Package 'word.alignment'

April 15, 2019

Type Package

Title Computing Word Alignment Using IBM Model 1 (and Symmetrization) for a Given Parallel Corpus and Its Evaluation

Version 1.1

Date 2019-04-04

Author Neda Daneshagr and Majid Sarmad.

Maintainer Neda Daneshgar<ne_da978@stu-mail.um.ac.ir>

Description For a given Sentence-Aligned Parallel Corpus, it aligns words for each sentence pair. It considers one-to-many and symmetrization alignments. Moreover, it evaluates the quality of word alignment based on this package and some other software. It also builds an automatic dictionary of two languages based on given parallel corpus.

Depends R(>= 3.2.2), data.table, openxlsx

License GPL (>= 2)

NeedsCompilation no

Repository CRAN

Date/Publication 2019-04-15 09:10:03 UTC

R topics documented:

word.alignment-package

Computing Word Alignment Using IBM Model 1 (and Symmetrization) for a Given Parallel Corpus and Its Evaluation

Description

For a given Sentence-Aligned Parallel Corpus, it aligns words for each sentence pair. It considers one-to-many alignment in the function `align.ibm1` and symmetric word alignment in the function `align.symmet`. Moreover, it evaluates the quality of word alignment from `align.ibm1` function or from some other software in the function `evaluation`. It also builds an automatic bilingual dictionary of two languages using the given corpus in the function `bidictionary`.

Details

Package:	word.alignment
Type:	Package
Version:	1.1
Date:	2019-04-04
License:	GPL (>= 2)

Author(s)

Neda Daneshgar and Majid Sarmad.

Maintainer: Neda Daneshgar<ne_da978@stu-mail.um.ac.ir>

References

Fraser F., Marcu D. (2007), "Measuring Word Alignment Quality for Statistical Machine Translation.", Computational Linguistics, 33(3), 293-303.

Koehn P. (2010), "Statistical Machine Translation.", Cambridge University, New York.

Lopez A. (2008), "Statistical Machine Translation.", ACM Computing Surveys, 40(3).

Peter F., Brown J., (1990), "A Statistical Approach to Machine Translation.", Computational Linguistics, 16(2), 79-85.

Supreme Council of Information and Communication Technology. (2013), Mizan English-Persian Parallel Corpus. Tehran, I.R. Iran. Retrieved from http://dadegan.ir/catalog/mizan.

http://statmt.org/europarl/v7/bg-en.tgz

Och F., Ney H. (2003), "A Systematic Comparison Of Various Statistical Alignment Models.", 2003 Association for Computational Linguistics, J03-1002, 29(1).

Wang X. "Evaluation of Two Word Alignment Systems.", Final Thesis, Department of Computer and Information Science.

Examples

```
# Since the extraction of  bg-en.tgz in Europarl corpus is time consuming,
# so the aforementioned unzip files have been temporarily exported to
# http://www.um.ac.ir/~sarmad/... .

## Not run:

ww = align.ibm1 ('http://www.um.ac.ir/~sarmad/word.a/euro.bg',
                     'http://www.um.ac.ir/~sarmad/word.a/euro.en',
                      n=2000, encode.sorc = 'UTF-8')

ss = align.symmet ('http://www.um.ac.ir/~sarmad/word.a/euro.bg',
                     'http://www.um.ac.ir/~sarmad/word.a/euro.en',
                      n = 50, encode.sorc = 'UTF-8')

## End(Not run)
```

`align.ibm1` *Computing One-to-Many and Symmetric Word Alignment Using IBM Model 1 for a Given Sentence-Aligned Parallel Corpus*

Description

For a given sentence-aligned parallel corpus, it calculates source-to-target and target-to-source alignments using IBM Model 1, as well as symmetric word alignment models such as intersection, union, or grow-diag in each sentence pair. Moreover, it calculates the expected length and vocabulary size of each language (source and taget language) and also shows word translation probability as a data.table.

