# A Survey on Statistical-based Parallel Corpus Alignment

Sulema Torres-Ramos, Raymundo E. Garay-Quezada

Universidad de Guadalajara, Departamento de Ciencias Computacionales, Jalisco, México
sulema.torres@cucei.udg.mx, raymundo.garay@alumnos.udg.mx

**Abstract.** The text alignment is an important process of different Machine Translation systems. This task consists in identifying correspondences between words, sentences or paragraphs of a source text and their translation (parallel corpus). There are two main approaches to perform parallel corpus alignment: the statistical-based methods and lexical-based methods. In this paper, the main statistical-based methods for align parallel corpus are presented.

**Keywords:** Statistical Corpus Alignment, Parallel Corpus, Machine Translation, Natural Language Processing.

## 1 Introduction

Natural Language Processing (NLP, also known as Computational Linguistics) is a field of Computer Science which aims to create and understand the language that human beings use to communicate. NLP has a lot of important tasks and applications; one of them is Machine Translation, which aims to automatically translate a text from one language to another one.

To make Machine Translation possible, there are several techniques based on dictionaries, statistics or examples. Even though these techniques have their own advantages and disadvantages, they share some methods like text alignment process [3].

Text alignment process consists in organize parallel corpus in order to establish a correspondence between paragraphs, sentences and/or words [16] of the source texts and their translations. Parallel corpus can be defined as two sets of texts in different languages where one of these sets is the source text and the other one is their translation.

There are two main approaches to perform parallel corpus alignment: the statistical-based methods and lexical-based methods. Lexical-based approaches rely on existing lexical knowledge such as antonyms and synonyms, translations of a word, etc.; while Statistical-based approaches rely on non-lexical information, such as sentence length, sentence position, co-occurrence frequency, sentence length ratio in two languages, etc. [13].

In this paper, it will be addressed some algorithms which use statistical information to align parallel corpora; from algorithms which can be considered the first in their field such as Gale and Church, Brown, and K-vec algorithm, their upgrades like Va-

nilla, Moore's Association-based algorithm, and DK-vec algorithm. And, some recent algorithms like Bleualign and its Iterative version.

## 2 Machine Translation

Translation is an ancient human activity that consists on communicates a message from an "original" language into a "terminal" language always taking care of not change the idea meaning. At late 1940s, once the digital computers were developed, Warren Weaver had the idea that, the computers could perform automatic translation; he conceives the problem of automatic translation as cryptography problem [30]. Since then, many efforts were made on this new task both in hardware, by improving memory and access store resources, as in software with the dictionary-based translation as its principal approach [29].

By 1966, a publication of the ALPAC reported, as a result of an investigation, that machine translation had no future but, the computational linguistic, and machine-aided translation look promising; the later resulted on a low investments on former task [25].

In despite of ALPAC report, many researchers kept their effort on developing machine translation systems, looking for the best translation arguing that, a post-edition was economically viable [29]. As a result, many system were developed during this period of time, most of them categorized as the Rule-based Machine Translate approach (RBMT), giving rise to the resurgence of machine translation task, during 1980, and beginning with the development of two new approaches, known as Example-based machine translation (EBMT) [22], and statistical machine translation (SMT) [1], the latter based on Weaver's very first ideas on machine translation.

### 2.1 Machine Translation Paradigms

#### *2.1.1 Ruled-based*

The Rule-based machine translation strategy has an explicit linguistic knowledge base, i.e. a linguistic expert builds the necessary grammatical rules to perform a better translation, however, the translation effectiveness vary depending on the deepness of the logical representation of the sentences. The deeper the rule abstraction, the more complex the task of mapping a sentence to its translation [29].

#### *2.1.2 Example-based*

This approach is based on tagged corpus used as an example-database. A sentence to be translated is compared with a database of examples to identify the similar sentences. In order to achieve this, a sentence is aligned against several examples-templates, then, the more similar template, based on the alignments, is used to retrieve the possible translations [29].

### *2.1.3 Statistical*

As the Example-based, the Statistical approach use corpora in order to achieve the translation but instead of compute the similarity among sentences, this approach employ two process: training and decoding. As all the supervised algorithms, this one use a set of examples in order to create a statistical model that represents the target language, then, when a translation is requested, the correct translation is search in the space of all the possible translation learned in the statistical model, until find the one with the better probability [29].

## 3 Text Alignment

To make the text alignment process possible, we need to use a *written corpus*[1] [17]. This can be defined as a collection of texts or a huge text that is used for research, especially for developing translation software and natural language processing [3]. Corpora can be labeled or unlabeled. On one hand, labeled corpora are annotated to identify various attributes or linguistic information such as the topics or themes of the documents contained in the corpora, or the part of speech of the words, among others. For example, the labeled attributes in a corpus for the word "roses" could be *noun*, *plural*, etc. Linguistic information that could be labeled would be its *lemma*[2], the correct sense of the word according to a specific dictionary, etc.; and, in other languages like Spanish, labels like *feminine noun* or *masculine noun* could be added. Figure 1 shows a sample of the SemEval-2015 task 13 corpus [20] which consists in four documents taken of European Medicines Agency documents, the KDE manual corpus and the EU bookshop corpus.

```
<?xml version="1.0" encoding="UTF-8" ?>
<corpus lang="en">
<text id="d001">
<sentence id="d001.s001">
<wf id="d001.s001.t001" pos="X">This</wf>
<wf id="d001.s001.t002" lemma="document" pos="N">document</wf>
<wf id="d001.s001.t003" lemma="be" pos="V">is</wf>
<wf id="d001.s001.t004" pos="X">a</wf>
<wf id="d001.s001.t005" lemma="summary" pos="N">summary</wf>
<wf id="d001.s001.t006" pos="X">of</wf>
<wf id="d001.s001.t007" pos="X">the</wf>
<wf id="d001.s001.t008" lemma="european" pos="J">European</wf>
<wf id="d001.s001.t009" lemma="public" pos="J">Public</wf>
<wf id="d001.s001.t010" lemma="assessment" pos="N">Assessment</wf>
<wf id="d001.s001.t011" lemma="report" pos="N">Report</wf>
<wf id="d001.s001.t012" pos="X">(</wf>
<wf id="d001.s001.t013" lemma="epar" pos="N">EPAR</wf>
<wf id="d001.s001.t014" pos="X">)</wf>
<wf id="d001.s001.t015" pos="X">.</wf>
</sentence>
```

**Figure 1.** A sample of the corpus used for the task 13 in SemEval-2015.

[1] From now on, written corpus will be addressed as corpus or corpora
[2] Is the canonical form, dictionary form, or citation form of a word

This corpus has been tagged with a special notation that identifies /sentence/word/lemma/pos/id. On the other hand, unlabeled corpora have no linguistic information and do not have a defined structure; it is often user-generated information such as email or instant messages, documents or social media postings.

