers of the boundaries.

Author(s)

Denis Arnold

Examples

```
## Not run:
library(tuneR)
Wave=readWave("MyWaveFile.wav")
Boundaries=getBoundary(Wave,800)

## End(Not run)
```

makeCues *Creates a string with the cues for each frequency band and segment seperated by "_"*

Description

Creates a string with the cues for each frequency band and segment seperated by "_"

Usage

```
makeCues(WAVE, IntensitySteps = 5, Smooth = 800)
```

Arguments

WAVE	A Wave object (see tuneR). Currently it is implemented for use with 16kHz sampling rate.
IntensitySteps	Number of steps that the intensity gets compressed to. Default is 5.
Smooth	A parameter for using the kernel smooth function provied by the package zoo.

Value

A string containing the coding. Each band and part is seperated by "_"

Author(s)

Denis Arnold

Examples

```
## Not run:

        library(tuneR)
        library(seewave)
        Wave=readWave("MyWaveFile.wav")
        if(Wave@samp.rate!=16000){
        Wave=resamp(Wave,f=Wave@samp.rate,g=16000,output="Wave")
        }
        Cues=makeCues(Wave,IntensitySteps=5,Smooth=800)


## End(Not run)
```

---

readESPSAnnotation *Reads a ESPS/Old Wavesurfer style annotation file and returns a data.frame with times and lables*

Description

Reads a ESPS/Old Wavesurfer style annotation file and returns a data.frame with times and lables

Usage

```
readESPSAnnotation(File, Encoding)
```

Arguments

File	Name (with full path, if not in wd) of the annotation file
Encoding	Encoding of the annotation file. Typically encodings are "ACSII","UTF-8" or "UTF-16"

Value

A data.frame with $Output for the lable $start and $end time of the lable.

Author(s)

Denis Arnold

Examples

```
## Not run:
# Assume that NameOfAnnotation is encoded in "UTF-8"
Data=readESPSAnnotation("NameOfTextGrid","UTF-8")

## End(Not run)
```

`readTextGridFast` *Reads a TextGrid made with praat and returns a list with a vector of all tier names and a data.frame for each tier.*

Description

Reads a TextGrid made with praat and returns a list with a vector of all tier names and a data.frame for each tier.

Usage

```
readTextGridFast(File, Encoding)
```

Arguments

`File`	Name (with full path, if not in wd) of the TextGrid
`Encoding`	Encoding of the TextGrid. Typically encodings are "ACSII","UTF-8" or "UTF-16"

Details

This method has sometimes problems with certain sequences like "\n" in the annotation file. If the method fails, try readTextGridRobust()

Value

A list containing a vectors with the names and data.frames for each tier in the TextGrid.

Author(s)

Denis Arnold

Examples

```
## Not run:
# Assume that NameOfTextGrid is encoded in "UTF-8"
Data=readTextGridFast("NameOfTextGrid","UTF-8")


## End(Not run)
```

`readTextGridRobust` *Reads a TextGrid made with praat and returns a list with a vector of all tier names and a data.frame for each tier*

Description

Reads a TextGrid made with praat and returns a list with a vector of all tier names and a data.frame for each tier

Usage

```
readTextGridRobust(File, Encoding)
```

Arguments

`File`	Name (with full path, if not in wd) of the TextGrid
`Encoding`	Encoding of the TextGrid. Typically encodings are "ACSII","UTF-8" or "UTF-16"

Value

A list containing a vectors with the names and data.frames for each tier in the TextGrid.

Author(s)

Denis Arnold

Examples

```
## Not run:
# Assume that NameOfTextGrid is encoded in "UTF-8"
Data=readTextGridRobust("NameOfTextGrid","UTF-8")


## End(Not run)
```

readWavesurfer *Reads a New Wavesurfer style annotation file and returns a data.frame with times and lables*

Description

Reads a New Wavesurfer style annotation file and returns a data.frame with times and lables

Usage

```
readWavesurfer(File, Encoding)
```

Arguments

File	Name (with full path, if not in wd) of the annotation file
Encoding	Encoding of the annotation file. Typically encodings are "ACSII","UTF-8" or "UTF-16"

Value

A data.frame with $Output for the lable $start and $end time of the lable.

Author(s)

Denis Arnold

Examples

```
## Not run:
# Assume that NameOfAnnotation is encoded in "UTF-8"
Data=readWavesurfer("NameOfTextGrid","UTF-8")

## End(Not run)
```

word_classification_data *Data of PLoS ONE paper*

Description

Dataset of a subject and modeling data for an auditory word identification task.

Usage

```
data(word_classification_data)
```

Format

Data from the four experiments and model estimates

`ExperimentNumber` Experiment identifier

`PresentationMethod` Method of presentation in the experiment: loudspeaker, headphones 3. Trial: Trial number in the experimental list

`TrialScaled` scaled Trial

`Subject` anonymized subject identifier

`Item` word identifier -german umlaute and special character coded as 'ae' 'oe' 'ue' and 'ss'

`Activation` NDL activation

`LogActivation` log(activation+epsilon)

`L1norm` L1-norm (lexicality)

`LogL1norm` log of L1-norm

`RecognitionDecision` recognition decision (yes/no)

`RecognitionRT` latency for recognition decision

`LogRecognitionRT` log recognition RT

`DictationAccuracy` dictation accuracy (TRUE: correct word reported, FALSE otherwise) 15. DictationRT: response latency to typing onset

References

Denis Arnold, Fabian Tomaschek, Konstantin Sering, Florence Lopez, and R. Harald Baayen (2017). Words from spontaneous conversational speech can be recognized with human-like accuracy by an error-driven learning algorithm that discriminates between meanings straight from smart acoustic features, bypassing the phoneme as recognition unit PLoS ONE 12(4):e0174623. https://doi.org/10.1371/journal.pone.01746

Index