

# **Cinema at the service of Natural Language Processing**

**Luís Carlos Cachapela Rosado**

Thesis to obtain the Master of Science Degree in

## **Information Systems and Computer Engineering**

Supervisors: Prof. Maria Luísa Torres Ribeiro Marques da Silva Coheur  
Prof. David Manuel Martins de Matos

### **Examination Committee**

Chairperson: Prof. José Carlos Alves Pereira Monteiro  
Supervisor: Prof. Maria Luísa Torres Ribeiro Marques da Silva Coheur  
Member of the Committee: Prof. Sara Alexandra Cordeiro Madeira

**May 2016**







# Acknowledgments

Ao longo do desenvolvimento deste trabalho pude contar com a preciosa ajuda e apoio de várias pessoas. Sem estas pessoas este caminho teria sido, certamente, mais difícil.

Agradeço aos meus orientadores, Prof. Luísa Coheur e Prof. David Matos, por toda a ajuda dada. Agradeço, não só, os esclarecimentos de dúvidas e feedback dado, como também o incentivo para que desse o meu melhor na realização deste projecto. As reuniões realizadas foram marcadas por momentos de boa disposição e a partilha de ideias essenciais ao desenvolvimento do projecto.

Agradeço também a todos os colegas de curso que, sempre que necessário, me ajudaram com opiniões ou sugestões sobre o trabalho a desenvolver. Em especial, quero agradecer a um grupo de colegas que sempre me motivou a fazer um bom trabalho: André, Cátia, Hugo, Iryna, Rita e Vasco.

Além destes colegas, quero agradecer a todos os que me ajudaram, participando na avaliação do projecto desenvolvido. Obrigado aos 36 anotadores que perderam o seu tempo para me ajudar.

Resta-me agradecer o apoio incondicional dos meus familiares, sem o qual teria sido difícil a realização deste projecto.







## **Abstract**

In the Natural Language Processing area, many systems require the existence of training data in order to improve their results. Typically, increasing the training data amount in a Natural Language Processing system leads to a great improvement on system performance. But, sometimes it can be difficult to find adequate corpora for training purposes.

Movie subtitles are a very important resource that is available for free and in large amounts for almost every language and can be used to obtain parallel corpora. Using different movie subtitle files for the same movie, we can extract useful information. In this work, we explore the potential of movie subtitles to produce aligned parallel corpora and extract information from them. This is done by correctly aligning the subtitle files, through the combination of state-of-the-art techniques that use not only the timing information present in subtitle files, but also the textual similarity between sentences. This represents a way of producing aligned parallel corpora that can be used in Machine Translation systems, due to the characteristics of movie dialogs. Besides the subtitle aligner, we have contributed with the creation of a subtitle dataset and reference alignments that can be used to evaluate any subtitle aligner. Using the created dataset and reference alignments, we have observed that the developed subtitle aligner successfully improves state-of-the-art results.

**Keywords:** Movie subtitles; Subtitle alignment; Information extraction; Building parallel corpora; Training data; Machine Translation; Reference alignments







## **Abstract**

Na área do Processamento de Língua Natural muitos sistemas requerem a existência de dados de treino para que possam melhorar os seus resultados. Tipicamente, aumentar a quantidade de dados de treino num sistema de Processamento de Língua Natural leva a uma grande melhoria dos resultados do sistema. Mas, por vezes, pode ser difícil encontrar corpora adequados a treinos.

As legendas são um recurso de muita importância, pois é gratuito e existe em grandes quantidades, para quase todas as linguagens e pode ser utilizado para obter corpora paralelos. Utilizando diferentes legendas para um mesmo filme, podemos extrair informação muito útil. Neste trabalho, exploramos o potencial das legendas de filmes para produzir corpora paralelos alinhados e, assim, extrair informação deles. Isto é feito alinhando correctamente as legendas, através da combinação de técnicas do estado-da-arte que utilizam, não só a informação temporal contida nos ficheiros de legenda, como também a semelhança textual entre frases. Isto representa uma forma de produzir corpora paralelos alinhados que podem ser utilizados em sistemas de tradução automática, devido às características dos diálogos de filmes. Além do alinhador de legendas, contribuímos com a criação de um dataset de legendas de filmes e de alinhamentos de referência que podem ser utilizados para avaliar qualquer alinhador de legendas. Utilizando este dataset e alinhamentos de referência, verificámos que o alinhador de legendas desenvolvido melhora resultados do estado-da-arte.

**Palavras-Chave:** Legendas de filmes; Alinhamento de legendas; Extração de informação; Construção de corpora paralelos; Dados de treino; Tradução automática; Alinhamentos de referência







# Contents

<b>List of Tables</b>	<b>xi</b>
<b>List of Figures</b>	<b>xiii</b>
<b>Acronyms</b>	<b>1</b>
<b>1 Introduction</b>	<b>3</b>
1.1 Objectives . . . . .	3
1.2 Characterization of the problem . . . . .	4
1.2.1 Lack of synchronization . . . . .	4
1.2.2 Scene description insertion . . . . .	4
1.2.3 Segmentation . . . . .	5
1.2.4 Subtitle omission and insertion . . . . .	6
1.3 Contributions . . . . .	6
1.4 Document outline . . . . .	6
<b>2 Related Work</b>	<b>9</b>
2.1 Pre-processing . . . . .	9
2.2 Subtitle alignment based on similarity . . . . .	10
2.3 Subtitle alignment based on timing information . . . . .	15
<b>3 Alignerzilla</b>	<b>21</b>
3.1 Architecture overview . . . . .	21
3.2 Pre-processing . . . . .	22
3.2.1 Encoding . . . . .	23
3.2.2 Normalization . . . . .	23
3.2.3 Sentence Concatenation . . . . .	24
3.2.4 Sentence Splitting . . . . .	24
3.3 Aligner . . . . .	25
3.3.1 Time based alignment . . . . .	26
3.3.2 Similarity based alignment . . . . .	27
3.3.3 Translator . . . . .	28
3.3.4 Configurations file . . . . .	28
<b>4 Building reference alignments</b>	<b>29</b>
4.1 Dataset . . . . .	29
4.2 Reference creation . . . . .	29
4.2.1 Web Application . . . . .	29
4.2.2 Alignments validation . . . . .	30



<b>5</b>	<b>Evaluation</b>	<b>33</b>
5.1	Results . . . . .	34
5.1.1	Correlation with sentence number differences . . . . .	35
5.1.2	Correlation with time offsets . . . . .	36
5.2	Comparison with aligner using time only . . . . .	37
5.3	Comparison with state-of-the-art aligner . . . . .	40
5.4	Summary . . . . .	42
<b>6</b>	<b>Conclusions</b>	<b>43</b>
6.1	Future Work . . . . .	44
	<b>Bibliography</b>	<b>46</b>
<b>A</b>	<b>Run instructions</b>	<b>49</b>
<b>B</b>	<b>Configurations file</b>	<b>51</b>
<b>C</b>	<b>Web Application</b>	<b>53</b>











# List of Tables

1.1	<i>Lack of synchronization</i> example taken from <i>Person of Interest S04E09</i> . After the intro, dialog timestamps do not overlap. . . . .	5
1.2	<i>Scene description insertion</i> example taken from <i>The Prestige</i> movie. The segment in bold represents information that is omitted in the English subtitle but included in the Portuguese subtitles. . . . .	5
1.3	<i>Segmentation</i> example taken from <i>The Prestige</i> movie. . . . .	5
1.4	<i>Subtitle omission and insertion</i> taken from <i>The Prestige</i> movie. The segment in bold is not present in both subtitle files. . . . .	6
2.1	Illustration of mismatching due to functional words [3]. . . . .	12
2.2	The F-score of subtitle alignment using different metrics [2]. . . . .	14
2.3	Adjusted priors for various alignment types (with probability $> 0.001$ ) [4]. . . . .	15
2.4	Scores of different alignment approaches [4]. . . . .	17
3.1	<i>Sentence concatenation</i> example taken from <i>A Beautiful Mind</i> movie. . . . .	24
3.2	<i>Sentence splitting</i> example taken from <i>A Beautiful Mind</i> movie. . . . .	25
4.1	List of the movie subtitles included in the dataset and the differences between the number of sentences in their <i>source</i> and <i>target</i> subtitles. . . . .	32
5.1	Results obtain after aligning the dataset subtitles with <i>Alignerzilla</i> . . . . .	34
5.2	Correlation between the sentence number difference percentage and F-measure. . . . .	35
5.3	Correlation between the offset times and F-measure. . . . .	36
5.4	Results obtained with Time Only Aligner and comparison with <i>Alignerzilla</i> . . . . .	38
5.5	Results obtained with Tiedemann aligner [4] and comparison with <i>Alignerzilla</i> . . . . .	41







# List of Figures

2.1	Representation of a subtitle screen in XML [4]. . . . .	10
2.2	Matching between $e$ and $f$ , where <i>English</i> is the <i>source</i> and <i>French</i> is the <i>target</i> language [3]. . . . .	12
2.3	Comparison between the F-measure and RFDM results [2]. The axis contain the subtitle segment numbers. . . . .	14
3.1	High-level view of the developed solution workflow. . . . .	21
3.2	UML representation of the problem domain. . . . .	22
3.3	Representation of the different classes that compose a Match. Example taken from <i>A Beautiful Mind</i> movie. . . . .	22
3.4	Overview of the alignment process for each iteration. . . . .	26
4.1	Screenshot of developed the web application where annotators aligned subtitles for reference. . . . .	30
5.1	Graphical representation of the F-measure calculated for each subtitle pair aligned using the time based approach and comparison with <i>Alignerzilla</i> . . . . .	37
5.2	Graphical representation of the results obtained with the aligner only using the time based approach and comparison with <i>Alignerzilla</i> . . . . .	37
5.3	Graphical representation of the F-measure calculated for each subtitle pair aligned using Tidemann [4] and comparison with <i>Alignerzilla</i> . . . . .	40
5.4	Graphical representation of the results obtained with the aligner only using Tidemann [4] and comparison with <i>Alignerzilla</i> . . . . .	40
C.1	Screenshot of the instructions and examples page of the developed web application. . . .	53
C.2	Screenshot of the page where annotators selected alignments between sentences. . . . .	54
C.3	Screenshot of an example of a 1:1 alignment selection. . . . .	55
C.4	Screenshot of an example of a 2:1 alignment selection. . . . .	56







# Acronyms

**BOW** bag of words. 13

**DTW** Dynamic Time Warping. 10, 13–15

**FN** False Negative. 33, 34, 38

**FP** False Positive. 33, 34

**NLP** Natural Language Processing. 3, 4, 9, 43, 44

**pp** percentage points. 38, 41

**RFDM** Relative-Frequency based Distance Metric. 13, 14

**SMT** Statistical Machine Translation. 3, 45

**TN** True Negative. 33, 34

**TOA** Time Only Aligner. 37, 38

**TP** True Positive. 33, 34

**XML** Extensible Markup Language. 9, 10







# Chapter 1

## Introduction

In the Natural Language Processing area, many systems require the existence of training data in order to improve their results. Typically, increasing the training data amount in a Natural Language Processing system leads to a great improvement on system performance [1]. But, sometimes it can be difficult to find adequate corpora for training purposes.

Movie subtitles are a very important resource that is available for free and in large amounts for almost every language and can be used to obtain parallel corpora. As each movie has a large number of subtitles associated and those various subtitles are usually created by different people, this is a potentially useful source of parallel corpora – different corpora that contain the same information, whether it is in a different language, whether it is written using different words. Natural Language Processing (NLP) systems have been using movie subtitles for quite a long time due to their potential [2]. There have been authors that studied how parallel corpora could be used to improve NLP systems [1]. This is widely used, for example, in Statistical Machine Translation (SMT) systems. In this kind of systems, typically, the more data is used to estimate the parameters of the translation model, the better the translation [1]. For instance, since the 90s, corpora alignment has been a topic of research due to its usefulness in training automatic machine translation systems [2] and it is considered, by many, an indispensable resource. While some authors researched methods to create parallel corpora that could be used to train NLP systems, others have studied how to make a better use of the available corpora (see Lü et al. [1] for more details).

### 1.1 Objectives

The main goal of this document is the creation of parallel corpora using movie subtitles. To accomplish this, it is necessary to develop a subtitle aligner. Furthermore, we also address the creation of a movie subtitle dataset and reference alignments that can be used to evaluate subtitle aligners.



## 1.2 Characterization of the problem

In order to correctly use movie subtitles as a source of parallel corpora, one needs to know how subtitle files are composed. The most common subtitle format is SubRip (.srt). A SubRip subtitle is composed by several *subtitle screens*. For each *subtitle screen*, it contains a block of plain text in the following format:

1	A unique identifier
00:00:57,057 --> 00:00:58,888	Start and end timestamps
Are you watching closely?	One or two lines containing the subtitle text
	Empty line indicating the end of the segment

During this document we will use “subtitle screen” to refer a single subtitle dialog and “subtitle” when referring the entire subtitle file.

Despite the huge potential of movie subtitles for NLP systems, multiple problems arise when trying to extract information from them. As different subtitles for the same movie are usually created by different people, it is not simple to align them due to their variations. Some subtitles may include information that others do not. While some subtitle creators tend to do exact transcriptions of the movie dialogs, others prefer to include less details. These and other variations in the subtitles make the alignment process complex. Some of the problems faced when working with subtitles are presented below.

### 1.2.1 Lack of synchronization

After knowing how subtitle files are constituted, we might think that aligning the subtitles will be an easy task due to the timestamps contained in each dialog. However, this task is not always that simple, because while some subtitles might be synchronized, most are not. Many subtitles for the same movie or series are created for slightly different video files – different cuts or inclusion of intros. This is a problem because the subtitles can be synchronized until a certain point and then be completely wrong, due to the delays caused by these differences in the video file. Hence, the timestamps present on subtitles might be useful, but cannot be used as the only criteria and must be used carefully. For example, in *Person of Interest S04E09* some subtitles files become desynchronized after the appearance of the series intro at minute 8:04. As we can see in Table 1.1, the dialogs were synchronized before the intro and are not afterwards.

### 1.2.2 Scene description insertion

The information in each subtitle might not be the same. For instance, some subtitles are made for hearing-impaired people, thus containing additional information that other subtitles do not. For instance, tags such as *[Laughing]* and *[Applause]* appear in a subtitle file for *The Prestige* movie.

Moreover, as subtitles are written by different people, some authors might include information that others decided not to transcribe. Also, if any informative text is displayed on the screen (e.g. signs), it may or may not appear in the subtitles. For example, if a *wet floor sign* appears in an English movie, it is not required to include it in English subtitles, whereas it might need translation in different language subtitles. An example can be seen in Table 1.2.



141 00:07:56,168 --> 00:07:57,501 If you know me at all, John,	113 00:07:56,401 --> 00:07:59,800 Se me conhece, John, sabe que há sempre outra saída.
142 00:07:57,503 --> 00:08:00,003 you know there's always another way out.	
INTRO OCURRS	
144 <b>00:08:18,712</b> --> 00:08:20,213 I got it from here, thanks.	116 <b>00:08:15,800</b> --> <b>00:08:18,400</b> - Assumo a partir daqui. - Precisa tirá-lo da cidade.
145 00:08:20,215 --> <b>00:08:21,214</b> You'll need help getting him out of the city.	

Table 1.1: *Lack of synchronization* example taken from *Person of Interest S04E09*. After the intro, dialog timestamps do not overlap.

113 00:08:17,196 --> 00:08:19,528 Well, my passion is equal to the task.	114 00:08:17,300 --> 00:08:19,700 Bem, o tamanho da minha paixão é igual ao da minha tarefa.
	<b>115</b> <b>00:08:24,100</b> --> <b>00:08:28,000</b> <b>"COLORADO SPRINGS"</b>
114 00:08:32,612 --> 00:08:36,139 Mr. Angier, welcome to Colorado Springs.	116 00:08:32,800 --> 00:08:36,300 Sr. Angier, bem-vindo ao Colorado Springs.

Table 1.2: *Scene description insertion* example taken from *The Prestige* movie. The segment in bold represents information that is omitted in the English subtitle but included in the Portuguese subtitles.

### 1.2.3 Segmentation

Alternative subtitles for the same movie may have different sentence breaks. Also, what is said in a few words in a language might need more words in another, thus requiring the use of more subtitle screens to say the same thing. Table 1.3 shows an example.

