




# word.alignment: an R package for computing statistical word alignment and its evaluation

Neda Daneshgar<sup>1</sup> · Majid Sarmad<sup>1</sup> 

Received: 5 November 2017 / Accepted: 10 March 2020 / Published online: 23 March 2020  
© Springer-Verlag GmbH Germany, part of Springer Nature 2020

## Abstract

Word alignment has lots of applications in various natural language processing (NLP) tasks. As far as we are aware, there is no word alignment package in the R environment. In this paper, `word.alignment`, a new R software package is introduced which implements a statistical word alignment model as an unsupervised learning. It uses IBM Model 1 as a machine translation model based on the use of the EM algorithm and the Viterbi search in order to find the best alignment. It also provides the symmetric alignment using three heuristic methods such as union, intersection, and grow-diag. It has also the ability to build an automatic bilingual dictionary applying an innovative rule. The generated dictionary is suitable for a number of NLP tasks. This package provides functions for measuring the quality of the word alignment via comparing the alignment with a gold standard alignment based on five metrics as well. It is easily installed and executable on the mostly widely used platforms. Note that it is easily usable and we show that its results are almost everywhere better than some other word alignment tools. Finally, some examples illustrating the use of `word.alignment` is provided.

**Keywords** Natural language processing (NLP) · IBM model 1 · EM algorithm · Symmetric word alignment · Parallel corpus · Evaluation · Gold standard alignment · Test set

---

✉ Majid Sarmad  
sarmad@um.ac.ir  
<http://www.um.ac.ir/~sarmad>  
Neda Daneshgar  
ne\_da978@mail.um.ac.ir

<sup>1</sup> Department of Statistics, Ferdowsi University of Mashhad, Mashhad, Iran

## 1 Introduction

Word alignment is a process that is used to determine the equivalent words in a bilingual sentence pair. The bilingual sentence pair contains two different languages. One is the source language which is the language the translation starts from and another one that is known as the target language is the language the translation ends in. In the literature, two different types of word alignment applications have been considered. One main goal is to produce lexical data for bilingual dictionaries, while another goal can be providing data for MT (Wang 2004). A number of word alignment applications are multi lingual lexicography, word sense disambiguation, translation connections in computational lexicography (Vulić and Moens 2010), patent retrieval that is a branch of information retrieval (Jochim et al. 2011), and cross-lingual information retrieval (Nie 2010). Word alignment is also a necessary step for almost all state-of-the-art translation systems including syntax-based machine translation (SBMT), statistical MT (SMT), hierarchical phrase-based systems (Brunning 2010), example-based MT (Vulić and Moens 2010) and many other multi lingual applications. Therefore, the task of word alignment is interesting in itself for plenty of applications (Koehn 2010).

From one point of view, there are two general approaches for alignment models including statistical and heuristic ones (Och and Ney 2003). Och and Ney (2003) have shown that the statistical model is better than the heuristic one, but Vulić and Moens (2010) have explained some advantages and disadvantages of these approaches. In the statistical model, no linguistic knowledge is required and only a large corpus of the bilingual sentence pair known as sentence-aligned parallel corpus is needed. The sentence-aligned parallel corpus is including bilingual sentence pairs of the source and the target languages so that each sentence pair of the two different languages are aligned sentence by sentence. The goal is to find a correspondence between the words in each sentence pair, i.e., constructing a word-aligned parallel corpus. To do this, firstly, an alignment function from target words positions into source words positions is defined (Koehn 2010). Thus, one-to-one or one-to-many alignment between the words are obtained such that it is possible to link each source word with none, one or several target words. In fact, many-to-many and many-to-one alignments are impossible and impose a limitation on this model. Och and Ney (2003) have proposed the symmetrization of word alignment so that the first training is performed in both translation directions (source-to-target and target-to-source), and then these are combined as various methods (union, intersection, or grow-diag) in order to intensify the quality of the word alignment and overcome this restriction. Therefore, this symmetric word alignment includes many-to-one and many-to-many correspondences.

In this paper a new R package named `word.alignment` (Daneshgar and Sarmad 2017) is described. In the package, a statistical word alignment model has been considered. It includes several functions to do one-to-one, one-to-many and symmetric alignments for each sentence pair, building an automatic dictionary based on two input languages (the source and the target languages) and measuring the quality of the word alignment using Precision, Recall, Alignment Error Rate (AER), and F-measure metrics. More details can be found in Fraser and Marcu (2007), Koehn (2010), Och and Ney (2003).

Two main functions (`align.ibm1` and `align.symmet`) in the package have been written in R using S3 classes and methods. These functions store the results in a specific class: `align`.

While there are a few packages like `quanteda` (Benoit et al. 2018), `tm` (Feinerer and Hornik 2015), and `NLP` (Hornik 2015) in the R environment (R Core Team 2015) for NLP tasks, they do not separately compute word alignment. Of course, some functions in the packages like `NLP` and `tm` do not work on Arabic Script (e.g., Persian) properly.

Meanwhile, there are some other open-source tools to compute word alignment. In Table 1, some advantages and disadvantages of some tools have been shown:

It should be noted that none of the tools of Table 1 except `word.alignment` has been written in R which seemed to be necessary in this environment. For example, **`pialign`**, **`UNL Aligner`**, **`Anymalign`** and **`Nile`** have been written in Python.

As pointed out in Table 1, **`NATools`** and the produced package compute some operations, simultaneously.

**`NATools`** builds a sentence-aligned parallel corpus, computes word alignment for the bilingual sentence-aligned parallel corpus and it creates a probabilistic dictionary.

The `word.alignment` package implements one-to-many and symmetric alignments. It also constructs an automatic bilingual dictionary. The package measures the quality of the word alignments as well by a new function evaluation introduced in Section 5. It is noteworthy that evaluating the quality of the word alignment is a very important issue that none of the previous mentioned tools carry out this case.

In Sect. 6.4, using the function evaluation, a brief comparison has been made between some tools with the package. The package's results are general better than some other tools.

This paper uses by `word.alignment` version 1.1 which is available on the Comprehensive R Archive Network. Note that Some of previous versions are available in R-Forge in <https://r-forge.r-project.org/>. In addition, there are two dependent packages `data.table` (Dowle 2017), and `openxlsx` (Walker 2017) which should be installed along with the package. The new version does not require the `quanteda` package which it is one of the dependent packages for some previous versions. which is the most important difference with the previous ones.

