Package 'JATSdecoder'

November 22, 2021

Title JATSdecoder

Version 1.1

Description This package contains a function collection to extract metadata, sectioned text and study characteristics from scientific articles. Its function JATSdecoder() converts NISO-JATS-tagged XML files to a structured list with elements containing title, author, journal, history, link, abstract, sectioned text and references. Studies in PDF format can be easily converted to NISO-JATS with the open source software CERMINE (https://github.com/CeON/CERMINE/). JATSdecoders function study.character() extracts multiple study characteristics like number of included studies, statistical methods used, alpha error, power, statistical results, correction method for multiple testing, software used. Based on different heuristics, it will perform a reliable estimation of studies sample size soon (in progress). The package contains a set of useful functions to unify and transform numerical representations within text.

Depends R (>= 3.1.1)

Imports utils,
stats,
rworldmap,
rorcid,
tm,
igraph,
countrycode,
geosphere,
rgeos,
RColorBrewer

License CC0

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

R topics documented:

allStats *allStats*

Description

Extract any statistical results from text with some uniformizations.

Usage

```
allStats(x)
```

Arguments

x	text to extract statistical results from

Examples

```
x<-c("The mean difference of scale A was significant (beta=12.9, t(18)=2.5, p<.05)",
"The ANOVA yielded significant results on
 faktor A (F(2,18)=6, p<.05, eta(g)2<-.22)",
"the correlation of x and y was r=.37.")
allStats(x)
```

est.ss *est.ss*

Description

Function to estimate studies sample size by maximizing different conservative estimates. Performs 4 different heuristic driven search tasks for reoprted sample size in abstract, text, stats and standardStats.

Usage

```
est.ss(
  abstract = NULL,
  text = NULL,
  quantileDF = 0.75,
  max.only = FALSE,
  max.parts = TRUE
)
```

Arguments

abstract	abstracts text
text	main text to process (usually method section)
quantileDF	quantile of (df1+1)+(df2+2) to extract
max.only	Logical. If TRUE only the final estimate will be returned, if FALSE all sub estimates are returned as well
max.parts	Logical. If FALSE outputs all captured sample sizes in sub inputs

Details

Sample size extraction from abstract:
- Extracts N= from abstract if possible

Sample size extraction from statistical results:
- Extracts "N=" in statistical result lines from get.stats() that contain p-value

Sample size extraction with result of standardStats(get.stats()):
- Extracts df1 and df2 if possible and neither containing a ".", than calculates quantile of (df1+1)+(df2+2) (at least 2 group comparison assumed)

Examples

```
a<-"One hundred twelve students participated in our study."
x<-"Our sample consists of three hundred twenty five undergraduate students.
   For one sub group the F-test indicates significant differences in means F(2,102)=3.21, p<.05."
est.ss(abstract=a,text=x)
```

---

get.abstract *get.abstract*

Description

Extract abstract tag from NISO-JATS coded XML file or text as vector of abstracts

Usage

```
get.abstract(
  x,
  sentences = FALSE,
  remove.title = TRUE,
  letter.convert = TRUE,
  cermine = FALSE
)
```

Arguments

x	a NISO-JATS coded XML file or text
sentences	Logical. If TRUE abstract is returned as vector of sentences
remove.title	Logical. If TRUE removes section titles in abstract
letter.convert	Logical. If TRUE converts hexadecimal and HTML coded characters to Unicode
cermine	Logical. If TRUE and letter.convert=TRUE performs CERMINE specific text correction

Examples

```
x<-"Some text <abstract>Some abstract</abstract> some text"
get.abstract(x)
x<-"Some text <abstract>Some abstract</abstract> TEXT <abstract with subsettings>
Some other abstract</abstract> Some text "
get.abstract(x)
```

---

get.aff *get.aff*

Description

Extract affiliation tag/s from NISO-JATS coded XML file or text as vector of affiliations

Usage

```
get.aff(x, remove.html = FALSE, letter.convert = TRUE)
```

Arguments

x	a NISO-JATS coded XML file or text
remove.html	Logical. If TRUE removes all html tags
letter.convert	Logical. If TRUE converts hexadecimal and HTML coded characters to Unicode