310 00:23:46,925 --> 00:23:48,126 Are either of you gentlemen sailors?	311 00:23:47,135 --> 00:23:49,721 - São ambos marinheiros? - Não.
311 00:23:48,214 --> 00:23:49,912 No.	

Table 1.3: *Segmentation* example taken from *The Prestige* movie.



### 1.2.4 Subtitle omission and insertion

Some subtitles do not transcribe exactly what has been said in the movie. Subtitles in the original movie language tend to be very close to the movie speech, but even when the language is the same, there are some speech parts that are not always included in the subtitles. Because of this, different subtitles usually have small differences in their texts, which makes it harder to correctly align them. Table 1.4 shows an example of this issue.

125 00:09:31,837 --> 00:09:33,134 That's why I'm here.	127 00:09:32,447 --> 00:09:33,740 É por isso que aqui estou.
<b>126</b> <b>00:09:40,379 --&gt; 00:09:42,609</b> <b>Whoa. Whoa.</b>	
127 00:09:44,317 --> 00:09:47,252 You'll have to walk the rest, I'm afraid, sir.	128 00:09:44,918 --> 00:09:47,838 Temo que terá de caminhar o resto do caminho, senhor.

Table 1.4: *Subtitle omission and insertion* taken from *The Prestige* movie. The segment in bold is not present in both subtitle files.

The previous variations found in subtitles represent serious problems when trying to extract informations and all of them must be dealt with in order to have decent results.

Furthermore, the process of aligning subtitle files has problems that also occur when aligning regular corpora. The alignment task establishes relations between segments of the corpus pairs. But, these relations are not always one-to-one correspondences. Thus, different types of alignments can be found in the alignment process. If one source segment matches with one target segment, it is called an 1:1 alignment. If one source segment has no correspondence in the target corpus, it is called 1:0 alignment. Similarly, if the target corpus has a segment that is not present in the source, it is a 0:1 alignment. The same logic is applied for segments that match with more than one segment in the other corpus, producing 1:N, N:1, or even N:M alignments.

## 1.3 Contributions

In order to achieve the aforementioned objectives, we have studied the different approaches explored by multiple authors and we have built an aligner capable of correctly aligning movie subtitles. As we will describe in Chapter 2, different authors have used different techniques to build subtitle aligners. By combining the existing approaches we have developed an aligner that improves state-of-the-art results, as we will see in Chapter 5. Besides the creation of a movie subtitle aligner, a movie subtitle dataset has been built and reference alignments have been created for each subtitle pair. With the created reference alignments, we have made it possible to evaluate any subtitle aligner.

All the developed work was made available in the Internet.

## 1.4 Document outline

This document is structured as follows. Chapter 2 describes how other authors have dealt with the aforementioned problems using different techniques for subtitle alignment. This section is divided in



three subsections: pre-processing, subtitle alignment based on similarity of segments and subtitle alignment based on timing information. Chapter 3 presents the approach taken to find our solution. Chapter 4 describes how we have created a dataset and reference to evaluate the developed aligner. Chapter 5 presents the evaluation methodology that will be used to infer the work results. Chapter 6 presents some final remarks.







# Chapter 2

## Related Work

The vast number of subtitles available, in multiple languages, makes them a powerful NLP resource. Due to movie subtitles multiple applications, it is of great importance to build systems that can extract information and use it properly. In the last years, several researchers developed systems to extract information from movie subtitles with different purposes.

The following subsections describe the solutions proposed by some authors to extract information from movie subtitles while dealing with the problems presented in the previous section.

### 2.1 Pre-processing

Movie subtitles cannot be used as a resource in their raw state, due to their variations. Thus, in order to use them, there is the need of doing some pre-processing of the subtitle files [3, 4, 2].

In order to solve the aforementioned *scene description insertion* problem, Lavecchia et al. [3] removed all the descriptions they could identify in the subtitles. They considered that these descriptions usually appear inside square brackets, between hashes, or in uppercase. As most of these descriptions are easily recognizable in the subtitle files, they removed them before going any further. Although this does not make the subtitles immediately alignable, it is a small step that helps in the alignment process.

To normalize the dataset, Tsiartas et al. [2] decided to clean subtitles from noisy symbols. They used an approach similar to what is usually done when normalizing text for statistical machine translation purposes, removing any non-alphanumeric symbols. Also, they removed all the timestamps and identifiers of each subtitle screen. In the sentence level, they removed all the punctuation and capitalized every character.

Tiedemann [4] went even further and completely changed the dataset format. First, he started by converting all the subtitle files to a uniform format. He decided to support two types of subtitles: *SubRip* and *microDVD* (.srt and .sub, respectively). If the subtitle files were not in .srt format, he used a tool to convert them (*sub2srt*<sup>1</sup>). Then, as different languages could use different text encodings, he decided to use a standalone Extensible Markup Language (XML) format using the UTF-8 encoding. This XML format consists of marking sentence boundaries and tokenizing its content, producing blocks of XML for each sentence tagged with their timestamps and identifiers. Table 2.1 shows a subtitle screen in the XML format used by Tiedemann.

Using regular expressions, Tiedemann tokenized and marked with sentence boundaries each subtitle file. Because this normalization was done automatically, it was ineffective for some subtitle files. In languages that sentence boundaries did not match with the patterns defined by the author, it produced

---

<sup>1</sup><https://github.com/robelix/sub2srt>



```

<s id="1">
  <time id="T1S" value="00:00:26,500"/>
  <w id="1.1">Spend</w>
  <w id="1.2">all</w>
  <w id="1.3">day</w>
  <w id="1.4">with</w>
  <w id="1.5">us</w>
  <w id="1.6">.</w>
  <time id="T1E" value="00:00:28,434"/>
</s>

```

Figure 2.1: Representation of a subtitle screen in XML [4].

errors in the XML file because no manual corrections were done. The author recognized that these errors in the preprocessing caused some of the mistakes in the alignment phase and that improving the sentence splitting method should greatly improve the final results.

Tiedemann also identified incorrect subtitle files as another source of errors. Some subtitles in his dataset contained corrupt character encodings or were tagged with the wrong language. In order to avoid this kind of errors, he decided to use a language classifier to check the contents of all subtitles. The tool used is available for free and it was designed for language identification (TextCat<sup>2</sup>). The language checker was used, by the author, after the conversion of all the input files to UTF-8 encoding, using the Unix tool `recode`. The classifier predicted the most likely language for each subtitle file and Tiedemann only used those that TextCat is certain of their language, discarding all other files.

## 2.2 Subtitle alignment based on similarity

The most complex and interesting part of this project is the alignment of the subtitle files, due to the problems described in the previous section. For some years, several authors have been aligning parallel corpora, using different methods. In this section, we describe some of the methods based on the segments similarity used by different authors to align subtitle files.

According to Lavecchia et al. [3] most of the work done in parallel corpora alignment is based on dynamic programming. The authors state that subtitle alignment can be seen as a typical problem of dynamic programming if one breaks the subtitle in smaller fragments, e.g. paragraphs, sentences or phrases.

By evaluating the closeness between segments and building a path of matches between them, it is possible to infer the correct alignment for two subtitle files. To calculate the best path between two subtitle files, the authors use a Dynamic Time Warping (DTW) algorithm based on F-measure, i.e., that uses the F-measure between *source* and *target* subtitle segments as cost function. If none or few segments on the source match with the target, it is said that its F-measure is weak and, therefore, subtitles are considered not aligned.

Considering that each segment of *source* and *target* subtitles are represented by  $s$  and  $t$  respectively, in order to have an alignment between subtitle files, we want a path that finds a  $t$  match for each  $s$ . A correct path aligns every segment in the source subtitle with a segment in the target subtitle, beginning at node  $(0, 0)$  and finishing at node  $(S, T)$  – where  $S$  and  $T$  are the last segments of each subtitle file. Nevertheless, as we have seen previously, *segmentation* is one of the variations frequently found in the subtitles and it would be difficult to find subtitle files where every match is one-to-one – each source segment matches exactly one target segment. Some segments in the source subtitle might match more

<sup>2</sup><http://odur.let.rug.nl/~vannoord/TextCat/>



than one segment in the target. Hence, Lavecchia et al. consider that, from a node, there are three shifts possible:

**Vertical** from  $(s, t)$  to  $(s, t + 1)$

the segment  $s$  matches with two consecutive segments  $t$ ;

**Diagonal** from  $(s, t)$  to  $(s + 1, t + 1)$

the segment  $s$  matches with the segment  $t$ ;

**Horizontal** from  $(s, t)$  to  $(s + 1, t)$

the segment  $t$  matches with two consecutive segments  $s$ ;

A matching score based on the F-measure ( $F_M$ ) is calculated with equation 2.1 for each node  $(e, f)$ .

$$S(s, t) = \max \begin{cases} S(s, t - 1) & + \beta_{F_M}(F_M(s, t) + \epsilon) \\ S(s - 1, t - 1) & + \alpha_{F_M}(F_M(s, t) + \epsilon) \\ S(s - 1, t) & + \lambda_{F_M}(F_M(s, t) + \epsilon) \end{cases} \quad (2.1)$$

Where  $\alpha_{F_M}$ ,  $\beta_{F_M}$  and  $\lambda_{F_M}$  are variable parameters chosen in order to find out the best alignment. In the previous formula, the F-measure is smoothed with an  $\epsilon$  value in order to prevent null values. Equations 2.2, 2.3 and 2.4 explain how the  $F_M$  is calculated for each node.

$$F_M(s, t) = 2 \times \frac{R(s, t) \times P(s, t)}{R(s, t) + P(s, t)} \quad (2.2)$$

$$R(s, t) = \frac{\text{match}(s, tr(t))}{N(s)} \quad P(s, t) = \frac{\text{match}(s, tr(t))}{N(t)} \quad (2.3)$$

$$\text{match}(s, tr(t)) = \sum_{i=1}^n \delta(s_i, tr(t_j)) \forall j \quad (2.4)$$

$tr(t)$  represents the translation of the target language to the source language, using a dictionary.  $N(x)$  is the number of words in the subtitle segment  $x$ .  $\text{match}(s, tr(t))$  is the number of words that match between  $s$  and  $tr(t)$ . The Kronecker  $\delta(x, y)$  is a function that returns 1 if  $x$  and  $y$  are equal and 0 otherwise.

As an example of how those values are calculated, see Figure 2.2.

In order to identify proper names, the *match* function has been improved and sees if two words match before doing the translation.

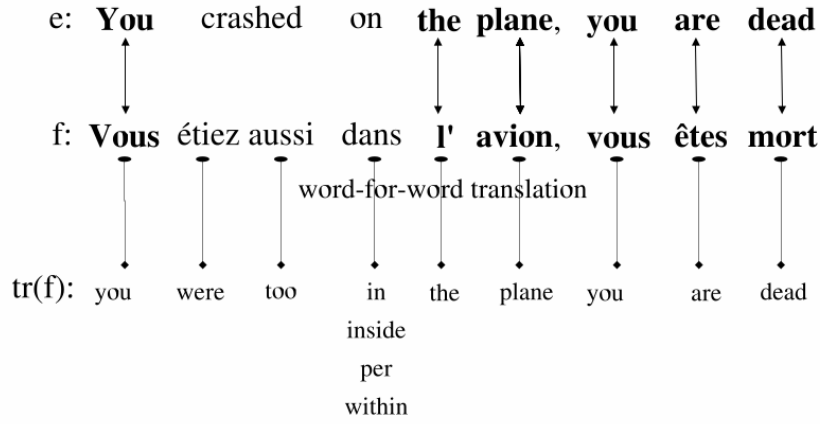
Following this method, Lavecchia et al. conducted a series of tests using a corpus extracted from 40 movies, where they manually aligned around 35 segments from each movie. They obtained 1364 (#A) pairs of English ( $s$ ) and French( $t$ ) aligned subtitle segments as reference.

After aligning the English and French corpora, they removed the alignments where the F-measure was zero. Considering #Tot the number of retrieved pairs and #C the number of correct alignments, equations in 2.5 are obtained.

$$\text{Precision} = \frac{\#C}{\#Tot} \quad \text{Recall} = \frac{\#C}{\#A} \quad (2.5)$$

The tests showed that  $\alpha_{F_M}$  parameter had a strong influence on the results. They noticed that  $F_M$  increases with  $\alpha_{F_M}$ , but only until a certain value. Given this, they decided to set  $\alpha_{F_M}$  to 9 in the following experiments. They realized that the number of aligned subtitle pairs was maximized when the precision was bigger, rather than F-measure. As the precision was increased, the number of incorrect alignments (False Positives) was decreased.





**match[e, tr(f)] = 6**  
**Precision(e, f) = 6/9**  
**Recall(e, f) = 6/8**  
**Fmeasure(e, f) = 0.70**

Figure 2.2: Matching between  $e$  and  $f$ , where *English* is the *source* and *French* is the *target* language [3].

After the alignment was done, only 94 pairs were incorrectly aligned. By analyzing the mismatched pairs, they concluded that the amount of functional words (such as prepositions, conjunctions) would incorrectly increase the F-measure, making F-measure positive even when the sentences did not match.

	E1 : Wallis <b>hold</b> on to this F1 : Wallace <b>tiens</b> moi cela	E1 : Wallis hold on <b>to</b> this F2 : Ulrich pense <b>à</b>
N(e)	5	5
N(f)	4	3
match	1	1
Precision	1/4	1/3
Recall	1/5	1/5
$F_M$	<b>0.22</b>	<b>0.23</b>

Table 2.1: Illustration of mismatching due to functional words [3].

As we can see in Table 2.1, in short sentences, the match of functional words increased the  $F_M$  value, matching the source sentence with the wrong one. Manually, we can see that the right alignment would be (E1, F2), but the F-measure value refutes this fact. The authors concluded that it would be wrong to give such influence to these small words and decided to correct the formula 2.4 in order to change the weight of functional words. The new match function is calculated as equation 2.6 shows.

$$match(s, tr(t)) = \sum_{i=1}^n \gamma \times \delta(s_i, tr(t_j)) \forall j \quad (2.6)$$

In equation 2.6,  $\gamma$  is smaller than 1 when  $s_i$  or  $t_j$  are functional words and equal to 1 otherwise. However, changing the weight of functional words did not improve the results. As the subtitles are usually formed by small segments, reducing the weight of functional words made the values of F-measure, Recall and Precision fall. When the  $\gamma$  value was set to 0, the number of aligned subtitles greatly decreased, because some pairs only matched due to the functional words.



Tsiartas et al. [2] also consider that the alignment can be seen as a dynamic programming problem where we want to find the best mapping between utterances in two languages, so that it reflects the best translations. But, instead of using a DTW algorithm based on F-measure that varies with a parameter depending on the path direction, like Lavecchia et al. [3], they decided to use a different approach to find the best mapping between subtitles. Since the authors consider that the results of solution [3] are influenced by the weight of the parameters  $(\alpha, \beta, \lambda)$ , high weights could neutralize the importance of the word matching metric. According to Tsiartas et al., such a strategy might not be the most adequate. Thus, the authors decided to approach this as a minimization problem, based on a distance metric. In order to calculate the best mapping between two subtitles, they computed its global distance ( $D$ ), by calculating the minimum distance between each of its utterances (equation 2.7).

$$D(S, T) = \sum_{i,j} m_{ij} DM(s_i, t_j) \quad (2.7)$$

In equation 2.7,  $DM(s_i, t_j)$  is the distance metric between  $s_i$  and  $t_j$  and  $m_{ij}$  is the mapping between two segments: if  $s_i$  matches  $t_i$ , then  $m_{ij} = 1$ , otherwise  $m_{ij} = 0$ .

The authors proposed a Relative-Frequency based Distance Metric (RFDM) to calculate the dissimilarity between two utterances. Similar subtitles have lower RFDM values. By using this metric, the distance between subtitles is easily minimized. In order to compute the RFDM, a bilingual dictionary is used to translate the segments. Using the dictionary, they obtain the translations for each word in the subtitle  $T$ , getting a bag of words (BOW) in the *source* language, which corresponds to each of the words in the *target* segment. Words that appear both in the BOW and in the *source* segment are counted and used to calculate the RFDM. Considering  $w_t$  a word in the *target* subtitle,  $tr(w_t)$  is the set of words translated using the dictionary. If the dictionary does not have a translation for a word, it produces an empty set.