There are different types of corpus (see Figure 2). In the field of machine translation, the classification of *monolingual* and *multilingual corpus* is important:

1. Monolingual corpora are texts in only one language.
2. Multilingual corpora are texts in multiple languages, and they can be divided in the following sub-categories:
   a. *Parallel corpus* can be defined as two sets of texts in different languages where one of these sets is the source texts and the other one is their translations. Each of these set of texts can be known as *bitexts* [11]. Parallel corpora can be uni-directional, bi-directional, or multi-directional [15]. For example, the Bible and its copies in different languages can be considered parallel corpus.
   b. On the other hand, a *comparable corpus* is a set of texts in different languages that share the same main topic but they differ in the way they address it. It means that a comparable corpus is not a source text and their translation. According to Simões Branão [28] "a set of new articles, from journals or news broadcast systems, as they refer the same event in different languages can be considered Comparable Corpora".

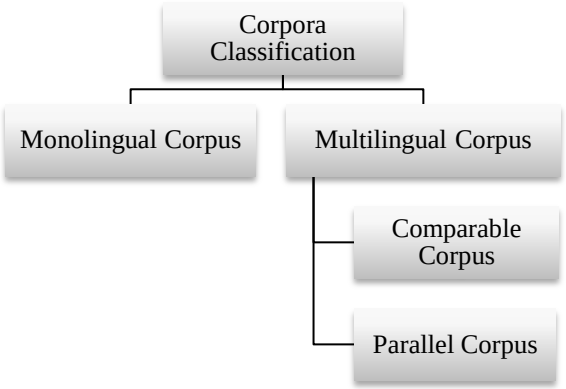


**Figure 2.** Corpora Classification

Corpus alignment consists in organize parallel corpus in order to establish a correspondence between paragraphs, sentences and/or words [16] of the source texts and their translations. Yet the automatic alignment of parallel corpora is not a trivial task for some language pairs [24].

E. Macklovitch [16], recognized 4 levels of alignments (see Figure 3):

- 1st level alignment: this level addresses the alignment of the whole text when the text is not long enough.

- 2nd level alignment: this level recounts the alignment of paragraphs.
- 3rd level alignment: this level describes the alignment of sentences.
- 4th level alignment: this level relates the alignment of words between bitexts.

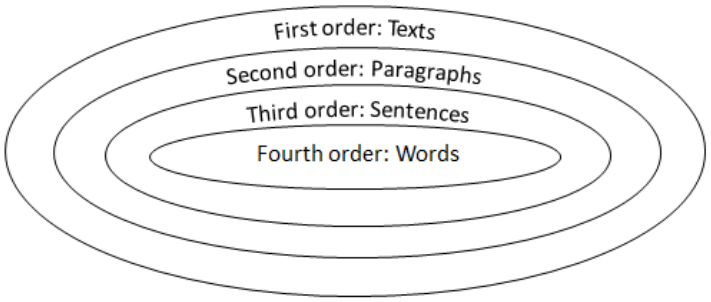


**Figure 3.** Alignment levels.

### 3.1 Corpus Alignment Approaches

There are two main approaches to perform parallel corpus alignment: one approach is based on statistical data, while the other one applies additional linguistic knowledge. The basis of this distinction is related with the kind of data being processed independently of the methods of processing [10]. Several techniques have been developed based on these approaches, each with their own advantages and disadvantages.

Lexical-based approaches rely on existing lexical resources, such as large-scale bilingual dictionaries and glossaries [13], to obtain information about the languages, such as antonyms and synonyms, translations of a word, etc. Some of these techniques are presented in [10][4][12][14][18][13]. These techniques tend to be slower than the techniques based on statistic information and are dependent language. The main disadvantage of these techniques is that their performance depends heavily on the lexical information used on the alignment process. However, many of these methods are being developed because are expected to generate better results than the statistical ones [6].

Statistical-based approaches, which are the basis of this study, rely on non-lexical information, such as sentence length, sentence position, co-occurrence frequency, sentence length ratio in two languages, etc. [13]. These techniques make the alignment process faster, and are, generally, independent language. However, the main disadvantage of these techniques is that their performance depends heavily on the structural similarity between target text and source text of the bitexts. According to [13] "the attraction of these resource-poor approaches arises from the sharp contrast between their poor resources and their rich outcomes".

Figure 4 shows the statistical-based methods discussed in this survey and some lexical-based methods as a reference.