Usage

```
align.ibm1(...,
              iter = 5, dtfile.path = NULL,
              name.sorc = 'f',name.trgt = 'e',
              result.file = 'result', input = FALSE)
align.symmet(file.sorc, file.trgt,
             n = -1L, iter = 4,
             method = c ('union', 'intersection', 'grow-diag'),
             encode.sorc = 'unknown', encode.trgt = 'unknown',
             name.sorc = 'f', name.trgt = 'e', ...)

## S3 method for class 'align'
print(x,...)
```

Arguments

`file.sorc`	the name of source language file.
`file.trgt`	the name of target language file.

n	the number of sentences to be read.If -1, it considers all sentences.
iter	the number of iterations for IBM Model 1.
method	character string specifying the symmetric word alignment method (union, intersection, or grow-diag alignment).
encode.sorc	encoding to be assumed for the source language. If the value is "latin1" or "UTF-8" it is used to mark character strings as known to be in Latin-1 or UTF-8. For more details please see scan function.
encode.trgt	encoding to be assumed for the target language. If the value is "latin1" or "UTF-8" it is used to mark character strings as known to be in Latin-1 or UTF-8. For more details please see scan function.
name.sorc	it is a notation for the source language (default = 'f').
name.trgt	it is a notation for the target language (default = 'e').
dtfile.path	if NULL (usually for the first time), a data.table will be created contaning cross words of all sentences with their matched probabilities. It saves into a file named as a combination of name.sorc, name.trgt, nrec and iter as 'name.sorc.name.trgt.n.iter.RData'. If specific file name is set, it will be read and continue the rest of the function, i.e. : finding the word alignments.
result.file	the output results file name.
input	logical. If TRUE, the output can be used by bidictionary and align.test functions.
x	an object of class 'align'.
...	further arguments passed to or from other methods and further arguments of function prepare.data.

Details

Here, word alignment is a map of the target language to the source language.

The results depend on the corpus. As an example, we have used English-Persian parallel corpus named Mizan which consists of more than 1,000,000 sentence pairs with a size of 170 Mb. If all sentences are considered, it takes about 50.96671 mins using a computer with cpu: intel Xeon X5570 2.93GHZ and Ram: 8*8 G = 64 G and word alignment is good. But for the 10,000 first sentences, the word alignment might not be good. In fact, it is sensitive to the original translation type (lexical or conceptual). The results can be found at

http://www.um.ac.ir/~sarmad/word.a/example.align.ibm1.pdf

Value

align.ibm1 and align.symmet returns an object of class 'align'.

An object of class 'align' is a list containing the following components:

If input = TRUE

dd1	A data.table.

Otherwise,

model	'IBM1'
initial_n	An integer.
used_n	An integer.
time	A number. (in second/minute/hour)
iterIBM1	An integer.
expended_l_source	A non-negative real number.
expended_l_target	A non-negative real number.
VocabularySize_source	An integer.
VocabularySize_target	An integer.
word_translation_prob	A data.table.
word_align	A list of one-to-many word alignment for each sentence pair (it is as word by word).
align_init	One-to-many word alignment for the first three sentences.
align_end	One-to-many word alignment for the last three sentences.
number_align	A list of one-to-many word alignment for each sentence pair (it is as numbers).
aa	A matrix (n*2), where n is the number of remained sentence pairs after preprocessing.
method	symmetric word alignment method (union, intersection or grow-diag alignment).

Note

Note that we have a memory restriction and so just special computers with a high CPU and a big RAM can allocate the vectors of this function. Of course, it depends on the corpus size.

Author(s)

Neda Daneshgar and Majid Sarmad.

References

Koehn P. (2010), "Statistical Machine Translation.", Cambridge University, New York.

Lopez A. (2008), "Statistical Machine Translation.", ACM Computing Surveys, 40(3).

Peter F., Brown J. (1990), "A Statistical Approach to Machine Translation.", Computational Linguistics, 16(2), 79-85.