Considering that the segments  $s$  and  $t$ , contain  $n$  and  $m$  words, respectively, they can be defined as the sets in equation 2.8.

$$s = \{w_{s_1}, w_{s_2}, \dots, w_{s_n}\} \quad t = \{w_{t_1}, w_{t_2}, \dots, w_{t_m}\} \quad (2.8)$$

The bag of translated words from the *target* language is a set of the translations and it is defined as in equation 2.9.

$$BOW(t) = \{tr(w_{t_1}), tr(w_{t_2}), \dots, tr(w_{t_m})\} \quad (2.9)$$

Supposing that there are  $N_w$  unique words in the BOW produced by a sequence and the word  $w_{t_k}$  appears  $C_{w_k}$  times in the collection  $k = 1, \dots, N_w$ , the RFDM is computed as equation 2.10.

$$DM(s, t) = \left( \sum_{k=1}^n \frac{I_{w_{s_k}}}{C_{w_{s_k}}} \right)^{-1} \quad (2.10)$$

In equation 2.10,

$$I_{w_{s_k}} = \begin{cases} 1 & \text{if } w_{s_k} \text{ is in both } s \text{ and } BOW(t) \\ 0 & \text{otherwise} \end{cases}$$

$I_w$  only has a value if the word  $w$  exists in the *source* segment and the translated words form the *target* language. Functional words, like *the* or *is* are very frequent and hence, their counts tend to be high. Due to the use of the inverse in equation 2.10, frequent words have a lower contribution to the global distance computation – because they produce a high distance metric and the goal is to minimize the distance. On the other hand, less frequent words have a smaller  $C_w$  and reduce the global distance value.

In order to use the DTW algorithm for solving the alignment problem, Tsiartas et al. [2] also use



back-tracking variables so that they can produce the best path correctly.

Tsiartas et al. [2] compared their results with the DTW based on F-measure (Lavecchia et al. [3] approach) and inverted the F-measure and used it as Distance Metric in equation 2.7 to obtain the best alignments (called it IF-measure). Table 2.2 shows that the RFDM metric gives 8-11% absolute improvements.

Metric	F-score
F-measure	0.609
IF-measure	0.681
RFDM	0.711

Table 2.2: The F-score of subtitle alignment using different metrics [2].

They also concluded that the path chosen by the maximization approach using the F-measure results in a path that is not close to the reference alignment (manually done) as the RFDM is (Figure 2.3).

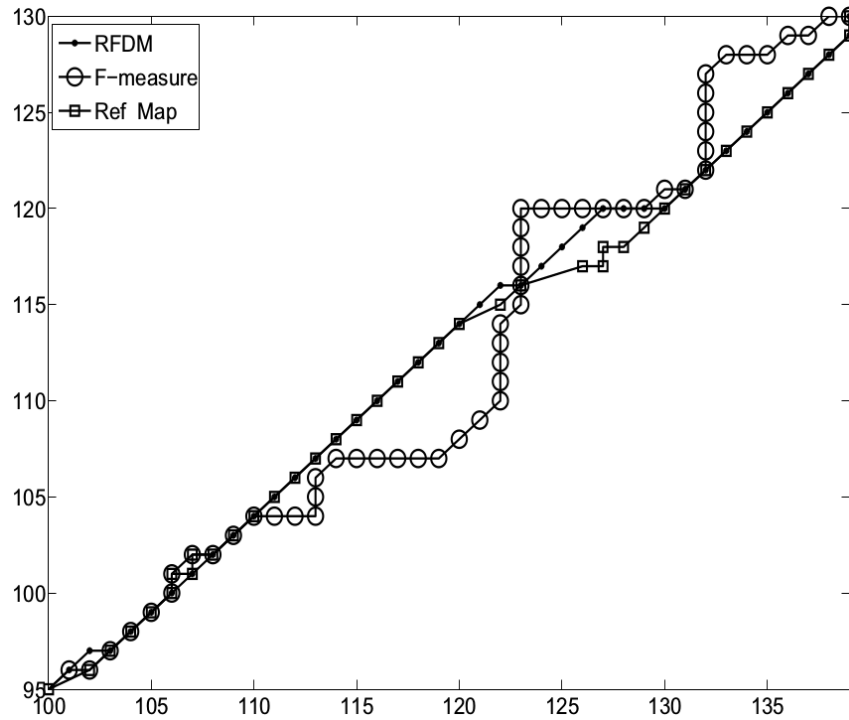


Figure 2.3: Comparison between the F-measure and RFDM results [2]. The axis contain the subtitle segment numbers.

Also, they observed that while the DTW based on F-measure requires a fine tuning of the diagonal weight ( $\alpha_{F_M} = 9$  in Lavechia et al. [3]), the minimization approach has better results without any tuning. Optimization of the diagonal weights in the minimization approach could improve even more the results. The authors also stated that the alignment performance could be improved by using timing information.



## 2.3 Subtitle alignment based on timing information

While the last two described approaches used alignment techniques based on the similarity of the translated subtitles with the source segments, other authors tried different approaches.

Tiedemann [4] based his alignment solution in time overlaps. The author started by choosing the segmentation level that he would align. One could align the subtitles using the subtitle screens that are present on the subtitle files – usually one or two lines – or the actual sentences. He opted for the latter, because he considers that sentences are linguistically motivated units and important for applications using the aligned data. On the contrary, the subtitle screens often include fragments uttered by different speakers and the contents of these subtitle screens vary between different subtitles and, therefore, are much harder to align without overlapping them with other screens. Hence, the author decided to align the subtitles at the sentence level. However, as subtitles do not come segmented into sentences, this required a sentence splitter. He assumed that his sentence splitter would work well for most of the languages.

According to Tiedemann, one of the popular approaches to align sentences in bilingual corpora is the length-based. It is based on the assumption that translations tend to have similar lengths in characters, with some variance. Using this assumption, it is possible to apply the DTW algorithm using the sentence sizes comparison as the distance and therefore, calculate the best mapping between the two languages corpora. It has been shown that this algorithm is flexible and robust, even without changing its parameters [4]. However, the author concluded that due to the characteristics of the subtitle files – described in the previous section – it is obvious that certain settings and assumptions of the algorithm are not appropriate for this kind of corpus. Subtitles contain many insertions and deletions and this difficult those algorithms, that require fine tuning of the parameters in order to work well in each case. To overcome this problem, the author adjusted the probabilities set in the length-based alignment approach. By manually aligning some example subtitles, the author obtain the probabilities in Table 2.3 for the different alignment types.

alignment type	count	probability
1:1	896	0.6829
2:1	100	0.0762
0:1	91	0.0694
1:0	74	0.0564
1:2	72	0.0549
1:3	24	0.0183
3:1	16	0.0122

Table 2.3: Adjusted priors for various alignment types (with probability > 0.001) [4].

Tiedemann omitted alignment types with probabilities below 0.001, for efficiency reasons. Despite calculating these probabilities and assuming that they are more appropriate than the default settings, he was aware that the sample used is not representative for the entire collection and the results may vary depending on the languages used.

After testing the algorithm, using the calculated values for the parameters, he obtained many erroneous alignments. In fact, most of the alignments obtained were wrong, which made the author consider a different approach.

Tiedemann concluded what he already suspected: a length-based approach is not very effective in subtitle alignment. Hence, he considered a different approach that uses the time information given in the subtitles. As each subtitle file has the timestamps that synchronize them with the original movie, different subtitles for the same movie should be shown roughly at the same time, even if they are in different languages. As we have seen in the previous section, subtitles are not always synchronized with each



other, but the overlap in time in which they are shown should be a good predictor for correspondence. Even if the same information is segmented in different ways, the time in which it appears in the screen should be similar, so that it corresponds to the movie dialogs.

To the author, the main problem in this approach is that the alignment is still done at sentence level and the information on the subtitle files is only for the subtitle screens. Some sentences start/end points correspond to a subtitle screen start/end, which makes it trivial to infer the start and end times of that sentences. On the other hand, some sentences end in the middle of a subtitle screen. In these cases, there is the need of approximating the time boundaries. To calculate the boundaries of sentences, Tiedemann uses the nearest time events and equation 2.11.

$$t_{new} = t_{before} + c_{before} * \frac{t_{after} - t_{before}}{c_{before} + c_{after}} \quad (2.11)$$

In equation 2.11,  $t_{before}$  and  $t_{after}$  are the nearest time events before and after the current position, respectively. Likewise,  $c_{before}$  and  $c_{after}$  are the lengths of the strings before and after the current position up to the nearest time events. This estimates the time for the sentences that end in the middle of subtitle screens, using the timestamps that are known and considering the length of the strings. After this time is calculated for the entire subtitle file, the consistency of the values is checked. By using this formula, some events can have identical or even decreasing values. In this case, he, iteratively, adds a dummy time of 0.0001 to remove the inconsistency.

After the times are fixed for every sentence in the subtitle, the sentences can be aligned by combining the length-based technique with the time overlaps. The length-based approach is used to find the possible mappings between the sentences to align and then, the algorithm runs through the pairs in a sliding window picking the pair with the highest time overlap. This can be done in linear time, because of the use of absolute time values – no need of recursion. By using the time to align the sentences, it makes it easy to handle cases where insertions or deletions occur. With this technique, most of the erroneous alignments are due to errors in the sentence splitting or lack of synchronization of the subtitles timestamps.

In preliminary evaluations, Tiedemann realized that the results using the time overlaps approach were very accurate or very poor – depending on the synchronization of the subtitles. By inspecting some problematic cases, he concluded that the errors were due to timing issues: a difference in speed and a difference in starting times. Therefore, the author found a way to adjust the timing differences, by computing two additional parameters: *time ratio* and *time offset*. The computation of these parameters requires the use of two anchor points that correspond to true alignment points, using the formulas 2.12 and 2.13.

$$time_{ratio} = \frac{(t_1 - t_2)}{(s_1 - s_2)} \quad (2.12)$$

$$time_{offset} = t_1 - s_2 \times time_{ratio} \quad (2.13)$$

In equations 2.12 and 2.13,  $s_1$ ,  $s_2$ ,  $t_1$  and  $t_2$  are the time values (in seconds) of the anchor points in the *source* and *target* subtitles, respectively. Having the  $time_{ratio}$  and  $time_{offset}$  calculated, all the time values are adjusted in the *source* subtitles. When the time overlap approach was used with the subtitles synchronized with this method, it produced very effective results. However, this time synchronization method requires two reliable anchor points that should be far away from each other. The most reliable approach is to define the anchor points manually, which is not doable for an entire collection. An alternative to restrict human intervention is to predict when erroneous alignments will occur, using some simple heuristics and define the anchor points only in those cases. For example, we can count the ratio



between sentences with no alignment and the ones aligned.

$$align_{ratio} = \frac{|alignments + 1|}{|empty alignments + 1|} \quad (2.14)$$

Assuming that we only consider two subtitles to be aligned when most of the sentences are aligned, we can use a threshold for this ratio to decide whether they are aligned or not. The ones that are not aligned can be inspected by humans and the time synchronized using the described method. Alternatively, the author proposes to scan the subtitle pairs in a sliding window in order to find anchor point candidates. Using the strings similarity, we can find the anchor points to use in the time synchronization. However, this depends on how similar the source and target languages are and requires dictionaries to every language in the dataset. Furthermore, it is not a straightforward task and may select wrong anchor points.

To reduce synchronization errors made by wrongly selected anchor points, Tiedemann proposes to try all possible combinations of anchor point candidates and use them iteratively. The one with higher alignment ratio is picked. The author only considered this a possible solution, because the time overlap approach is fast enough to make this feasible. In another article, Tiedemann [5] more thoroughly describes the process of synchronizing movie subtitles.

To evaluate the alignment, Tiedemann considered four different approaches:

1. length-based approach with the adjusted priors (*length*)
2. standard time-overlap alignment (*time1*)
3. time-overlap alignment with similarity filter (threshold = 0.8) applied in cases where  $align_{ratio} < 2.0$  (*time2*)
4. time-overlap alignment with similarity filter (threshold = 0.6) and iterative selection of candidate pairs in cases where the  $align_{ratio} < 2.0$  (*time3*)

The similarity filter is the same for both *time2* and *time3*, but the threshold is different so that the iterative anchor point selection in *time3* is more flexible. It is not advised to use such a relaxed threshold in *time2* in order to avoid false positives. The results obtained by the author are presented in Table 2.4, where the scores are split into three categories: *correct* for exact matches; *partial* for alignments where source segments overlap correctly with target segments, despite not being totally correct; and *wrong* for all other alignments. The counted scores are only those present in the reference data (manually aligned).

approach	correct	partial	wrong
length	0.515	0.119	0.365
time1	0.608	0.105	0.286
time2	0.672	0.136	0.192
time3	0.732	0.144	0.124

Table 2.4: Scores of different alignment approaches [4].

As we can see in Table 2.4, the time-overlap approaches present major improvements, compared to the length-based alignment. Also, we can see that the recursive anchor point approach outperforms the other approaches for time synchronization.

In further tests, the author concluded that the time-overlap approach greatly improves the length-based results, because it handles much better insertions and deletions in the subtitles. Tiedemann concluded that the time synchronization is helpful for almost all types of alignment.



After testing each of the dataset subtitles separately for each movie, the author observed that the *time3* approach did not have better scores for every movie, but the overall performance was higher using this strategy compared to the others.

Also, the author verified that the results of the length-based approach were better when using Dutch-German subtitles, rather than Dutch-English. This can be explained by the differences in style, because English subtitles tend to be more detailed whereas German and Dutch more compressed. On the contrary, the time-overlap methods showed a better performance in Dutch-English alignments. As English subtitles are usually created for the hearing impaired, it contains many insertions that can be better handled by the time-overlap approaches, which explains the difference in scores.

Itamar et al. [6] tried an approach that combines solutions based on the sentences character length with subtitle timing information. They considered that subtitle alignment is a similar task to regular sentence alignment. The authors approach disregards many-to-many alignments because they found them quite rare to occur, allowing only 1:1, 0:1, 1:0, 2:1 and 1:2 alignments, which are the most common.

They used a dynamic programming algorithm to align the movie subtitles that looks for a minimal cost alignment, according to the aforementioned constraints. The matching cost is defined in the equation 2.15.

$$C(i, j) = \min \begin{cases} C(i-1, j-1) + c(s_i, t_j) \\ C(i-1, j) + c(s_i, \emptyset) \\ C(i, j-1) + c(\emptyset, t_j) \\ C(i-2, j-1) + c(s_{i-1} || s_i, t_j) \\ C(i-1, j-2) + c(s_i, t_{j-1} || t_j) \end{cases} \quad (2.15)$$

Where  $C(s, t)$  is the cost of aligning the segments  $s_0 \dots s_i$  with  $t_0 \dots t_j$ ,  $c(s, t)$  is the cost of aligning  $s$  with  $t$ ,  $\emptyset$  is an empty string (in case of insertion or deletion) and  $||$  is a concatenation – used when one segment aligns with more than one.

To calculate the cost of the alignment  $c(s, t)$ , the authors followed the method used by Gale et al. [7], that consists of the relative normalized length of sentences, namely  $\frac{l(s)}{l(S)}$  and  $\frac{l(t)}{l(T)}$  where  $l(S)$  and  $l(T)$  are the total lengths of the *source* and *target* subtitle files, respectively. This was used by Itamar et al. [6] as a metric for similarity between sentences, but it was combined with an additional resource that is available on the subtitle files – timing information. The use of timing information combined with the sentences relative length can be a powerful way of producing alignments. Therefore, the authors defined  $d(s)$  as the duration of a segment  $s$ , as shown in equation 2.16.

$$d(s) = \text{end}(s) - \text{begin}(s) \quad (2.16)$$

But, they stated that in order to have better results, this information should be used as relative to the total subtitle duration. Hence, the cost function is defined in the equation 2.17.

$$c(s, t) = \lambda \left( \frac{d(s)}{d(S)} - \frac{d(t)}{d(T)} \right)^2 + (1 - \lambda) \left( \frac{l(s)}{l(S)} - \frac{l(t)}{l(T)} \right)^2 \quad (2.17)$$

Where  $d(S)$  and  $d(T)$  are the total durations of the *source* and *target* subtitles, respectively. The  $\lambda$  parameter should vary between 0 and 1 and represents the relative importance of the timing information. The cost function for insertions and deletions is calculated considering that it does not depend on segments length or subtitle duration, as it is defined in equation 2.18.

$$c(e, \emptyset) = c(\emptyset, f) = \delta \quad (2.18)$$



Equation 2.18 represents the cost of all 0:1 and 1:0 alignments, which has the purpose of penalizing this type of alignments.