The rest of the article is organized as follows. In Sect. 2, a general concept of word alignment is defined. Statistical alignment model is briefly outlined in Sect. 3.1. In Sect. 3.2, symmetric word alignment is explained. The concept evaluation of the quality of the word alignment is defined in Sect. 4. The detailed usages are described in Sect. 5. Some examples of the functions are provided in Sect. 6. Finally, conclusion and future works are presented in Sect. 7.

## 2 The concept of word alignment

The concept of word alignment between two sentence pairs can be quite complicated. Typically, a word alignment is associated with reorderings, droppings, insertions, and one-to-many alignments. Formally, the following definition is considered in this paper.

**Table 1** Comparison of some word alignment tools

Tool name	Advantages	Disadvantages
<b>pialign (Python)</b> (Neubig et al. 2011, 2012)	Usable results in the Moses software for automatic translation Déchelotte et al. (2007)	Not working directly on windows platform Too slow according to user guide. It claims that it takes a time about 1 to 2 h for only 10,000 sentence pairs
<b>UNL Aligner (Python)</b> (Ildefonso and Lopes 2005)	Convenient installation	Ambiguous outputs. An example of its output is shown in <a href="http://research.variancia.com/unl-aligner/">http://research.variancia.com/unl-aligner/</a> Not working on windows platform
<b>Anymalign (Python)</b> (Lardilleux and Lepage 2009)	Aligning multiple languages at the same time Executable on the three most widely used platforms	Too slow. Its speed is good up to 100,000 sentence pairs. But more than that, it takes time The latest version (2011)
<b>NATools (C)</b> (Simes and Almeida 2003)	Performing some other operations simultaneously, which will be explained after the table	Difficult installation according to user guide
The Berkely Word aligner (Java) Moore (2005)	Performing both methods of supervised and unsupervised word alignment	Not working on windows platform Relatively difficult to install
<b>Nile (Python)</b>	–	Relating to Riesa's PhD project in 2012. Therefore, it cannot be used any more
<b>Giza++ (C++)</b> Och (2000)	The widely used software to do word alignment Faster than other mentioned word alignment tools for large corpus Suitable for large corpus in terms of speed and memory	Not working directly on windows Relatively difficult to install
word.alignment (R)	Executable on the three most widely used platforms Convenient installation. It is one package in the R environment which is easily installed in the three most widely used platforms Easy to use. It is enough to run a command line in the R environment which will be described later in Sects. 6.1 and 6.2 Faster than any other mentioned alignment tools for small corpus Performing some other operations simultaneously, which will be explained after the table	Slower than <b>Giza++</b> for large corpus but faster than many other tools like <b>pialign</b> or <b>Anymalign</b> Memory limit for large corpus. More details have been completely explained in Sect. 6

**Table 1** continued

Tool name	Advantages	Disadvantages
	Performing some preprocessing on the input sentence pairs. The preprocessing includes separating or removing punctuation marks or converting the first character of each sentence into lowercase. It is important to note that the preprocessing can affect the quality of the word alignment	

Assume that we have a sentence pair  $(f, e)$  that we want to align.  $f = f_1, \dots, f_i, \dots, f_I$  is a source language string and  $e = e_1, \dots, e_j, \dots, e_J$  is a target language string. A word alignment  $A$  can be defined as a subset of the Cartesian product of the word positions acquired by the following relation:

$$A \subseteq \{(j, i) : j = 1, \dots, J; i = 1, \dots, I\} \quad (1)$$

This model is quite general as an arbitrary connection between source and target languages positions and of course it is computationally hard as there are  $2^{IJ}$  alignments. Hence, often additional constraints have been supposed on word alignment models (Och and Ney 2003).

A usual restriction is letting each target word be linked to exactly one source word. This rule does not hold in the other direction, i.e., a source word can be assigned to multiple target words or none of them. Therefore, the alignment mapping in such models is a map from a target word at position  $j$  to a source word at position  $i$  with a function  $a : j \rightarrow i$ . In other words, we consider alignment variable  $\mathbf{a} = a_1, \dots, a_j, \dots, a_J$  as  $j \rightarrow i = a_j$ . This alignment may contain some alignments  $a_j = 0$ . It is used to align the target word  $e_j$  to the “empty” word  $f_0$  which is called the “null token” (Koehn 2010).

In the next section, statistical word alignment will be presented.

### 3 Statistical word alignment

MT is an automated translation of a source language transcript to a target language one with or without human assistance (Chérâgui 2012). SMT is a statistical approach to MT that is specified by using statistical models and does not need any prior knowledge of linguistics. It was proposed by Brown et al. (1990) for the first time. In fact, the concept of statistical word alignment is associated with SMT. Statistical word alignment using IBM Model 1 and symmetric word alignment will be demonstrated in the following section.

### 3.1 Statistical word alignment using IBM model 1

The concept of statistical word alignment is associated with SMT as mentioned before. The objective of SMT is to find the most probable target sentence  $e = e_1, \dots, e_j, \dots, e_J$  of a given source sentence  $f = f_1, \dots, f_i, \dots, f_I$  as the following.

$$\hat{e} = \underset{e}{\operatorname{argmax}} P(e|f) \quad (2)$$

The  $P(e|f)$  is determined by training a statistical model using the sentence-aligned parallel corpus (Brunning 2010). The relationship between this probability and the alignment model is given by

$$P(e|f) = \sum_a P(e, a|f) \quad (3)$$

where  $a = a_1, \dots, a_j, \dots, a_J$  is the hidden alignment variable which specifies the position of a source word aligned with each target word.

One the other hand,  $P(e, a|f)$  is decomposed to three components  $P(J|f)$ ,  $P(a|J, f)$ , and  $P(e|a, J, f)$ . This is the story of generative mode, the simplest and most primary one with the following assumptions which is the so-called IBM Model 1.

1.  $P(J|f)$  is as a constant like  $\epsilon$ .
2.  $P(a|J, f) = \frac{1}{(I+1)^J}$ .
3.  $P(e|a, J, f) = \prod_{j=1}^J t(e_j|f_{a_j})$  where  $t(e_j|f_{a_j})$  is the probability of translating a source word  $f_i$  into a target word  $e_j$  with a map  $j \rightarrow i = a_j$ .