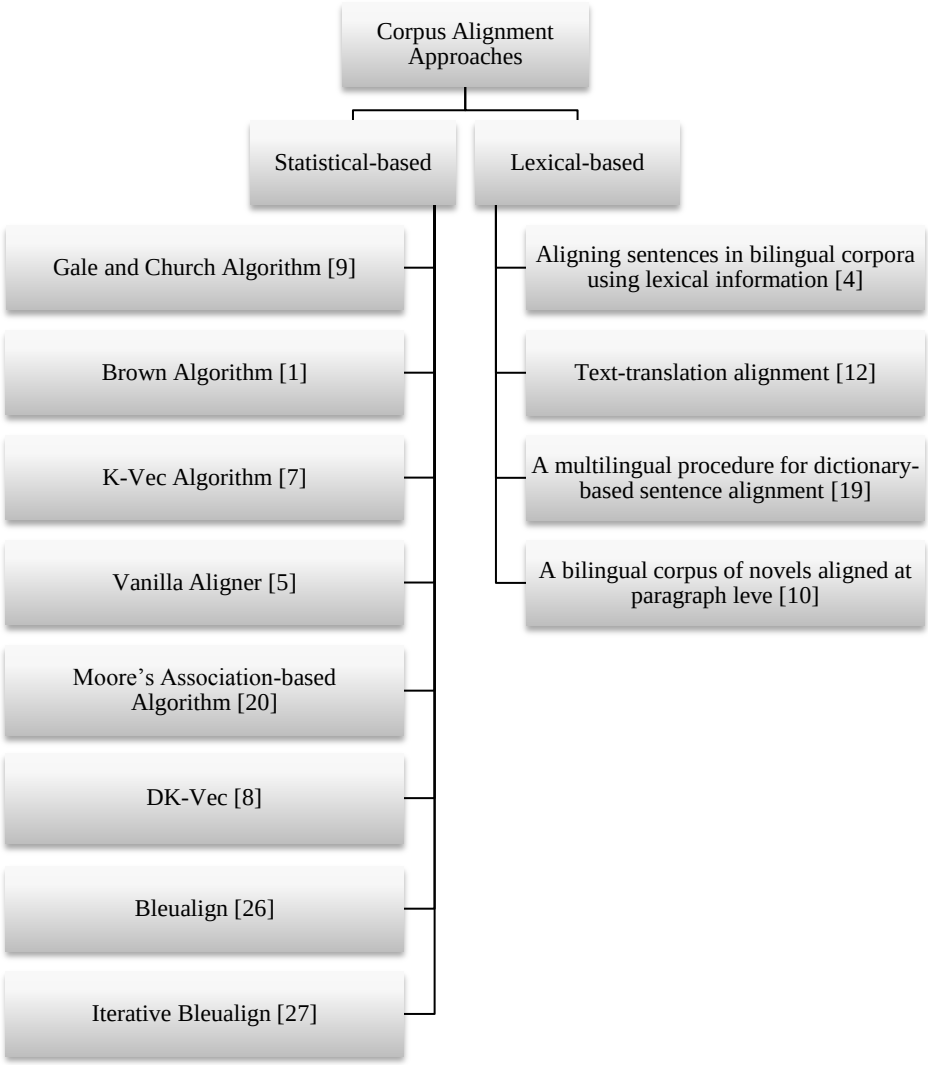


**Figure 4.** Some of the Statistical- and Lexical-based methods for parallel corpus alignment

# 4 Statistical-based Methods for Parallel Corpus Alignment

## 4.1 Gale and Church Algorithm

The main idea behind this sentence level aligner is "longer sentences in one language tend to be translated into longer sentences in the other language, and shorter sentences tend to be translated into shorter sentences" [9]. A parallel corpus already aligned in paragraphs is required by this algorithm.

This algorithm considers the length of the sentences (in characters) to align them. These are used to calculate a value called *distance measure* for each pair of sentences (one of the source texts and one of the target texts). The lower the distance measure, the higher the probability that the sentences correspond themselves. The algorithm considers the following alignment categories [3]:

1. 1-to-1 alignment: this is the best possible scenario, where one sentence of a language (source text) is translated into exactly one sentence in the other language (target text).
2. 1-to-2 alignment: one sentence in the source text is divided into two sentences in its translation.
3. 1-to-0 alignment: one sentence in the source text is not translated in the target text.
4. 0-to-1 alignment: one sentence in the target text was added by the translator.
5. 2-to-2 alignment: two sentences in the source text correspond with two sentences in its translation.
6. 2-to-1 alignment: two sentences in the source text correspond with only one sentence in its translation. This is because the two sentences were merged by the translator.

Each alignment is considered to find the correspondence of each sentence. For instance, as Figure 5 shows, the length of the sentence *s1* (15 characters) is not similar than the length of *t1* (45 characters) nor the length of the combination *t1* y *t2* (84 characters). However, the length of the combination *s1* y *s1* (47 characters) and the length of *t1* (45 characters) are more alike. Therefore, the best option, is this case, is the 2-to-1 alignment.

| | | | **Target text** | | |
|---|---|---|---|---|---|
| | | | *t1* | *t2* | *t3* |
| | | | Mi nombre es Guillermo y puedo leer en inglés y español | Sin embargo quiero aprender otros idiomas | Esto es un texto en inglés |
| **Source text** | *s 1* | My name is William | 1 to 1 alignment / 2 to 1 alignment | 1 to 2 alignment | |
| | *s 2* | I can read English and Spanish texts | | 2 to 2 alignment | |
| | *s 3* | This is an English text | | | |

**Figure 5.** Alignment example using Gale and Church Algorithm

Even though the number of characters in a sentence is used as distance measure in the example, the algorithm uses the following distance measure ($D$) [9]:

$$D = -100 * \log(2 * (1 - Prob(\delta))) \quad (1)$$

Where $Prob(\delta)$ is defined as:

$$Prob(\delta) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\delta} e^{\frac{z^2}{2}} dz \quad (2)$$

And $\delta$ is defined as:

$$\delta = \frac{l_2 - l_1 c}{\sqrt{l_1 s^2}} \quad (3)$$

Where:

- $l_2$ and $l_1$ are the lengths of the portions of text
- $c$ also known as *mean,* is the expected number of characters in $l_2$ per character in $l_1$. This value is equal to 1 to simplify the algorithm.
- $s^2$ is the variance of the number of characters in $l_2$ per character in $l_1$. This value is equal to 6.8.

The values of mean and variance are estimated to European languages (English, French, German, Spanish, and etcetera). They could change in other pairs of languages that are not alike, for example, Spanish and Japanese.

### 4.2 Brown Algorithm

This sentence level aligner was developed by Brown *et. al*. [1]. Similar to Gale and Church, this algorithm considers the length of the sentences to align them, but the difference is that the measure sentence length is in words. They use the tags included in the TEX format of the Canadian Hansard corpus as anchor points to help in the alignment process. The algorithm considers major and minor anchor points and performs the alignment in two steps:

1. Aligning the major anchors:
   (a) To each possible section alignment, a cost is assigned, rewarding the exact matchings and penalizing omissions and inexact matchings.
   (b) Determine the alignment with the least total cost, using dynamic programming. The output is an aligned sequence of sections.
   (c) Sections that not contains the same number of minor anchors in the same order for each corpus are eliminated
2. Aligning the minor anchors:
   (a) Divide the remaining sections (not accepted in the previous step) into subsections.