The authors determined empirically values for  $\delta$  and  $\lambda$ . After determining the values for English-Hebrew and English-Spanish alignments, Itamar et al. [6] concluded that as the obtained  $\lambda$  value was similar for both language pairs, it is not language dependent. In the other hand, the obtained  $\delta$  value was slightly higher for the latter language pair, which might indicate that it is better to have higher penalties for insertions and deletions in similar languages.

To evaluate their system, the authors compared the performance of their cost function with the one used by Gale et al. [7] – only uses the character length metric. The comparisons were made on a document mainly composed by 1:1 alignments (called “easily alignable” by the authors) and one with many 1:0, 2:1 and 2:2 alignments (called “noisy”). On the “easily alignable” document the performance of both systems was roughly equal. However, in the “noisy” document, the length based method had a worse performance. Thus, the author concluded that timing information is very helpful when the alignment is not trivial.

Other authors have used similar approaches to the ones previously described. For instance, Fishel et al. [8] tried a solution that was based in Tiedemann [4] work. Mangeout et al. [9] also developed a subtitle alignment solution based on the timing information. As those solutions are similar to the previously presented, no further description is done here.

Besides the previously described approaches, many authors have been doing work in the corpora alignment area. However, as subtitle files are not regular corpora, many of the traditional approaches are not quite good enough when compared to approaches that make use of subtitle specific characteristics. Even Tiedemann [10], one of the most respected authors in sentence alignment, considers that traditional sentence alignment approaches are not appropriate for this kind of data [4].

Authors like Gale et al. [7], Brown et al. [11] and Songyot et al. [12] have done work in the corpus alignment topic, which was useful to better understand the alignment process and its inherent problems. Also, Singh and Husain [13] have compared several of the alignment methods. However, since the described approaches to align subtitle files are already based on those researches, no further description of them is done here.







## Chapter 3

# Alignerzilla

In order to achieve the aforementioned goals, *Alignerzilla* was created. The developed solution has been built having in mind the presented problems and taking into account the research presented in Chapter 2.

Our work has been divided in two major phases: the pre-processing and the subtitle alignment phase. In this implementation we have used different tools and languages, such as `shell` scripting, `Perl` and `Java`. The implementation was performed in a generic way, so that it could be used in different scenarios or extended for further use. We have decided to include a *configuration file*, where the user can specify some options before running the module, as can be seen in Appendix B. The development followed an incremental approach, where functionalities were added one by one, when the previous one was completed.

### 3.1 Architecture overview

As we want to create parallel corpora, using movie subtitles, we can see the solution as a production line where the starting product (*input*) are movie subtitles in their raw state and the final product are the aligned subtitles (*output*). The product passes through several steps until it reaches the end of the line. Figure 3.1 represents the high-level view of the developed solution workflow.

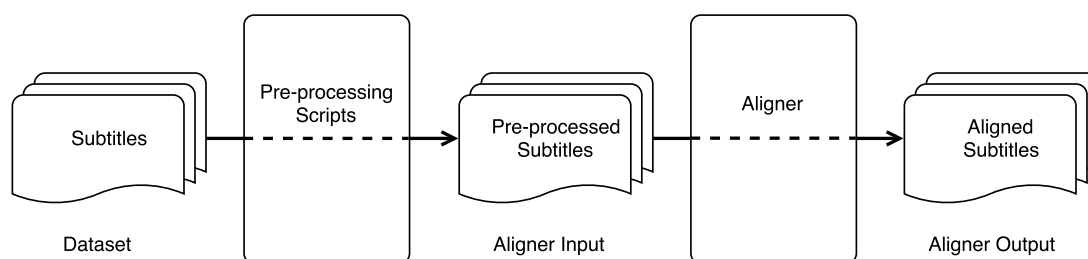


Figure 3.1: High-level view of the developed solution workflow.

In the following sections we will explain in detail how each of the components were developed and the effect they produce in the dataset.

The domain of the problem can be broken down in the classes represented as the diagram in Figure 3.2 and described as follow:

#### SubtitlePair

Represents the *subtitle pair* to align. It is composed by two subtitle files – *source* and *target*. To produce correct alignments it is expected that the two subtitles are from the same movie. In



order to obtain translations as the output, the subtitles must be of different languages. To obtain paraphrases and synonyms, the subtitle pair must be of the same language.

### Subtitle

Represents each subtitle file. It is composed by all the sentences that form the subtitle.

### Sentence

Represents each sentence that composes a subtitle file. Besides the *text* that forms a sentence, contains the *identifier* from the subtitle screen it belongs, as well as *start* and *end* times. This is the smallest element in the domain.

### SentenceGroup

A *sentence group* is an aggregate of zero ou more sentences. As the same dialog in a movie can be written using a different number of sentences, there is the need of grouping sentences in order to align them correctly – important for N:M alignments. When an alignment is 1:1, each Sentence Group contains a single sentence. In N:M alignments, each Sentence Group contains multiple sentences. In 1:0 alignments, the second Sentence Group contains no sentences.

### Match

This element is composed by two sentence groups – *source* and *target* – that match. It corresponds to an alignment. The output of the aligner is a list of Matches. An example of the domain classes that compose a Match can be seen in Figure 3.3.

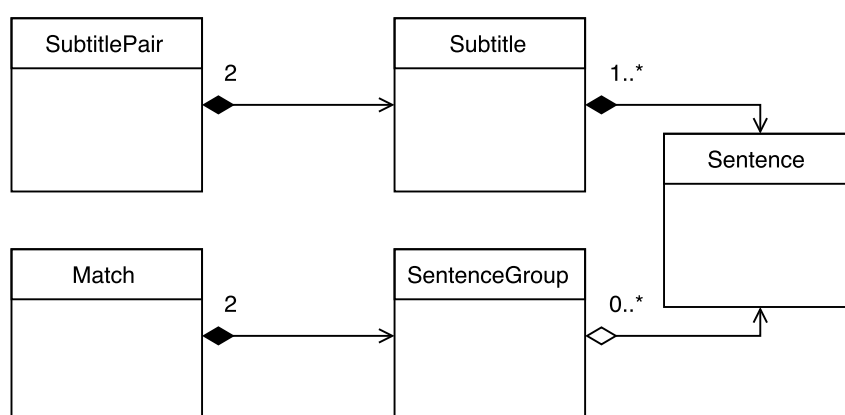


Figure 3.2: UML representation of the problem domain.

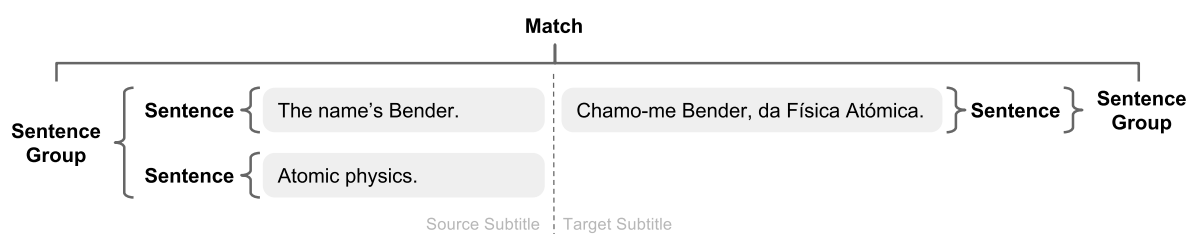


Figure 3.3: Representation of the different classes that compose a Match. Example taken from *A Beautiful Mind* movie.

## 3.2 Pre-processing

As we have seen, pre-processing is an important step when working with this kind of data. Therefore, before trying to align two subtitle files, it is necessary to pre-process the input files.



### 3.2.1 Encoding

The first step in the pre-processing phase is the conversion of the character encodings. Following an approach similar to Tiedemman [4], a script was written to ensure the subtitle files are using UTF-8 encoding. If the original subtitle files are using a different encoding format, the script converts them to UTF-8, before going any further. This is an important step, because the developed aligner expects all the files to be using this encoding. If the subtitle files have different character encodings, the aligner can behave erroneously and, therefore, produce bad results. This script was developed using `bash` commands and the `iconv` tool.

### 3.2.2 Normalization

This is the core of the pre-processing phase. A Perl script has been developed in order to normalize the subtitle files. It removes noise from the subtitles and formats the files in a way that makes it simpler to find alignments. It consists of several steps, such as:

**Tag removal** Some subtitle files include formatting tags derived from HTML, in order to change the visual aspect of the subtitles on screen (e.g. `<b>...</b>` or `<font color="#FFFF00">`). Given our purpose, this represents noise. Therefore, this scripts removes all the tags it finds. To find this tags, the following regular expression is used:

```
# word character(s) with or without '/' between '<>'  
<[\\/\w\s="#">+>
```

**Descriptions removal** As we have seen in a previous chapter, *scene description insertion* represents a problem that we have to deal with when working with subtitles. Therefore, we decided to remove all the descriptions we can find, so that they do not represent a problem during the alignment phase. To identify these descriptions and remove them, we use the following regular expression:

```
# sentence with capital words that may be between '[]'  
# occupying one or more lines  
\[?[A-Z].+\]?\\n?.+[A-Z ]{5,}.+\]?
```

**Sentences forming** As we have seen, each *subtitle screen* may contain one or two text lines. In the developed script we append both lines, because we do not need them to be separated and it facilitates the next phase.

**Unfinished sentences marking** As we have seen previously, most of the state-of-the-art approaches [3, 2, 4, 6] decided to align subtitles at the sentence level. Given our purpose, we also decided to align at the sentence level. Thus, in order to ease the alignment process, the pre-processing script tries to identify unfinished sentences and mark them with a distinctive symbol (in this case `>>` was used). Later on, the aligner takes this information into account and uses it to complete sentences accordingly, as explained in 3.2.3.

The following regular expression is used to find unfinished sentences:

```
# sentence that ends with '...' or without '.' or '!' or '?'  
([\\.] {3}|[^\.!?])$
```



### 3.2.3 Sentence Concatenation

As we have seen in 3.2.2, the normalization script marks the unfinished sentences. Afterwards, using this information, the developed aligner concatenates consecutive subtitle screens that complete each other, so that no sentences are left unfinished. An example can be seen in Table 3.1.

Before concatenation	193 00:13:38,340 -- > 00:13:41,930 Gentlemen, might I remind you that my odds of success...>>
	194 00:13:42,010 -- > 00:13:44,220 ...dramatically improve with each attempt?
After concatenation	00:13:38,340 -- > 00:13:44,220 Gentlemen, might I remind you that my odds of success... dramatically improve with each attempt?

Table 3.1: *Sentence concatenation* example taken from *A Beautiful Mind* movie.

When text from different *subtitle screens* is concatenated, the timestamps of the resulting sentence are obtained from the corresponding original sentences. However, before two sentences are concatenated, two conditions must be met. The first sentence must be classified as *unfinished sentence* and the second sentence must be classified as *continuing sentence*. In order to classify a sentence as *continuing*, we use the following regular expression:

```
# sentence starts with lowercase or reticences or quotes
# or special cases like 'I' or an acronym
(^[a-z\\.\\" ]+|^I|^ [A-Z]{3,}).+
```

However, there are still some cases where it is impossible for the program to decide correctly. For instance, if a sentence starts with a proper name, that sentence will never be considered as *continuing*, because of the capital letter in the beginning.

### 3.2.4 Sentence Splitting

In order to align subtitles at the sentence level, not only we need to concatenate sentences that appear in different *subtitle screens*, but also to split sentences that appear together in the same *subtitle screen*. In Table 3.2 we can see an example.

In order to correctly split sentences, the sentence splitter needed a reliable way to identify sentence ends. Therefore, the splitter looks for final punctuation but takes into account special cases that may appear, such as “Mr.”, “Dr.” or “St.” – the special cases that are considered have been manually added as we have seen them occur. As this special cases are language dependent, new ones can easily be added using the *configuration file*. By splitting the text of a single *subtitle screen*, the start and end timestamps of each sentence need to be recalculated. To calculate those times, we have used equation 3.1 [4].

$$t_{new} = t_{before} + c_{before} * \frac{t_{after} - t_{before}}{c_{before} + c_{after}} \quad (3.1)$$

In Equation 3.1,  $t_{before}$  and  $t_{after}$  are the nearest time events before and after the current position, respectively. Likewise,  $c_{before}$  and  $c_{after}$  are the lengths of the strings before and after the current position up to the nearest time events. This estimates the time for the sentences found in the middle of *subtitle screen*.



Before splitting	611 00:50:49,930 -- > 00:50:51,730 - No! A human girl? - Homo sapien.
After splitting	611 00:50:49,930 -- > 00:50:50,340 - No!  00:50:50,340 -- > 00:50:50,032 A human girl?  00:50:51,032 -- > 00:50:51,730 - Homo sapien.

Table 3.2: *Sentence splitting* example taken from *A Beautiful Mind* movie.

### 3.3 Aligner

The developed aligner tries to find alignments between two subtitle files from the same movie. The alignments are made at the sentence level and we consider not only one-to-one alignments, but also one-to-many. As other authors have stated and can be seen at Table 2.3, many-to-many alignments are quite rare and they affect the performance of the aligners, so we have decided to only consider 0:1, 1:0, 1:1, 1:2 and 2:1 alignments, disregarding N:M.

As we have seen in the research done by other authors and described in Section 2.3, **alignment techniques based on the timing information** available on the subtitle files represent a powerful and reliable way of aligning subtitles [4, 6]. This type of approach usually allows to have well aligned subtitles with little effort, but only if the subtitle pair is synchronized in time. When that does not happen it is necessary to synchronize the subtitles before trying to align them.

On the contrary, as described in Section 2.2, subtitle **alignment approaches based on the similarity of segments** are more flexible regarding time, but are more susceptible to produce erroneous results [3, 2]. Approaches based on the similarity often require more complex tasks – e.g. translation of words using dictionaries –, thus increasing the overhead of the aligning process and making this kind of approaches worse than time based approaches, execution performance-wise.

Therefore, in order to obtain a module that aligns subtitles with a good performance and is flexible enough to align subtitle pairs that are not synchronized, **we have combined the two techniques**. The time based approach is the main provider of trustworthy alignments, while the similarity based technique is only used when the first cannot find alignments with a good confidence level.

If neither method is able to obtain a valid alignment, *Alignerzilla* assumes there is no possible match for the given sentence.

Figure 3.4 represents how we have combined time and similarity techniques to provide good alignments.

In summary, when the subtitle pair is synchronized, none or very few segments require the similarity based approach to be used, thus granting the system a good execution performance. On the other hand, when that does not happen, the similarity based aligner will be used to find the alignments. In this cases the execution performance is slightly worse, but surely the results are better than if only one method was used.



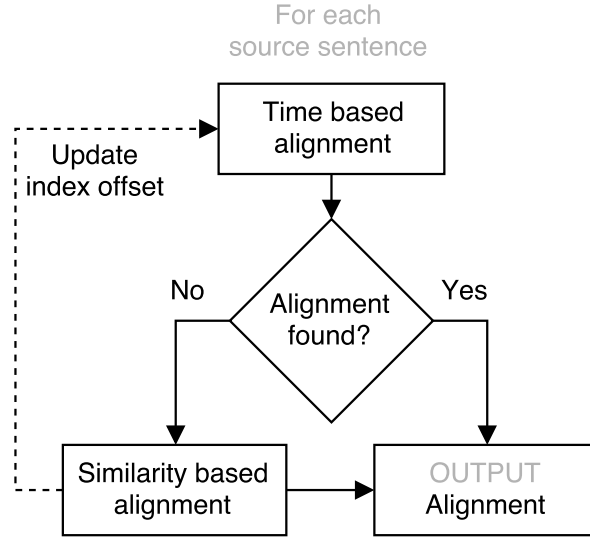


Figure 3.4: Overview of the alignment process for each iteration.