Supreme Council of Information and Communication Technology. (2013), Mizan English-Persian Parallel Corpus. Tehran, I.R. Iran. Retrieved from http://dadegan.ir/catalog/mizan.

http://statmt.org/europarl/v7/bg-en.tgz

See Also

align.test, align.symmet, bidictionary, scan

Examples

```
# Since the extraction of  bg-en.tgz in Europarl corpus is time consuming,
# so the aforementioned unzip files have been temporarily exported to
# http://www.um.ac.ir/~sarmad/... .
## Not run:

w1 = align.ibm1 ('http://www.um.ac.ir/~sarmad/word.a/euro.bg',
                     'http://www.um.ac.ir/~sarmad/word.a/euro.en',
                      n = 30, encode.sorc = 'UTF-8')

w2 = align.ibm1 ('http://www.um.ac.ir/~sarmad/word.a/euro.bg',
                     'http://www.um.ac.ir/~sarmad/word.a/euro.en',
                      n = 30, encode.sorc = 'UTF-8', remove.pt = FALSE)

S1 = align.symmet ('http://www.um.ac.ir/~sarmad/word.a/euro.bg',
                     'http://www.um.ac.ir/~sarmad/word.a/euro.en',
                      n = 200, encode.sorc = 'UTF-8')

S2 = align.symmet ('http://www.um.ac.ir/~sarmad/word.a/euro.bg',
                     'http://www.um.ac.ir/~sarmad/word.a/euro.en',
                      n = 200, encode.sorc = 'UTF-8', method = 'grow-diag')


## End(Not run)
```

---

align.test *Computing One-to-Many Word Alignment Using a Parallel Corpus for a Given Test Set*

Description

For a given parallel corpus based on IBM Model 1, it aligns the words of a given sentence-aligned test set.

Usage

```
align.test(file.sorc, file.trgt, test.sorc, test.trgt,
           n.train = -1, n.test = -1, minlen.train = 5, maxlen.train = 40,
           minlen.test = 5, maxlen.test = 40, null.tokens = TRUE,
           dtfile.path = NULL, file.align = 'alignment',
           name.sorc='f',name.trgt='e',iter = 3, ...)
```

Arguments

file.sorc	the name of source language file in training set.
file.trgt	the name of target language file in training set.
test.sorc	the name of source language file in test set.
test.trgt	the name of target language file in test set.
n.train	the number of sentences in the training set to be read. If -1, it considers all sentences.
n.test	the number of sentences in the test set to be read. If -1, it considers all sentences.
minlen.train	a minimum length of sentences in training set.
maxlen.train	a maximum length of sentences in training set.
minlen.test	a minimum length of sentences in test set.
maxlen.test	a maximum length of sentences in test set.
null.tokens	logical. If TRUE, "null" is added at the first of each source sentence of the test set.
dtfile.path	if NULL (usually for the first time), a data.table will be created contaning cross words of all sentences with their matched probabilities. It saves into a file named as a combination of name.sorc, name.trgt, n and iter as "f.e.n.iter.RData". If specific file name is set, it will be read and continue the rest of the function, i.e. : finding the word alignments for the test set.
file.align	the output results file name.
name.sorc	it is a notation for the source language (default = 'f').
name.trgt	it is a notation for the target language (default = 'e').
iter	the number of iterations for IBM Model 1.
...	Further arguments to be passed to prepare.data.

Details

If dtfile.path = NULL, the following question will be asked:

"Are you sure that you want to run the align.ibm1 function (It takes time)? (Yes/ No: if you want to specify word alignment path, please press 'No'.)

Value

an RData object as "file.align.n.iter.Rdata".

Note

Note that we have a memory restriction and so just special computers with a high CPU and a big RAM can allocate the vectors of this function. Of course, it depends on the corpus size.

Author(s)

Neda Daneshgar and Majid Sarmad.

References

Koehn P. (2010), "Statistical Machine Translation.", Cambridge University, New York.

Lopez A. (2008), "Statistical Machine Translation.", ACM Computing Surveys, 40(3).

Peter F., Brown J. (1990), "A Statistical Approach to Machine Translation.", Computational Linguistics, 16(2), 79-85.