(b) The algorithm aligns the sentences using a Hidden Markov Model based on the number of words.

### 4.3 K-vec Algorithm

This algorithm is a word level aligner developed by Pascale Fung and Kenneth Ward Church [7]. K-vec has a very important characteristic; it does not depend on sentence boundaries (for example, periods "." are boundaries in European languages). Using this feature, the algorithm aims to align languages that are not alike; for example, English and Japanese, as well as European languages. The main idea of this algorithm is "if two words are translations of each other, they are more likely to occur in the same segments than two words which are not" [8].

In order to determine if a word is the translation of another one in a bitexts, this algorithm focuses on the similarity of their distributions in the corresponding text. First, the algorithm splits the parallel texts into k number of segments. For each word in the text, a K-dimensional binary vector is created and it indicates the presence (value 1) or absence (value 0) of that word in each segment. Note that the first position of the vector corresponds to the first segment of the text, the second position to the second segment, and so on; also, the frecuency or position of the word in each segment is not important. For example, Figure 6 shows parallel texts (Spanish and English) divided into 3 segments (k=3). In this case, the Spanish text has the words "casa" and "computadora", while English text has "house" and "computer". The word "casa" appears in segments 1 and 3 of the Spanish corpus, so the corresponding vector would be **V***spanish-casa* = <1, 0, 1>. The word "computadora" appears in segment 2 of the Spanish corpus, so the corresponding vector would be **V***spanish-computadora* = <0, 1, 0>. The word "house" appears in segments 1 and 3 of the English corpus, so the corresponding vector would be **V***english-house* = <1, 0, 1>. The word "computer" appears in segment 2 of the English corpus, so the corresponding vector would be **V***english-computer* = <0, 1, 0>.

The resulting vectors represent the distribution of each word in their corresponding corpus.

Once the distributions of the words have been calculated, the algorithm computes the similarity for each pair of vectors (one of the source text and one of the target text). The higher the similarity, the higher the probability that the words correspond themselves.

In the previous example the vectors **V***spanish-casa* and **V***english-house* are identical so it can be considering that the word "house" is the English translation of the Spanish word "casa", and vice-verse. On the other hand, the vectors corresponding to "casa" and "computer" are different, therefore it can be determined that "computer" is not the translation of "casa". Nevertheless, calculating the similarity of pairs of vectors goes far beyond simply identifying whether they are equal or not. In the original algorithm Pointwise Mutual Information and T-Score are used as measures of association in order to compute the similarity between pair of vectors (words) [7].

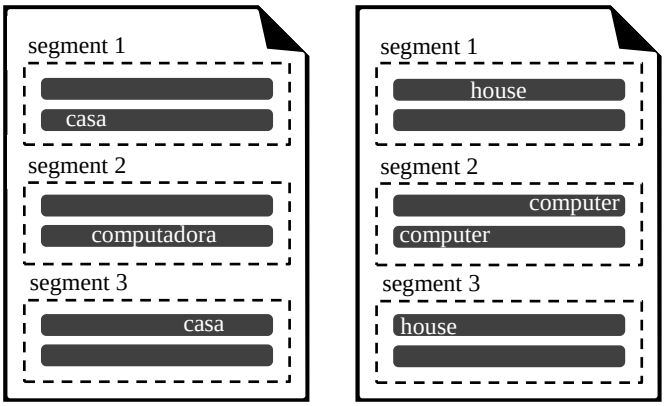


**Figure 6.** Example of the K-vec algorithm.

### 4.4 Vanilla Aligner

Vanilla aligner was presented by Pernilla Danielson and Daniel Ridings in 1997 [5], and is an upgrade of Gale and Church algorithm [9]. The same way that its predecessor this is a sentence level aligner and dependent on sentences boundaries. The main contribution of this aligner is the compatibility to work with bitexts in SGML format. One of the benefits to using bitexts in SGML format is that a standard form or structure can be settled. Therefore it will help to identify sentence boundaries more easily.

Even though Vanilla aligner is based on the Gale and Church aligner using the sentences length (character count) to find correspondences between them, this algorithm implements some modifications to the alignment process:

1. Parsing the bitexts in SGML format: Searches for all spaces and replaces them with an end of line character.
2. Reading through a normalized SGML file and creates input files for the aligner:
   (a) Concatenates all the lines of a paragraph within a <BODY> element to one single line
   (b) Looks for sentence ending punctuation marks and stick an "end of sentence" after each one
   (c) Removes the SGML labels and the double spaces resulting
   (d) Adds the "end of paragraph" code.
3. Finally two files are created. One per language. The words are separated by line and contain labels to point out the sentence and paragraph ending.

The two files generated are the input for the aligner. The Gale and Church aligner used the same input data. Vanilla Aligner also uses the same sentence alignment categories: 1-to-1, 1-to-2, 1-to-0, 0-to-1, 2-to-2, and 2-to-1.

Besides the pre-processing to work with SGML format, another difference between Vanilla aligner and Gale and Church algorithm is the output file, post-

processing and the access to the results. First, Vanilla aligner generates only one output file while Gale and Church algorithm creates two output files. According to Danielsson and Ridings [5] "having two separate files to work with instead of only one, makes it slightly more inconvenient to check the results and look for possible errors". Second, they address some ways of dealing with results. For example, the use of some labeled text indicating the correspondence between sentences, or the use of databases like mSQL.

### 4.5 Association-based Bilingual Word Alignment

There are two main motivations in this work: first, they strongly believed that word-based alignment could be a good startup for phrase-based alignment; second, the algorithms presented until then, for example, Brown *et. al.*[1] presents the disadvantage of high computational complexity, and that it was able to find low computational complexity strategies, but with a proportional good accuracy.