### 3.3.1 Time based alignment

Consider  $S = \{S_1, \dots, S_{|s|}\}$  and  $T = \{T_1, \dots, T_{|t|}\}$  the *source* and *target* subtitles, respectively. *Alignerzilla* starts by iterating over  $S$  looking for matches in  $T$ .

Using the starting time of  $S_i$ , the aligner looks for the sentence in  $T$  that starts closer to that time,  $T_j$ . If  $S_i$  and  $T_j$  have similar **starting times** and similar **duration** it is considered to be a  $match(S_i, T_j)$ .

To compute a sentence  $s$  duration, Equation 3.2 is used, similarly to what has been seen in Section 2.3.

$$d(s) = end(s) - start(s) \quad (3.2)$$

To validate if two sentences can be a match duration-wise, Equation 3.3 is used.

$$|d(S_i) - d(T_j)| < threshold \quad (3.3)$$

Where the mentioned *threshold* is a previously specified value and can be customized in the *configuration file*. However, if starting times match but the durations do not, it means one of the sentences is shorter than the other. Hence, when this occurs the aligner considers the possibility of a 1:2 or 2:1 alignment. If  $S_i$  has smaller duration than  $T_j$ , the aligner tries the  $match(S_i + S_{i+1}, T_j)$  and considers the new duration. Analogously, the  $match(S_i, T_j + T_{j+1})$  is considered if  $T_j$  is the one lacking duration.

After using the time based alignment approach a match is issued and, afterwards, one of two things happens. Either the match is considered valid and everything is okay, or the match is not valid – the sentences are too different – and the textual similarity must be used in order to find a trustworthy alignment.

A limitation of this approach is that it requires the sentences to be synchronized to work properly. Also, the current implementation only supports 1:0, 1:1, 1:2 and 2:1 correspondences. Through preliminary experimentations, we have observed that allowing other types of N:M alignments, such as 1:3 or 2:3, deteriorated the alignments quality. Since those cases happen very few times in most of the subtitle pairs [4], as can be seen in Table 2.3, we have decided to disregard them for the sake of the overall alignment quality, knowing it is a limitation.



### 3.3.2 Similarity based alignment

The developed solution to align subtitle sentences that could not be aligned using timing information is similar to the work done by Tsiartas et al. [2], in the sense that uses a Distance Metric that we try to minimize. In other words, we have defined the differences between two sentences as their **distance**. *Alignerzilla* chooses as an alignment the **sentence pair that has the smaller distance**. However, instead of using dynamic programming to do so (like Tsiartas et al. [2]), the aligner uses a sliding window approach, since we only want to use this technique on sentences that could not be aligned using the timing information. Thus, in order to find an alignment for a given sentence  $S_i$ , the aligner chooses a sentence  $T_j$  based on the last alignments information. Then, it looks to a window before and after that sentence, comparing the distances between those sentences and  $S_i$ . In other words, the aligner computes the distance between  $S_i$  and sentences from  $T_{j-k}$  to  $T_{j+k}$ , where  $k$  is the *sliding window size*.

To calculate the **distance between two sentences** it uses two metrics: **word** and **length** similarity. Nevertheless, the two metrics may have different weights. We have considered that sentences composed by the same words are a match for sure, while having the same length does not guarantee they are a match. Hence, the Distance ( $D$ ) between two sentences ( $S_i, T_j$ ) is calculated as Equation 3.4 considering word similarity and sentence length and varies between 0 and 1. This equation can be seen as a variation of Itamar [6] cost function (Equation 2.17).

$$D(S_i, T_j) = \lambda \left( 1 - \frac{M_w}{N_w} \right)^2 + (1 - \lambda) \left( \frac{\min(L(S_i), L(T_j))}{\max(L(S_i), L(T_j))} \right) \quad (3.4)$$

Where  $M_w$  is the number of *words that match* between the two sentences,  $N_w$  the number of words in the biggest sentence and  $L(s)$  the length of sentence  $s$  – in this case, the number of words of a sentence is used as its length. The  $\lambda$  parameter should vary between 0 and 1 and represents the relative importance of the word similarity. This parameter is used to increase or decrease the influence of the word similarity in the final distance. To decide if two words match, we chose not to use a direct comparison, because that would reject words with irrelevant differences (e.g. color vs colour). Hence, the Levenshtein distance<sup>1</sup> – metric to measure the difference between two character sequences – is calculated and used to decide if the words match. Obviously, if the two subtitles used are in different languages, we need to translate them before any comparison is made, as explained in subsection 3.3.3.

The cost function for insertions and deletions is calculated considering that it does not depend on segments similarity, as it is defined in Equation 3.5.

$$D(S_i, \emptyset) = D(\emptyset, T_j) = 1 \quad (3.5)$$

Equation 3.5 represents the cost of all 0:1 and 1:0 alignments, which has the purpose of penalizing this type of alignments, thus giving those alignments the maximum distance value.

A limitation of our similarity based aligner implementation is that it only supports 1:1 alignments. During the implementation, we have observed that when we gave the aligner the chance of matching  $N$  *source sentences* with  $M$  *target sentences*, the results were less accurate. Thus, knowing that 1:1 correspondences are the most common [4], as can be seen in Table 2.3, we have limited the similarity based technique to only consider this type of alignments. This way, we make it possible for the aligner to make better choices in the majority of the cases, while neglecting the cases where the correspondences should be N:M.

<sup>1</sup>[https://en.wikipedia.org/wiki/Levenshtein\\_distance](https://en.wikipedia.org/wiki/Levenshtein_distance)



### 3.3.3 Translator

To use the sentence similarity approach to align subtitles of different languages, the sentences cannot be compared as they appear, because there should be little similarity between them even if they correspond to the same dialog. Thus, one of the sentences needs to be translated so that they can be compared.

In order to translate sentences, a bilingual dictionary has been built. This dictionary is composed by words in the target language followed by their translation in the source language. Since the developed solution focus the alignment of subtitles in Portuguese and English, the created dictionary is for those languages. To build it, a list of the top 5000 English words has been downloaded from the Word Frequency website <sup>2</sup>. Then, using an API for Google Translate <sup>3</sup> all the words have been translated and stored as a bilingual lexicon – list of words that correspond. To enhance the dictionary, some words have been manually corrected, afterwards. This is a simple way to create dictionaries and the same process can be used to obtain dictionaries for other languages. However, a dictionary that contains only a translation for each word has its limitations and may not be sufficient in some cases. But, as the word comparison is not the only metric used, it should be enough to help align the majority of sentences that require this approach.

After having the bilingual dictionary, it was used to translate sentences word by word. Since the dictionary contains the most used English words, almost every sentence contains words present in that list. Nevertheless, not every word is on the dictionary. In those cases, the word is left unchanged and compared as it is. We have considered that it is better to compare the words even when they are in different languages rather than to ignore them. For instance, some of the words that cannot be translated may be proper names, which usually helps the alignment process. When a proper name is left unchanged, it appears in both sentences, which will increase the alignment chance.

### 3.3.4 Configurations file

As mentioned in the previous sections, a *configurations file* was included in our project, so that some parameters of *Alignerzilla* could be tuned. Using this file, the following configurations are available:

- Specify if the aligner should use *only the time based technique* to align subtitles;
- Adjust *threshold* values to obtain different results;
- Specify what *special cases* should be taken in account by the sentence splitter;
- Choose if the similarity based technique should *translate* the sentences before comparing them – no translations are needed if same language subtitles are used;
- Specify what *dictionary file* should be used to translate the sentences.

This file can be seen in Appendix B.

The developed pre-processing scripts and *Alignerzilla* are available in the repository:

<https://github.com/b2rosado/subtitle-alignment>.

Instructions on how to use the pre-processing scripts and the subtitle aligner are included in the repository *read me* file and can be seen in Appendix A.

---

<sup>2</sup><http://www.wordfrequency.info/>

<sup>3</sup><https://code.google.com/archive/p/java-google-translate-text-to-speech/>



## Chapter 4

# Building reference alignments

### 4.1 Dataset

In order to properly evaluate the developed solution, we needed enough subtitles for testing purposes. Given this, OpenSubtitles <sup>1</sup> administrators were kind enough to provide us, for free, a great amount of SubRip (.srt) subtitles in English and Portuguese languages.

Afterwards, inspired by the evaluation processes used by other authors [3, 2, 4], we have selected 40 movies to evaluate our solution. The movie selection was manually done and we tried to pick different types of movies.

Preliminary observations showed us that when the number of sentences in a subtitle pair for the same movie, varied too much, it meant one of the subtitles was much more descriptive and contained a lot of sentences that were not even present in the other one. This characteristic makes it difficult to find alignments either manually, either with a subtitle aligner. Thus, while selecting the subtitles to include in the dataset, we discarded subtitles pairs where the differences between English and Portuguese subtitles were above 40% – for instance, in one of the pairs found, the *source subtitle* contained 2615 sentences and the *target subtitle* 1702. Although these subtitle pairs could be used, we decided to discard them, because we were aware that they would act like outliers, influencing negatively the performance of any aligner. Table 4.1 shows the movie subtitles selected to be included in the dataset, the number of sentences in *source* and *target* subtitles, as well as the difference percentage between them.

### 4.2 Reference creation

For the 40 subtitle pairs selected, reference alignments have been created between English and Portuguese languages. The reference alignments were created with the help of 36 different annotators. All the annotators were Portuguese with a good comprehension of the English language.

#### 4.2.1 Web Application

A web application was developed so that the annotators could easily create the reference alignments. Each annotator was given a different subtitle pair to align using the developed web application. In order to ease the process of creating reference alignments, the subtitle files were pre-processed and split into sentences. Then, we took 75 sentences from the beginning, middle and end of each subtitle file, leaving us with a total of 225 sentences per subtitle file. This division was made to grant that the reference

---

<sup>1</sup><http://www.opensubtitles.org>



alignments comprised different sections of each subtitle file. In the web application, the sentences taken from the subtitle pairs would show up, side by side, and the annotators had to select sentences that matched and submit them, one at a time. Figure 4.1 shows how the web application looks like.

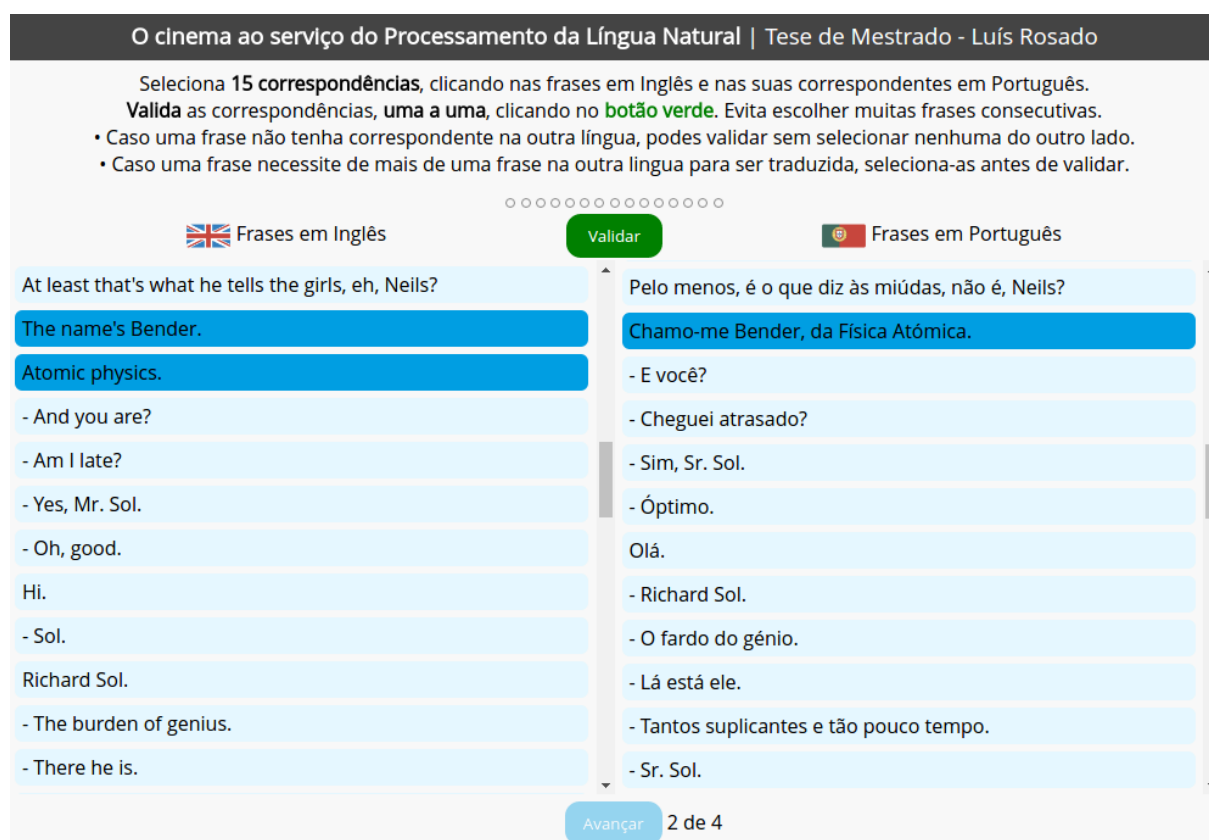


Figure 4.1: Screenshot of developed the web application where annotators aligned subtitles for reference.

This process was divided in four different steps. The first step was the presentation of some instructions to the annotators. This step was of great importance, because not only it explained to the annotators what we expected them to do, but also how they should select the alignments in the web application. Also, some examples were shown. The other three steps comprised the creation of the reference alignments. These three steps contained the sentences taken from the beginning, middle and end of the subtitle files, respectively. In the end of the process, each annotator had manually select 45 alignments – 15 from each part.

By using this process, we have obtained impartially chosen alignments that can pose as reference for any subtitle aligner.

## 4.2.2 Alignments validation

In order to grant inter-rater agreement, after obtaining the reference alignments for every subtitle pair, different annotators checked their correctness. To do this, we randomly selected 200 reference alignments. Then, two different annotators, analyzed the alignments and registered alignments that they considered to be incorrect. After this was done, only 5 out of 200 alignments were considered incorrect, which shows us that the annotators agree in almost every alignment.

Afterwards, a different annotator checked the resulting alignments and corrected cases where the annotator decisions were not correct – very few cases, probably caused by small distractions. Some of



the incorrections found were:

- Similar sentences appeared in a subtitle file and the annotator selected one that was not the most accurate match – verifiable by looking at the sentence context.
- The annotator did not select the minimum group of sentences that formed an alignment – i.e. a N:N alignment was selected when multiple 1:1 alignments should have been selected instead.
- The selected sentences were simply not an alignment – the annotator selected sentences that were definitely not a match.

We have manually corrected these cases. Due to this, not every movie has exactly 45 reference alignments, because some were deleted or separated in more than one, which resulted in a total of 1789 reference alignments.

The web application where the reference alignments were created is depicted with more detail in Appendix C.

The used subtitle dataset and reference alignments can be found at the aforementioned repository (<https://github.com/b2rosado/subtitle-alignment>).