The IBM Model 1 parameters are  $t(e_j|f_i)$  which can be estimated using the EM algorithm with the following steps:

1. Initialize  $t(e_j|f_i)$ , usually with uniform distributions.
2. E-step: Compute the probability of an alignment given a sentence pair in the training data, i.e.,  $P(a|e, f)$  using the current parameters.
3. M-step: "Compute a count function  $c(e_j|f_i; \mathbf{e}, \mathbf{f})$  that collects evidence from a sentence pair  $(f, e)$  that a particular input word  $f_i$  translates into the output word  $e_j$ " (Koehn 2010). Then, new parameter values are estimated using the following equation given these counts:

$$t(e_j|f_i; \mathbf{f}, \mathbf{e}) = \frac{\sum_{e, f} c(e_j|f_i; \mathbf{f}, \mathbf{e})}{\sum_e \sum_{e, f} c(e_j|f_i; \mathbf{f}, \mathbf{e})} = \frac{\operatorname{count}(e|f)}{\operatorname{total}(f)} \quad (4)$$

4. The E and M steps are iterated until the algorithm is converges.

More details and the pseudo code for "the EM algorithm for IBM Model 1" can be found in Koehn (2010).

Now, the best alignment can be obtained by using these estimated parameters. In the IBM models (Brown et al. 1993), the best alignment  $\hat{a}$  for a given sentence pair  $(f, e)$  is acquired by

$$\hat{a} = \operatorname{argmax}_a P(a, e|f) \quad (5)$$

The alignment  $\hat{a}$  is also known as the Viterbi alignment of the sentence pair (Och and Ney 2003). For Model 1, it can be simply summarized as shown in the following equation.

$$\hat{a}_j = \operatorname{argmax}_i (e_j | f_i); \quad j = 1, \dots, J \quad (6)$$

The pseudo code for the best alignment is provided by Jochim et al. (2011).

Note that this alignment model only includes one-to-one and one-to-many alignments while many-to-one or many-to-many alignments between languages are usual and this fact imposes a major restriction on IBM models.

### 3.2 Symmetric word alignment

In the preceding section, it was mentioned that the statistical alignment model has some limitations. One way to lift these limitations is the symmetrization of word alignments. First, we run IBM model training in both directions and then we combine the results in various ways. Although several methods of symmetrization are proposed (Koehn 2010; Och and Ney 2004; Wu and Wang 2007), only three of them have been considered in this paper.

Let  $A_1$  and  $A_2$  be alignments according to source-to-target and target-to-source translation directions, respectively. The matrices of  $A_1$  and  $A_2$  named alignment matrix can be created, separately. The rows are related to the initial target words and the columns are referred to the initial source words. Note that in both matrices, rows and columns are alike. The alignment between words are represented by points in the matrix.

The above-mentioned symmetrization methods are defined in the following:

1. Intersection:  $A = A_1 \cap A_2$ ,
2. Union:  $A = A_1 \cup A_2$ ,
3. Grow-diag: It starts with the alignments in the intersection set.

Then it grows by adding adjacent alignment points in the union sets but not at the intersection. These neighbors include left, right, top, or bottom and also diagonally adjacent points in the alignment matrix.

## 4 Evaluation of the word alignment quality

Several methods have been presented in the literature to compute word alignments (Jochim et al. 2011). Therefore, the measurement of alignment quality is an important issue. For this purpose, firstly a small sentence-aligned parallel corpus is considered as a test set. Note that a few number of these test sets for the variety of paired European

**Table 2** Precision and recall concepts

	Relevant	Not relevant
Retrieved	True positive (tp)	False positive (fp)
Not retrieved	False negative (fn)	True negative (tn)

languages is available at the Europarl website in <http://statmt.org/europarl/>. Then, each sentence pair should be aligned word by word by human annotators. It means that, in each sentence pair the source language words is linked to the corresponding target language words by some experts. The final result of these aligned words is based on integration of at least the two experts' comments. This result is called "gold standard alignment" or "reference alignment" (Vulić and Moens 2010). It should be noted that there are different alignment guidelines to create a good gold standard (Vulić and Moens 2010).

To evaluate the quality of the word alignment, the alignments computed by an algorithm will be compared to the gold standard. The comparison would be based on one or more desired metrics. In this work, the words of the test set has been aligned based on the model made by a training set. After that, the result is compared to the gold standard with respect to the following metrics.

Precision and Recall are two common metrics in clustering, binary classification and various branches of NLP like information retrieval (IR). Precision is the ratio of true relevant samples among all retrieved ones. While, Recall is the ratio of true relevant samples among all the relevant ones (There is a similarity concept between these two metrics and type I and II in hypothesis testing). In Table 2, these concepts have been displayed.

Now, their harmonic average ( $\frac{2PR}{P+R}$ ) can be considered as a combined measure named *F-measure* (Sasaki 2007). Therefore,  $1 - \frac{2PR}{P+R}$  can be as a measure of error called Alignment Error Rate (AER).

Note that usually a weighted *F-measure* as the following is considered:

$$F\text{-measure} = \frac{1}{\alpha \frac{1}{P} + (1 - \alpha) \frac{1}{R}} \quad (0 \leq \alpha \leq 1) \quad (7)$$

For  $\alpha = \frac{1}{2}$ ,  $AER = 1 - (F\text{-measure})$ .

Now, these measures can be utilized for the evaluation of the quality of the word alignment as follows.

$$Precision(A, A_r) = \frac{|A \cap A_r|}{|A|} \quad (8)$$

$$Recall(A, A_r) = \frac{|A \cap A_r|}{|A_r|} \quad (9)$$

Here,  $A$  is the result of the word alignment using the assumed algorithm and  $A_r$  is the alignment in the gold standard which is determined by the experts. Note that



$A$ ,  $A_r$  and  $A \cap A_r$  have been taken as the retrieved samples, relevant samples and true positive ones, respectively.

During constructing of the gold standard (making links between word pairs of the sentences in the test set), there is possibility that the experts are not sure about their decision. Therefore, Och and Ney (2000) proposed using “Sure alignment points” and “Possible alignment points” (referred to as S and P) to align the word pairs. S and P are related to unambiguous and ambiguity links, respectively. In fact, P is corresponding to the aligned words in the idiomatic expressions, free translation and etc.