Examples

```
x<-"Some text <aff>Some affiliation</aff> some text"
get.aff(x)
x<-"Some text <aff>Some affiliation</aff> TEXT <aff>Some other affiliation</aff> Some text "
get.aff(x)
```

get.alpha.error *get.alpha.error*

Description

Extract reported and corrected alpha error from text and 1-alpha confidence intervalls

Usage

```
get.alpha.error(x, p2alpha = FALSE, output = "list")
```

Arguments

x	text to process
p2alpha	Logical. If TRUE detects and extracts alpha errors denoted with critical p-value (what may lead to some false positive detections)
output	One of "list" (list with elements: alpha_error, corrected_alpha, alpha_from_CI, alpha_max, alpha_min), vector with unique alpha errors but no distinction of types

Examples

```
x<-c("The threshold for significance was adjusted to .05/2",
"Type 1 error rate was alpha=.05.")
get.alpha.error(x)
x<-c("We used p<.05 as level of significance.",
     "We display .95 CIs and use an adjusted alpha of .10/3.",
     "The effect was significant with p<.025.")
get.alpha.error(x)
```

get.assumptions *get.assumptions*

Description

Extract mentioned assumptions in text out of list with 22 statistical assumptions

Usage

```
get.assumptions(x, hits_only = TRUE)
```

Arguments

x	text to process
hits_only	Logical. If TRUE returns the detected assumtions only, else a hit matrix with all potential assumptions

Examples

```
x<-"Sphericity assumption and gaus-marcov was violated."
get.assumptions(x)
```

`get.author` *get.author*

Description

Extract author tag/s from NISO-JATS coded XML file or text as vector of authors

Usage

```
get.author(x, paste = "", short.names = FALSE, letter.convert = FALSE)
```

Arguments

`x`	a NISO-JATS coded XML file or text
`paste`	if "" author list is exported as vector with length of number of authors, else collapsed to one cell
`short.names`	Logical. If TRUE fully available first names will be reduced to one letter abbreviation
`letter.convert`	Logical. If TRUE converts hexadecimal and HTML coded characters to Unicode

---

`get.category` *get.category*

Description

Extract category tag/s from NISO-JATS coded XML file or text as vector of categories

Usage

```
get.category(x)
```

Arguments

`x`	a NISO-JATS coded XML file or text

Examples

```
x<-"Some text <article-categories>Some category</article-categories> some text"
get.category(x)
```

get.contrib *get.contrib*

Description

Extract contrib tag/s from NISO-JATS coded XML file or text as vector of contributers

Usage

```
get.contrib(x, remove.html = FALSE, letter.convert = FALSE)
```

Arguments

x	a NISO-JATS coded XML file or text
remove.html	Logical. If TRUE removes all HTML tags
letter.convert	Logical. If TRUE converts hexadecimal and HTML coded characters to Unicode

get.country *get.country*

Description

Extract country tag from NISO-JATS coded XML file or text as vector of unique countries

Usage

```
get.country(x, unifyCountry = TRUE)
```

Arguments

x	a NISO-JATS coded XML file or text
unifyCountry	Logical. If TRUE replaces country name with standardised country name

Examples

```
x<-"Some text <country>UK</country> some text <country>England</country>
    Text<country>Berlin, Germany</country>"
get.country(x)
```

get.doi *get.doi*

Description

Extract articles doi from NISO-JATS coded XML file or text

Usage

```
get.doi(x)
```

Arguments

x	a NISO-JATS coded XML file or text

get.editor *get.editor*

Description

Extract editor tag from NISO-JATS coded XML file or text as vector of editor/s

Usage

```
get.editor(x, role = FALSE, short.names = FALSE, letter.convert = FALSE)
```

Arguments

x	a NISO-JATS coded XML file or text
role	Logical. If TRUE adds role to editor name, if available
short.names	Logical. If TRUE reduces fully available first names to one letter abbreviation
letter.convert	Logical. If TRUE converts hexadecimal and HTML coded characters to Unicode

get.history *get.history*

Description

Extract available publishing history tags from NISO-JATS coded XML file or text and compute pubDate and pubyear

