Package ‘word.alignment’

February 21, 2018

Type Package

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

Version 1.0.9

Date 2018-02-19

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, quanteda

License GPL (>= 2)

NeedsCompilation no

Repository CRAN

Date/Publication 2018-02-21 14:10:52 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 `word_alignIBM1` and symmetric word alignment in the function `Symmetrization`. Moreover, it evaluates the quality of word alignment from `word_alignIBM1` function or from some other software in the function `Evaluation1`. It also builds an automatic bilingual dictionary of two languages using the given corpus in the function `mydictionary`.

Details

Package:	word.alignment
Type:	Package
Version:	1.0.9
Date:	2018-02-19
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 = word_alignIBM1 ('http://www.um.ac.ir/~sarmad/word.a/euro.bg',
                     'http://www.um.ac.ir/~sarmad/word.a/euro.en',
                      nrec=2000, encode.sorc = 'UTF-8')

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

## End(Not run)
```

align_test.set *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.set(file_train1, file_train2,
                tst.set_sorc, tst.set_trgt,
                nrec = -1, nlen = -1,
                encode.sorc = 'unknown', encode.trgt = 'unknown',
                minlen1 = 5, maxlen1 = 40, minlen2 = 5, maxlen2 = 40,
                removePt = TRUE, all = FALSE, null.tokens = TRUE,
                iter = 3, f1 = 'fa', e1 = 'en',
                dtfile_path = NULL, file_align = 'alignment')
```

Arguments

`file_train1` the name of source language file in training set.

`file_train2` the name of target language file in training set.

`tst.set_sorc` the name of source language file in test set.

`tst.set_trgt` the name of target language file in test set.

`nrec` the number of sentences in the training set to be read. If -1, it considers all sentences.

`nlen` the number of sentences in the test set 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.

`minlen1` a minimum length of sentences in training set.

`maxlen1` a maximum length of sentences in training set.

`minlen2` a minimum length of sentences in test set.

`maxlen2` a maximum length of sentences in test set.

`removePt` logical. If `TRUE`, it removes all punctuation marks.

`all` logical. If `TRUE`, it considers the third argument (`lower = TRUE`) in `culf` function.

`null.tokens` logical. If `TRUE`, "null" is added at the first of each source sentence of the test set.

`iter` the number of iterations for IBM Model 1.

`f1` it is a notation for the source language (default = `'fa'`).

`e1` it is a notation for the target language (default = `'en'`).

`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 `f1`, `e1`, `nrec` and `iter` as "f1.e1.nrec.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.

Details

If `dtfile_path = NULL`, the following question will be asked:

"Are you sure that you want to run the word_alignIBM1 function (It takes time)? (Yes/ No: if you want to specify word alignment path, please press 'No'.)

Value

an RData object as "file_align.nrec.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

word_alignIBM1, Evaluation1, 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.set ('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',
                       nrec = 100,nlen = 5, encode.sorc = 'UTF-8',)

## End(Not run)
```

cons.agn *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 `nrec` 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

```
cons.agn(tst.set_sorc, tst.set_trgt, nrec = -1,
        encode.sorc = 'unknown', encode.trgt = 'unknown',
        minlen = 5, maxlen = 40, removePt = TRUE,
        all = FALSE, null.tokens = TRUE, Format = c('R', 'Excel'),
        file_align = 'alignment')
```

Arguments

`tst.set_sorc` the name of source language file in test set.

`tst.set_trgt` the name of target language file in test set.

`nrec` 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.

`minlen` a minimum length of sentences.

`maxlen` a maximum length of sentences.

`removePt` logical. If `TRUE`, it removes all punctuation marks.

`all` logical. If `TRUE`, it considers the third argument (`lower = TRUE`) in `culf` function.

`null.tokens` logical. If `TRUE`, "null" is added at the first of each source and target sentence, when we use R format.

`Format` character string including two values. If `R`, it creates a cross table of the source language words vs the target language words of a given sentence pair. Then, it constructs a list of them. If `Excel`, it makes an excel file with `nrec` sheets of a test set including the source and the target languages. Each sheet includes the words of the source sentence in its first rows and the words of the target sentence in its first columns.

`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 Format as 'R'.

If ypu assign 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 Evaluation1 function, don't forget to use ExcelToR 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

Evaluation1, ExcelToR, scan

Examples

```
## Not run:

cons.agn('http://www.um.ac.ir/~sarmad/word.a/source1.txt',
          'http://www.um.ac.ir/~sarmad/word.a/target1.txt',
           nrec = 5, encode.sorc = 'UTF-8')

cons.agn('http://www.um.ac.ir/~sarmad/word.a/source1.txt',
          'http://www.um.ac.ir/~sarmad/word.a/target1.txt',
           nrec = 5, encode.sorc = 'UTF-8', Format = 'Excel')

## End(Not run)
```

culf — *Make a String's First n Characters Lowercase*

Description

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

Usage

```
culf(x, n = 1, first = TRUE, second = FALSE, lower = 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.