Moore [20] presents three different strategies for word alignment: 1) 1-to-1 word type alignment, 2) n-to-1 alignment, and 3) Token alignment selection. Each strategy has two or more methods to overcome several problems. And all of them are based on Log-Likelihood-Ratio (LLR) association measure which has been used in constructing lexicon translation.

1. 1-to-1 word alignment: The following methods only permit one-to-one alignment and do not take position into account.

   (a) Method 1

   It used the Competitive linking algorithm [18] and use the LLR score as a measure. The algorithm consists in three steps:

   i. Find the word type pair that have the highest association score (LLR) of any pair of words type have not been linked.
   ii. Add one to the count of linked occurrences of this pair of word types, and subtracts one to the unlinked count instances.
   iii. Repeat while words can be linked

   (b) Method 2

   The problem with the later method is that the sentence alignment decision, given a pair of words, is taken independently for the same pair of word in different sentences. This method considers this, and to solve this problem, a second alignment is applied using the conditional probability of a pair of words as an alignment score. The new alignment is defined as follows:

   i. Count the number of links in the training corpus for each pair of words linked in any sentence pair.
   ii. Count the number of co-occurrences in the training corpus for each pair of words linked in any sentence pair by Method 1
   iii. Compute the link probability score [20] for each pair of words linked in any sentence pair by Method 1.

iv. Align sentences pairs by competitive linking using link probability score.

(c) Method 3

The method 2 fails to display monotonic relationships between precision and recall. So, a discounted link probability (LPd) [20] is applied. The algorithm is the same that the one in method 2 but using LPd.

2. n-to-1 alignment: This technique is based on the method 3 of the One-to-one alignment. It is shown that iterative application of such a method could create Many-to-One clusters by building of clusters, incrementally.

3. Token alignment selection methods: This method is a complement to the 1-to-1 and n-to-1 alignment. This addresses the problem of selecting the best word token alignment for a given word type alignment, *i.e*., the incorporation of positional information into associated-based word alignment.

(a) Method A

Make a random choice (without replacement) for each word type, in the alignment, from among tokens of that type.

(b) Method B

A word token alignment consisted with a given word type alignment that is the most monotonic is found, by minimizing the degree of no monotonicity of such alignment. If there is more than one word token with the less degree of no monotonicity, then it is picked arbitrarily.

### 4.6 DK-vec Algorithm

The Dynamic K-vec algorithm, DK-vec for now on, is based on its ancestor, the K-vec algorithm, which works under the assumption that two words are more likely to appear in the same segment if they are translations of each other. However, it usually does not happen with languages that are not alike. In addition, k-vec algorithm does not consider *a priori* information from the language or corpus characteristics reducing its performance [8].

To overcome those disadvantages, the authors propose the DK-vec algorithm, which includes two important characteristics. First, they define the concept of arrival interval, which is the difference between the initial position of a word in a segment and the next appearance of the same word. The set of arrival intervals is known as *recency information*. Second, a pattern matching technique is proposed to gives the algorithm the capability of align vectors of different lengths.

The algorithm works as follow, first, it is necessary to define the position of a word, defined as the number of characters counted from the beginning of the document until the first character of the intended word. For example, if we have the text "This is an example" the position of the word "*an*" is the ninth given that there are nine characters between the first character in the text, which is "t", and the first char-

acter of the word "*an*" which is "a", see Figure 7. It is worth knowing that, when the length of a text is computed, each blank character is counted as one. Then, the *recency information* is calculated using the position vector computed before.

| Text: | ***T*** | ***h*** | ***i*** | ***s*** | | ***i*** | ***s*** | | ***a*** | ***n*** | | ***E*** | ***x*** | ***a*** | ***m*** | ***p*** | ***l*** | ***e*** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| No. Character: | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 |
| Word Position: | ✔ | | | | | ✓ | | | ✓ | | | ✓ | | | | | | |

**Figure 7.** Position of the words (DK-vec algorithm).

Now, suppose that we want to compute the arrival vector for the word "*example*" of a given corpus. First, it is necessary to build the position vector of the word which is [12, 100, 250, 500, 700, 800], i.e., the first character of the word "*example*" appears in the positions 12 of the corpus, the second in position 100 and so on. Then, the recency information vector is built based on the position vector computed before obtaining the following vector [88, 150, 250, 200, 100]. The length of the first vector is equal to the number of times that the word appears in the whole corpus, and the length of the second vector is the length of the first vector minus 1, see Figure 8.

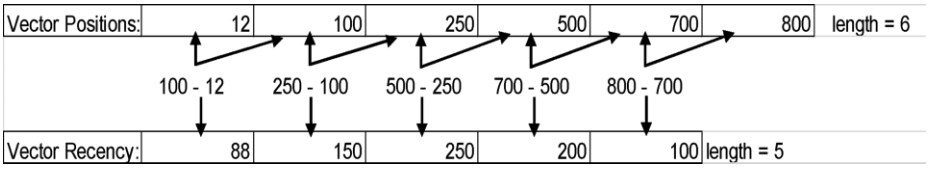


**Figure 8.** Vector positions and Vector recency of the word "example"

To analyze the information it is possible to represent the two vectors, position vector and arrival vector, as a signal creating a graph where the vertical axe is the recency information and the horizontal axe is the word position information.

Now suppose there are three pairs of vectors corresponding to three different words, being two of them its corresponding translation. Their vectors and graphical representation are seen as follows.