Usage

```
get.history(x, remove.na = FALSE)
```

Arguments

x	a NISO-JATS coded XML file or text
remove.na	Logical. If TRUE hides non available date stamps

get.journal *get.journal*

Description

Extract journal tag from NISO-JATS coded XML file or text

Usage

```
get.journal(x)
```

Arguments

x	a NISO-JATS coded XML file or text

Examples

```
x<-"Some text <journal-title>PLoS One</journal-title> some text"
get.journal(x)
```

get.keywords *get.keywords*

Description

Extract keyword tag/s from NISO-JATS coded XML file or text as vector of keywords

Usage

```
get.keywords(
  x,
  paste = "",
  letter.convert = TRUE,
  include.max = length(keyword)
)
```

Arguments

x	a NISO-JATS coded XML file or text
paste	if paste!="" author vector is collapsed to one cell
letter.convert	Logical. If TRUE converts hexadecimal and HTML coded characters to Unicode
include.max	a maximum number of keywords to extract

Examples

```
x<-"Some text <kwd>Keyword 1</kwd>, <kwd>Keyword 2</kwd> some text"
get.keywords(x)
get.keywords(x,paste(", "))
```

get.method *get.method*

Description

Extract statistical methods mentioned in text

Usage

```
get.method(x, add = NULL, cermine = FALSE)
```

Arguments

x	text to extract statistical methods from
add	possible new end words of method as vector
cermine	Logical. If TRUE CERMINE specific letter conversion will be performed

Examples

```
x<-"We used multiple regression analysis and
two sample t tests to evaluate our results."
get.method(x)
```

`get.multi.comparison` *get.multi.comparison*

Description

Extracts alpha-/p-value correction method for multiple comparisons from list with 15 correction methods

Usage

```
get.multi.comparison(x)
```

Arguments

x	text to process

Examples

```
x<-"We used Bonferroni corrected p-values."
get.multi.comparison(x)
```

`get.n.studies` *get.n.studies*

Description

Extract n studies/experiments from section titles or abstract text

Usage

```
get.n.studies(x, tolower = TRUE)
```

Arguments

x	section titles or abstract text to process
tolower	Logical. If TRUE lowerises text and search patterns for processing

get.outlier.def *get.outlier.def*

Description

Extract outlier/extreme value definition/removal in standard deviations, if present in text

Usage

```
get.outlier.def(x)
```

Arguments

x	text to process

Examples

```
x<-"We removed 4 extreme values that were 3 SD above mean."
get.outlier.def(x)
```

get.power *get.power*

Description

Extract a priori power, empirial power values and 1-betaerror

Usage

```
get.power(x)
```

Arguments

x	text to process

Examples

```
x<-"We used G*Power 3 to calculate the needed sample with
beta error rate set to 12% and alpha error to .05."
get.power(x)
```

get.R.package *get.R.package*

Description

Extract mentioned R package from text

Usage

```
get.R.package(x, update.package.list = FALSE)
```

Arguments

x	text to process
update.package.list	Logical. If TRUE update of list with available packages is downloaded from CRAN with available.packages()

Examples

```
get.R.package("We used the R Software packages lme4 (and psych).")
```

get.references *get.references*

Description

Extract reference list from NISO-JATS coded XML file or text as vector of references

Usage

```
get.references(
  x,
  letter.convert = FALSE,
  remove.html = FALSE,
  extract = "full"
)
```

Arguments

x	a NISO-JATS coded XML file or text
letter.convert	Logical. If TRUE converts hexadecimal and HTML coded characters to Unicode
remove.html	Logical. If TRUE removes all HTML tags
extract	part of refernces to extract (one of "full" or "title")

get.sentence.with.pattern

get.sentence.with.pattern

Description

Return lines with search term patterns

Usage

```
get.sentence.with.pattern(x, patterns = c(""), tolower = TRUE)
```

Arguments

x	text to process
patterns	search terms
tolower	Logical. If TRUE converts search terms and text to lower case