Movie	#Source	#Target	Difference
12 Years a Slave (2013)	1323	1349	2,0%
A Beautiful Mind (2001)	1498	1436	4,3%
Alien (1979)	962	1111	15,5%
Despicable Me (2010)	1121	1080	3,8%
Fifty Shades of Grey (2015)	1480	1490	0,7%
Fight Club (1999)	2400	2413	0,5%
Finding Nemo (2003)	2152	2078	3,6%
Forrest Gump (1994)	1668	1436	16,2%
Gladiator (2000)	1434	1358	5,6%
Gravity (2013)	1097	1009	8,7%
Inside Out (2015)	2061	1746	18,0%
Interstellar (2014)	2343	2167	8,1%
Law Abiding Citizen (2009)	1378	1445	4,9%
Mad Max: Fury Road (2015)	924	832	11,1%
Million Dollar Baby (2004)	1695	1655	2,4%
Raiders of the Lost Ark (1981)	799	745	7,2%
Real Steel (2011)	2356	2013	17,0%
Shakespeare in Love (1998)	1698	1624	4,6%
Short Term 12 (2013)	1185	1119	5,9%
Slumdog Millionaire (2008)	1119	1139	1,8%
Star Wars: Episode II - Attack of the Clones (2002)	1181	1229	4,1%
The Avengers (2012)	1673	1517	10,3%
The Dark Knight (2008)	2137	1903	12,3%
The Day After Tomorrow (2004)	1456	1367	6,5%
The Departed (2006)	2498	2240	11,5%
The Fast and the Furious (2001)	1238	1193	3,8%
The Godfather (1972)	1755	2040	16,2%
The Hobbit: An Unexpected Journey (2012)	1990	1866	6,6%
The Hunger Games (2012)	1842	1749	5,3%
The Imitation Game (2014)	1718	1636	5,0%
The Intouchables (2011)	1945	1997	2,7%
The Lego Movie (2014)	2185	2057	6,2%
The Lion King 1 1/2 (2004)	1458	1061	37,4%
The Lord of the Rings: The Fellowship of the Ring (2001)	1792	1366	31,2%
The Notebook (2004)	1935	1576	22,8%
The Prestige (2006)	1762	1786	1,4%
The Usual Suspects (1995)	1542	1554	0,8%
Up (2009)	1107	1117	0,9%
V for Vendetta (2005)	1816	1837	1,2%
Whiplash (2014)	982	991	0,9%
<b>Average</b>	<b>1618</b>	<b>1533</b>	<b>8.2%</b>
Standard Deviation	449	412	8.27

Table 4.1: List of the movie subtitles included in the dataset and the differences between the number of sentences in their *source* and *target* subtitles.



# Chapter 5

## Evaluation

In order to evaluate *Alignerzilla*, we proceeded with experiments similar to what other authors have done [3, 2, 4, 6]. Using the dataset described in Chapter 4, we have obtained the alignments for those subtitles using the developed aligner. Afterwards, those alignments were manually analyzed and compared with the reference alignments.

To be able to properly evaluate the aligner performance, we have calculated the following values:

**True Positive (TP)** the aligner finds an alignment that matches the reference;

**False Positive (FP)** the aligner finds an alignment but does not match the reference;

**True Negative (TN)** the aligner does not find an alignment and neither does the reference;

**False Negative (FN)** the aligner does not find an alignment but the reference does.

After defining those variables, we have calculated the Precision and Recall of each aligned subtitle pair with the formulas in 5.1.

$$Precision = \frac{TP}{TP + FP} \quad Recall = \frac{TP}{TP + FN} \quad (5.1)$$

In order to compare the results more easily, we have also calculated the traditional F-measure for each aligned subtitle pair, using the formula in equation 5.2.

$$F_M = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (5.2)$$

The previously presented calculus provided us enough information to evaluate any aligner. Besides calculating the Precision, Recall and F-measure of the alignments found by *Alignerzilla*, we have done some comparisons using different options of our aligner, so that we could evaluate the impact of certain decisions we made. We have also compared the developed aligner with a state-of-the-art aligner, in order to validate our approach against other authors.

While developing the aligner, our focus was on its reliability to find alignments between movie subtitles, rather than in execution performance (for instance, time complexity). Hence, during the development process, we tried to keep the aligner as fast as possible, but no optimizations were made. Therefore, in the evaluation process no measurements were made in this matter.



## 5.1 Results

Following the methodology proposed in the beginning of Chapter 5, we have aligned the dataset subtitles with *Alignerzilla*. As we can see in Table 5.1, the average Precision, Recall and F-measure for the obtained alignments is 82.7%, 89.8% and 85.8%, respectively. In Section 5.3 the obtained results are compared with a state-of-the-art aligner, so that we can infer their quality.

Movie	TP	FP	TN	FN	#A	Precision	Recall	F-measure
12 Years a Slave (2013)	39	3	0	3	45	92.9%	92.9%	92.9%
A Beautiful Mind (2001)	44	0	3	0	47	100.0%	100.0%	100.0%
Alien (1979)	37	4	0	1	42	90.2%	97.4%	93.7%
Despicable Me (2010)	36	4	0	5	45	90.0%	87.8%	88.9%
Fifty Shades of Grey (2015)	38	7	0	1	46	84.4%	97.4%	90.5%
Fight Club (1999)	31	14	0	2	47	68.9%	93.9%	79.5%
Finding Nemo (2003)	29	15	0	1	45	65.9%	96.7%	78.4%
Forrest Gump (1994)	36	3	0	6	45	92.3%	85.7%	88.9%
Gladiator (2000)	43	2	0	1	46	95.6%	97.7%	96.6%
Gravity (2013)	25	10	0	7	42	71.4%	78.1%	74.6%
Inside Out (2015)	26	10	0	7	43	72.2%	78.8%	75.4%
Interstellar (2014)	35	6	0	5	46	85.4%	87.5%	86.4%
Law Abiding Citizen (2009)	37	5	0	3	45	88.1%	92.5%	90.2%
Mad Max: Fury Road (2015)	20	13	0	12	45	60.6%	62.5%	61.5%
Million Dollar Baby (2004)	21	17	0	7	45	55.3%	75.0%	63.6%
Raiders of the Lost Ark (1981)	39	2	0	4	45	95.1%	90.7%	92.9%
Real Steel (2011)	29	9	0	7	45	76.3%	80.6%	78.4%
Shakespeare in Love (1998)	36	5	0	3	44	87.8%	92.3%	90.0%
Short Term 12 (2013)	27	12	0	6	45	69.2%	81.8%	75.0%
Slumdog Millionaire (2008)	40	4	0	1	45	90.9%	97.6%	94.1%
Star Wars: Episode II - Attack of the Clones (2002)	36	6	0	3	45	85.7%	92.3%	88.9%
The Avengers (2012)	39	7	0	2	48	84.8%	95.1%	89.7%
The Dark Knight (2008)	43	2	0	2	47	95.6%	95.6%	95.6%
The Day After Tomorrow (2004)	37	5	0	3	45	88.1%	92.5%	90.2%
The Departed (2006)	40	2	0	3	45	95.2%	93.0%	94.1%
The Fast and the Furious (2001)	26	18	0	1	45	59.1%	96.3%	73.2%
The Godfather (1972)	38	3	0	3	44	92.7%	92.7%	92.7%
The Hobbit: An Unexpected Journey (2012)	43	1	0	0	44	97.7%	100.0%	98.9%
The Hunger Games (2012)	30	6	1	5	42	83.3%	85.7%	84.5%
The Imitation Game (2014)	27	9	0	9	45	75.0%	75.0%	75.0%
The Intouchables (2011)	36	4	0	5	45	90.0%	87.8%	88.9%
The Lego Movie (2014)	40	3	0	2	45	93.0%	95.2%	94.1%
The Lion King 1 1/2 (2004)	27	14	0	4	45	65.9%	87.1%	75.0%
The Lord of the Rings: The Fellowship of the Ring (2001)	36	1	0	5	42	97.3%	87.8%	92.3%
The Notebook (2004)	27	12	0	4	43	69.2%	87.1%	77.1%
The Prestige (2006)	34	10	0	1	45	77.3%	97.1%	86.1%
The Usual Suspects (1995)	31	11	0	3	45	73.8%	91.2%	81.6%
Up (2009)	31	9	0	4	44	77.5%	88.6%	82.7%
V for Vendetta (2005)	38	3	0	3	44	92.7%	92.7%	92.7%
Whiplash (2014)	33	8	0	2	43	80.5%	94.3%	86.8%
<b>Average</b>						<b>82.7%</b>	<b>89.8%</b>	<b>85.8%</b>
Standard Deviation						12.00	7.91	9.20

Table 5.1: Results obtain after aligning the dataset subtitles with *Alignerzilla*.



### 5.1.1 Correlation with sentence number differences

As we have stated in Chapter 4, some of the subtitle pairs in the dataset have considerable differences in the number of sentences that compose *source* and *target* subtitle files. In order to verify how that influences the results, we have calculated the correlation between the difference percentage and the F-measure of each alignment. Table 5.2 shows the number of sentences difference percentage and obtained F-measure for each subtitle pair in the dataset.

Movie	# Sentences difference	F-measure
12 Years a Slave (2013)	2,0%	92.9%
A Beautiful Mind (2001)	4,3%	100.0%
Alien (1979)	15,5%	93.7%
Despicable Me (2010)	3,8%	88.9%
Fifty Shades of Grey (2015)	0,7%	90.5%
Fight Club (1999)	0,5%	79.5%
Finding Nemo (2003)	3,6%	78.4%
Forrest Gump (1994)	16,2%	88.9%
Gladiator (2000)	5,6%	96.6%
Gravity (2013)	8,7%	74.6%
Inside Out (2015)	18,0%	75.4%
Interstellar (2014)	8,1%	86.4%
Law Abiding Citizen (2009)	4,9%	90.2%
Mad Max: Fury Road (2015)	11,1%	61.5%
Million Dollar Baby (2004)	2,4%	63.6%
Raiders of the Lost Ark (1981)	7,2%	92.9%
Real Steel (2011)	17,0%	78.4%
Shakespeare in Love (1998)	4,6%	90.0%
Short Term 12 (2013)	5,9%	75.0%
Slumdog Millionaire (2008)	1,8%	94.1%
Star Wars: Episode II - Attack of the Clones (2002)	4,1%	88.9%
The Avengers (2012)	10,3%	89.7%
The Dark Knight (2008)	12,3%	95.6%
The Day After Tomorrow (2004)	6,5%	90.2%
The Departed (2006)	11,5%	94.1%
The Fast and the Furious (2001)	3,8%	73.2%
The Godfather (1972)	16,2%	92.7%
The Hobbit: An Unexpected Journey (2012)	6,6%	98.9%
The Hunger Games (2012)	5,3%	84.5%
The Imitation Game (2014)	5,0%	75.0%
The Intouchables (2011)	2,7%	88.9%
The Lego Movie (2014)	6,2%	94.1%
The Lion King 1 1/2 (2004)	37,4%	75.0%
The Lord of the Rings: The Fellowship of the Ring (2001)	31,2%	92.3%
The Notebook (2004)	22,8%	77.1%
The Prestige (2006)	1,4%	86.1%
The Usual Suspects (1995)	0,8%	81.6%
Up (2009)	0,9%	82.7%
V for Vendetta (2005)	1,2%	92.7%
Whiplash (2014)	0,9%	86.8%
<b>Correlation</b>		<b>-0.11</b>

Table 5.2: Correlation between the sentence number difference percentage and F-measure.

With our dataset, the obtained value for correlation was -0.11. As we have obtained a negative value we have signs that when the difference increases, the F-measure tends to decrease. However, as this value is close to zero, we cannot state there is a direct impact. The dependency is little and that means the registered differences in the sentence number do not have a big impact on the results.



## 5.1.2 Correlation with time offsets

As we have stated in the previous chapters, subtitle files are not always synchronized in time. Sometimes, subtitles for the same movie have differences in the timestamps present in each dialog. Manually, we have checked the offset values for each subtitle pair, through the comparison of subtitle dialogs that corresponded and, yet, had different start or end timestamps. To simplify the measurements, the smallest unit considered was 100ms. Therefore, every value is a multiple of 100. On Table 5.3 we have registered the offset values and compared them with the obtained F-measure for each subtitle pair.

Movie	# Sentences difference	F-measure
12 Years a Slave (2013)	500	92.9%
A Beautiful Mind (2001)	200	100.0%
Alien (1979)	200	93.7%
Despicable Me (2010)	400	88.9%
Fifty Shades of Grey (2015)	300	90.5%
Fight Club (1999)	700	79.5%
Finding Nemo (2003)	1100	78.4%
Forrest Gump (1994)	300	88.9%
Gladiator (2000)	800	96.6%
Gravity (2013)	500	74.6%
Inside Out (2015)	800	75.4%
Interstellar (2014)	400	86.4%
Law Abiding Citizen (2009)	400	90.2%
Mad Max: Fury Road (2015)	1600	61.5%
Million Dollar Baby (2004)	2000	63.6%
Raiders of the Lost Ark (1981)	600	92.9%
Real Steel (2011)	500	78.4%
Shakespeare in Love (1998)	700	90.0%
Short Term 12 (2013)	1400	75.0%
Slumdog Millionaire (2008)	800	94.1%
Star Wars: Episode II - Attack of the Clones (2002)	600	88.9%
The Avengers (2012)	600	89.7%
The Dark Knight (2008)	300	95.6%
The Day After Tomorrow (2004)	100	90.2%
The Departed (2006)	400	94.1%
The Fast and the Furious (2001)	1000	73.2%
The Godfather (1972)	300	92.7%
The Hobbit: An Unexpected Journey (2012)	400	98.9%
The Hunger Games (2012)	100	84.5%
The Imitation Game (2014)	2500	75.0%
The Intouchables (2011)	800	88.9%
The Lego Movie (2014)	200	94.1%
The Lion King 1 1/2 (2004)	800	75.0%
The Lord of the Rings: The Fellowship of the Ring (2001)	400	92.3%
The Notebook (2004)	900	77.1%
The Prestige (2006)	1200	86.1%
The Usual Suspects (1995)	400	81.6%
Up (2009)	700	82.7%
V for Vendetta (2005)	400	92.7%
Whiplash (2014)	300	86.8%
<b>Correlation</b>		<b>-0.68</b>

Table 5.3: Correlation between the offset times and F-measure.

Looking at Table 5.3 we can see that every movie in the dataset shows some offset in the timestamps. Despite this fact, some subtitles can be considered synchronized because small offset values are often below the used thresholds.

Also, the obtained value for correlation with our dataset was -0.68. As this value is negative, we can conclude that the F-measure tends to decrease as the time offset is higher. The observed dependency is not linear, but we can say, with some confidence, that the obtained F-measure values may depend on subtitle synchronization level.



## 5.2 Comparison with aligner using time only

In order to evaluate the importance of the combination of time and similarity techniques, we have decided to compare the results of *Alignerzilla* with a version of the aligner that uses only the time based approach. To do this, using the *configuration file* we have set the aligner to use the *time only* option.

Despite not measuring the time of any execution, we can tell that while using the Time Only Aligner (TOA) the alignments are performed much quicker than with *Alignerzilla*, due to not using the most complex technique.

The obtained results and the comparison between TOA and *Alignerzilla* can be seen in Table 5.4. To better visualize the differences, Figures 5.1 and 5.2 depict the results of both aligners.

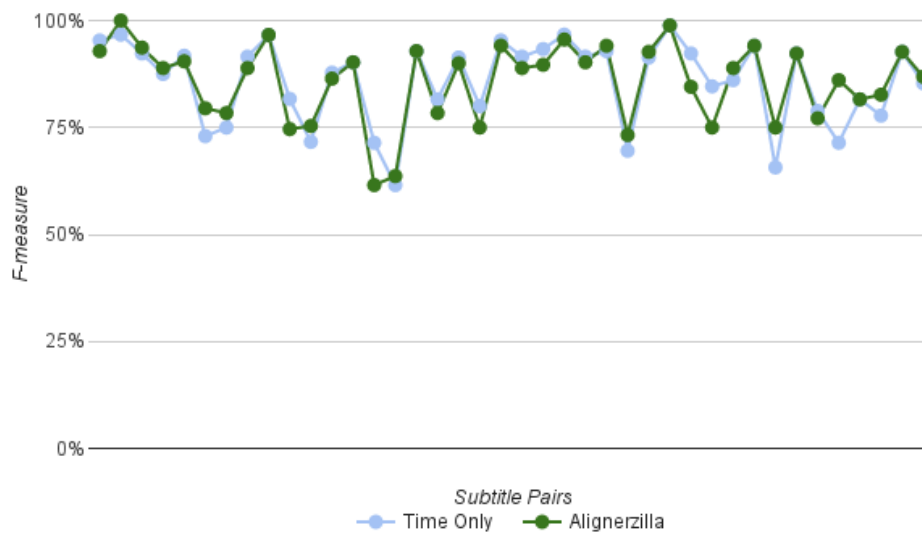


Figure 5.1: Graphical representation of the F-measure calculated for each subtitle pair aligned using the time based approach and comparison with *Alignerzilla*.