Supreme Council of Information and Communication Technology. (2013), Mizan English-Persian Parallel Corpus. Tehran, I.R. Iran. Retrieved from http://dadegan.ir/catalog/mizan.

http://statmt.org/europarl/v7/bg-en.tgz

See Also

align.ibm1, evaluation, scan

Examples

```
# Since the extraction of  bg-en.tgz in Europarl corpus is time consuming,
# so the aforementioned unzip files have been temporarily exported to
# http://www.um.ac.ir/~sarmad/... .
# In addition, in this example we use the first five sentence pairs of training set as the
# test set.
## Not run:

ats = align.test ('http://www.um.ac.ir/~sarmad/word.a/euro.bg',
                      'http://www.um.ac.ir/~sarmad/word.a/euro.en',
                      'http://www.um.ac.ir/~sarmad/word.a/euro.bg',
                      'http://www.um.ac.ir/~sarmad/word.a/euro.en',
                       n.train = 100,n.test = 5, encode.sorc = 'UTF-8')

## End(Not run)
```

---

bidictionary *Building an Automatic Bilingual Dictionary*

Description

It builds an automatic bilingual dictionary of two languages based on given sentence-aligned parallel corpus.

Usage

```
bidictionary (..., n = -1L, iter = 15, prob = 0.8,
              dtfile.path = NULL, name.sorc = 'f', name.trgt = 'e')
```

Arguments

...	Further arguments to be passed to prepare.data.
n	Number of sentences to be read.
iter	the number of iterations for IBM Model 1.
prob	the minimum word translation probanility.
dtfile.path	if NULL (usually for the first time), a data.table will be created contaning cross words of all sentences with their matched probabilities. It saves into a file named as a combination of name.sorc, name.trgt, n and iter as "f.e.n.iter.RData". If specific file name is set, it will be read and continue the rest of the function, i.e. : finding dictionary of two given languages.
name.sorc	source language's name in mydictionary.
name.trgt	traget language's name in mydictionary.

Details

The results depend on the corpus. As an example, we have used English-Persian parallel corpus named Mizan which consists of more than 1,000,000 sentence pairs with a size of 170 Mb. For the 10,000 first sentences, we have a nice dictionary. It just takes 1.356784 mins using an ordinary computer. The results can be found at

http://www.um.ac.ir/~sarmad/word.a/bidictionary.pdf

Value

A list.

time	A number. (in second/minute/hour)
number_input	An integer.
Value_prob	A decimal number between 0 and 1.
iterIBM1	An integer.
dictionary	A matrix.

Note

Note that we have a memory restriction and just special computers with high cpu and big ram can allocate the vectors of this function. Of course, it depends on corpus size.

In addition, if dtfile.path = NULL, the following question will be asked:

"Are you sure that you want to run the align.ibm1 function (It takes time)? (Yes/ No: if you want to specify word alignment path, please press 'No'.)

Author(s)

Neda Daneshgar and Majid Sarmad.

References

Supreme Council of Information and Communication Technology. (2013), Mizan English-Persian Parallel Corpus. Tehran, I.R. Iran. Retrieved from http://dadegan.ir/catalog/mizan.

http://statmt.org/europarl/v7/bg-en.tgz

See Also

scan

Examples

```
# Since the extraction of  bg-en.tgz in Europarl corpus is time consuming,
# so the aforementioned unzip files have been temporarily exported to
# http://www.um.ac.ir/~sarmad/... .

## Not run:

dic1 = bidictionary ('http://www.um.ac.ir/~sarmad/word.a/euro.bg',
                     'http://www.um.ac.ir/~sarmad/word.a/euro.en',
                      n = 2000, encode.sorc = 'UTF-8',
                      name.sorc = 'BULGARIAN', name.trgt = 'ENGLISH')

dic2 = bidictionary ('http://www.um.ac.ir/~sarmad/word.a/euro.bg',
                     'http://www.um.ac.ir/~sarmad/word.a/euro.en',
                      n = 2000, encode.sorc = 'UTF-8',
                      name.sorc = 'BULGARIAN', name.trgt = 'ENGLISH',
                      remove.pt = FALSE)

## End(Not run)
```

---

cross.table *Constructing Cross Tables of the Source Language Words vs the Target Language Words of Sentence Pairs*

Description

It is a function to create the cross tables of the source language words vs the target language words of sentence pairs as the gold standard or as the alignment matrix of another software. For the gold standard, the created cross table is filled by an expert. He/she sets '1' for Sure alignments and '2' for Possible alignments in cross between the source and the target words. For alignment results of another software, '1' in cross between each aligned source and target words is set by the user.