Now, Precision, Recall, AER and F-measure can be used to evaluate the quality of the word alignment as follows.

$$\textit{Precision}(A, P) = \frac{|A \cap P|}{|A|} \quad (10)$$

$$\textit{Recall}(A, S) = \frac{|A \cap S|}{|S|} \quad (11)$$

$$\textit{AER}(A, P, S) = 1 - \frac{|A \cap P| + |A \cap S|}{|S| + |A|} \quad (12)$$

$$\textit{F-measure}(A, P, S, \alpha) = \frac{1}{\frac{\alpha}{\textit{Precision}(A, P)} + \frac{(1-\alpha)}{\textit{Recall}(A, S)}} \quad (13)$$

In Eq. 13, the trade-off between Precision and Recall is assigned by  $\alpha$  ( $0 < \alpha < 1$ ). Koehn (2010) had shown that the best results can be obtained for  $\alpha$  between 0.2 and 0.4.

If we have two experts to create the gold, then we have S1, S2, P1 and P2. The final S is the intersection of S1 and S2, while final P is the union of P1 and P2.

Fraser and Marcu (2007) introduced a more precise measure as follows.

$$\textit{F-measure}(A, S, \alpha) = \frac{1}{\frac{\alpha}{\textit{Precision}(A, S)} + \frac{(1-\alpha)}{\textit{Recall}(A, S)}} \quad (14)$$

Note that, it is defined only by S.

All the five above measures can be obtained by the package for a computed word alignment.

## 5 The main functions in the word.alignment packages

The current section offers a brief description of word.alignment to make it more clear for the end-user. The package aims to provide a general framework for computing the statistical word alignment, symmetric alignments, building an automatic bilingual dictionary and evaluating the quality of the word alignment.

Table 3 presents a brief summary of the seven major functions in the package.

**Table 3** Summary of functions in the `word.alignment` package

Function	Description	Main arguments
<code>prepare.data()</code>	Prepares the input (sentence pairs) from a given sentence-aligned parallel corpus for <code>align.ibm1()</code> and <code>evaluation()</code> functions	<p><code>file.sorc (file.trgt)</code>: source (target) language file name</p> <p><code>n</code>: Number of sentences to be read from the corpus</p> <p><code>encode.sorc (encode.trgt)</code>: the encoding of the source (the target) language</p> <p><code>min.len (max.len)</code>: a minimum (maximum) length of each sentence</p> <p><code>remove.pt</code>: if TRUE, all punctuation marks will be removed</p> <p><code>word.align</code>: if FALSE, it divides each sentence into its words. If TRUE, the sentences are not tokenized, but they prepare to the function <code>align.ibm1()</code></p>
<code>align.ibm1()</code>	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 (the source and the target) and also shows word translation probability as a <code>data.table</code>	<p>Most of its arguments are the same as <code>prepare.data()</code></p> <p><code>iter</code>: the number of iterations for IBM Model 1</p> <p><code>name.sorc (name.trgt)</code>: a description for the source (target) language</p> <p><code>dtfile.path</code>: to run this function for the first time, <code>dtfile.path</code> must be assigned to NULL. In this case, the function will automatically store the required <code>data.table</code> (to obtain MLE of parameters) with a name which is a combination of <code>f1</code>, <code>e1</code>, <code>n</code>, and <code>iter</code>, i.e., 'f1.e1.n.iter.RData'</p>
<code>align.symmet()</code>	It finds source-to-target and target-to-source alignments using IBM Model 1, and it merges alignments as intersection, union, or grow-diag	<p>Most of its arguments are the same as <code>prepare.data()</code> and <code>align.ibm1()</code> except for 'method'</p> <p><code>method</code>: its options are 'union', 'intersection' and 'grow-diag' according to method of the symmetric alignment ('union', 'intersection', or 'grow-diag') as described in Sect. 3.2</p>
<code>bidictionary()</code>	It builds an automatic dictionary of two languages based on a given sentence-aligned parallel corpus	<p>Almost all of its arguments are the same as <code>prepare.data()</code> and <code>align.ibm1()</code> functions</p> <p><code>prob</code>: it is the minimum probability of word translation which is used to build the dictionary</p>

Table 3 continued

Function	Description	Main arguments
cross.table()	It is a function to create cross tables of the source language words versus the target language words of each sentence pair as gold standard or as 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	<p>Rather than the arguments similar to <code>prepare.data()</code>, there are a few more arguments as follows:</p> <p><code>out.format</code>: a character string including two options. For "rdata" format, it constructs a cross table of the source language words versus 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</p> <p><code>null.tokens</code>: logical. If TRUE, "null" is added at the first of each source and target sentence, when we use "rdata" format</p>
align.test ()	It aligns the source language words with the target language words for a bilingual sentence pair in a given sentence-aligned test set. This process is based on the results of IBM Model 1 which implemented in the training parallel corpus	<p>Almost all its arguments are the same as the arguments of the functions <code>prepare.data()</code> and <code>align.ibm1()</code> functions</p> <p><code>file.sorc (file.trgt)</code>: source (Target) language file name in the training set</p> <p><code>test.sorc (test.trgt)</code>: source (Target) language file name in the test set</p> <p><code>null.tokens</code>: logical. If TRUE, "null" is added at the first of each source sentence of the test set</p>
evaluation()	It measures precision, recall, AER, and F-measures as five metrics to evaluate the quality of the word alignment	<p><code>file.gold</code>: The gold standard file name</p> <p><code>file.align</code>: the alignment file name</p> <p><code>agn</code>: a character string including two options. If "my.agn", the user wants to evaluate one-to-many word alignment using the function <code>align.ibm1()</code> in this package. If "an.agn", the user wants to evaluate word alignment results which are obtained by another software</p>

**Table 4** Comparison between the ability of aligning for three different computers

RAM	CACHE	CPU	The minimum of the number of words in the used sentence pairs
8 G	Intel	i5 6400	3,000,000
8*8 G = 64 G	Intel	i7 7700k	13,000,000
8*8 G = 64 G	Intel Xeon X5570 2.93 GHz	i7	27,000,000

## 6 An example

In the `word.alignment` package, there is a memory limitation for the number of sentence pairs. It depends to the number of words in sentence pairs as well as the computer's specifications (RAM, CPU, cache, etc.). In the following, there is a comparison between the ability to do word alignment for three different computers in terms of the minimum of the number of words in the used sentence pairs (Table 4).