Examples

```
text<-c("This demo demonstrates how get.sentence.with.pattern works.","The is a simple 0, 1.")
get.sentence.with.pattern(text,c("Demo","example","work"))
get.sentence.with.pattern(text,c("Demo","example","work"),tolower=TRUE)
```

get.sig.adjectives

get.sig.adjectives

Description

Extract adjectives used for in/significance out of list with 37 potential adjectives

Usage

```
get.sig.adjectives(x, unique_only = FALSE)
```

Arguments

x	text to process
unique_only	Logical. If TRUE returns unique hits only.

Examples

```
get.sig.adjectives(
 x<-"We found very highly significance for type 1 effect"
)
```

get.software *get.software*

Description

Extract mentioned software from text by dictionary search for 63 software names (object: .software_names)

Usage

```
get.software(x, add.software = NULL)
```

Arguments

x	text
add.software	a text vector with additional software name patterns to search for

Examples

```
get.software("We used the R Software and Excel 4.0 to analyse our data.")
```

get.stats *get.stats*

Description

Extract statistical results from plain text, XML, CERMXML, HTML, HTM or DOCX files. The result is a list with a vector containing all identified sticked results and a matrix with containing reported standard statistics and recalculated p-values if computation is possible.

Usage

```
get.stats(
  x,
  output = "both",
  stats.mode = "all",
  recalculate.p = TRUE,
  alternative = "undirected",
  estimateZ = FALSE,
  T2t = FALSE,
  R2r = FALSE,
  select = NULL,
  rm.na.col = TRUE,
  cermine = FALSE
)
```

Arguments

x	text or JATScoded XML file to extract statistical results from
output	Select the desired output. One of c("both","allStats","standardStats")
stats.mode	Select subset of standardStats. One of: "all", "checkable", "computable", "uncomputable"
recalculate.p	Logical. If TRUE recalculates p-values of standardStats if possible
alternative	Character. Select sidedness of recomputed p-values from t-, r- and beta-values. One of c("undirected","directed","both")
estimateZ	Logical. If TRUE detected beta-/d-value is divided by reported standard error "SE" to estimate Z-value ("Zest") for observed beta/d and recompute p-value. Note: This is only valid, if Gauss-Marcov assumptions are met and a sufficiently large sample size is used. If a Z- or t-value is detected in a report of a beta-/d-coefficient with SE, no estimation will be performed, although set to TRUE.
T2t	Logical. If TRUE capital letter T is treated as t-statistic
R2r	Logical. If TRUE capital letter R is treated as correlation
select	Select specific standard statistics only (e.g.: c("t","F","Chi2"))
rm.na.col	Logical. If TRUE removes all columns with only NA from standardStats
cermine	Logical. If TRUE CERMINE specific letter conversion will be peformed on allStats results

Examples

```
x<-c("The mean difference of scale A was significant (beta=12.9, t(18)=2.5, p<.05).",
"The ANOVA yielded significant results on
 faktor A (F(2,18)=6, p<.05, eta(g)2<-.22)",
"the correlation of x and y was r=.37.")
get.stats(x)
```

---

get.subject *get.subject*

Description

Extract subject tag/s from NISO-JATS coded XML file or text as vector of subjects

Usage

```
get.subject(x, letter.convert = TRUE, paste = "")
```

Arguments

x	a NISO-JATS coded XML file or text
letter.convert	Logical. If TRUE converts hexadecimal and HTML coded characters to Unicode
paste	if "" author list is exported as vector with length of number of authors, else collapsed to one cell

Examples

```
x<-"Some text <subject>Some subject</subject> some text"
get.subject(x)
x<-"Some text <subject>Some subject</subject> TEXT <subject>Some other subject</subject> Some text "
get.subject(x)
get.subject(x,paste=", ")
```

`get.tables` *get.tables*

Description

extracts HTML tables as vector of tables

Usage

```
get.tables(x)
```

Arguments

x	HTML file or html text

`get.test.direction` *get.test.direction*

Description

Extract mentioned test direction/s (one sided, two sided, one and two sided) from text