`lower` logical. If `TRUE`, it works similar to `tolower` in base R.

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 exceptions 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.')
```

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

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

culf(y) # only run for the first sentence

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

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

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

Evaluation1 *Evaluation of Word Alignment Quality*

Description

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

Usage

```
Evaluation1(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 `word_alignIBM1` 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 R 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

`cons.agn`, `align_test.set`, `word_alignIBM1`

`ExcelToR` *Converting Excel Files Into Required R Format*

Description

This function converts the excel files into required R format.

Usage

```
ExcelToR(file_align = 'alignment.xlsx', null.tokens = TRUE, len = len)
```

Arguments

`file_align` the excel file name which we want to convert it into required R 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 R 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 Evaluation1, don't forget to use ExcelToR function to convert the excel file into required R format.

Author(s)

Neda Daneshgar and Majid Sarmad.

See Also

cons.agn, Evaluation1

mydictionary *Building an Automatic Bilingual Dictionary*

Description

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

Usage

```
mydictionary(file_train1, file_train2, nrec = -1,
             encode.sorc = 'unknown', encode.trgt = 'unknown',
         iter = 15, prob = 0.8, minlen = 5, maxlen = 40,
             lang1 = 'Farsi', lang2 = 'English', removePt = TRUE,
             dtfile_path = NULL, f1 = 'fa', e1 = 'en',
             result_file = 'mydictionaryResults')
```

Arguments

`file_train1`	the name of source language file in training set.
`file_train2`	the name of target language file in training set.
`nrec`	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.
`iter`	the number of iterations for IBM Model 1.

`prob` the minimum word translation probanility.

`minlen` a minimum length of sentences.

`maxlen` a maximum length of sentences.

`lang1` source language's name in mydictionary.

`lang2` traget language's name in mydictionary.

`removePt` logical. If `TRUE`, it removes all punctuation marks.

`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 `f1`, `e1`, `nrec` and `iter` as "f1.e1.nrec.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.

`f1` it is a notation for the source language (default = `'fa'`).

`e1` it is a notation for the target language (default = `'en'`).

`result_file` the output results file name.

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/mydictionary.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 word_alignIBM1 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 = mydictionary ('http://www.um.ac.ir/~sarmad/word.a/euro.bg',
                     'http://www.um.ac.ir/~sarmad/word.a/euro.en',
                      nrec = 2000, encode.sorc = 'UTF-8', lang1 = 'BULGARIAN')

dic2 = mydictionary ('http://www.um.ac.ir/~sarmad/word.a/euro.bg',
                     'http://www.um.ac.ir/~sarmad/word.a/euro.en',
                      nrec = 2000, encode.sorc = 'UTF-8', lang1 = 'BULGARIAN',
                      removePt = FALSE)

## End(Not run)
```

`prepareData` *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 `word_alignIBM1` and `Evaluation1` functions in this package.

Usage

```
prepareData(file1, file2, nrec = -1,
   encode.sorc = 'unknown', encode.trgt = 'unknown',
            minlen = 5, maxlen = 40, all = FALSE,
            removePt = TRUE, word_align = TRUE)
```

Arguments

`file1` the name of source language file.

`file2` the name of target language file.

`nrec` 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.

`minlen` a minimum length of sentences.

`maxlen` a maximum length of sentences.

`all` logical. If '`TRUE`', it considers the third argument ('`lower = TRUE`') in `culf` function.

`removePt` 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 `Symmetrization`, `cons.agn`, `align_test.set` and `Evaluation1` 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 `culf` 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 preprocessing.

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

`Evaluation1`, `culf`, `word_alignIBM1`, `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 = prepareData ('http://www.um.ac.ir/~sarmad/word.a/euro.bg',
                   'http://www.um.ac.ir/~sarmad/word.a/euro.en',
                    nrec = 20, encode.sorc = 'UTF-8')

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

aa3 = prepareData ('http://www.um.ac.ir/~sarmad/word.a/euro.bg',
                   'http://www.um.ac.ir/~sarmad/word.a/euro.en',
                    nrec = 20, encode.sorc = 'UTF-8', removePt = FALSE)

## End(Not run)
```

---

squareN *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

```
squareN(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
squareN (fe, ef, n_row)
```

`Symmetrization` *Calculating Symmetric Word Alignment*

Description

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.