Several corpus like Bulgarian–English, Swedish–English, French–English and Persian–English [the three first have been taken from the Europarl corpus and the last has been taken from Mizan (Supreme Council of Information and Communication Technology 2013)] have been used to compare the three computers. Then, the minimum of the number of words in the used sentence pairs which the computer can be align have been reported. Notice that the difference between the various corpus is the difference in the number of prepositions in them.

### 6.1 align.ibm1 example

As described in Sect. 1, a large corpus of aligned sentences is required in two separated files as the input. Here, the Swedish–English corpus has been considered. Note that it is selected because its gold standard is available in Holmqvist and Ahrenberg (2011) while one can easily set each desired language pairs as a sentence-aligned parallel corpus.

In order to compute the word alignment using the function `align.ibm1` for Swedish–English, it is sufficient to write the code lines in the following. The arguments `file.sorc` and `file.trgt` have been achieved from Swedish–English Europarl corpus.

```
R> install.packages('word.alignment', repos = 'https://cran.r-project.org/')
R> library(word.alignment) ### load the package word.alignment
R> # the source and the target language filenames as inputs'
R> file.sorc <- url('http://www.um.ac.ir/~sarmad/word.a/sv-en10000.sv')
R> file.trgt <- url('http://www.um.ac.ir/~sarmad/word.a/sv-en10000.en')
R> align.ibm1(file.sorc, file.trgt, n = 10000, encode.sorc = 'UTF-8')
```

Note that as a default, `min.len` is 5 and `max.len` is 40. Because, typically according to Déchelotte et al. (2007), Okita (2009) sentences with length longer than `X1` and shorter than `X2` are removed for word alignment, especially in its application for SMT. `X1` and `X2` are optionally minimum and maximum limitation to the length of a sentence, respectively. This mechanism is known as Sentence Cleaning Algorithm

(Okita 2009). Meanwhile, here, this function is called for the first time. Therefore, `dtfile.path` is assigned to `NULL`. Then, the constructed `data.table` including all of the combinations of words in each sentence pair and their corresponding probability is saved with the name `'sv.en.10000.5.RData'`. In future calls to the function, the saved file name can be set in the `dtfile.path` argument. In fact, the function `align.ibm1` picks up the results in `'sv.en.10000.5.RData'` and will not construct it again, so this would save us some time.

```
R> align.ibm1(file.sorc, file.trgt,
+ dtfile.path = url('http://www.um.ac.ir/~sarmad/word.a/f.e.10000.5.RData'))
# for next times
```

The results are as follows:

```
C:/Users/Asus/Documents/f.e.9919.5.RData created
result.f.e.9919.5.RData created
Time difference of 37.37314 secs
The model is IBM1
The number of input sentence pairs is 9919
The number of used sentence pairs is 8033
The number of iterations for EM algorithm is 5
Word alignment for some sentence pairs are
```

```
1: null jag förklarar Europaparlamentets session återupptagen
efter avbrottet den17 december Jag vill på nytt önska er ett
gott nytt år och jag hoppas att ni haft en trevlig semester
```

```
[1] i --> jag           declare --> förklarar
[3] resumed --> återupptagen the --> null
[5] session --> session   of --> den
[7] the --> null          European --> Europaparlamentets
[9] Parliament --> Europaparlamentets adjourned --> session
[11] on --> förklarar     Friday --> återupptagen
[13] 17 --> 17            December --> december
[15] 1999 --> semester   and --> och
[17] I --> Jag            would --> vill
[19] like --> vill         once --> nytt
[21] again --> nytt       to --> att
[23] wish --> önska       you --> ni
[25] a --> ett            happy --> semester
[27] new --> nytt         year --> år
[29] in --> på            the --> null
[31] hope --> hoppas     that --> att
[33] you --> ni           enjoyed--> semester
[35] a --> ett            pleasant --> semester
[37] festive --> semester period --> semester
```

2: null som ni kunnat konstatera ägde den stora år 2000-buggen aldrig rum Däremot har invånarna i ett antal av våra medlemsländer drabbats av naturkatastrofer som verkligen varit förskräckliga

[1] although --> ägde	as --> som
[3] you --> ni	will --> rum
[5] have --> har	seen --> ägde
[7] the --> null	dreaded --> ägde
[9] millennium --> ägde	bug --> ägde
[11] failed --> ägde	to --> null
[13] materialise --> ägde	still --> ägde
[15] the --> null	people --> ägde
[17] in --> i	a --> ett
[19] number --> antal	of --> av
[21] countries --> ägde	suffered --> drabbats
[23] a --> ett	series --> ägde
[25] of --> av	natural --> naturkatastrofer
[27] disasters --> naturkatastrofer	that --> konstatera
[29] truly --> ägde	were --> ägde
[31] dreadful --> ägde	

3: null ni har begärt en debatt i ämnet under sammanträdesperiodens kommande dagar

[1] you --> ni	have --> har
[3] requested --> begärt	a --> en
[5] debate --> debatt	on --> ämnet
[7] this --> null	subject --> ämnet
[9] in --> i	the --> null
[11] course --> sammanträdesperiodens	of --> null
[13] the --> null	next --> kommande
[15] few --> dagar	days --> dagar
[17] during --> under	this --> null
[19] part-session --> sammanträdesperiodens	

8031: null skulle ni kunna ge ett svar på detta

```
[1] could --> kunna   you --> ni           please --> svar   give --> ge
[5] an --> ett        answer --> svar  on --> på        this --> detta
[9] issue --> svar
```

8032: null ni oroar er över att få veta om vi skall finansiera en anläggning av kärnkraftverk någonstans med strukturfonder Jag förenklar

```
[1] you --> ni                are --> någonstans
[3] concerned --> oroar      about --> kärnkraftverk
[5] whether --> veta         the --> null
[7] Structural --> strukturfonder Funds --> strukturfonder
[9] will --> förenklar       be --> skall
[11] used --> förenklar      to --> att
[13] finance --> finansiera   the --> null
[15] installation --> anläggning of --> av
[17] nuclear --> kärnkraftverk plants --> förenklar
[19] anywhere --> någonstans
```

8033: null man kan säga att det inte är fråga om det

```
[1] quite --> säga   simply --> säga   there --> det   is --> är
[8] no --> inte      question --> fråga of --> null     this --> det
```

The results are stored as an object of align class except the two first lines. They are defined during the program.