Usage

```
get.test.direction(x)
```

Arguments

x	text to process

get.text *get.text*

Description

Extract main textual content from NISO-JATS coded XML file or text as sectioned text

Usage

```
get.text(
  x,
  sectionsplit = "",
  grepsection = "",
  letter.convert = TRUE,
  greek2text = FALSE,
  sentences = FALSE,
  cermine = "auto",
  rm.table = TRUE,
  rm.formula = TRUE,
  rm.xref = TRUE,
  rm.media = TRUE,
  rm.graphic = TRUE,
  rm.ext_link = TRUE
)
```

Arguments

x	a NISO-JATS coded XML file or text
sectionsplit	search patterns for section split (forced to lower case), e.g. c("intro","method","result","discus")
grepsection	search pattern to reduce text to specific section namings only
letter.convert	Logical. If TRUE converts hexadecimal and HTML coded characters to Unicode
greek2text	Logical. If TRUE some greek letters and special characters will be unified to textual representation. (important to extract stats)
sentences	Logical. IF TRUE text is returned as sectioned list with sentences
cermine	Logical. If TRUE CERMINE specific error handling and letter conversion will be applied. If set to "auto" file name ending with 'cermxml$' will set cermine=TRUE
rm.table	Logical. If TRUE removes <table> tag from text
rm.formula	Logical. If TRUE removes <formula> tags
rm.xref	Logical. If TRUE removes <xref> tag (citing) from text
rm.media	Logical. If TRUE removes <media> tag from text
rm.graphic	Logical. If TRUE removes <graphic> and <fig> tag from text
rm.ext_link	Logical. If TRUE removes <ext link> tag from text

get.title *get.title*

Description

Extract articles title from NISO-JATS coded XML file or text

Usage

```
get.title(x)
```

Arguments

x	a NISO-JATS coded XML file or text

get.type *get.type*

Description

Extract article-type tag from NISO-JATS coded XML file or text

Usage

```
get.type(x)
```

Arguments

x	a NISO-JATS coded XML file or text

get.vol *get.vol*

Description

Extract volume, first and last page from NISO-JATS coded XML file or text

Usage

```
get.vol(x)
```

Arguments

x	a NISO-JATS XML coded file or text

grep2 *grep2*

Description

Extension of grep(). Allows to identify and extract cells with/without multiple search patterns that are connected with AND.

Usage

```
grep2(pattern, x, value = TRUE, invert = FALSE, perl = FALSE)
```

Arguments

pattern	Character vector containing regular expression as cells to be matched in the given character vector.
x	A character vector where matches are sought, or an object which can be coerced by as.character to a character vector. Long vectors are supported.
value	Logical. if FALSE, a vector containing the (integer) indices of the matches determined by grep2 is returned, and if TRUE, a vector containing the matching elements themselves is returned.
invert	Logical. If TRUE return indices or values for elements that do not match.
perl	Logical. Should Perl-compatible regexps be used?

Examples

```
x<-c("ab","ac","ad","bc","bad")
grep2(c("a","b"),x)
grep2(c("a","b"),x,invert=TRUE)
grep2(c("a","b"),x,value=FALSE)
```

---

has.interaction *has.interaction*

Description

Identify interaction/moderator/mediator effect in text

Usage

```
has.interaction(x)
```

Arguments

x	text to process

has.pattern *has.pattern*

Description

Return search term hit vector for all search patterns

Usage

```
has.pattern(x, patterns = c(""), tolower = TRUE)
```

Arguments

x	text to process
patterns	search terms as vector
tolower	Logical. If TRUE converts search terms and text to lower case

Examples

```
text<-c("This demo demonstrates how has.pattern() works.",
        "The result is a simple 0, 1 coded vector for all search patterns.")
has.pattern(text,c("Demo","example","work"))
has.pattern(text,c("Demo","example","work"),tolower=TRUE)
```