1. Word "example"
   (a) $V_{\text{Positions}} =< 12,100,250,500,700,800 >$
   (b) $V_{\text{Recency}} =< 88,150,250,200,100 >$
2. Word "ejemplo"
   (a) $V_{\text{Positions}} =< 50,140,280,520,710,805,860 >$
   (b) $V_{\text{Recency}} =< 90,140,240,190,95,55 >$
3. Word "ver"
   (a) $V_{\text{Positions}} =< 10,100,540,900,1500 >$
   (b) $V_{\text{Recency}} =< 90,440,360,600 >$

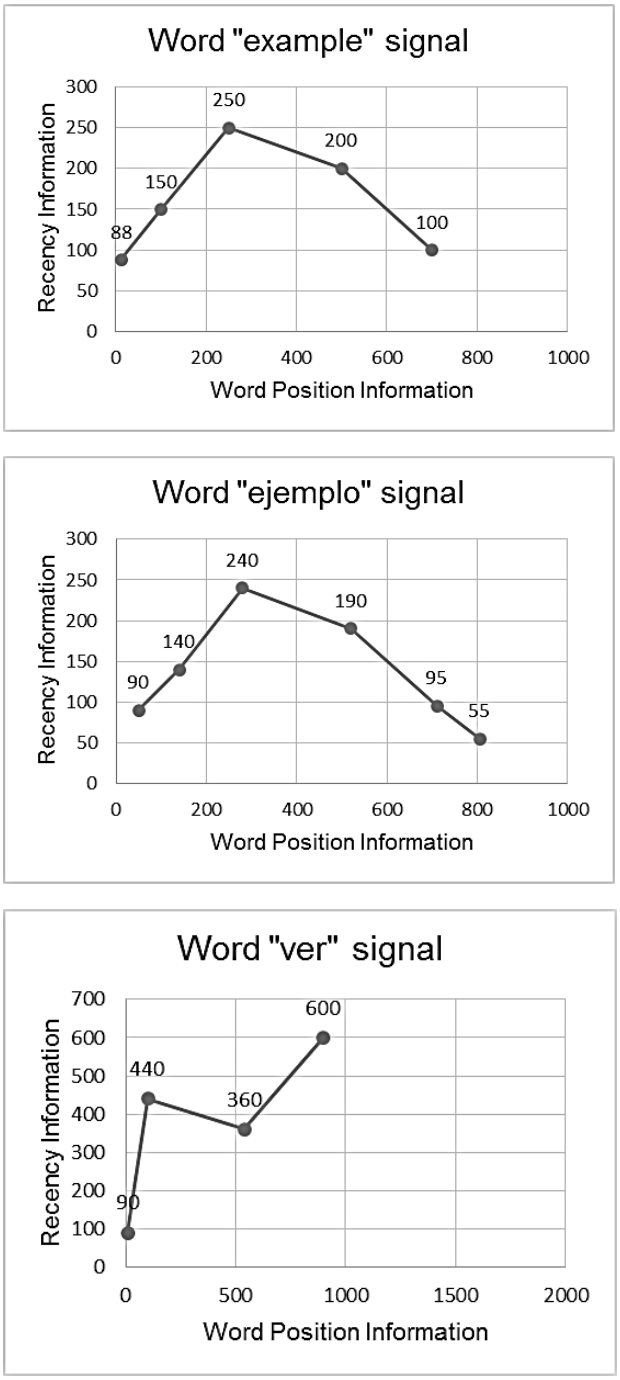


**Figure 9.** Graphs of the words “example”, “ejemplo” and “ver”

In the graphical representation (Figure 9), it is visually clear that two of them are more alike, the ones corresponding to the words "example" and "ejemplo", while they are very different from the graph representing the "ver" word. In order to make a quantitative representation, the Dynamic Time Warping algorithm is used [8].

### 4.7 Bleualign

This sentence aligner was created by Rico Sennrich and Martin Volk [26]. Its main idea is to use a Machine Translation (MT) system and a translation evaluator (BLEU, hence its name) to help in the alignment process.

In order to better understand the aligner, it is necessary to know about BLEU. BLEU is an algorithm which evaluates the quality of an MT output. For the purpose of measure that quality, BLEU obtains a score by comparing the MT output against one or more reference human translations. "The closer a machine translation is to a professional human translation, the better it is" [23]. The BLEU score goes from *0* to *1*. A score *1* means that the MT output is identical to its reference.

This alignment algorithm receives the following input data: source text and target text. The input data have delimiters to divide the text. The algorithm performs the following steps [26]:

1. Translate the source text to the corresponding target language using an MT system.
2. Based on the BLEU algorithm, each sentence of the translated source text is compared to each sentence of the target text to determine the similarity score between pair of sentences.
3. The algorithm keeps the 3 best-scoring alignment candidates for each source sentence.
4. Choose the combination of 1-to-1 alignments that maximizes the BLEU score, using dynamic programming. The result is an ordered list of 1-to-1 alignments.

In the last steps 1-to-1 alignments are made, therefore there may be sentences in both texts (source and target) that remained unaligned. In order to align those sentences, the algorithm does the following:

5. Determine if the 1-to-1 alignments are n-to-1 alignments: For each pair of aligned sentences (*i*, *j*), the sentence of the translated source text (*i*) is concatenated with its nearest unaligned sentences neighbors (according to the ordered list obtained in step 4, the all possible 1-, 2- or 3- sentence sequences). Then, those sentence sequences are compared with the target sentence (*j*). If one of those comparisons obtains a best BLEU score against the (*i*, *j*) score, the algorithm chooses a new n-to-1 alignment (instead of 1-to-1 alignment). Else, the process is performed, analogous, with the sentence of the target source text (*j*) against the sentence sequences (concatenations with sentences neighbors) of the translated source text. The latter determines the 1-to-n alignments.

6. As final resource, the algorithm tries to align the remaining sentences using the Gale and Church algorithm [9]. The input to this algorithm is the automatic translation of the source text and the target text because "this gives slightly better results, and should be more robust for unrelated language pairs, for which a length-based comparison is less suited" [26].

### 4.8 Iterative Bleualign

This sentence-aligner algorithm was created in 2011 by Rico Sennrich and Martin Volk [27] as a result of a deeper analysis of the disadvantage on using machine translating system in the process of alignments. Sennrich and Volk established that MT-based alignments strongly depends on the correct translation of the source text, and given that MT systems are generally fed with aligned texts, then it is evident the existence of a circular dependency.