It works with two formats:

For the first format, it constructs a cross table of the source language words vs the target language words of a given sentence pair. Then, after filling as mentioned above sentence by sentence, it builds a list of cross tables and finally, it saves the created list as "file.align.RData".

In the second format, it creates an excel file with n sheets. Each sheet includes a cross table of the two language words related each sentence pair. The file is as "file.align.xlsx". The created file to be filled as mentioned above.

Usage

```
cross.table( ...,
             null.tokens = TRUE,
             out.format = c('rdata','excel'),
             file.align = 'alignment')
```

Arguments

`...`	Further agguments to be passed to `prepare.data` and `align.test`
`null.tokens`	logical. If `TRUE`, "null" is added at the first of each source and target sentence, when we use RData format.
`out.format`	a character string including two options.For `"rdata"` format, it constructs a cross table of the source language words vs the target language words of a given sentence pair. Then, after filling it as mentioned in the description sentence by sentence, it builds a list of cross tables and finally, it saves the created list as "file.align.RData". In the `"excel"` format, it creates an excel file with n sheets. Each sheet includes a cross table of the two language words related to each sentence pair. The file is as "file.align.xlsx". The created file to be filled as mentioned in description.
`file.align`	the output file name.

Value

an RData object as "file.align.RData" or an excel file as "file.align.xlsx".

Note

If you have not the non-ascii problem, you can set `out.format` as `'rdata'`.

If ypu assign `out.format` to `'excel'`, it is necessary to bring two notes into consideration. The first note is that in order to use the created excel file for `evaluation` function, don't forget to use `excel2rdata` function to convert the excel file into required R format. The second note focouses on this: ocassionally, there is a problem with 'openxlsx' package which is used in the function and it might be solved by 'installr::install.rtools() on Windows'.

Author(s)

Neda Daneshgar and Majid Sarmad.

References

Holmqvist M., Ahrenberg L. (2011), "A Gold Standard for English-Swedish Word Alignment.", NODALIDA 2011 Conference Proceedings, 106 - 113.

Och F., Ney H.(2003), "A Systematic Comparison Of Various Statistical Alignment Models.", 2003 Association for Computational Linguistics, J03-1002, 29(1).

See Also

`evaluation`, `excel2rdata`, `scan`

Examples

```
## Not run:

cross.table('http://www.um.ac.ir/~sarmad/word.a/euro.bg',
           'http://www.um.ac.ir/~sarmad/word.a/euro.en',
           n = 10, encode.sorc = 'UTF-8')

cross.table('http://www.um.ac.ir/~sarmad/word.a/euro.bg',
           'http://www.um.ac.ir/~sarmad/word.a/euro.en',
           n = 5, encode.sorc = 'UTF-8', out.format = 'excel')

## End(Not run)
```

---

evaluation *Evaluation of Word Alignment Quality*

Description

It measures Precision, Recall, AER, and F_measurs metrics to evaluate the quality of word alignment.

Usage

```
evaluation(file.gold = 'gold.RData',
           file.align = 'alignment.-1.3.RData',
           agn = c('my.agn','an.agn'), alpha = 0.3)
```

Arguments

`file.gold`	the gold standarad file name.
`file.align`	the alignment file name.
`agn`	character string including two values. If `"my.agn"`, the user wants to evaluate one-to-many word alignment using the `align.ibm1` function in this package. If `"an.agn"`, the user wants to evaluate word alignment results which are obtained by another software.
`alpha`	is a parameter that sets the trade-off between Precision and Recall.

Details

To evaluate word alignment quality, we need to a "reference alignment" (a gold standard for the word alignment) of a test set. In order to read the gold into RData format and to compare it with the word alignment results, the gold standard file name must be set in `file.gold`.

Value

A list.

Recall	A decimal number.
Precision	A decimal number.
AER	A decimal number.
F_measure.PS	A decimal number.
F_measure.S	A decimal number.

Author(s)

Neda Daneshgar and Majid Sarmad.