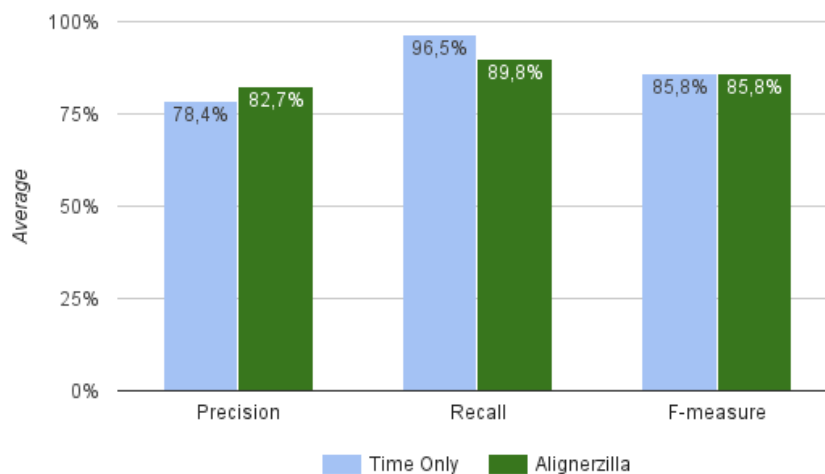


Figure 5.2: Graphical representation of the results obtained with the aligner only using the time based approach and comparison with *Alignerzilla*.



Movie	TOA			Alignerzilla		
	Precision	Recall	F-measure	Precision	Recall	F-measure
12 Years a Slave (2013)	93.2%	97.6%	95.3%	-0.3	-4.8	-2.5
A Beautiful Mind (2001)	93.6%	100.0%	96.7%	+6.4	0.0	+3.3
Alien (1979)	85.7%	100.0%	92.3%	+4.5	-2.6	+1.4
Despicable Me (2010)	83.3%	92.1%	87.5%	+6.7	-4.3	+1.4
Fifty Shades of Grey (2015)	84.8%	100.0%	91.8%	-0.3	-2.6	-1.3
Fight Club (1999)	58.7%	96.4%	73.0%	+10.2	-2.5	+6.5
Finding Nemo (2003)	60.0%	100.0%	75.0%	+5.9	-3.3	+3.4
Forrest Gump (1994)	86.4%	97.4%	91.6%	+5.9	-11.7	-2.7
Gladiator (2000)	93.5%	100.0%	96.6%	+2.1	-2.3	0.0
Gravity (2013)	70.7%	96.7%	81.7%	+0.7	-18.5	-7.1
Inside Out (2015)	60.0%	88.9%	71.6%	+12.2	-10.1	+3.7
Interstellar (2014)	80.0%	97.3%	87.8%	+5.4	-9.8	-1.4
Law Abiding Citizen (2009)	86.0%	94.9%	90.2%	+2.0	-2.4	0.0
Mad Max: Fury Road (2015)	62.5%	83.3%	71.4%	-1.9	-20.8	-9.9
Million Dollar Baby (2004)	47.6%	87.0%	61.5%	+7.6	-12.0	+2.1
Raiders of the Lost Ark (1981)	90.7%	95.1%	92.9%	+4.4	-4.4	0.0
Real Steel (2011)	72.1%	93.9%	81.6%	+4.2	-13.4	-3.2
Shakespeare in Love (1998)	88.1%	94.9%	91.4%	-0.3	-2.6	-1.4
Short Term 12 (2013)	66.7%	100.0%	80.0%	+2.6	-18.2	-5.0
Slumdog Millionaire (2008)	91.1%	100.0%	95.3%	-0.2	-2.4	-1.2
Star Wars: Episode II (2002)	84.4%	100.0%	91.6%	+1.3	-7.7	-2.7
The Avengers (2012)	89.4%	97.7%	93.3%	-4.6	-2.6	-3.7
The Dark Knight (2008)	95.7%	97.8%	96.7%	-0.1	-2.2	-1.1
The Day After Tomorrow (2004)	86.4%	97.4%	91.6%	+1.7	-4.9	-1.3
The Departed (2006)	92.9%	92.9%	92.9%	+2.4	+0.2	+1.3
The Fast and the Furious (2001)	53.3%	100.0%	69.6%	+5.8	-3.7	+3.7
The Godfather (1972)	84.1%	100.0%	91.4%	+8.6	-7.3	+1.3
The Hobbit (2012)	97.7%	100.0%	98.9%	0.0	0.0	0.0
The Hunger Games (2012)	87.8%	97.3%	92.3%	-4.5	-11.6	-7.8
The Imitation Game (2014)	76.7%	94.3%	84.6%	-1.7	-19.3	-9.6
The Intouchables (2011)	77.3%	97.1%	86.1%	+12.7	-9.3	+2.8
The Lego Movie (2014)	88.9%	100.0%	94.1%	+4.1	-4.8	0.0
The Lion King 1 1/2 (2004)	51.2%	91.7%	65.7%	+14.7	-4.6	+9.3
The Lord of the Rings(2001)	92.3%	92.3%	92.3%	+5.0	-4.5	0.0
The Notebook (2004)	65.1%	100.0%	78.9%	+4.1	-12.9	-1.7
The Prestige (2006)	56.8%	96.2%	71.4%	+20.5	+1.0	+14.6
The Usual Suspects (1995)	68.9%	100.0%	81.6%	+4.9	-8.8	0.0
Up (2009)	66.7%	93.3%	77.8%	+10.8	-4.8	+4.9
V for Vendetta (2005)	90.5%	95.0%	92.7%	+2.2	-2.3	0.0
Whiplash (2014)	74.4%	100.0%	85.3%	+6.1	-5.7	+1.5
<b>Average</b>	<b>78.4%</b>	<b>96.5%</b>	<b>85.8%</b>	<b>+4.3</b>	<b>-6.6</b>	<b>-0.06</b>
Standard Deviation	14.11	3.98	9.72	5.15	5.68	4.59

Table 5.4: Results obtained with Time Only Aligner and comparison with *Alignerzilla*.

By analyzing the results, we can see that on average, TOA and *Alignerzilla* perform very similarly. For instance, **the average F-measure only changes by 0.06 percentage points (pp)**. We can also observe that, in some cases, the TOA obtains better results and in other cases, the opposite occurs. In fact, 47.5% of the results were better when TOA was used, while *Alignerzilla* improved 45% of the alignments.

When we look at the values obtained for Precision and Recall we can see that *Alignerzilla* has a better Precision average, while TOA has a better Recall average. This happens because the implementation of the time based aligner always outputs an alignment, reducing drastically the number of FN, thus increasing Recall. While the implemented similarity method only outputs an alignment when a certain level of confidence is attributed to that alignment.

Before comparing these two aligners, we expected *Alignerzilla* to improve all the results since it combines both aligning techniques. However, the obtained results proved us wrong. This can be explained because when the similarity approach is used it means timing information was not enough to provide a trustworthy alignment. In those cases, we consider that the similarity approach will provide us with a better alignment. However, as we have stated before, the similarity approach is not as reliable as the time based technique and, therefore, it can produce erroneous alignments. Furthermore, the used dictionary



only contains a translation for each word, but some words may have different meanings depending on the context they appear. For example, the English word *accent* can be translated to Portuguese as *acento*, *acentuar* or *sotaque* depending on the context. However, the used dictionary does not comprise this fact, which limits the performance of the similarity based technique.

So, this means that **on subtitles where the time based approach is enough to find alignments, *Alignerzilla* may not provide improvements**. The main conclusion we can take from this comparison is that the performance of each aligner depends on the subtitle pairs and their characteristics and that the similarity based technique may still be improved.



### 5.3 Comparison with state-of-the-art aligner

In order to see how the *Alignerzilla* performed relatively to a state-of-the-art subtitle aligner, we have decided to compare it with Tiedemann aligner [4]. Tiedemann is one of the most relevant authors in this area and has, recently, made available a great collection of corpus created from movie subtitles[14]<sup>1</sup>.

To compare Tiedemann subtitle aligner with the developed one, we have followed the evaluation methodology proposed in the beginning of Chapter 5. The goal of this comparison was to see if the developed aligner outperformed a state-of-the-art aligner and if so, to quantify the improvements.

To do so, we have downloaded *subalign*<sup>2</sup> from Tiedemann repository. Then, following the indicated steps, we have pre-processed the dataset subtitles using the tools *dos2unix* and *srt2xml*. Afterwards, we have used *stralign* with the required options in order to align the subtitles in the best possible way. We have replicated the options of the approach that provided better results for Tiedemann (called *time3* by the author).

The obtained results and the comparison between Tiedemann aligner and *Alignerzilla* can be seen in Table 5.4. To better visualize the differences, Figures 5.3 and 5.4 depict the results of both aligners.

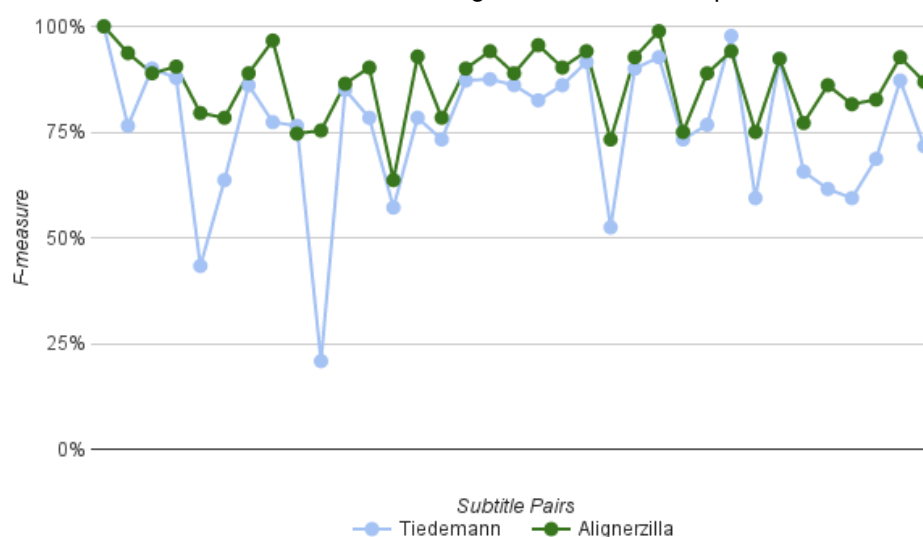


Figure 5.3: Graphical representation of the F-measure calculated for each subtitle pair aligned using Tiedemann [4] and comparison with *Alignerzilla*.

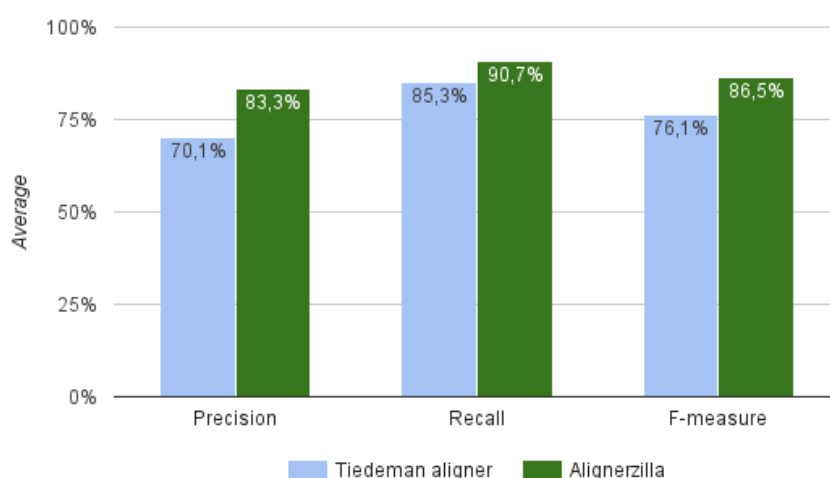


Figure 5.4: Graphical representation of the results obtained with the aligner only using Tiedemann [4] and comparison with *Alignerzilla*.

<sup>1</sup><http://opus.lingfil.uu.se/OpenSubtitles2016.php>

<sup>2</sup><https://bitbucket.org/tiedemann/subalign>



Movie	Tiedemann aligner			Alignerzilla improvement		
	Precision	Recall	F-measure	Precision	Recall	F-measure
12 Years a Slave (2013)						
A Beautiful Mind (2001)	100.0%	100.0%	100.0%	0.0	0.0	0.0
Alien (1979)	66.7%	89.7%	76.5%	+23.6	+7.7	+17.2
Despicable Me (2010)	85.7%	94.7%	90.0%	+4.3	-6.9	-1.1
Fifty Shades of Grey (2015)	80.0%	97.3%	87.8%	+4.4	+0.1	+2.7
Fight Club (1999)	39.4%	48.1%	43.3%	+29.5	+45.8	+36.2
Finding Nemo (2003)	50.0%	87.5%	63.6%	+15.9	+9.2	+14.7
Forrest Gump (1994)	85.0%	87.2%	86.1%	+7.3	-1.5	+2.8
Gladiator (2000)	64.4%	96.7%	77.3%	+31.1	+1.1	+19.3
Gravity (2013)	66.7%	89.7%	76.5%	+4.8	-11.5	-1.8
Inside Out (2015)	14.3%	38.5%	20.8%	+57.9	+40.3	+54.5
Interstellar (2014)	81.0%	89.5%	85.0%	+4.4	-2.0	+1.4
Law Abiding Citizen (2009)	69.0%	90.6%	78.4%	+19.0	+1.9	+11.9
Mad Max: Fury Road (2015)						
Million Dollar Baby (2004)	46.2%	75.0%	57.1%	+9.1	0.0	+6.5
Raiders of the Lost Ark (1981)	72.5%	85.3%	78.4%	+22.6	+5.4	+14.5
Real Steel (2011)	65.0%	83.9%	73.2%	+11.3	-3.3	+5.1
Shakespeare in Love (1998)	77.3%	100.0%	87.2%	+10.5	-7.7	+2.8
Short Term 12 (2013)						
Slumdog Millionaire (2008)	87.5%	87.5%	87.5%	+3.4	+10.1	+6.6
Star Wars: Episode II (2002)	81.0%	91.9%	86.1%	+4.8	+0.4	+2.8
The Avengers (2012)						
The Dark Knight (2008)	89.2%	76.7%	82.5%	+6.4	+18.8	+13.1
The Day After Tomorrow (2004)	79.1%	94.4%	86.1%	+9.0	-1.9	+4.2
The Departed (2006)	86.4%	97.4%	91.6%	+8.9	-4.4	+2.6
The Fast and the Furious (2001)	38.1%	84.2%	52.5%	+21.0	+12.1	+20.8
The Godfather (1972)	83.7%	97.3%	90.0%	+9.0	-4.6	+2.7
The Hobbit (2012)	92.7%	92.7%	92.7%	+5.0	+7.3	+6.2
The Hunger Games (2012)						
The Imitation Game (2014)	66.7%	81.3%	73.2%	+8.3	-6.3	+1.8
The Intouchables (2011)	66.7%	90.3%	76.7%	+23.3	-2.5	+12.2
The Lego Movie (2014)	95.6%	100.0%	97.7%	-2.5	-4.8	-3.6
The Lion King 1 1/2 (2004)	47.5%	79.2%	59.4%	+18.4	+7.9	+15.6
The Lord of the Rings(2001)	90.0%	94.7%	92.3%	+7.3	-6.9	0.0
The Notebook (2004)	55.3%	80.8%	65.6%	+14.0	+6.3	+11.5
The Prestige (2006)	48.8%	83.3%	61.5%	+28.5	+13.8	+24.5
The Usual Suspects (1995)	70.4%	51.4%	59.4%	+3.4	+39.8	+22.2
Up (2009)	59.0%	82.1%	68.7%	+18.5	+6.4	+14.0
V for Vendetta (2005)	85.0%	89.5%	87.2%	+7.7	+3.2	+5.5
Whiplash (2014)	66.7%	77.4%	71.6%	+13.8	+16.9	+15.2
<b>Average</b>	<b>70.1%</b>	<b>85.3%</b>	<b>76.1%</b>	<b>+13.3</b>	<b>+5.4</b>	<b>+10.4</b>
Standard Deviation	18.95	14.11	16.56	11.61	13.45	11.68

Table 5.5: Results obtained with Tiedemann aligner [4] and comparison with *Alignerzilla*.

While aligning the dataset subtitles using Tiedemann aligner, some of the subtitle files failed to be processed. Thus, only 35 subtitle pairs were used in this comparison.