In order to overcome this dependency, this algorithm presents a bootstrapping approach to do the alignment. The sentence alignment consists on the following steps:

1. Align parallel text.
   (a) First iteration: An implementation of any sentence alignment tool that does not require additional source is used. This work uses the Gale and Church algorithm [9].
   (b) Subsequent iteration:
      (i) The translation of the source text is made using the alignment of the previous step and a SMT system,
      (ii) Then, with the later translation, Bleualign is used for the alignment.
2. Train the SMT system on the sentences-aligned corpus

In SMT, it is common that the alignment algorithms produce several misaligned sentence pairs, however, it is not a big problem given that wrong phrases translation tends to be less probable than the corrects ones. Nevertheless, the latter is not true for an iterative approach where the training test is also the to-be-translated text. In order to overcome this problem, a pruning strategy was implemented. It consist on computing whether the occurrence frequency of phrase pairs in the SMT is statistically significant, or to be expected by chance

## 5 Discussion

As it was mentioned before, the algorithms presented through this work use statistical information to perform corpus alignment. During the next section, the main idea, the input and output parameters, and the assumptions of each algorithm, will be analyzed.

The Gale and Church algorithm and Brown algorithm were ones of the first corpus aligners that used statistical information. The simplistic idea was used both, as a platform for posterior algorithms or as an alternative to others. To assess their performance, Gale and Church algorithm uses a trilingual corpus of the Union Bank of Switzerland of economic results. Gale and Church reported the following results us-

ing a character sentence-length: "there was a 4.2% error rate on 1316 alignments, averaged over both English-French and English-German data". In addition, an alternative test was performed using a word sentence-length; however, the results were not good enough showing a 6.5% error rate against the 4.2% of the ones of character sentence-length. In regard to the initial condition and parameters, the algorithm requires some probabilistic values that were assigned based on Canadian Hansard bitexts but these values didn't change a lot on others kind of corpus, so they remain as constants. Moreover, the input data used in this algorithm requires a very specific structure, so it needs some pre-processing step on the parallel corpus.

On the other hand, Brown uses the Canadian Hansard corpus to evaluate its algorithm. Similar to Gale and Church, Brown algorithm have achieved remarkably good outputs for language pairs like English-French with error rates of 4% on an average. However, these algorithms are not robust with respect to non-literal translations and deletions; and they depend heavily on the delimiters (paragraphs or anchor points). Also, the algorithms present bad performance with languages that are not alike.

The Vanilla aligner is very similar to the Gale and Church aligner; in fact, it is used as a base with some adjustments. The main differences were the compatibility with a labeled or formatted text and the way the output data is presented. The former difference refers to the pre-processing step in which labels are removed, allowing to the algorithm have more parallel corpus to work with. The second refers to the way the algorithm presents the output information. The Gale and Church aligner creates two files with the alignment results while in Vanilla aligner one file is generated. According to [5] "this algorithm gets it right more than 95% of the time. When it does go wrong, it is usually when it tries to find a 0-1 alignment (or a 1-0) that should be 3-1 or 1-3, for example."

The basic idea of Gale and Church aligner works when the parallel corpus to be aligned are European languages, for example; English, Spanish French, among others. Nevertheless, it appears to be false when we want to align languages like English against Chinese, i.e., from different language familiy. Several important characteristics have been identified, for example, their alphabet, which makes the sentence-length be different; the sentence boundaries, like the linguistic boundaries and so on. All these reasons may lead to other researchers to create new aligner algorithms, for example, Moore's Association-based algorithm, K-vec and DK-vec algorithm.

Moore made two main contributions, the first was the development of faster algorithm, and second, he used word-association statistics for sentence alignment. In addition, Moore claims that, even though he cannot ensure that the word-association heuristics are better than the well-funded alignment approaches, this work give an insight on that word-association heuristics is still a good research opportunity.

The K-vec algorithm is a word level aligner that is not dependent to linguistic boundaries. It aims to align texts with different language ancestor, for instance, Japanese and Spanish. In fact, this aligner creates its own boundaries in the bitexts called segments. These K segments are the information vectors representing each word in the text, i.e., one vector per word. The corpus used to assess the performance of the algorithm was Canadian Hansards in order to have a previous comparison reference.

Finally, as the authors say, one of the principal contributions of this work is that "could be used as a starting point for more detailed alignment algorithm…" given that it generates "a quick-and-dirty estimate of a bilingual lexicon"[7].

Shortly afterwards, a new algorithm was developed, called DK-vec, which is based on the K-vec algorithm and its principal contribution is the addition of statistical information, as a new vector known as arrival vector, in order to improve the parallel corpus alignment. Unlike the K-vec algorithm, DK-vec divides the bitexts into K pieces and for each word, two vectors are computed, the vector position and the arrival vector, with a dimension k and k-1, respectively. These vectors are treated as signals and used to measure the signal similarity using dynamic time warping. As a result, it presents a better performance on corpus alignment of different root languages like Japanese against English corpora.

On 2010, the Bleualign does two main contributions: First, a new approach, known as MT-based sentence alignment, is created and consists of the use of an MT system to improve the sentence alignment. Second, Bleualign first uses the BLEU score as a similarity measure for sentence alignment. However, this new approach has a high dependency on the MT-System used for the translation. Problems such as dependency rise when the MT-System cannot translate the pair of languages resulting in worst performance (from 50% to 61% of accuracy) than Gale and Church algorithm (from 68% to 80% of accuracy) over the Text+Berg corpus. However, if it is used a reliable MT-System, the performance of this aligner is about 81% to 95% of accuracy in the same bitexts. Also, it has been shown that if we try to align a translation made by MT-System and the target text in Gale and Church aligner, the performance gets better from 68% - 80% to 72% - 83% of accuracy [26].

One year later, the same authors of the Bleualing algorithm created a new version of the algorithm. In this case, the authors point out that Bleualing is not 100% reliable with a pair of languages that does not have a good MT-System and they tried to solve it with this new algorithm. In this case, they use the same three elements of the previous algorithm; MT-System, BLEU-score and Gale and Church aligner. However, they change the order of these elements to solve this dependency problem. The authors did the experiments over the same bitexts (Text+Berg corpus). This new aligner had a performance of 69.5% to 94.4% accurate [27] and overcome the problem of use an MT system for sentence alignment problem, which is a basic tool of almost every MT system.

## 6 Conclusion

In this paper, some of the most representative statistical-based algorithms to parallel corpus alignment were described and a discussion about their results, advantages and disadvantages was presented.

There is not one statistical-based approach that works for all kinds of languages in the scope of parallel corpus alignment, i.e., some methods perform better when the languages to be alignment share a common ancestor, but others are more robust under this condition.