Usage

```
Symmetrization(file_train1, file_train2,
               method = c('union', 'intersection', 'grow-diag'),
               nrec = -1, encode.sorc = 'unknown', encode.trgt = 'unknown',
       iter = 4, minlen = 5, maxlen = 40, removePt = TRUE,
               all = FALSE, f1 = 'fa', e1 = 'en')

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

Arguments

`file_train1` the name of source language file in training set.

`file_train2` the name of target language file in training set.

`method` character string specifying the symmetric word alignment method (union, intersection, or grow-diag alignment).

`nrec` 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.

iter the number of iterations for IBM Model 1.

minlen a minimum length of sentences.

maxlen a maximum length of sentences.

removePt logical. If TRUE, it removes all punctuation marks.

all logical. If TRUE, it considers the third argument (lower = TRUE) in culf function.

f1 it is a notation for the source language (default = 'fa').

e1 it is a notation for the target language (default = 'en').

x an object of class 'symmet'.

... further arguments passed to or from other methods.

Details

Here, word alignment is not only a map of the target language to the source language and it is considered as a symmetric alignment such as union, or intersection, or grow-diag alignment.

Value

Symmetrization returns an object of class 'symmet'.

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

time A number. (in second/minute/hour)

method symmetric word alignment method (union, intersection, or grow-diag alignment).

alignment A list of alignment for each sentence pair .

aa a vector of source sentences.

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.

Author(s)

Neda Daneshgar and Majid Sarmad.

References

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

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

See Also

word_alignIBM1, 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:

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

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

## End(Not run)
```

---

word_alignIBM1 *Computing One-to-Many Word Alignment Using IBM Model 1 for a Given Parallel Corpus*

Description

For a given sentence-aligned parallel corpus, it aligns words 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

```
word_alignIBM1(file_train1, file_train2, nrec = -1,
               encode.sorc = 'unknown', encode.trgt = 'unknown',
               iter = 5, minlen = 5, maxlen = 40,
               removePt = TRUE, all = FALSE,
               dtfile_path = NULL, f1 = "fa", e1 = "en",
               result_file = 'myResultIBM1', input = FALSE)

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

Arguments

file_train1	the name of source language file in training set.
file_train2	the name of target language file in training set.
nrec	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.

`iter` the number of iterations for IBM Model 1.

`minlen` a minimum length of sentences.

`maxlen` a maximum length of sentences.

`removePt` logical. If `TRUE`, it removes all punctuation marks.

`all` logical. If `TRUE`, it considers the third argument (`lower = TRUE`) in `culf` function.

`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 `f1`, `e1`, `nrec` and `iter` as 'f1.e1.nrec.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.

`f1` it is a notation for the source language (default = `'fa'`).

`e1` it is a notation for the target language (default = `'en'`).

`result_file` the output results file name.

`input` logical. If `TRUE`, the output can be used by `mydictionary` and `align_test.set` functions.

`x` an object of class `'alignment'`.

`...` further arguments passed to or from other methods.

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_wordalignIBM1.pdf`

Value

`word_alignIBM1` returns an object of class `'alignment'`.

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

If '`input = TRUE`'

`dd1` A data.table.

Otherwise,

`n1` An integer.

`n2` 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).

`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.

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.set`, `Symmetrization`, `mydictionary`, `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 = word_alignIBM1 ('http://www.um.ac.ir/~sarmad/word.a/euro.bg',
                     'http://www.um.ac.ir/~sarmad/word.a/euro.en',
                      nrec = 30, encode.sorc = 'UTF-8')

w2 = word_alignIBM1 ('http://www.um.ac.ir/~sarmad/word.a/euro.bg',
                     'http://www.um.ac.ir/~sarmad/word.a/euro.en',
                      nrec = 30, encode.sorc = 'UTF-8', removePt = FALSE)

## End(Not run)
```

Index