By comparing the results on Table 5.5 we can see that, in average, the developed aligner performs slightly better than Tiedemann *subalign* in every metric used. In fact, **the average F-measure of our alignments is better by 10.4pp**. The obtained values are a good indicator that make us confident in the performance of *Alignerzilla*. While analyzing the results for each movie, we can see that Tiedemann solution still has better results in some of the subtitle pairs. **Approximately 8.6% of the subtitle pairs from the dataset have a better F-measure when aligned with *subalign***. This means that our aligner still has room for improvements that should be looked into in future work.

However, we must recall that these results are relative to our dataset, that only comprises alignments between English and Portuguese subtitles. We cannot predict how the results would be if a different dataset was used.



## 5.4 Summary

By calculating the correlation between the obtained results and the time offsets registered in the subtitle pairs, we have observed that the lack of synchronization of subtitle pairs tends to negatively influence the results. Also we have seen that the similarity alignment technique does not provide improvements in every case. This can be explained by some limitations that our implementation still has. By comparing *Alignerzilla* with a state-of-the-art aligner, we have observed that, in average, the developed aligner obtains better results for the used dataset.

In Chapter 6 some conclusions are taken from the obtained results. Furthermore, some possible paths on how to improve the developed system are presented.



## Chapter 6

# Conclusions

In this document we addressed the creation of parallel corpora from movie subtitles, that could be used as training data in NLP systems. In order to accomplish the proposed objectives, it was necessary to research the state-of-the-art in this topic and several approaches were explored. While researching information, many articles were found about corpus alignment. However, since the corpora we are using are not regular corpora – as we have seen, subtitle files have specific characteristics that we can make use of –, some of those articles were not described. Nevertheless, we have explored what has been done in the subtitle alignment area and we have concluded that there are two main techniques available to use when aligning subtitle files. These two techniques are based in the *timing information* and *textual similarity*, respectively. Taking into account the work done by other authors and considering the pros and cons of each approach, we have decided to combine those two techniques to obtain an aligner capable of aligning movie subtitles better than state-of-the-art aligners. In the development of *Alignerzilla* an effort was made in order to keep the aligner generic, so that it works with a high variety of languages. However, our focus was in correctly aligning English and Portuguese subtitles.

After developing *Alignerzilla* we needed to compare it with existing solutions so that we could evaluate its performance. In order to do this, we have built a movie subtitle dataset that would be used to test the aligner. Furthermore, we have created a web application where annotators would help us build reference alignments that could be used to evaluate any subtitle aligner.

After obtaining reference alignments for 40 movie subtitles we have followed the evaluation methodology described in Chapter 5 and analyzed the results.

By calculating the correlation between time offsets and our results, we have concluded that the results were influenced by the time offsets between subtitles. This was expected and during the development process we have tried to find a reliable way to synchronize the subtitle pairs before aligning them. However, through experimentation, we have observed that our attempt to synchronize subtitles was not improving the results and we have decided to align the subtitles despite the timing offsets registered. If a proper way to synchronize subtitle files were used, we believe the results could be improved.

By comparing *Alignerzilla* with a version of the developed aligner that only uses timing information to align the subtitle files, we have concluded that the similarity based approach still needs improvements. While the combination of approaches improved some of the results, that did not happen in every case. We are aware that similarity based techniques are less reliable than time based techniques, but we believe we could achieve better results if our similarity aligner implementation was enhanced. As we have stated in previous chapters, our implementation of the similarity based approach has some limitations. The fact that it only supports 1:1 alignments and that the used dictionary only contains a translation for each word – a bilingual lexicon – are limitations that probably are influencing negatively the obtained



results.

After the comparison between *Alignerzilla* and Tiedemann *subalign*, we have observed that, on average, the developed aligner outperformed Tiedemann subtitle aligner. However, as we have stated before, these results are relative to our dataset and we cannot declare that our aligner is better than Tiedemann's in every scenario. We did not test *Alignerzilla* with different languages. Despite improving most of the results, we can see that *subalign* still has better results in some cases. We believe this happens because of the aforementioned limitations.

In summary, we believe the developed aligner achieves the proposed objectives, despite still having room for improvements. Our aligner is capable of creating parallel corpora that can be used by NLP systems as training data. Nevertheless, we know that the quality of the created parallel corpora depends on the quality of the used subtitles – i.e., if a subtitle file contains bad translations for the original movie dialogs, the obtained alignments will not represent good translations.

Furthermore, the creation of the reference alignments was a useful contribution to this topic because it makes possible for everyone to test a subtitle aligner.

## 6.1 Future Work

As described in the previous sections, we have identified some limitations in our implementation. Hence, in this section we propose some paths that could be looked into to enhance the aligner performance.

In the pre-processing phase, one of the limitations found were the language dependent abbreviations (such as “Mr.”, “Dr.” or “St.”) that may appear in a subtitle file. These abbreviations influence the sentence splitting process when they are not taken into account. If a complete list of abbreviations for every existing language was used in the pre-processing phase, this should grant the sentence splitter means to work correctly. Besides, by doing this, there would not be the need of adding these special cases in the *configurations file*, one by one, thus simplifying the alignment process.

As we have stated before, the time based alignment technique depends on the time synchronization of subtitle files. Thus, if a reliable way to synchronize the subtitles was implemented, this could represent an enhancement to the current aligner. Others authors have done this and it should be looked into in future work.

Another possible path to improve the aligner performance includes fixing the limitations present in the implemented similarity based alignment technique. One possible way to improve the similarity based alignments is to include better dictionaries. For instance, a dictionary with more entries, so that more words could be translated. Furthermore, the dictionary used could provide more than one translation per word. This would require the aligner to be modified, in order to make use of the different translations possible for a given word. Tsiartas et al. [2] already take this into account.

In what regards evaluation, it would be valuable if *Alignerzilla* was tested with a different dataset and different languages. We already know how the aligner performs with the dataset and reference we have created, but until further tests are done we cannot predict how the aligner will perform. One of the possible use cases for a subtitle aligner is the alignment between subtitles in the same language, in order to extract paraphrases and synonyms. However, the dataset and reference alignments created do not comprise this case and it would be interesting to test *Alignerzilla* in this scenario. It would be interesting to see how the developed aligner performed in multiple scenarios and to explore the different use cases possible. By testing *Alignerzilla* in different scenarios we would obtain more data to correctly evaluate the developed aligner, but also more information about its limitations. This would provide us the possibility to keep improving the aligner by solving the found problems.



Besides evaluating the system in different ways, it would also be valuable to test how the output data – i.e., parallel corpora – would help SMT systems. For instance, it would be interesting to use data provided by our aligner as training data for a SMT system and see how that would improve its translations.







# Bibliography

- [1] Y. Lü, J. Huang, and Q. Liu, “Improving statistical machine translation performance by training data selection and optimization.”
- [2] A. Tsiartas, P. K. Ghosh, P. G. Georgiou, and S. Narayanan, “Context-driven automatic bilingual movie subtitle alignment,” in *Proceedings of Interspeech 2009*, (Brighton, UK), Sept. 2009.
- [3] L. Caroline, S. Kamel, and L. David, “Building parallel corpora from movies.” The 4th International Workshop on Natural Language Processing and Cognitive Science - NLPCS 2007, Jun 2007. Funchal, Madeira, Portugal.
- [4] J. Tiedemann, “Improved sentence alignment for movie subtitles,” in *In Proceedings of RANLP, Borovets*, 2007.
- [5] J. Tiedemann, “Synchronizing translated movie subtitles,” in *LREC*, 2008.
- [6] E. Itamar and A. Itai, “Using movie subtitles for creating a large-scale bilingual corpora,” in *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC’08)* (B. M. J. M. J. O. S. P. D. T. Nicoletta Calzolari (Conference Chair), Khalid Choukri, ed.), (Marrakech, Morocco), European Language Resources Association (ELRA), May 2008. <http://www.lrec-conf.org/proceedings/lrec2008/>.
- [7] W. A. Gale and K. W. Church, “A program for aligning sentences in bilingual corpora,” in *Proceedings of the 29th Annual Meeting on Association for Computational Linguistics*, ACL ’91, (Stroudsburg, PA, USA), pp. 177–184, Association for Computational Linguistics, 1991.
- [8] M. Fishel, Y. Georgakopoulou, S. Penkale, V. Petukhova, M. Rojc, M. Volk, and A. Way, “From subtitles to parallel corpora,” in *Proceedings of the 16th Annual Conference of the European Association for Machine Translation EAMT*, pp. 3–6, 2012.
- [9] M. Mangeot and E. Giguet, “Multilingual aligned corpora from movie subtitles,” 2005.
- [10] J. Tiedemann, *Bitext Alignment*. Synthesis digital library of engineering and computer science, Morgan & Claypool, 2011.
- [11] P. F. Brown, J. C. Lai, and R. L. Mercer, “Aligning sentences in parallel corpora,” in *Proceedings of the 29th annual meeting on Association for Computational Linguistics*, pp. 169–176, Association for Computational Linguistics, 1991.
- [12] T. Songyot and D. Chiang, “Improving word alignment using word similarity,” in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1840–1845, 2014.



- [13] A. K. Singh and S. Husain, “Comparison, selection and use of sentence alignment algorithms for new language pairs,” in *Proceedings of the ACL Workshop on Building and Using Parallel Texts*, ParaText '05, (Stroudsburg, PA, USA), pp. 99–106, Association for Computational Linguistics, 2005.
- [14] P. Lison and J. Tiedemann, “Opensubtitles2016: Extracting large parallel corpora from movie and tv subtitles,”



# Appendix A

## Run instructions

The following sections explain how to use the pre-processing scripts and *Alignerzilla*. The same instructions are included in the repository *read me* file.

### 1) How use pre-processing scripts

The **pre-processing-scripts** require an environment with **Bash** and **Perl**.

- First, navigate to `pre-processing-scripts/` directory;
- Then, you should place the raw subtitle files that you want to pre-process inside the folder `input/`;
- Finally, execute the following command: `$ sh normalizeAll.sh` or `$ ./normalizeAll.sh`.

If everything runs as expected, you should now have all the pre-processed subtitle files inside 'output/' folder.

**Note:** the run time of the *pre-processing-scripts* depends on the used machine and how many subtitles you want to normalize. It might take a while for many subtitles.

### 2) How to use Alignerzilla

**Alignerzilla** requires an environment with **Apache Ant** and **Java 7**. Also, make sure the subtitles you provide as input are already pre-processed by **pre-processing-scripts**.

- First, make sure you've followed the instructions in **1**);
- Then, navigate to `alignerzilla/` directory;
- Now, you should place the pre-processed subtitle files inside the folder `data/`;
  - The subtitles placed in `data/` directory **should all be named as** `MovieName-LANGUAGE.srt`. For example: `lotr-EN.srt` or `TheDarkKnight-PT.srt`.
  - As the goal is to align subtitle pairs, **every movie must have two different subtitle files, with different language tags**. For example: `lotr-EN.srt` and `lotr-PT.srt`. If you want to use same language subtitles `lotr-EN1.srt` and `lotr-EN2.srt` can be used.



- Then, make sure `list.txt` contains a list with the filenames of the subtitles you want to align.
  - One subtitle file per line.
  - If you want to aligner every subtitle in `data/` you can simply use the following command:  

```
$ ls data/ > list.txt
```

- Finally, execute one of the following commands: `$ sh run-alignerzilla.sh` or `$ ./run-alignerzilla.sh`

If everything runs as expected, you should now have the aligned subtitles inside `results/` folder.

You can configure additional parameters editing `alignerzilla.config`.

**Note:** the run time of *Alignerzilla* depends on the used machine and how many subtitles you want to align. It might take a while for many subtitles.



## Appendix B

# Configurations file

```
#### Alignerzilla Configurations File ####
#####
# Change the values at your own risk to obtain different results

##### ALIGNER CONFIGURATION #####
#####

# Use only time based alignments?
# Must be a boolean (true or false)
# If "true" does not use sentence similarity to align the subtitles
# Default value is "false" (w/o quotes)
TIME_ONLY = false

# Maximum time difference accepted in an alignment
# Must be between 0.00 e 1.00
# Lower values make the alignments more precise, but will reduce the number of alignments
# Default value is 0.95
TIME_THRESHOLD = 0.95

# Maximum distance accepted in an alignment; Will be ignored if TIME_ONLY = true;
# Must be between 0.00 e 1.00
# Lower values make the alignments more precise, but will reduce the number of alignments
# Default value is 0.55
DISTANCE_THRESHOLD = 0.55

# List of special cases that use punctuation but should not interrupt a sentence.
# Examples: "Mr. Dr. St."
# Must follow this format: "case1|case2|case3|case4" without the quotes.
# Will not work if other format is used.
SPECIAL_PUNCTUATION_CASES = Mr|mr|Mrs|Dr|Av|St|Sr|Sra|D|R|Exmo

# Print alignment confidence next to the alignments?
# Must be a boolean (true or false)
# Default value is "false" (w/o quotes)
PRINT_CONFIDENCE = false

##### TRANSLATOR CONFIGURATION #####
#####

# Use a dictionary?
# Must be a boolean (true or false)
# Use "false" if same language alignments
# Default value is "true" (w/o quotes).
USE_DICTIONARY = true

# Location of the dictionary file; Will be ignored if USE_DICTIONARY = false.
# Must be relative to project root folder
# Must contain a text file with two rows, as follow:
# word [tab] translation
# Default value is "lib/translatorEN-PT.txt" (w/o quotes)
DICTIONARY_FILE = lib/translatorEN-PT.txt
```









## Appendix C

# Web Application

O cinema ao serviço do Processamento da Língua Natural | Tese de Mestrado - Luís Rosado

Olá! Antes de mais, obrigado por aceitares ajudar-me na minha tese de mestrado.  
Peço-te que me ajudes a encontrar **correspondências entre frases** tiradas de legendas de filmes.  
Vou mostrar-te frases em Inglês e Português e deves seguir os passos de forma a indicar-me correspondências entre elas.  
O tempo que demoras não é importante, mas 15 minutos devem ser suficientes para terminares.  
Antes de começares observa os seguintes exemplos, para melhor entenderes o que pretendo.

 Frases em Inglês

 Frases em Português

**Exemplo 1: correspondência 1 frase com 1 frase**

The stated goal of the Soviets is global Communism.	Os matemáticos venceram a guerra.
Mathematicians won the war.	Não, tem de ter tudo só para ele.
Mathematicians, like you.	Ele ficou aborrecido.

**Exemplo 2: correspondência 2 frases com 1 frase**

Helped rid the world of Fascism.	Chamava-se Johnny Walker.
The name's Bender.	Chamo-me Bender, da Física Atómica.
Atomic physics.	E você?

Podem surgir frases sem correspondência do outro lado.      Ou seja, 1 frase com 0 frases.

Avançar 1 de 4

Figure C.1: Screenshot of the instructions and examples page of the developed web application.



[illegible]

54







O cinema ao serviço do Processamento da Língua Natural | Tese de Mestrado - Luís Rosado

Selecione 15 correspondências, clicando nas frases em Inglês e nas suas correspondentes em Português.

Valida as correspondências, uma a uma, clicando no botão verde. Evita escolher muitas frases consecutivas.

- Caso uma frase não tenha correspondente na outra língua, podes validar sem selecionar nenhuma do outro lado.
- Caso uma frase necessite de mais de uma frase na outra língua para ser traduzida, seleciona-as antes de validar.

Frases em Inglês

Validar

Frases em Português

Hansen's used to being picked first.

Yeah, he's wasted on math.

He should be running for president.

There could be a mathematical explanation for how bad your tie is.

Thank you.

Neilson, symbol cryptography.

Neils here broke a Jap code.

Helped rid the world of Fascism.

At least that's what he tells the girls, eh, Neils?

The name's Bender.

Atomic physics.

- And you are?

- Am I late?

- Yes, Mr. Sol.

- Oh, good.

Hi

Habitou-se a ser escolhido primeiro.

É mal empregue em matemática.

Devia candidatar-se a presidente.

Pode haver uma explicação matemática para a fealdade da sua gravata.

Obrigado.

Neilson, de Criptografia.

Decifrou um código japonês e ajudou a acabar com o fascismo.

Pelo menos, é o que diz às miúdas, não é, Neils?

Chamo-me Bender, da Física Atómica.

- E você?

- Cheguei atrasado?

- Sim, Sr. Sol.

- Ótimo.

Olá.

- Richard Sol.

Avançar

2 de 4

56