Now, the above outputs have been explained:

- The stored associated data.table path.
- The main outputs filename (stored). In case of asking to access all the results, they will easily be found with the given address.
- Time difference is the program runtime.
- A model that utilizes the word align. Here IBM Model 1 is implemented.
- The number of input sentence pairs which is 10,000 here.
- The remained sentence pairs after removing the sentences with shorter than min.len and longer than max.len. In the example, 8000(?) sentence pairs remain.
- The number of EM algorithm iterations in order to estimate the IBM Model 1 parameters.
- The word alignment results for the first three and the last three sentence pairs. For instance, i in English sentence is aligned to jag in Swedish sentence in the first sentence pair.

The align class has been utilized for the sake of better and more suitable output illustration of the functions align.ibm1() and align.symmet(). When the class is not defined, in addition to the above components and a series of other outputs, all alignments performed for all sentence pairs are displayed which is not proper.

## 6.2 align.symmet example

Here, the symmetric word alignment for Swedish-English has been run by calling `align.symmet()`, e.g., ‘intersection’ method. This method and other methods have been explained in Sect. 3.2.

```
R> align.symmet (file.sorc, file_train2, n = 10000,
+               encode.sorc = 'UTF-8',
+               method ='intersection')
```

The results are as follows:

```
C:/Users/Asus/Documents/f.e.9919.4.RData created
result.f.e.9919.4.RData created
C:/Users/Asus/Documents/e.f.9919.4.RData created
result.e.f.9919.4.RData created
C:/Users/Asus/Documents/symmetric.intersection.10000.4.RData created
Time difference of 1.183084 mins
The model is symmetric intersection
The number of input sentence pairs is 10000
The number of used sentence pairs is 8033
The number of iterations for EM algorithm is 4
Word alignment for some sentence pairs are
```

1: null jag förklarar Europaparlamentets session återupptagen efter  
avbrottet den 17 december Jag vill på nytt önska er ett gott nytt  
år och jag hoppas att ni haft en trevlig semester

```
[1] i --> jag           declare --> förklarar session --> session
[1] 17 --> 17           December --> december on --> på
[5] a --> ett           year --> år           and --> och
[9] hope --> hoppas     that --> att          you --> ni
[13] pleasant --> semester
```

2: null som ni kunnat konstatera ägde den stora år 2000-buggen aldrig  
rum Däremot har invånarna i ett antal av våra medlemsländer drabbats  
av naturkatastrofer som verkligen varit förskräckliga

```
[1] as --> som          you --> ni           dreaded --> ägde     have --> har
[5] in --> i            a --> ett           number --> antal    of --> av
[9] suffered --> drabbats natural --> naturkatastrofer
```

3: null ni har begärt en debatt i ämnet under sammanträdesperiodens  
kommande dagar

```
[1]you --> ni          have --> har          requested --> begärt
[5] a --> en           debate --> debatt    in --> i
[7] during --> under   part-session --> sammanträdesperiodens
[9]next --> kommande  days --> dagar
```



8031: null skulle ni kunna ge ett svar på detta

```
[1] you --> ni          could --> kunna   give --> ge       an --> ett
[5] answer --> svar    on --> på          this --> detta
```

8032: null ni oroar er över att få veta om vi skall finansiera en anläggning av kärnkraftverk någonstans med strukturfonder Jag förenklar

```
[1] you --> ni          to --> att
[3] be --> skall        finance --> finansiera
[5] installation --> anläggning of --> av
[7] anywhere --> någonstans
```

8033: null man kan säga att det inte är fråga om det

```
[1] simply --> säga    there --> det       no --> inte
[4] is --> är          question --> fråga
```

The above outputs have been demonstrated as follows:

- The stored associated data.table path for one-sided alignment (source-to-target).
- The main outputs filename (stored) for one-sided alignment (source-to-target).
- The stored associated data.table path for one-sided alignment (target-to-source).
- The main outputs filename (stored) for one-sided alignment (target-to-source).
- The main outputs filename (stored) for symmetric alignment (here; intersection).
- The other results are stored as an object of align class like the function align.ibm1().

### 6.3 bidictionary example

The following results have been obtained by running the function on the 10,000 first sentence pairs of the Swedish–English Europarl corpus with `prob = 0.8` and `iter = 15` as an example:

```
R> bidictionary (file.sorc, file.trgt, n = 10000,
+               name.sorc = 'Swedish',
+               name.trgt = 'English',
+               encode.sorc = 'UTF-8')
```

The results are as follows (the brief results have been displayed):

```
C:/Users/Asus/Documents/sv.en.10000.15.RData created
```

```
$time
```

```
Time difference of 2.551829 mins
```

```
$number_input
```

```
1 10000
```

```
$Value_prob
```

```
1 0.8
```

```
$iterIBM1
```

```
1 15
```

```
$Source_Language
```

```
1 "Swedish"
```

```
$Target_Language
```

```
1 "English"
```

```
$dictionary
```

```
[1] "1:1"           "11:11"
[3] "12:12"         "123:123"
[5] "13:13"         "15:15"
[7] "17:17"         "18:18"
[9] "1993:1993"     "1994:1994"
[11] "1996:1996"     "1997:1997"
[13] "1998:1998"     "1999:1999"
[15] "2:2"           "20:20"
[17] "2000:2000"     "2000-2006:2000-2006"
[19] "25:25"         "26:26"
[21] "28:28"         "29:29"
[23] "30:30"         "31:31"
[25] "34:34"         "39:39"
[27] "40:40"         "41:41"
[29] "42:42"         "44:44"
```