JATSdecoder *JATSdecoder*

Description

Function to extract and restructure NISO-JATS coded XML file or text into a list with metadata and text as selectable elements

Usage

```
JATSdecoder(
  x,
  sectionsplit = c("intro", "method", "result", "study", "experiment", "conclu",
    "implica", "discussion"),
  grepsection = "",
  sentences = FALSE,
  abstract2sentences = TRUE,
  output = "all",
  letter.convert = TRUE,
  unify.country.name = TRUE,
  greek2text = FALSE,
```

```
  warning = TRUE,
  countryconnection = FALSE,
  authorconnection = FALSE
)
```

Arguments

x	a NISO-JATS coded XML file or text
sectionsplit	search patterns for section split of text parts (forced to lower case), e.g. c("intro","method","result","discus
grepsection	search pattern in regex to reduce text to specific section only
sentences	Logical. IF TRUE text is returned as sectioned list with sentences
abstract2sentences	Logical. IF TRUE abstract is returned as vector with sentences
output	selection of specific results to output c("all","title","author","affiliation","journal","volume","editor","doi' "abstract", "sections", "text", "tables", "captions", "references")
letter.convert	Logical. If TRUE converts hexadecimal and HTML coded characters to Unicode
unify.country.name	Logical. If TRUE tries to unify country name/s with list of country names from worldmap()
greek2text	Logical. If TRUE converts and unifies several greek letters to textual representation, e.g.: alpha
warning	Logical. If TRUE outputs a warning if processing CERMINE converted PDF files
countryconnection	Logical. If TRUE outputs country connections as vector c("A - B","A - C", ...)
authorconnection	Logical. If TRUE outputs connections of a maximum of 50 involved authors as vector c("A - B","A - C", ...)

---

letter.convert *letter.convert*

Description

Convert and unify most hexadecimal and some HTML coded letters in text to Unicode characters and correct CERMINE specific errors in captured statistical results.

Usage

```
letter.convert(x, cermine = FALSE, greek2text = FALSE, warning = TRUE)
```

Arguments

x	text to process
cermine	Logical. If TRUE CERMINE specific error handling and letter conversion will be applied
greek2text	Logical. If TRUE some greek letters and special characters will be unified to textual representation. (important to extract stats)
warning	Logical. If TRUE prints warning massage if CERMINE specific letter conversion was performed

Examples

```
x<-c("five &#x0003c; ten","five &lt; ten")
letter.convert(x)
```

---

ngram *ngram*

Description

Extract ngram bag of words around a pattern match in a text vector. Note: If an input contains the search pattern twice, only the ngram bag of words of the last hit is detected. Consider individual text splitting with text2sentences() or strsplit2() before applying ngram().

Usage

```
ngram(
  x,
  pattern,
  ngram = c(-3, 3),
  tolower = FALSE,
  split = FALSE,
  exact = FALSE
)
```

Arguments

x	vector of text to process
pattern	a search term pattern to extract the ngram bag of words
ngram	a vector of length=2 that defines the number of words to extract from left and right side of pattern match
tolower	Logical. If TRUE converts text and pattern to lower case
split	Logical. If TRUE splits text input at "[.,;:] " before processing. Note: You may consider other text splits before
exact	Logical. If TRUE only exact word matches will be proceses

Examples

```
text<-"One hundred twenty-eight students participated in our Study,
that was administred in thirteen clinics."
ngram(text,pattern="study",ngram=c(-1,2))
```

`standardStats` *standardStats*

Description

Extract and restructure standard statistical results like Z, t, Cohen's d, F, eta^2, r, R^2, chi^2, BF_10, Q, U, H, OR, RR, beta values to a matrix. Performs a recomputation of p-values if possible.