References

Fraser F., Marcu D. (2007), "MeasuringWord Alignment Quality for Statistical Machine Translation.", Computational Linguistics, 33(3), 293-303.

Koehn P. (2010), "Statistical Machine Translation.", Cambridge University, New York.

Och F., Ney H.(2003)."A Systematic Comparison Of Various Statistical Alignment Models.", 2003 Association for Computational Linguistics, J03-1002, 29(1).

Wang X. "Evaluation of Two Word Alignment Systems.", Final Thesis, Department of Computer and Information Science.

See Also

cross.table, align.test, align.ibm1

---

excel2rdata *Converting Excel Files Into Required R Format*

Description

This function converts the excel files into required RData format.

Usage

```
excel2rdata(file.align = 'alignment.xlsx', null.tokens = TRUE, len = len)
```

Arguments

file.align	the excel file name which we want to convert it into required RData format.
null.tokens	logical. If 'TRUE', 'null' is added at the first of each source sentence of the test set.
len	the number of sheets of the excel file to be converted into RData format. It must be assigned by the user.

Value

an RData object as 'file.align.RData'.

Note

Note that in order to use the created excel file for the function `evaluation`, don't forget to use `excel2rdata` function to convert the excel file into required `RData` format.

Author(s)

Neda Daneshgar and Majid Sarmad.

See Also

`cross.table`, `evaluation`

`neighbor` *Finding Neighborhood Locations*

Description

Starting with the intersection of ef and fe alignment one by one and finding the square neighbors including the union and intersection, recursively.

Usage

```
neighbor(fe, ef, n.row)
```

Arguments

`fe`	an integer vector.
`ef`	an integer vector.
`n.row`	an integer. Number of rows of an initial matrix.

Value

An integer vector.

Author(s)

Neda Daneshgar and Majid Sarmad.

References

Koehn P. (2010), "Statistical Machine Translation.", Cambridge University, New York.

Examples

```
fe = c(1,4,2,4,2)
ef = c(3,2,1,5)
n.row = 4
neighbor (fe, ef, n.row)
```

`nfirst2lower` *Make a String's First n Characters Lowercase*

Description

Converts uppercase to lowercase letters for the first n characters of a character string.

Usage

```
nfirst2lower(x, n = 1, first = TRUE, second = FALSE)
```

Arguments

`x`	a character string.
`n`	an integer. Number of characters that we want to convert.
`first`	logical. If `TRUE`, it converts the n first characters into lowercase.
`second`	logical. If `TRUE`, it checks if the second letter of `x` is uppercase, the whole word will be converted to lower.

Details

It is a function to convert some uppercase letters into lowercase for which words with uppercase second letter. If `tolower` in base R is used, it will be sometimes created a problem for proper nouns. Because, as we know, a name or proper noun starts with capital letter and we do not want to convert them into lowercase. But sometimes there are some words which are not a name or proper noun and displayed in capital letters. These words are the target of this function.

If we have a text of several sentences and we want to convert the first n letters of every sentence to lowercase, separately. We have to split text to sentences, furthermore we should consider `first=TRUE` and apply the function for each sentence (see the examples below).

If we have a list, it works fine.

Value

A character string.

Note

Because of all sentences begin with uppercase letters, `first=TRUE` is considered as a default. But, if the second character of a word be capital, it is usually concluded that all its characters are capital. In this case, you can consider `second=TRUE`. Of course, there are some exceptations in these cases that they can be ignored (see the examples below).

In general, if there are not a lot of proper nouns in your text string, we suggest you to use `tolower` in base R. As an ability of this function, `lower` is considered as a third argument.

Author(s)

Neda Daneshgar and Majid Sarmad.