[31]	"50:50"	"6:6"
[33]	"7:7"	"70:70"
[35]	"75:75"	"8:8"
[37]	"80:80"	"9:9"
[39]	"98:98"	"accepterar:accept"
[41]	"Agenda:Agenda"	"aldrig:never"
[43]	"alla:all"	"alltför:too"
[45]	"alltid:always"	"allvarliga:serious"
[47]	"ämnen:substances"	"än:than"
[49]	"antalet:number"	"antingen:either"
[51]	"användningen:use"	"arbete:work"
[53]	"arbetslösheten:unemployment"	"året:year"
[55]	"artikel:Article"	"åtta:eight"
[57]	"avfall:waste"	"avtalet:agreement"
[59]	"både:both"	"Barak:Barak"
[61]	"Barnier:Barnier"	"bästa:best"
[63]	"bedrägeri:fraud"	"bedrägerier:fraud"
[65]	"behöver:need"	"behovet:need"
[67]	"Belgien:Belgium"	"betänkande:report"
[69]	"betänkandet:report"	"bilar:cars"
[71]	"Bryssel:Brussels"	"BSE:BSE"
[73]	"budget:budget"	"budgeteten:budget"
[75]	"CNS:CNS"	"dag:today"
[77]	"dagordningen:agenda"	"däribland:including"
[79]	"debatt:debate"	"debatten:debate"
[81]	"december:December"	"deltagande:participation"
[83]	"demokrati:democracy"	"demokratisk:democratic"
[85]	"demokratiska:democratic"	"denna:this"
[87]	"deras:their"	"dess:its"
[89]	"dessa:these"	"detta:this"
[91]	"dialog:dialogue"	"direktiven:directives"
[93]	"direktivet:directive"	"diskriminering:discrimination"
[95]	"dock:however"	"egen:own"
[97]	"egenskap:as"	"eget:own"
[99]	"egna:own"	"ekologiska:ecological"
[101]	"ekonomin:economy"	"eller:or"
[103]	"emellertid:however"	"endast:only"

The outputs of the function `bidictionary()` are as a list of the following components:

- As the previous functions, the corresponding `data.table` path.
- The number of input sentence pairs.
- The minimum probability of word translation which is used to build the dictionary. The default is 0.8.
- The number of EM algorithm iterations to perform IBM Model 1. The default is 15.
- The source language name.

- The target language name.
- A dictionary of the source and the target languages. It is found based on the above minimum probability criterion.

#### 6.4 evaluation example

In this subsection, an applied example is explained to show how the function evaluation has been used. As explained in the previous sections, the training set is the Swedish–English Europarl corpus and the source and the target languages are Swedish and English, respectively.

In order to evaluate the quality of the any word alignment, the gold standard which is explained in Sect. 4 is required. It will be constructed with the following steps:

1. Using the function `cross.table`, an excel file with  $n$  (number of sentences in the test set explained in Sect. 4) sheets of the two languages is built. Each sheet includes a cross table of the two language words related each sentence pair.
2. Then, the file is given to two annotators to be filled by Sure/Possible (1/2). The final gold will be the intersection of the Sure alignments and the union of the Possible alignments provided by the two annotators.
3. Then using the function `excel2rdata`, it is converted to `rdata` format.

For the current example, as previously mentioned, the gold standard for Swedish–English was available in Holmqvist and Ahrenberg (2011) named `ep-ensv-alignref.v2015-10-12/test/test.ensv.naacl`. Therefore, instead of the above step 2, we fill up the cross table based on the gold results in `test.ensv.naacl`.

The gold has been already created (the created gold standard has been located in '<http://www.um.ac.ir/~sarmad/word.a/finalgold.sv.en.RData>'). Now, two modes are considered for the alignments. In the first one, it is assumed that the alignments between words in each sentence pairs have been computed using the function `align.ibm1` in the package. In this case, using the function `align.ibm1`, the alignment between words are constructed with the following code:

```
R> # the source and the target language filenames in the training set as inputs.
R> file.sorc <- url('http://www.um.ac.ir/~sarmad/word.a/sv-en10000.sv')
R> file.trgt <- url('http://www.um.ac.ir/~sarmad/word.a/sv-en10000.en')
R> # the source and the target language filenames in the test set as inputs.
R> test.sorc <- url('http://www.um.ac.ir/~sarmad/word.a/test.sv.naacl')
R> test.trgt <- url('http://www.um.ac.ir/~sarmad/word.a/test.en2.naacl')
R> # the source and the target language filenames in the test set as inputs.
R> align.test(file.sorc, file.trgt, test.sorc, test.trgt, n.train = 1000,
+           minlen.train = 1, maxlen.train = 100,
+           minlen.test = 1, maxlen.test = 100,
+           encode.sorc = 'UTF-8', iter = 5)
```

The result of the above command line is as follows:

```
alignment.f.e.1000.5.RData created
```

therefore, having the gold and the alignment files, the code for the above example is:

```
R> ## the gold standard and the computed alignment filenames as inputs.
R> file.gold <- url('http://www.um.ac.ir/~sarmad/word.a/finalgold.sv.en.RData')
R> file.align <- url('http://www.um.ac.ir/~sarmad/word.a/alignment.f.e.1000.5.RData')
R> evaluation(file.gold, file.align, agn = 'my.agn')#Then press Enter.
```

**Table 5** Evaluation of the quality of the word alignment by five metrics

Method	Recall	Precision	AER	F-measure.PS ( $\alpha = 0.3$ )	F-measure.S ( $\alpha = 0.3$ )
word.alignment (IBM 1)	0.31	0.03	0.81	0.07	0.33
word.alignment (union)	<b>0.50</b>	0.02	0.79	0.07	0.42
word.alignment (intersection)	0.34	0.02	0.76	0.06	0.41
word.alignment (grow-diag)	0.44	0.03	<b>0.73</b>	0.10	<b>0.48</b>
<b>pialign</b> (one-to-one)	0.25	<b>0.05</b>	0.84	<b>0.12</b>	0.26
<b>pialign</b> (one-to-many)	0.27	<b>0.05</b>	0.83	<b>0.12</b>	0.27
<b>pialign</b> (many-to-many)	0.37	<b>0.05</b>	0.85	<b>0.12</b>	0.31
<b>Giza++</b> (IBM 1)	0.45	0.02	0.81	0.05	0.4

The second mode is when the word alignments have been computed by another method (symmetric alignments described in Sect. 3.2) or another word alignment tool. For example, **pialign** tool. In this issue:

- Concatenating the training set and the test set.
- Run **pialign** over both of them.
- Extracting word alignments.
- Selecting the part corresponding to the test set and evaluating the quality of the word alignment. Notice that using the function `cross.table`, an excel file with  $n$  sheets of the two languages have been created. then, we fill up the cross table based on **pialign** results for the test set (it has been located in '<http://www.um.ac.ir/~sarmad/word.a/result.pialign.RData>').

The same steps are taken for the rest of the methods.

The code for measuring the quality of the word alignment is:

```
R> ## the gold standard and the computed alignment filenames as inputs.
R> file.gold = url('http://www.um.ac.ir/~sarmad/word.a/finalgold.sv.en.RData')
R> file.align = url('http://www.um.ac.ir/~sarmad/word.a/result.pialign.RData')
R> evaluation(file.gold, file.align, agn = 'an.agn')#Then press Enter.
```

The results of these evaluations are summarized in Table 5.