Usage

```
standardStats(
  x,
  stats.mode = "all",
  recalculate.p = TRUE,
  alternative = "undirected",
  estimateZ = FALSE,
  T2t = FALSE,
  R2r = FALSE,
  select = NULL,
  rm.na.col = TRUE
)
```

Arguments

`x`	result of allStats()
`stats.mode`	Select subset of standard stats. One of: "all", "checkable", "computable", "uncomputable"
`recalculate.p`	Logical. If TRUE recalculates p values (for 2 sided test) if possible
`alternative`	Character. Select sidedness of recomputed p-values from t-, r- and beta-values. One of c("undirected","directed","both")
`estimateZ`	Logical. If TRUE detected beta-/d-value is divided by reported standard error "SE" to estimate Z-value ("Zest") for observed beta/d and recompute p-value. Note: This is only valid, if Gauss-Marcov assumptions are met and a sufficiently large sample size is used. If a Z- or t-value is detected in a report of a beta-/d-coefficient with SE, no estimation will be performed, although set to TRUE.
`T2t`	Logical. If TRUE capital letter T is treated as t-statistic
`R2r`	Logical. If TRUE capital letter R is treated as correlation
`select`	Select specific standard statistics only (e.g.: c("t","F","Chi2"))
`rm.na.col`	Logical. If TRUE removes all columns with only NA

Examples

```
x<-c("t(38.8)<=>1.96, p=.002","F(2,39)<=>4, p<=>.05",
"U(2)=200, p>.25","Z<=>2.1, F(20.8,22.6)=200, p<.005,
BF(01)<=>4","chi=3.2, r(34)<=>-.7, p<.01, R2=76%.")
standardStats(x)
```

`strsplit2` *strsplit2*

Description

Extension of strsplit(). Makes it possible to split lines "before" or "after" a pattern match

Usage

```
strsplit2(x, split, type = "remove", perl = FALSE)
```

Arguments

`x`	text to process
`split`	pattern to split text at
`type`	one out of "remove", "before", "after"
`perl`	Logical. If TRUE uses perl expressions

Examples

```
x<-"This is some text, where text is the split pattern of the text."
strsplit2(x,"text","after")
```

`study.character` *study.character*

Description

Extracts study characteristics out of a NISO-JATS coded XML file or JATSdecoder result

Usage

```
study.character(
  x,
  stats.mode = "all",
  recalculate.p = TRUE,
  alternative = "auto",
  estimateZ = FALSE,
  T2t = FALSE,
  R2r = FALSE,
  selectStandardStats = NULL,
  p2alpha = FALSE,
  alpha_output = "list",
  captions = TRUE,
  text.mode = 1,
  update.package.list = FALSE,
  add.software = NULL,
  quantileDF = 0.75,
  N.max.only = FALSE,
  output = "all",
  rm.na.col = TRUE
)
```

Arguments

x	JATS coded XML file or JATSdecoder result
stats.mode	Character. Select subset of standard stats. One of: "all", "checkable", "computable"
recalculate.p	Logical. If TRUE recalculates p values (for 2 sided test) if possible
alternative	Character. Select sidedness of recomputed p-values for t-, r- and Z-values. One of c("auto","undirected","directed","both"). If set to "auto" 'alternative' will be be set to 'both' if get.test.direction() detects one-directional hypotheses/tests in text. If no directional hypotheses/tests are dtected only "undirected" recomputed p-values will be returned
estimateZ	Logical. If TRUE detected beta-/d-value is divided by reported standard error "SE" to estimate Z-value ("Zest") for observed beta/d and recompute p-value. Note: This is only valid, if Gauss-Marcov assumptions are met and a sufficiently large sample size is used. If a Z- or t-value is detected in a report of a beta-/d-coefficient with SE, no estimation will be performed, although set to TRUE.
T2t	Logical. If TRUE capital letter T is treated as t-statistic when extracting statistics with get.stats()
R2r	Logical. If TRUE capital letter R is treated as correlation when extracting statistics with get.stats()
selectStandardStats	Select specific standard statistics only (e.g.: c("t","F","Chi2"))
p2alpha	Logical. If TRUE detects and extracts alpha errors denoted with critical p-value (what may lead to some false positive detections)