See Also

`tolower`

Examples

```
# x is a list

x=list('W-A for an English-Persian Parallel Corpus (Mizan).','ALIGNMENT is a link between words.')

nfirst2lower(x, n=8) ## nfirst2lower(x, n=8) is not a list

y='MT is the automatic translation. SMT is one of the methods of MT.'

nfirst2lower(y) # only run for the first sentence

u1=unlist(strsplit(y, ". ", fixed = TRUE))
sapply(1:length(u1),function(x)nfirst2lower(u1[x])) ## run for all sentences

h = 'It is a METHOD for this function.'
nfirst2lower (h, second = TRUE) #only run for the first word

h1 = strsplit(h, ' ')[[1]]
nfirst2lower(h1, second = TRUE) # run for all words
```

`prepare.data` *Initial Preparations of Bitext before the Word Alignment and the Evaluation of Word Alignment Quality*

Description

For a given Sentence-Aligned Parallel Corpus, it prepars sentence pairs as an input for `align.ibm1` and `evaluation` functions in this package.

Usage

```
prepare.data(file.sorc, file.trgt, n = -1L,
             encode.sorc = 'unknown' , encode.trgt = 'unknown',
             min.len = 5, max.len = 40, remove.pt = TRUE, word.align = TRUE)
```

Arguments

file.sorc	the name of source language file.
file.trgt	the name of target language file.
n	the number of sentences to be read.If -1, it considers all sentences.
encode.sorc	encoding to be assumed for the source language. If the value is "latin1" or "UTF-8" it is used to mark character strings as known to be in Latin-1 or UTF-8. For more details please see scan function.
encode.trgt	encoding to be assumed for the target language. If the value is "latin1" or "UTF-8" it is used to mark character strings as known to be in Latin-1 or UTF-8. For more details please see scan function.
min.len	a minimum length of sentences.
max.len	a maximum length of sentences.
remove.pt	logical. If 'TRUE', it removes all punctuation marks.
word.align	logical. If 'FALSE', it divides each sentence into its words. Results can be used in align.symmet, cross.table, align.test and evaluation functions.

Details

It balances between source and target language as much as possible. For example, it removes extra blank sentences and equalization sentence pairs. Also, using nfirst2lower function, it converts the first letter of each sentence into lowercase. Moreover, it removes short and long sentences.

Value

A list.

if word_align = TRUE

len1	An integer.
aa	A matrix (n*2), where 'n' is the number of remained sentence pairs after pre-processing.

otherwise,

initial	An integer.
used	An integer.
source.tok	A list of words for each the source sentence.
target.tok	A list of words for each the target sentence.

Note

Note that if there is a few proper nouns in the parallel corpus, we suggest you to set `all=TRUE` to convert all text into lowercase.

Author(s)

Neda Daneshgar and Majid Sarmad.

References

Koehn P. (2010), "Statistical Machine Translation.", Cambridge University, New York.

See Also

`evaluation`, `nfirst2lower`, `align.ibm1`, `scan`

Examples

```
# Since the extraction of  bg-en.tgz in Europarl corpus is time consuming,
# so the aforementioned unzip files have been temporarily exported to
# http://www.um.ac.ir/~sarmad/... .
## Not run:

aa1 = prepare.data ('http://www.um.ac.ir/~sarmad/word.a/euro.bg',
                    'http://www.um.ac.ir/~sarmad/word.a/euro.en',
                     n = 20, encode.sorc = 'UTF-8')

aa2 = prepare.data ('http://www.um.ac.ir/~sarmad/word.a/euro.bg',
                    'http://www.um.ac.ir/~sarmad/word.a/euro.en',
                     n = 20, encode.sorc = 'UTF-8', word.align = FALSE)

aa3 = prepare.data ('http://www.um.ac.ir/~sarmad/word.a/euro.bg',
                    'http://www.um.ac.ir/~sarmad/word.a/euro.en',
                     n = 20, encode.sorc = 'UTF-8', remove.pt = FALSE)

## End(Not run)
```

---

remove.punct *Tokenizing and Removing Punctuation Marks*

Description

It splits a given text into separated words and removes its punctuation marks.

Usage

```
remove.punct(text)
```

Arguments

`text`	an object.

Details

This function also considers numbers as a separated word.

Note that This function removes "dot"" only if it is at the end of the sentence, separately. Meanwhile, it does not eliminate dash and hyper.Because it is assumed that words containing these punctuations are one word.

Value

A vector of character string.

Author(s)

Neda Daneshgar and Majid Sarmad

Examples

```
x = "This is an  example-based MT!"
remove.punct (x)
```

Index