The most precise numbers in each column have been bolded.

From Table 5, it can be seen that if the optimal criteria explained in Sect. 4 have been intended, the grow-diag is the best. According to the results, in general, `word.alignment` shows the best overall performance in comparison to the other packages tested.

## 7 Future work

We introduced a new R package named `word.alignment` for computing the statistical word alignment, symmetric alignment, as well as constructing the automatic bilingual dictionary and evaluating the quality of the word alignment by considering five metrics.

The current version of `word.alignment` provides the most applicable statistical word alignment based on IBM Model 1 and the package can be used for it. Note that there are other statistical word alignments like IBM Models 2 to 5 which have other applications in NLP such as SMT. The package `word.alignment` can be considered as an appropriate starting point in R for this family. Ultimately, it is our plan to solve the memory restriction in the next versions of this package.

**Acknowledgements** We sincerely thank Associate Editor for comments that greatly improved the manuscript. We hereby acknowledge that parts of this computation was performed on the HPC center of Ferdowsi University Of Mashhad (Grant No. 28609).

## References

- Benoit K, Watanabe K, Wang H, Nulty P, Obeng A, Müller S, Matsuo A (2018) `quanteda`: an R package for the quantitative analysis of textual data. *J Open Source Softw* 3(30):774. <https://doi.org/10.21105/joss.00774>
- Brown PF, Cocke J, Pietra SAD, Pietra VJD, Jelinek F, Lafferty JD, Mercer RL, Roossin PS (1990) A statistical approach to machine translation. *Comput Linguist* 16(2):79–85
- Brown PF, Pietra VJD, Pietra SAD, Mercer RL (1993) The mathematics of statistical machine translation: parameter estimation. *Comput Linguist* 19(2):263–311
- Brunning JJJ (2010) Alignment models and algorithms for statistical machine translation. Doctoral dissertation. University of Cambridge
- Chéragui MA (2012) Theoretical overview of machine translation. In: *Proceedings ICWIT*, pp 160–169
- Daneshgar N, Sarmad M (2019) `word.alignment`: computing word alignment using IBM model 1 (and symmetrization) for a given parallel corpus and its evaluation. R package version 1.1
- Déchelotte D, Schwenk H, Bonneau-Maynard H, Allauzen A, Adda G (2007) A state-of-the-art statistical machine translation system based on mooses. In: *MT Summit*, pp 127–133
- Dowle M, Srinivasan A, Short T, Lianoglou S, Saporta R, Antonyan E (2017) `data.table`: extension of data.frame. R package version 1.10.4-3
- Feinerer I, Hornik K (2015). `tm`: text mining package. R package version 0.6-1
- Fraser A, Marcu D (2007) Measuring word alignment quality for statistical machine translation. *Comput Linguist* 33(3):293–303
- Holmqvist M, Ahrenberg L (2011) A gold standard for English–Swedish word alignment. In: *Proceedings of the 18th Nordic conference of computational linguistics (NODALIDA 2011)*, pp 106–113
- Hornik K (2015). `NLP`: natural language processing infrastructure. R package version 0.1-7
- Ildefonso T, Lopes GP (2005) Longest sorted sequence algorithm for parallel text alignment. *International conference on computer aided systems theory*. Springer, Berlin, pp 81–90
- Jochim C, Lioma C, Schütze H (2011) Expanding queries with term and phrase translations in patent retrieval. *Information retrieval facility conference*. Springer, Berlin, pp 16–29
- Koehn P (2010) *Statistical machine translation*. Cambridge University Press, Cambridge
- Lardilleux A, Lepage Y (2009) Sampling-based multilingual alignment. In: *International conference on recent advances in natural language processing (RANLP 2009)*. Borovets, Bulgaria
- Moore RC (2005) A discriminative framework for bilingual word alignment. In: *Proceedings of the conference on human language technology and empirical methods in natural language processing*. Association for Computational Linguistics, pp. 81–88

- Neubig G, Watanabe T, Sumita E, Mori S, Kawahara T (2011) An unsupervised model for joint phrase alignment and extraction. In: Proceedings of the 49th annual meeting of the association for computational linguistics: human language technologies, vol 1. Association for Computational Linguistics, pp. 632–641
- Neubig G, Watanabe T, Mori S, Kawahara T (2012) Machine translation without words through substring alignment. In: Proceedings of the 50th annual meeting of the association for computational linguistics: long papers, vol 1. Association for Computational Linguistics, pp. 165–174
- Nie JY (2010) Cross-language information retrieval. *Synth Lect Hum Lang Technol* 3(1):1–125
- Och FJ (2000) Giza++: training of statistical translation models. Technical report, RWTH Aachen, University of Technology
- Och FJ, Ney H (2000) A comparison of alignment models for statistical machine translation. In: COLING 2000, volume 2: the 18th international conference on computational linguistics
- Och FJ, Ney H (2003) A systematic comparison of various statistical alignment models. *Comput Linguist* 29(1):19–51
- Och FJ, Ney H (2004) The alignment template approach to statistical machine translation. *Comput Linguist* 30(4):417–449
- Okita T (2009) Data cleaning for word alignment. In: Proceedings of the ACL-IJCNLP 2009 student research workshop. Association for Computational Linguistics, pp. 72–80
- Sasaki Y (2007) The truth of the F-measure. *Teach Tutor Mater* 1(5):1–5
- Simes A, Almeida JJ (2003) NATools-a statistical word aligner workbench. *Proces Leng Nat* 31(septiembre 2003), 217–224
- Supreme Council of Information and Communication Technology (2013) Mizan English–Persian Parallel Corpus
- R Core Team (2015) R: a language and environment for statistical computing R Foundation for statistical computing, Vienna, Austria. ISBN 3-900051-07-0
- Vulić I, Moens MF (2010) Term alignment, state of the art overview. Technical report, Katholieke Universiteit Leuven LIIR (Language Intelligence and Information Retrieval)
- Walker A (2017) openxlsx: read, write and edit XLSX files. R package version 4.0.17
- Wang X (2004) Evaluation of two word alignment systems. Institutionen för datavetenskap, Umeå
- Wu H, Wang H (2007) Comparative study of word alignment heuristics and phrase-based SMT. In: Proceedings of the MT Summit XI

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.