alpha_output	One of "list" (list with elements: alpha_error, corrected_alpha, alpha_from_CI, alpha_max, alpha_min), vector with unique alpha errors but no distinction of types
captions	Logical. If TRUE captions text will be scanned for statistical results
text.mode	text parts to extract statistical results from (text.mode=1: abstract and full text, text.mode=2: method and result section, text.mode=3: result section only)
update.package.list	if TRUE updates available R packages with available.packages() function
add.software	additional software names to detect as vector
quantileDF	quantile of (df1+1)+(df2+1) to extract for estimating sample size
N.max.only	return only maximum of estimated sample sizes
output	output selection of specific results c("all", "doi", "title", "year", "Nstudies", "methods", "alpha_error", "power", "multi_comparison_correction", "assumptions", "OutlierRemovalInSD", "InteractionModeratorMediatorEffect", "test_direction", "sig_adjectives", "software", "Rpackage", "stats", "standardStats", "estimated_sample_size")
rm.na.col	Logical. If TRUE removes all columns with only NA in extracted standard statistics

---

`study.type` *study.type*

Description

function to identify type of study by list of study types

Usage

```
study.type(title = NULL, text = NULL)
```

Arguments

title	articles title text
text	main text to process

Examples

```
study.type("We performed a randomized treatment control trail with waiting group")
```

text2num *text2num*

Description

Converts special annotated number and written numbers in a text string to a fully digit representation. Can handle numbers with exponent, fraction, percent, e+num, products and written representation (e.g. 'fourtys-one') of all absolut numbers till 99,999 (Note: gives false returns for higher numbers). Process is performed in the same order as its arguments.

Usage

```
text2num(
  x,
  exponent = TRUE,
  percentage = TRUE,
  fraction = TRUE,
  e = TRUE,
  product = TRUE,
  words = TRUE
)
```

Arguments

x	text to process
exponent	Logical. If TRUE values with exponent are converted to a digit representation
percentage	Logical. If TRUE percentages are converted to a digit representation
fraction	Logical. If TRUE fractions are converted to a digit representation
e	Logical. If TRUE values denoted with num e+num (e.g. '2e+2') or num*10^num are converted to a digit representation
product	Logical. If TRUE values products are converted to a digit representation
words	Logical. If TRUE written numbers are converted to a digit representation

Examples

```
x<-c("numbers with exponent: -2^3, .2^-2, -.3^.2, 49^-.5, 2^10.",
     "numbers with percentage: 2%, 15 %, 25 percent.",
     "numbers with fractions: 1/100, -2/5, -7/-.1",
     "numbers with e: 10e+2, -20e3, .2E-2, 2e4",
     "numbers as products: 100*2, -20*.1, 2*10^3",
     "written numbers: twenty-two, one hundred fourty five",
     "mix: one hundred ten is not 1/10 is not 10^2 nor 10%/5")
text2num(x)
```

text2sentences *text2sentences*

Description

Converts floating text to a vector with sentences via fine-tuned regular expressions

Usage

```
text2sentences(x)
```

Arguments

x	text to process

Examples

```
x<-"Some text with result (t(18)=1.2, p<.05). This shows how text2sentences works."
text2sentences(x)
```

vectorize.text *vectorize.text*

Description

convert vector of text to a list of words within each cell. Note: punctuation will be removed

Usage

```
vectorize.text(x)
```

Arguments

x	text to vectorize

Examples

```
text<-"One hundred twenty-eight students participated in our Study,
that was administred in thirteen clinics."
vectorize.text(text)
```

which.term *which.term*

Description

Returns search element/s from vector that is/are present in text or returns search term hit vector for all terms

Usage

```
which.term(x, terms, tolower = TRUE, hits_only = FALSE)
```

Arguments

x	text to process
terms	search terms
tolower	Logical. If TRUE converts search terms and text to lower case
hits_only	Logical. If TRUE returns search pattern/s, that were found in text and not a search term hit vector

Examples

```
text<-c("This demo demonstrates how which.term works.",
       "The result is a simple 0, 1 coded vector for all search patterns or
        a vector including the identified patterns only.")
which.term(text,c("Demo","example","work"))
which.term(text,c("Demo","example","work"),tolower=TRUE,hits_only=TRUE)
```

Index