Even though these approaches have achieved remarkably good results, considering the poor resources used, there is much to improve. More recent works suggest using the lexical and statistical information to improve the performance of the parallel corpus alignment.

**Acknowledgments**. Work done under partial support of Mexican Government (CONACYT, SNI) and PRODEP (project 227835).

## References

1. Brown, P. F., Lai, J. C., & Mercer, R. L. (1991, June). Aligning sentences in parallel corpora. In *Proceedings of the 29th annual meeting on Association for Computational Linguistics* (pp. 169-176). Association for Computational Linguistics.
2. Brown, P. F., Cocke, J., Della Pietra, S., Della Pietra, V. J., Jelinek, F., Mercer, R. L., & Roossin, P. S. (1988, June). A Statistical Approach to French/English Translation. In *RIAO* (pp. 810-829).
3. Cendejas Castro, E.A. (2013). *Alineación automática de textos paralelos a nivel de palabras información lingüística diversa* (Ph.D. thesis). Centro de Investigación en Computación-IPN, México.
4. Chen, S. F. (1993, June). Aligning sentences in bilingual corpora using lexical information. In *Proceedings of the 31st annual meeting on Association for Computational Linguistics* (pp. 9-16). Association for Computational Linguistics.
5. Danielsson, P., & Ridings, D. (1997, February). Practical presentation of a "vanilla" aligner. In *TELRI Workshop in alignment and exploitation of texts*, February.
6. Félix, J. Á. V., & Sidorov, G. (2004). Proyecto de preparación del corpus paralelo alineado español-inglés. *Memorias del 5o encuentro internacional de computación ENC-2004*, Colima, México, 235-242.
7. Fung, P., & Church, K. W. (1994, August). K-vec: A new approach for aligning parallel texts. In *Proceedings of the 15th conference on Computational linguistics*-Volume 2 (pp. 1096-1102). Association for Computational Linguistics.
8. Fung, P., & McKeown, K. (1994). Aligning noisy parallel corpora across language groups: Word pair feature matching by dynamic time warping. *arXiv preprint cmp-lg/9409011*.
9. Gale, W. A., & Church, K. W. (1993). A program for aligning sentences in bilingual corpora. *Computational linguistics*, *19*(1), 75-102.
10. Gelbukh, A., Sidorov, G., & Vera-Félix, J. Á. (2006). A bilingual corpus of novels aligned at paragraph level. In *Advances in Natural Language Processing* (pp. 16-23). Springer Berlin Heidelberg.
11. Harris, B. (1988). Bi-text, a new concept in translation theory. *Language Monthly*, 54, 8-10.
12. Kay, M., & Röscheisen, M. (1993). Text-translation alignment. *Computational Linguistics*, 19(1), 121-142.
13. Kit, C., Webster, J. J., Sin, K. K., Pan, H., & Li, H. (2004). Clause alignment for Hong Kong legal texts: A lexical-based approach. *International Journal of Corpus Linguistics*, *9*(1), 29-51.
14. Langlais, P., Simard, M., & Véronis, J. (1998, August). Methods and practical issues in evaluating alignment techniques. In *Proceedings of the 36th Annual Meeting of the Asso-*

*ciation for Computational Linguistics and 17th International Conference on Computational Linguistics-Volume 1* (pp. 711-717). Association for Computational Linguistics
15. McEnery, A., & Xiao, R. Z. (2008). Paralell and comparable corpora: what are they up to?. *Incorporating Corpora: Translation and the Linguist. Translating Europe. Clevendon: Multilingual Matters*.
16. Macklovitch, E., & Hannan, M. L. (1998). Line 'em up: advances in alignment technology and their impact on translation support tools. *Machine Translation*, 13(1), 41-57.
17. Marín, F. M. (1993). La Biblioteca Electrónica en el Archivo Digital de Manuscritos y Textos Españoles. *Lexis: Revista de lingüística y literatura*, *17*(1), 33-56.
18. Melamed, I. D. (2000). Models of translational equivalence among words. *Computational Linguistics*, *26*(2), 221-249.
19. Meyers, A., Kosaka, M., & Grishman, R. (1998, October). A multilingual procedure for dictionary-based sentence alignment. In *Conference of the Association for Machine Translation in the Americas* (pp. 187-198). Springer Berlin Heidelberg.
20. Moore, R. C. (2005, June). Association-based bilingual word alignment. In *Proceedings of the ACL Workshop on Building and Using Parallel Texts* (pp. 1-8). Association for Computational Linguistics.
21. Moro, A., & Navigli, R. (2015). SemEval-2015 task 13: multilingual all-words sense disambiguation and entity linking. *Proc. of SemEval-2015*.
22. Nagao, M. (1984). A framework of a mechanical translation between Japanese and English by analogy principle. *Artificial and human intelligence*, 351-354.
23. Papineni, K., Roukos, S., Ward, T., & Zhu, W. J. (2002, July). BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics* (pp. 311-318). Association for Computational Linguistics.
24. Piao, S. (2000). *Sentence and Word Alignment between Chinese and English* (Ph.D. thesis). Lancaster University, Lancaster.
25. Pierce, J. R., & Carroll, J. B. (1966). *Language and machines: Computers in translation and linguistics*.
26. Sennrich, R., & Volk, M. (2010, November). MT-based sentence alignment for OCR-generated parallel texts. In *The Ninth Conference of the Association for Machine Translation in the Americas (AMTA 2010)*, Denver, Colorado.
27. Sennrich, R., & Volk, M. (2011, May). Iterative, MT-based sentence alignment of parallel texts. In *18th Nordic Conference of Computational Linguistics, NODALIDA*
28. Simões, A. (2004). *Parallel corpora word alignment and applications* (master thesis). Universidade do Minho, Braga.
29. Way, A., & Hearne, M. (2011). On the Role of Translations in State-of-the-Art Statistical Machine Translation. *Language and Linguistics Compass*, 5(5), 227-248.
30. Weaver, W. (1955). Foreword: the new tower. In W.N. Locke and A.D. Booth (Editors), *Machine Translation of Languages: Fourteen Essays*, Cambridge, MA.